THE SYNTHETIC SOUND EDITOR

This is the editor, which is used to design simple pieces of waveforms and to "program" the volume and pitch altering sequences. Synthetic sounds don't take much memory (that's the main advantage of them). They can use the whole 5-octave range. Although they are quite simple, they usually sound quite "cool" (and remind me of the good old C-64).

Getting started

First load OctaMED, select the Instr-panel, click the Synth-gadget to initialize a new synthsound. Then click the Synth-display selection gadget to display the synthsound editor. Please turn editing off. You can play the current sound with keyboard, like ordinary samples.

Waveform editing

There are two waveform displays, the left one is the "master waveform display" and the right one is for intermediate editing (it also serves as a copy buffer). Between the waveform displays are some gadgets, that are used for transferring waveforms between displays.

First there are two copy gadgets. If you press the left copy gadget, the right waveform will be copied to the left one (and vice versa). Gadget "<- EXCHANGE ->" exchanges the waveforms on the left and right. "<- MIX" mixes the left waveform to the right waveform. It is a powerful tool that can be needed quite often. "<-ADD" is similar to the mix, but it doesn't produce any average between the wforms (= waveforms, from now on). The wforms are simply added together. It's possible, that the waveforms exceed the upper and lower limits (in that case, they are truncated). Two "<- EDIT" "EDIT->" gadgets allow you to select either of the displayed wforms as a current waveform, which is then used in some editing functions. Sometimes they're automatically set (when you draw the wform with mouse, for example). "UNDO" undoes your last editing operation. It can be quite useful. "RANGE ALL" selects the current wform, so that any editing operations affect the whole wform.

Freehand drawing

A very simple way to create wforms is to draw them from scratch. This is easily done. Just press the left mouse button and draw... There are two drawing modes. The default is "pixel". The other is "line". This can be used for drawing straight lines. There's also a mix-mode, that works in both "line" and "pixel" -mode. It mixes the drawed pixels or lines with the already existing data. Mode-selection gadgets are located below "RANGE ALL".

Waveform length

Each waveform in synthsound can be 2 - 128 bytes long (it must be even, though). The shorter the waveform, the higher the pitch (it also depends on the wform itself). Usually, to make the pitches compatible with other instruments, you should use length 2, 4, 8, 16, 32, 64 or 128. The length can be changed with "LEN:" gadgets below the left wform, by typing in a new value or by clicking the single-step gadgets.

Multiple waveforms

Each synthsound can consist of more than one wform (the maximum is actually 64). On the left, below the left wform, there's a set of gadgets:

```
CURR.  WFORM  #
   |
   0   <  >  /  00
       |/      \|
NEXT/PREV    NUMBER  OF  THE  LAST  WAVEFORM
```

The arrow gadgets select the next/previous waveform. As this is a operation that is needed often, there are keyboard shortcuts:

Alt-cursor left = previous
Alt-cursor right = next

There are two Add gadgets that allow you to increase the number of waveforms.
"Last" adds a new waveform after the last wform.
"Here" inserts a new wform here, and shifts the others.

Del gadgets:
"Last" deletes the last wform.
"Here" deletes the current wform.

Preset waveforms

There are some simple, often needed preset waveforms available at a single click. First make sure, that the destination wform is the current wform (selected with the EDIT <-/-> gadgets). Then just click the preset waveform icon. The icons are located at the bottom of the editor.

The available waveforms are: sine, saw up/down, pulse, random and triangle.
The CLR-gadget clears the current wform.
The ALL-gadget clears the whole synthsound (so be careful..)

Range operations

These are some basic operations, that are applied to the selected range (highlighted white). You can select the range by dragging the mouse over the wform with the right mouse button down. You can select the whole wform by clicking "RANGE ALL". The Range start/Range end gadgets at the lower right corner of the synthsound editor can be used to make small corrections to the area.

When there's no range selected, there's a single horizontal white line. This is the cursor. Some editing operations need it. The cursor position can be set by clicking the right mouse button on the wform area. There are also three narrow gadgets (on the left of the preset wform gadgets) for positioning the cursor:

```
CURSOR TO       CURSOR TO END
START    |        |
         <   +    >
             |
     CURSOR TO THE MIDDLE (OF THE WFORM)
```

On the middle of the screen, there's a word "RNG", followed by a strip of gadgets:

RNG Cut Copy Paste Clr Dbl Rev < >

Cut (works only on left wform). It "cuts" the selected area, which is then moved to the right wform display.
Copy copies the selected range to the right wform, but doesn't cut it away.
Paste copies the contents of the right wform to the position pointed by the cursor.
Clr clears the range.
Dbl "doubles" the range, making it play one octave higher.
Rev reverses the range.
< or > shift the ranged data to left or right.

There are two other editing operations: stretch and volume change.

Stretch allows you to stretch a point of the wform towards some other point (understand?? no?? then try this:)
For example, select a sine wave. Then position the cursor on the middle of the waveform (there's a gadget for this purpose, see above). Now activate the "STRETCH:"-gadget (on the left lower corner of the synthsound ed). Type in the amount of moving, e.g. 32. This should demonstrate it... If the number you type in is negative, the point is stretched to left.

Volume change allows you to increase/decrease the volume of selected range: First select a range. Then activate the "VOL.CHG:"-gadget. Now you must enter the amount of volume change (in %'s). For example, to halve the volume, you should type 50. If you want to double the volume, type 200. The volume will be changed when you press Return.

Transformation

This is quite a powerful feature. It allows you to change a waveform to another smoothly (by creating the waveforms between them). Demonstration: Allocate 9 new waveforms (click Add/Last 9 times). Now you have 10 wforms. Move to wform 00, and select a pulse waveform, for example. Then click Transform: Start to mark the beginning of transformation range. Move to wform 09 (the last one). Put here a sine wform. Click Transform: Do, and the magic is done. If you view the wforms 01 - 08, you'll notice that there's now a smooth transition from the pulse wave to the sine wave.

THE PROGRAMMING LANGUAGE

Note: please read this section carefully before experimenting, because you can lock up your machine with the synthsound programming language (as with any other language)

The programming language is used for controlling the volume/pitch/vibrato/ waveforms/arpeggio/etc... of the synthsound. It consist of simple keywords, of which some have an argument.

The programming is done using two lists of commands/numbers. These lists are displayed through a small window below the right waveform display. First there are line numbers in decimal and hex (two leftmost columns). The middle column contains the volume sequence, while the rightmost is the waveform/pitch sequence.

```
       VOLUME CTRL SEQ
          |
LINE      |    WAVEFORM CTRL SEQ
  |       |     |
  00 00   40    00
  01 01  END   END
```

These lists are both max. 127 ($7F) entries long. The list is always terminated with "END" instruction (it's automatically there so you don't have to do that). You can scroll the list with cursor up/down -keys. The cursor can be moved horizontally using the cursor left/right -keys. There are six possible cursor locations (3 for both lists). When entering commands, the cursor should be on the leftmost position of that list. Values are entered by pointing the cursor over the value to be changed and entering a new value. Commands and numbers are entered with keyboard. You can insert entries to the list with the Return-key (or clicking Ins-gadget), and delete using Del (or clicking Del-gadget). JMP commands are renumbered when entries are inserted or deleted.

ALL NUMBERS IN THE SEQUENCE LISTS ARE HEXADECIMAL.
Just remember this, and you have no problems (fingers crossed :^)

First, let's examine some example sequences:

This is a volume sequence:

```
00  40 <= SET INITIAL VOLUME TO 64 (HEX $40, DID YOU ALREADY FORGET?)
01 CHD <= COMMAND, THAT MEANS "SET VOLUME CHANGE DOWN SPEED"
02  03 <= ARGUMENT (SPEED = 3)
03 EN
```

Here's another:

```
00  00 <= INITIAL VOLUME = 0
01 CHU <= COMMAND: SET VOLUME CHANGE UP
02  07 <= SPEED = 7
03 WAI <= COMMAND: WAIT
04  10 <= WAIT 10 PULSES
05 CHU <= SET VOLUME CHANGE UP
```

```
06  00 <= SPEED = 0 -> STOP CHANGING VOLUME
07 WAI <= WAIT
08  70 <= 70 PULSES
09 CHD <= CMD: SET VOLUME CHANGE DOWN
0A  01 <= SPEED = 1 (SLOW)
0B END
```

Third:

```
00  40 <= VOL = $40
01  30 <= VOL = $30 (WITHOUT COMMAND, THE VALUES ARE VOLUME CHANGES)
02  20 <= VOL = $20
03
```

And fourth, finally:

```
00  40 <= VOL = $40
01 CHD <+ <= CHANGE DOWN SPEED..
02  01  | <= ..= 1
03 WAI  | <= WAIT..
04  20  | <= ..20
05 CHU  | <= CHANGE UP
06  01  | <= ..1
07 WAI  | <= WAIT..
08  20  | <= ..20 AGAIN
09 JMP  | <= JUMP (= GOTO)
0A  01 -+ <= TO LINE NUMBER 01
0B END
```

Now some wform/pitch sequences. Wform/pitch sequence is the "master sequence" while the volume sequence is a kind of "slave sequence".

First: (the simplest case)

```
00  00 <= SET WAVEFORM #00
01 END
```

A bit more complex:

```
00 VBS <= SET VIBRATO SPEED
01  40 <= SPEED = $40
02 VBD <= SET VIBRATO DEPTH
03  02 <= DEPTH = 2
04  00 <= WAVEFORM #00
05 END
```

And very complex:

```
00 ARP <= START ARPEGGIO SEQUENCE
01  00 \
02  03  ARPEGGIO VALUES 0, 3, 7 (MINOR CHORD)
03  07 /
04 ARE <= END ARPEGGIO SEQUENCE
05 VBD <= VIBRATO DEPTH
06  06 <= 6
07 VBS <= VIBRATO SPEED
08  40 <= $40
09  00 <= SET WAVEFORM #0
0A  01 <= SET WAVEFORMS 01 - 0A (ONE TIMING PULSE/WAVEFORM)
0B  02    | |
0C  03   \   /
0D  04    \_/
0E  05
0F  06
10  07
```

```
11   08
12   09
13   0A AND BACK TO #01...
14   08
15   07
16   06
17   05
18   04
19   03
1A   02
1B   01
1C JMP <= JUMP
1D   09 <= TO POSITION 09 (RESTART WAVEFORM CHANGING)
1E END
```

You can learn a lot by examining the example synthsounds and by experimenting.

Now, some reference...

About timing

For both sequence lists, most of the commands are fetched, executed, and the next command is immediately fetched. There are also some commands that wait for the next timing pulse, this is important because the computer would otherwise send all its time executing the sequence lists (and would hang up). You should take care that all loops contain a command, that waits for the next timing pulse. These commands are WAI (Wait), vol chg (plain number in the volume list) and set waveform (plain number in pitch/wform list).

For example, the following loops will hang up your computer:

```
00 JMP           00   CHU <-+ COMMAND CHU DOESN'T WAIT
01   00          01    02   |
   ...           02   JMP   |
                 03    00 --+
```

While the following would not:

```
00   20          00   WAI
01 JMP           01    02
02   00          02   JMP
                 03    00
```

Execution speed

The synthsound handling routine is called once every interrupt. This means that the handling is usually done 6 times/note (can be changed with secondary tempo, see main docs). Both lists have an execution speed, that is the speed of the operation. When the speed is 1, the next command is fetched/pitch changed etc. on every interrupt. If the speed was 2, it would happen every second interrupt (runs two times slower). The initial execution speed can be set with the two speed gadgets. During executing, the speed can be changed with command SPD.

VOLUME SEQUENCE LIST COMMANDS

1. Set volume

Command: ---
Keyboard: --- (key needed to enter the command)

This is the default command (no command identifier). It sets the absolute volume of the synthsound. It should be 00 - 40. Note that the relative track volumes are not used in synthsounds (mostly because of performance reasons).

Example:

```
00  30 ;VOLUME = $30
01  10 ;VOLUME = $10
...
```

2. End sequence

Command: END
Keyboard: ---

This command terminates the volume sequence list. It's always there and automatically inserted. You can't insert commands past this one.

3. Set volume change down speed

Command: CHD
Keyboard: D

This command sets the speed, in which volume is decreased each timing pulse. The volume starts changing automatically after this command. To stop automatic volume sliding, issue this command with speed 00.

Example:

```
00  CHD
01   05 ;SPEED = 5
...
10  CHD
11   00 ;SPEED = 0 -> STOP SLIDING
```

4. Set volume change up speed

Command: CHU
Keyboard: U

This command is like CHD, except it sets the volume change up.

5. Wait

Command: WAI
Keyboard: W

This command waits for specified amount of timing pulses (pause).

Example:

```
03   WAI
04    10  ;WAIT FOR 16 ($10) PULSES T
```

---

6. Jump

Command: JMP
Keyboard: J

Causes an immediate jump to another position of the volume list.

Example:

```
05   JMP
06    0A  ;JUMP FORWARD TO LINE 0A
```

---

7. Jump waveform sequence

Command: JWS
Keyboard: Shift-J

This command causes a jump in the waveform sequence. This can be used for example, to trigger a pitch change at the end of the vol seq list. Note that this DOESN'T cause a jump TO waveform sequence.

Example:

```
04   JWS
05    0F     ;CAUSE JUMP IN
```

---

8. Halt

Command: HLT
Keyboard: H

This has the same effect with command END (halt execution), but it can be inserted in the middle of the sequence list.

Example:

```
03   HLT
04    04  ;SOME OTHER CODE (CAN BE ACCESSED WITH JMP INSTRUCTION, FOR
...     EXAMPLE)
```

---

9. Set speed

Command: SPD
Keyboard: S

Sets the execution speed.

Example:

```
0A   SPD
0B    01  ;SPEED = 1 (FASTEST)
...
```

(The following commands require OctaMED V2.00 (MED V3.20) or later.)

10. One-shot envelope

Command: EN1
Keyboard: E

This command allows you to draw the shape of the envelope with the mouse. When the end of the envelope is reached, nothing occurs.

```
02    40
03   EN1
04    05
```

Waveform 05 is used as an envelope. Note that the envelope execution starts on next interrupt, so the volume is initialized to $40.
The envelope waveform must always be 128 bytes long!!

---

11. Looping envelope

Command: EN2
Keyboard: Shift-E

This works like command EN1, except that when the end is reached, execution will start again from the beginning.

WAVEFORM/PITCH SEQUENCE LIST COMMANDS

1. Set waveform

Command: ---
Keyboard: ---

This command is used to indicate the waveform number (starting from 00). After this instruction, the execution stops until next timing pulse occurs. Don't use waveform numbers that are higher than the actual number of the last wform.

Example:

```
00   00  ;WFORM 00
01   01  ;WFORM 01
...
```

2. End sequence

Command: END
Keyboard: ---

This command terminates the wform/pitch sequence list. It's always there and automatically inserted. You can't insert commands past this one.

3. Set pitch change down speed

Command: CHD
Keyboard: D

This command sets the sliding speed for sliding pitch down. The sliding automatically starts after this command and stops, when speed is set to zero.

Example:

```
00 CHD
01   03  ;SET  SPEED  TO  3
...
```

4. Set pitch change up speed

Command: CHU
Keyboard: U

Just like previous, but slides pitch up.

5. Wait

Command: WAI
Keyboard: W

This command waits for specified amount of timing pulses (pause).

Example:

```
03   WAI
04    02  ;WA
```

6. Jump

Command: JMP
Keyboard: J

Causes an immediate jump to another position of the wform/pitch list.

Example:

```
05   JMP
06    0A  ;JUMP FORWARD T
```

7. Jump volume sequence

Command: JVS
Keyboard: Shift-J

This command causes a jump to happen in the volume sequence. Can be used e.g. for triggering volume changes after some wform event.

Example:

```
09   JVS
0A    00  ;START
```

8. Halt

Command: HLT
Keyboard: H

This has the same effect with command END (halt execution), but it can be inserted in the middle of the sequence list.

Example:

```
03   HLT
04    04  ;SOME OTHER CODE (CAN BE ACCESSED WITH JMP INSTRUCTION, FOR
...     EXAMPLE)
```

9. Set speed

Command: SPD
Keyboard: S

Sets the execution speed.

Example:

```
0A   SPD
0B    01  ;SPEED = 1 (FASTEST)
...
```

10. Begin arpeggio definition

Command: ARP
Keyboard: A

This command starts the arpeggio sequence. The following values are the arpeggio offsets from the base note. The arpeggio sequence is terminated with ARE-command. The arpeggio starts automatically after the sequence is defined.

Example:

```
03   ARP  ;START ARPEGGIO
04    00  ;OFFSET VALUES
05    04
06    07
07    0A
08   ARE  ;END ARPEGGIO DEFINITION
```

11. End arpeggio definition

Command: ARE
Keyboard: E

Ends an arpeggio definition. See above.

12. Set vibrato depth

Command: VBD
Keyboard: V

This command is used to set the vibrato depth (00 - 7F).

Example:

```
02   VBD
03    04  ;SET DEPTH TO 4
```

13. Set vibrato speed

Command: VBS
Keyboard: Shift-V

This command sets the vibrato speed (00 - 7F). Both speed and depth must be nonzero for vibrato to occur.

Example:

```
02   VBD
03    04  ;DEPTH = 4
04   VBS
05    30  ;SPEED = 30
```

14. Reset pitch

Command: RES
Keyboard: R

This command resets the pitch of the note to its initial pitch.

(The following command requires OctaMED V2.00 (MED V3.20) or later.)

15. Set vibrato waveform

```
Command: VWF
Keyboard: Shift-W
```

Sets the vibrato waveform. The argument is the number of wform.
The waveform should always be 32 bytes long!! Note that it's
actually played reversed (use the Rev gadget to reverse it).
By default, sine wave is used.

Example:

```
00   VBD
01    06
02   VBS
03    40
04   VWF
05    04  ;USE WFORM NUMBER 04 AS VIBRATO WAVEFORM
```

Synthsound handling commands in the songs:

Command E in the songs controls the MIDI pan, if used with MIDI, but with synthsounds, it is used to trigger a jump in the wform/pitch sequence list.

For example, if you wanted to decrease the pitch of the sound after a certain point, your wform/pitch sequence could look like this:

```
00   VBS
01    40
02   VBD
03    06
04    00     ;PLAY WFORM 00
05   HLT
06   CHD     ;PITCH CHANGING ENTRY POINT
07    02
08   END
```

Now you could compose a track like this:

```
C-2 3000   ;THIS IS THE PREVIOUS SYNTHSOUND
--- 0000
--- 0000
--- 0E06   ;CAUSE A JUMP TO POSITION 06 (PITCH STARTS TO SLIDE DOWN)
--- 0000
...
```

With command JVS, you could make the command affect volume sequence too.

Hold/Decay:

Hold/Decay works well with synthsounds too. The decay value in synthsounds is, however, a jump address in the volume sequence list. When the decay should start, the execution will jump to this entry in the vol seq list. This means that you can handle the decay in any way you want. You can also make it affect the pitch/wforms using the command JWS.

Example volume sequence list:

```
00  40     ;VOLUME
01 HLT     ;END
02 CHD     ;DECAY HANDLING (ENTRY POINT) -> CAUSE DECAY
03  03
04 END
```

The decay value for this synthsound should be 2. When you save/load synthsounds, the decay values are saved/loaded with the instrument.

Saving/loading synthsounds

Synthsounds are saved using the file requester panel, "save instr". They usually take very little disk space. Synthsounds can be loaded just like normal samples (filerequester, sample list, typing sample name).

HYBRID SOUNDS

Hybrid sounds are much like synthsounds, except that there are no waveform pieces. Instead, there's a normal sample. All commands of the synthsound handling programming language can be used with hybrid sounds. There's only one thing you should know: Don't use "Set waveform". Because there's only a single waveform, there's no need for that (and it wouldn't work anyway).

Final note:

Don't be afraid if you didn't understand everything described in this file (I hardly understand it myself ;-) All this information is not necessary for using the synthsounds (only for designing them).

References:
Original document
Text refactored by KONEY 2022