Klangexperimente mit der Soundkarte

Harald Wiltsche

Einleitung

Welche Veranlassung sollte es geben, mit den einfachen Mitteln einer Programmierspache der Soundkarte eines Personalcomputers Töne und Klänge entlocken zu wollen, findet man doch in Zeiten der Multimedia-PCs eine Vielzahl fertiger, leistungsfähiger, zum Teil sogar als Shareware erhältlicher Musikprogramme?

Abgesehen davon, dass es für jeden interessierten Programmierer reizvoll ist, Peripheriegeräte wie z.B. Plotter, Drucker oder Maus mittels selbstgeschriebener kleiner Pascal- oder Basicprogramme anzusteuern, gibt es vor allem für die Tonausgabe über die Soundkarte zahlreiche interessante Anwendungsbereiche im Bereich des Physik- oder Musikunterrichts, als Grundlage für Fachbereichsarbeiten oder als Bestandteil fächerübergreifender Projekte.

Im Physikunterricht wäre es zum Beispiel bei der Demonstration der Schwebung eine interessante Ergänzung zum Experiment mit den beiden Stimmgabeln, wenn man (als Lehrer oder auch als Schüler) die Möglichkeit hätte, den Effekt mit Frequenzpaaren, die man beliebig wählen kann, am Computer nachzuvollziehen. Gesetzmäßigkeiten und Zusammenhänge könnten so von den Schülern experimentell erforscht werden.

Viele interessante Phänomene der Schallwahrnehmung, wie das Hören von Residuumtönen, der Maskierungseffekt oder das Frequenzauflösungsvermögen unseres Gehörs erfordern es, ein- oder zweistimmige Tonfolgen erzeugen zu können. Auch zum Vergleich der verschiedenen musikalischen Stimmungssysteme, wie der diatonschen und der wohltemperierten Stimmung, ist es notwendig, Töne ganz bestimmter Frequenzen wiederzugeben.

Für all diese Zwecke erweisen sich die eingangs erwähnten Musikprogramme größtenteils als unbrauchbar. Vielmehr hat man aber die Möglichkeit, mit erstaunlich einfachen, selbstgeschriebenen Basic- oder Pascalprogrammen gezielt an die Problemstellung herangehen.

War es aber bei einem der "Urväter" der Heimcomputer, dem Commodore 64, mit Hilfe des im Gerät integrierten Soundchips geradezu trivial und ohne nennenswerten Aufwand möglich, zweistimmige Klänge in verschiedenen Klangfarben zu erzeugen, so gestaltet sich dies beim PC nicht ganz so einfach.

Mit der in den Programmiersprachen Basic und Pascal verfügbaren SOUND-Anweisung ist es leider nur möglich, einstimmige Töne zu erzeugen, die direkt über den PC-Speaker ausgegeben werden. Vordefinierte Funktionen oder Prozeduren zur Ansteuerung der Soundkarte sind nicht verfügbar.

In VisualBasic ist es wiederum recht einfach, Wave-Dateien zu bearbeiten; zur Erzeugung einzelner Sinustöne wäre man jedoch gezwungen, in die undurchschaubaren Tiefen der Windowsprogrammierung vorzudringen.

Das Programmieren der Soundkarte macht es erforderlich, sich einen kleinen Überblick über deren Funktionsweise zu schaffen. Um auch ohne dieses Vorwissen auf den Geschmack zu kommen, und um die Lust am selbständigen Experimentieren zu wecken habe ich für Neugierige ein Pascal-Programmbeispiel zur Schwebung den Erklärungen vorangestellt. Die späteren Erläuterungen beziehen sich soweit wie möglich auf die Befehlszeilen des Programms (die Zeilennummern sollen dabei die Orientierung erleichtern).

```
1: PROGRAM Schwebung;
2: USES CRT;                {für Delay}
3: VAR  Status:Byte;
4:      i:Integer;
5: CONST
6:   Adr_Reg:Word=$388;    {Adress-Register}
7:   Dat_Reg:Word=$389;    {Daten-Register}

{---------- Ausgabe-Prozedur -----------}

8:  Procedure Sound(Adr,Data:Byte);
9:  Begin
10:   Port[Adr_Reg]:=Adr;
11:   Delay(1);             {Verzögerung}
12:   Port[Dat_Reg]:=Data;
13:   Delay(1);
14: End;

{----------- Hauptprogramm ------------}

15: Begin

{--- Einstellung der Generatorzellen ----}

16:  Sound($23,$01); {EGT=0}
17:  Sound($24,$01);
18:  Sound($E3,$00); {Wellenform: Sinus}
19:  Sound($E4,$00);
20:  Sound($43,$00); {Level: 0db Dämpfung}
21:  Sound($44,$00);
22:  Sound($63,$F0); {Attack=F, Decay=0}
23:  Sound($64,$F0);
24:  Sound($83,$F1); {Sustain=F, Release=1}
25:  Sound($84,$f1);

{--- Einstellung der Stimmen (Kanäle) ---}

26:  Sound($A0,$44); {LSBs Fnummer Stimme 1}
27:  Sound($B0,$32); {Block,MSBs Fn Stimme1}
28:  Sound($A1,$3f); {LSBs Fnummer Stimme 2}
29:  Sound($B1,$32); {Block,MSBs Fn Stimme2}

{------- Ausschalten der Stimmen --------}

30:  Sound($B0,$12);    {Key-Off,Stimme1}
31:  Sound($B1,$12);    {Key-Off,Stimme2}
32: End.
```

Prof. Mag. Harald Wiltsche unterrichtet an der Pädagogischen Akademie des Bundes, Hubertusstr. 1, 9020 Klagenfurt

Das Programm müßte unter jeder TurboPascal-Version ab 5.0 funktionieren. Aufgrund der Wahl der Portadressen dürfte es auch keine Probleme mit verschiedenartigen Soundkarten geben. Sollte die Ausgabe verzerrt oder zu laut erfolgen, sind in den Programmzeilen 20 und 21 die Inhalte der Register 43h(exadezimal) und 44h zu ändern. Bei zu leiser Ausgabe liegt das Problem an einem zu geringen Pegel des NF-Verstärkers der Soundkarte. Hier hilft ein Griff zum Lautstärkeregler der (dann notwendigen) Aktivlautsprecher.

Vorschlag zum Experimentieren: die Einstellung der Tonhöhen erfolgt in den Programmzeilen 26 bis 29. Eine Änderung der Tonhöhe kann durch Variieren der Inhalte der Register A0h (Zeile 26) und A1h (Zeile 28) erzielt werden. Dazu ist jeweils der zweite Wert innerhalb der Klammer zu verändern.

Ein Übertragen des Programms auf QuickBASIC wäre relativ einfach möglich. Dem PORT-Befehl von Pascal entspricht dann die OUT-Anweisung in Basic.

Und nun zu den angekündigten technischen Details:

Grundfunktionen der Soundkarten

Sie können grob in drei Gruppen zusammengefaßt werden:

- Die synthetische Tonerzeugung, wobei die gebräuchlichste Form jene der Frequenzmodulation (FM, FM-Synthese) darstellt.
- Das Sampling auf Basis eines Digitalen Sound Processors (DSP). Darunter versteht man die Analog/Digital- bzw. Digital/Analog-Wandlung beim Aufzeichnen und Wiedergeben von Musiksignalen.
- Der Mixer mit der Aufgabe, die einzelnen Eingangs- und Ausgangsquellen zu steuern, zu verbinden und die Lautstärke zu definieren.

Vielfach verfügen Soundkarten zusätzlich noch über eine MIDI-Schnittstelle (*M*usical *I*nstrument *D*ata *I*nterface) zur Ansteuerung externer elektronischer Musikinstrumente.

Wenn Soundkarten auch von vielen Herstellern angeboten werden, so hat sich doch ein Standard entwickelt, der von der Firma Creative Labs mit ihren "Soundblaster"-Karten vorgegeben wurde. Von nahezu jedem Produkt anderer Herstellerfirmen kann man daher "Soundblaster-Kompatibilität" erwarten.

FM - Synthese

Technische Basis der Tonerzeugung bilden bei praktisch allen Soundkartenherstellern sogenannte OPL-Chips von Yamaha. Am weitesten verbreitet ist der OPL2. Auf diesen Bauteil beziehen sich auch die folgenden Erläuterungen. Mit OPL3 oder OPL4 Chips bestückte Soundkarten sind generell abwärtskompatibel zum OPL2.

Der OPL2 kann in zwei Einstellungen betrieben werden und verfügt dann entweder über neun Melodiestimmen (Kanäle) oder im Percussion-Mode über sechs Melodiestimmen und fünf Rhythmusstimmen.

Prinzipiell beruht die Tonerzeugung beim OPL2 auf Sinusgeneratoren, die als Generatorzellen oder Operatoren bezeichnet werden.

Jede Generatorzelle besteht im wesentlichen aus drei Komponenten:

- dem Oszillator zur Erzeugung einer Schwingung
- dem Hüllkurvengenerator, dessen Funktionsweise noch genauer beschrieben wird
- dem Level-Controller zur Festlegung der Amplitude

Der OPL2 besitzt insgesamt 18 solcher Generatorzellen, je zwei davon werden zu einem Kanal zusammengefaßt, woraus sich auch die Maximalzahl von neun Melodiestimmen ergibt.

Die beiden Generatorzellen eines Kanals können dabei auf zwei verschiedene Arten verknüpft werden:

Bei der additiven Synthese werden die Ausgangssignale einfach addiert:

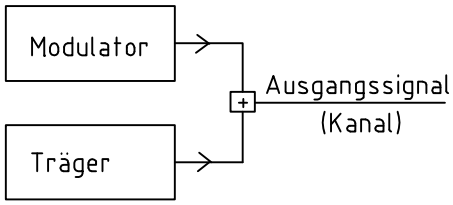


Abb. 1: Additive Verknüpfung zweier Generatorzellen: Modulator und Träger sind funktionell gleichwertig.

$$y_1 = a_1 \cdot \sin(2\pi f_1 t),\ y_2 = a_2 \cdot \sin(2\pi f_2 t)$$

$$y = y_1 + y_2 = a_1 \cdot \sin(2\pi f_1 t) + a_2 \cdot \sin(2\pi f_2 t)$$

Geht es darum, bestimmte Klangbilder beispielsweise von Instrumenten nachzubilden, so sind die Möglichkeiten, die sich aus der additiven Synthese von nur zwei Teilschwingungen ergeben, zwangsläufig eher dürftig. Annähernd wirklichkeitsgetreue Klänge sind kaum zu erzielen.

Wesentlich mehr Variationsmöglichkeiten ergeben sich, wenn man die beiden Generatoren hintereinanderschaltet; d.h. das Ausgangssignal des ersten Generators als Eingangssignal des zweiten verwendet. Diese Art der Verknüpfung wird als Frequenzmodulation (oder Frequenzmodulationssynthese) bezeichnet.

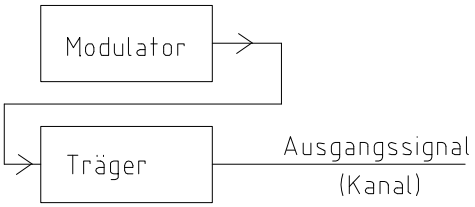


Abb. 2: FM-Synthese: Das Signal der ersten Generatorzelle (Modulator) beeinflußt den Träger.

Obwohl zum Aufbau einer Melodiestimme üblicherweise zwei Generatorzellen verwendet werden, läßt sich ein reiner Sinuston auch mit nur einer Generatorzelle erzeugen. Dabei muß es sich aber dann unbedingt um eine Trägerzelle handeln. Bei der alleinigen Ansteuerung einer Modulatorzelle wird kein Ausgangssignal erzeugt.

Im Programmbeispiel wurden zur Erzeugung einer Schwebung ebenfalls nur die jeweiligen Trägerzellen für die erste

und zweite Stimme (das sind die Generatorzellen 3 und 4) verwendet.

Eine Alternative zur Erzeugung der Schwebung durch Programmierung zweier Kanäle mit je einer Trägerzelle wäre die Verwendung eines Kanals mit additiv verknüpfter Modulator- und Trägerzelle (z.B. Generatorzelle 0 zusammen mit Generatorzelle 3).

Ansteuerung der Generatorzellen

Bei der Programmierung des OPL-Chips geht es darum, in bestimmte Speicherplätze (Register) des Chips Werte einzutragen, die daraufhin seine Funktionsweise bestimmen. Der Chip besitzt ca. 120 dieser Register; nur ein kleiner Teil davon wird für die Programmierung benutzt. Wie gelingt es nun, Werte in die Register des OPL-Chips zu schreiben?

Wie auch bei anderen Peripheriegeräten erfolgt der Zugriff auf die Soundkarte über sogenannte Portadressen. Es sind dies den entsprechenden Geräten zugeordnete Speicherplätze von Ein/Ausgabebausteinen des PCs.

Zur Programmierung des OPL2 benötigt man zwei Ports. Ein Port (jenes mit der niedrigeren Adresse) ist dabei einem Adreßregister des OPL2, das andere einem Datenregister des OPL2 zugeordnet. Wie schon aus der Bezeichnung hervorgeht, wird in das Adreßregister eine Speicheradresse, und in das Datenregister der für diese Adresse bestimmte Wert eingetragen.

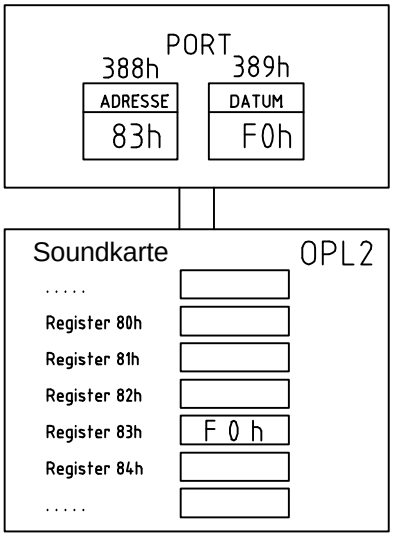


Abb. 3: Zusammenhang zwischen den Ports und den Registern des OPL2

Für die Wahl der beiden Ports hat man zwei Möglichkeiten:

- die Ports 388h und 389h (Portadressen , wie auch Registerinhalte werden i.a. in hexadezimaler Form, gekennzeichnet durch ein nachgestelltes h, angegeben). Diese beiden Adressen waren ursprünglich der AdLib-Soundkarte zugeordnet, wurden aber für Soundblasterkarten übernommen und müßten bei fast allen installierten Soundkarten funktionieren.
- Anstelle der Ports 388h und 389h, für die keine Programmanpassungen notwendig sind, kann ein Zugriff auch immer über die Basisadresse (und Basisadresse +1) der jeweils installierten Soundkarte erfolgen. Die Basisadresse kann dabei aus der Datei AUTOEXEC.BAT entnommen werden. Bei der Installation einer Soundkarte wird üblicherweise automatisch eine Umgebungsvariable bestimmt. Der Eintrag lautet dann z.B. SET BLASTER = A220 I5 D1 T1. Der erste Parameter enthält dabei immer die Basisadresse, d.h. in diesem Fall 220h. Der Nachteil der Wahl dieser Portadressen ist der, dass Programme, die für eine unter dieser Basisadresse konfigurierte Soundkarte geschrieben werden, bei Soundkarten, die über andere Basisadressen angesprochen werden, nicht mehr funktionieren.

Im Programmbeispiel wurden die Ports 388h und 389h verwendet (siehe Programmzeile 6 und 7). Mit dem Befehl `PORT[Portadresse]:=Wert` wird der Inhalt der Variablen Wert dem entsprechenden Port zugeordnet. (z.B.: `PORT[$388]:=$83` belegt den Port 388h mit dem Wert 83h; Hexadezimalwerte werden in Pascal durch ein vorgestelltes "$"-Zeichen gekennzeichnet).

Die Register der Generatorzellen

Zur Einstellung jeder der insgesamt 18 verfügbaren Generatorzellen (durchnumeriert von 0 bis 17) werden je fünf Register verwendet. Um nun auf ein ganz bestimmtes Register einer gewählten Zelle zugreifen zu können, verwendet man sogenannte Basis- und Offsetadressen. So besitzt z.B. das Register zur Bestimmung der Sustain- und Releaseparameter der ersten Generatorzelle (Generatorzelle 0) die Adresse 80h (=Basisadresse), jenes der zweiten Generatorzelle (Generatorzelle 1) die Adresse 81h (=Basisadresse + Offset 1) usw.

Welche der Generatorzellen als Modulator, welche als Trägerzelle verwendet werden kann, der entsprechende Offset, sowie die Zuordnung der Zellen zu den 9 Kanälen (Modulator = 1 bedeutet z.B. Modulator für Stimme 1), ergeben sich aus folgender Tabelle:

Generator	Offset	Modulator	Träger
0	00h	1	
1	01h	2	
2	02h	3	
3	03h		1
4	04h		2
5	05h		3
6	08h	4	
7	09h	5	
8	0Ah	6	
9	0Bh		4
10	0Ch		5
11	0Dh		6
12	10h	7	
13	11h	8	
14	12h	9	
15	13h		7
16	14h		8
17	15h		9

Im Programmbeispiel wurde die Generatorzelle 3 als Trägerzelle des ersten Kanals und die Generatorzelle 4 als Träger-

zelle zweiten Kanals verwendet. Zu den Basisregistern wird daher jeweils der Offset 3 bzw. 4 addiert.

Das Basisregister 20h
(Programmzeilen Nr. 16 und 17)

Es dient zur Festlegung des Multiplikationsfaktors, der Hüllkurvenverkürzung und des Hüllkurventyps (EGT, siehe Anhang), sowie von Vibrato und Tremolo.

BASISREGISTER 20h

Bit	Bedeutung
0 - 3	Multiplikationsfaktor
4	Hüllkurvenverkürzung
5	Hüllkurventyp
6	Vibrato
7	Tremolo

Der Multiplikationsfaktor bewirkt eine Vervielfachung der im Basisregister B0h eingestellten Frequenz (Standardeinstellung = 1).

Die Hüllkurvenverkürzung führt zu einer Erhöhung der Attack- und Releasewerte für höhere Frequenzen.

Da im Programmbeispiel nicht die Generatorzelle 0, sondern die Zellen 3 und 4 verwendet werden, sind daher die Register 23h und 24h anstelle des Basisregisters 20h mit Werten zu belegen. Dies passiert in den Programmzeilen 16 und 17. Mit dem Befehl `Sound($23,$01)` wird der Wert 01h in das Register 23h eingetragen. Wie schon aus der Beschreibung des Basisregisters 20h hervorgeht, sind die Register achtstellig, d.h. sie besitzen eine Größe von 8 Bit (1 Byte). Da nun einer Hexadezimalziffer jeweils eine vierstellige Binärzahl (Halbbyte) entspricht, bedeutet der Eintrag 01h eine Belegung der oberen vier Bit des Registers 23h mit 0 0 0 0 und eine Belegung der unteren vier Bit mit 0 0 0 1.

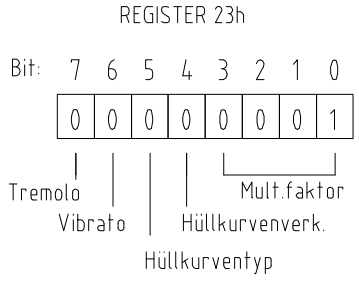


Abb. 4: Belegung des Registers 23h

Die aktuelle Registerbelegung bedeutet somit: Multiplikationsfaktor = 1, keine Hüllkurvenverkürzung, Hüllkurventyp 0, kein Vibrato und keinTremolo.

Wie man beim Multiplikationsfaktor erkennt, können mehrere Bits zusammen den Wert eines Parameters enthalten, es kann aber auch ein einzelnes Bit eine Art Schalterfunktion besitzen (wie z.B. das Bit 7 für Tremolo). Zur Aktivierung des Tremolo müßte als Bit 7 auf 1 gesetzt werden, die Registerbelegung wäre dann `1000 0001` und der entsprechende Programmbefehl müßte in `Sound($23,$81)` geändert werden (81h deshalb, da die vier oberen Bits den Wert `1000` binär = 8h und die unteren vier Bits den Wert `0001` =1h besitzen).

Noch ein Beispiel:
Wollte man zusätzlich ein Vibrato einstellen, wäre dafür das 6-te Bit auf 1 zu setzen, der Registerinhalt daher `1100 0001`, die entsprechende Hexadezimalzahl C1h und der Programmbefehl `Sound($23,$C1)`.

Das Basisregister 40h

dient zur Einstellung der Lautstärke durch Angabe eines Dämpfungsfaktors, wobei Dämpfung = Dämpfungsfaktor * 0,75 dB.

Unabhängig von der gewählten Lautstärke können hohe Töne leiser wiedergegeben werden als tiefe Töne. Die Einstellung erfolgt durch den Parameter "Hochtondämpfung". Im Programm wurde die maximale Amplitude ohne Hochtondämpfung gewählt (Zeilen 20, 21).

BASISREGISTER 40h

Bit	Bedeutung
0 - 5	Dämpfungsfaktor
6 - 7	Hochtondämpfung

Das Basisregister 60h

Damit werden Attack und Decay (siehe Anhang) der Hüllkurve eingestellt. Bei großen Werten erfolgt ein schneller Anstieg bzw. Abfall, bei kleinen Werten ein langsamer.

REGISTER 60h

Bit	Bedeutung
0 - 3	Decay der Hüllkurve
4 - 7	Attack der Hüllkurve

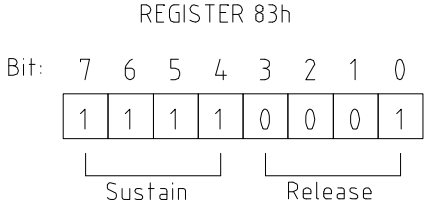
Im Programmbeispiel (Zeilen 22 und 23) wurde ein schneller Anstieg und langsamer Abfall `Sound($63,$F0)` bzw. `Sound($64,$F0)` gewählt.

Das Basisregister 80h

dient zur Einstellung von Sustain und Release. Wird für Release der Wert Null eingetragen, so klingt der Ton nicht aus, bis die Einstellung wieder geändert wird.

REGISTER 80h

Bit	Bedeutung
0 - 3	Release der Hüllkurve
4 - 7	Sustain der Hüllkurve



`Sound($83,$F1)`

Das Basisregister E0h

Von diesem Register werden nur die beiden niedrigstwertigen Bits benutzt. Sie dienen zur Einstellung der Wellenformen.

REGISTER E0h

Bit	Bedeutung
0 - 1	Wellenform
2 - 7	nicht benutzt

Jede Generatorzelle kann nicht nur Sinustöne, sondern noch drei weitere Klangformen erzeugen. Typ1 liefert anstelle der negativen Halbwelle der Sinusschwingung einen Nullpegel, Typ2 gibt den Absolutwert der Sinusschwingung aus, Typ3 setzt das Signal in der fallenden Flanke von Typ2 auf Null. Es gilt folgende Zuordnung: 00 ... Sinus, 01 ... Typ1, 10 ... Typ2, 11 ... Typ3.

Das erzielbare Klangerlebnis ist dabei nicht allzu berauschend; alle drei Klangtypen wirken im Vergleich zum Sinuston eher verzerrt und übersteuert. (Programmzeilen 18 und 19)

Mit der Programmierung der Generatorzellen sind nun die Rahmenbedingungen für die Klangausgabe festgelegt. Zum Erklingen eines Tons fehlt aber noch die

Einstellung der Kanäle (Stimmen)

Pro Kanal müssen in drei Registern fünf Parameter eingestellt werden:

- Blocknummer
- Frequenz
- Verknüpfung
- Rückkopplung
- An/Ausschalten der Stimme

Die Zuordnung der Register zu den einzelnen Kanälen erfolgt ähnlich wie bei den Generatorzellen durch Basisadresse plus Offset, wobei die Nummer des Kanals nun gleichzeitig den Offset darstellt (d.h. Kanal 0: Basisadresse; Kanal 1: Basisadresse + Offset 1, usw.).

Das wichtigste Merkmal eines Tons, seine Tonhöhe, ist auch gleichzeitig der am schwierigsten einzustellende Parameter, da eine direkte Eingabe der Frequenz nicht möglich ist. Ein Register besteht aus 8 Bits, es können daher nur $2^8 = 256$ Werte angenommen werden, was für eine Frequenzangabe nicht ausreichen würde. Vielmehr errechnet sich die Frequenz aus zwei Parametern, der Block- und der Frequenznummer, die auf zwei verschiedene Register aufgeteilt sind. Die Bezeichnung Frequenznummer wurde gewählt, um zu keiner Verwechslung mit der tatsächlichen Frequenz des Tons zu verleiten.

REGISTER A0h

Bit	Bedeutung
0 - 7	LSBs der Frequenznummer

REGISTER B0h

Bit	Bedeutung
0 - 1	MSBs der Frequenznummer
2 - 4	Block
5	Stimme ein/aus
6 - 7	nicht verwendet

Das Basisregister A0h enthält die unteren acht Bits (LSBs ... Least Significant Bits), das Register B0h die oberen zwei Bits (MSBs ... Most Significant Bits) der Frequenznummer. Bei 10 Bits darf die Frequenznummer somit Werte von 0 bis 1023 (000h bis 3FFh) annehmen. Zusätzlich stehen 3 Bits für die Blocknummer zur Verfügung (die daher acht Werte von 0 bis 7 annehmen kann).

Soll nun ein Ton der Frequenz f ausgegeben werden, so errechnet sich die dazugehörige Frequenznummer (*Fn*) nach der Formel

$$Fn = \frac{f \cdot 2^{19}}{49{,}72kHz \cdot 2^{Block-1}}$$

Löst man die Formel nach *f* auf, wird auch die Bedeutung der Blocknummer deutlich:

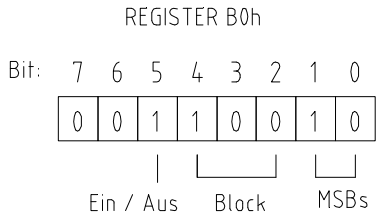
$$f = \frac{49{,}72kHz \cdot Fn \cdot 2^{Block-1}}{2^{19}}$$

Eine Erhöhung der Blocknummer um eins bewirkt jeweils eine Frequenzverdopplung, schaltet daher um eine Oktave hinauf, die Verminderung um eins eine Halbierung der Frequenz, wechselt also um eine Oktave nach unten.

Für die Frequenznummer des Kammertons A mit einer Frequenz von 440 Hz ergibt sich, wenn man für Block den Wert vier einsetzt und rundet:

$$Fn = \frac{440Hz \cdot 2^{19}}{49720Hz \cdot 2^{3}}$$

Wie man sich leicht überlegen kann, erfordert dies einen Eintrag von 44h (= `0100 0100`) in das Register A0h und einen Eintrag von 2h (=10 binär) in die untersten Bits des Registers B0h.



Hätte man für Block den Wert 5 verwendet, so würde man den ebenfalls erlaubten Wert Fn=122h erhalten; dh. zur Codierung einer Tonhöhe existieren i.a. mehrere gültige Wertepaare von Frequenz- und Blocknummer.

Überschreitet ein Ergebnis aus obiger Formel die Höchstgrenze von 3FFh, so ist einfach die Blocknummer zu erhöhen, um wieder in den gültigen Wertebereich der Frequenznummer zu gelangen.

Mit folgenden Registerbelegungen würde der Kammerton A daher eingeschaltet (Programmzeilen 26 und 27) bzw. ausgeschaltet (Zeile 30) werden:

	Register	Inhalt
	A0h	44h = `0100 0100`
EIN	B0h	32h = `0011 0010`
AUS	B0h	12h = `0001 0010`

Im Programmbeispiel wird in den Zeilen 26 und 27 der Ton mit einer Frequenz von 440 Hz, in den Zeilen 28 und 29 der Ton mit der Frequenznummer 23Fh und somit der Frequenz 436 Hz eingeschaltet.

Wird wie in diesem Beispiel nur eine Trägerzelle zur Tonausgabe verwendet, ist es nicht notwendig, die zwei noch verbleibenden Parameter, Verknüpfung und Rückkopplung, einzustellen. Soll ein Kanal jedoch aus Modulator- und Treiberzelle gebildet werden, so hat die Einstellung im Register C0h zu erfolgen.

REGISTER C0h

Bit	Bedeutung
0	Verbindung
1 - 3	Rückkopplung
4 - 7	nicht verwendet

Verbindung = 1 bedeutet additive Verknüpfung der Generatorzellen, Verbindung = 0 bedeutet FM-Synthese, wobei die Einstellung der Rückkopplung nur für die FM-Synthese von Bedeutung ist.

Hüllkurve

Zum Abschluß und als weitere Anregung zum Experimentieren noch eine genauere Beschreibung der Hüllkurve.

Die Hüllkurve bestimmt den zeitlichen Verlauf der Amplitude eines Klanges. In der Sythesizertechnik ist es üblich, den Amplitudenverlauf in vier Phasen zu teilen, die mit Attack, Decay, Sustain, und Release (ADSR) bezeichnet werden.

- Der Attack-Parameter bestimmt die Geschwindigkeit des Ampitudenanstiegs. Bei einem hohen Wert erreicht ein Ton schneller seine volle Lautstärke als bei einem niedrigen.
- Der Decay-Parameter gibt die Dauer bis zum Erreichen des Haltepegels (Sustain-Wert) an. Ein hoher Wert führt zum schnellen Absenken auf den Sustain-Pegel.
- Der Sustain-Parameter legt keine Dauer, sondern jenen Amplitudenwert fest, ab dessen Erreichen der Übergang von der Decay- in die Release-Phase (Ausklingphase) stattfindet. Von Bedeutung ist der Sustainwert vor allem bei kontinuierlichen Hüllkurven.
- Der Release-Parameter bestimmt die Dauer der Ausklingphase.

Zusätzlich kann noch zwischen zwei verschiedenen Hüllkurventypen unterschieden werden:

Bei der kontinuierlichen Hüllkurve verbleibt das Signal bis zum Ausschalten (Key Off) auf dem Sustain-Level. Erst mit dem Ausschaltbefehl wird die Release-Phase eingeleitet. Dieser Hüllkurventyp wäre vor allem zur Simulation von Instrumenten, bei denen Töne gehalten werden können (z.B. Blasinstrumente) zu wählen. Aktiviert wird die kontinuierliche Hüllkurve durch Setzen des EGT-Bits (EGT=1 ... Envelope Generator Type im Basisregister 20h).

Bei der abnehmenden Hüllkurve (EGT=0) wird die Ausklingphase vom FM-Chip selbst eingeleitet. Zu beachten ist, dass die entsprechende Generatorzelle auch dann noch arbeitet, wenn die Amplitude schon auf Null abgefallen ist. Ein Abschalten der Zelle ist also auch in diesem Fall notwendig (Programmzeilen 30 und 31). Gewählt wird diese Hüllkurve zur Simulation von Instrumenten, die nach dem Anschlagen von selbst ausklingen (z.B. Saiteninstrumente)

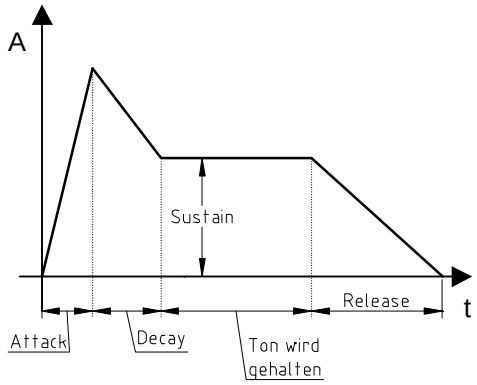


Abb. 4: Kontinuierliche Hüllkurve. Der Ton wird gehalten und klingt erst nach dem Ausschalten ab.

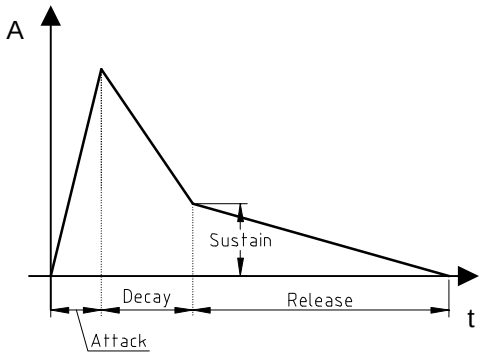


Abb. 5: Abnehmende Hüllkurve. Der Ton klingt nach dem "Anschlagen" automatisch aus.

Da neben der Amplitude und der Frequenz gerade die Hüllkurve einen Klang entscheidend beeinflußt, ist es der Experimentierfreudigkeit des Programmierers überlassen, durch Veränderung der Hüllkurvenparameter den gewünschten "Sound" zu erzielen. Viel Spaß dabei!

Literatur

M.Tischer, *PC intern 4*, Data Becker 1994
K. Dembowski, *Soundkarten*, Hanser 1996
Töne, Klänge, Gefühle, Chip Spezial 2/96