Instructions for *Ere's Fokker organ console*

by Ere Lievonen

Introduction

Ere's Fokker organ console software is designed to provide assistance when performing on the 31-tone Fokker organ, or any virtual version of the Fokker organ. The most important functions of the software are *registration changes* and *keymapping*. It can be used by anyone who plays the Fokker organ (or a virtual Fokker organ), and is especially useful for anyone who doesn't play on a microtonal keyboard, as with this software you can program any keys on a regular 12-tone MIDI keyboard to play any 31-tone pitches.

The features of *Ere's Fokker organ console* include the following functions:
– Assign the traditional meantone tuning scheme to the keys of a regular 12-tone MIDI keyboard, and modulate that scheme to any tonality (see "Meantone modulator")
– Assign preset Euler-Fokker genera to 12-tone MIDI keyboards (see "Preset keymaps")
– Assign any 31-tone pitches to any keys of 12-tone MIDI keyboards (see "Custom keymaps")
– Transpose up or down by 1/5 tones (dieses) (see "Transposer")
– Store and recall register combinations using a registration list (see "Registration lists")
– Include commands for keymap changes and other functions in a registration list (see "Commands in registration lists")
– Trigger events stored in a registration list using MIDI controllers (see "Control changes")
– Automatically trigger events stored in a registration list by synchronizing them with notes played by the performer (see "Synchronizing registration events with MIDI input")
– Route MIDI data from any MIDI keyboard(s) or an external MIDI source to the two manuals and pedal of the (virtual) Fokker organ (see "Settings")
– Add new (artificial) couplers to the Fokker organ's disposition (see "Virtual couplers")
– Split one keyboard to play both manual I and manual II (see "Split manual I")
– Use a MIDI controller as a general crescendo pedal (see "Crescendo pedal")
– Program the buttons, knobs and sliders of a VMK-88 MIDI keyboard to control almost any of the available functions (see "VMK-88 controls editor")
– Record Fokker organ performances into MIDI files (see "MIDI recorder")
– Control a virtual organ with a different (user-specified) disposition (see "Custom console")

This guide is still a work in progress, and does not yet cover all the topics nor describe all the available functions.

Table of contents

Installation and setting up

System requirements for *Ere's Fokker organ console* are the same as for the *Max* software. See https://cycling74.com/downloads/sys-reqs for more details.

1. Download and install the *Max* software from https://cycling74.com/downloads.
Max is available for both Windows and Mac. It is not mandatory to buy Max; you may continue running *Ere's Fokker organ console* on an unpaid installation of Max also after the trial period has ended.
Ere's is now compatible with Max versions 7 and 8, but on older computers, you could (also) use older versions of *Max Runtime*, version 5 or 6. For Windows users, Max Runtime version 6.1.9 has been proven to work best; for Mac users, Max Runtime version 5.1.9 used to be the best one. The older versions are also available for download on the above-mentioned webpage.

2. Download *Ere's Fokker organ console* from www.erelievonen.eu/documents/.

3. Connect your MIDI keyboard(s) to your computer (or start your MIDI source software).

4. If you are at the Fokker organ in the Muziekgebouw, connect your computer to the organ via a MIDI interface. Connect both the input and the output plugs!
If you are not at the Fokker organ, start the virtual Fokker organ software you want to use (e.g. Hauptwerk and Fokker Emulator).

5. Open file "Ere's 3.x.x.mxf" (whichever is the latest version), and wait for it to start.
A separate version of this file is provided for users of Max 5 or 6.
(If Max doesn't start automatically, right-click on *Ere's*, and select "Open With Max" from the context menu.)

6. The window that first opens is the grey Setup window.
In the appropriate pop-up menus, select the names of your MIDI input device(s), and your MIDI output device (normally, use the one labeled "output to Fokker organ").
Note that when using *Ere's*, all the MIDI data from your MIDI keyboards or other devices should be routed via *Ere's*. Do not send any MIDI data directly to the (virtual) Fokker organ.

7. To modify the input channel numbers – in case your keyboards (or MIDI software) transmit on other channels than the default ones – click on Advanced settings, and make the necessary changes to the MIDI channel numbers.

8. To avoid having to redo these settings every time you start *Ere's*, you can save your settings (click on "Save settings"). Next time, instead of steps 6 and 7, you can just load your settings file (click on "Load settings").

9. After the set-up is done, click "Go to console window". On the Console window you will see a representation of the Fokker organ's register knobs, as well as many other controls for using *Ere's Fokker organ console*.

Key commands on the computer keyboard

The following commands for *Ere's Fokker organ console* are available on the computer keyboard by default:

Space bar : All notes off
shift-Space bar : Force all notes off (On Mac, also **alt-Space bar**)

Registers / virtual couplers:
– Keys **1** – **9** correspond to registers 1–9.
– To control registers (virtual couplers) 10–99, hold down **shift** (or use **caps lock**) and type register number.
– To control registers (virtual couplers) 100–119, hold down **shift** (or use **caps lock**) and type register number, but with **A** instead of the initial 1 (i.e., for register 100, type **shift-A00**, and so on). On the numeric keypad, you can also use the decimal separator key (comma or dot) instead of the initial 1.

The Shift key may not work for the above-mentioned commands on some operating systems. On Mac, you can also use the ctrl key instead of Shift.
For registers (virtual couplers) 10–119, the number keys with shift / caps lock / ctrl need to be pressed within 1.5 seconds of each other in order to be recognized.

0 (zero) : All registers off
shift-0 (shift-zero) or **ctrl-0** (Mac) : All registers off, reset registration event display to 0
T : Tutti (registers 1–9 on)

V : Show all virtual couplers (open "Virtual couplers" window)
C : All virtual couplers off
shift-V : Hide virtual couplers window

Registration controls:
→ or **+** : Next registration
← or **–** : Previous registration
shift- → : Go directly to next registration, ignoring sync hold and delay (On Mac, also **alt-** →)
' (apostrophe) or **Home** : Go to first registration
Enter : Restore current registration

shift-S : Store current registration in registration list
shift-D : Store current registration at previous registration change

Backspace : Cancel sync hold / delay

Euler-Fokker genera:
shift-Q : Euler-Fokker genus 1 [33355]
shift-W : Euler-Fokker genus 2 [33555]
shift-E : Euler-Fokker genus 3 [55577]
shift-R : Euler-Fokker genus 4 [55777]
shift-T : Euler-Fokker genus 5 [33377]
shift-Y : Euler-Fokker genus 6 [3357]
shift-U : Euler-Fokker genus 7 [3557]
shift-I : Euler-Fokker genus 8 [3577]

Meantone modulator:
, (comma) : Modulate sharpwards
M : Meantone Eb–G# (default)
N : Modulate flatwards
shift-M : Reactivate last used meantone

You might want to change the key assignment of some of the command keys, in order to better suit your computer keyboard layout. This might even be imperative in some cases, when some of the commands would otherwise fall on the same keys.
By using the *Command keys editor* utility, you can reassign the keys for Next registration, Previous registration, Go to first registration, Modulate sharpwards and Modulate flatwards commands. When doing this, make sure that no two commands get assigned to the same key!
The command key assignments will be saved with the rest of the settings when you save your settings. To do this, choose "Save settings" from the File menu. To recall saved settings, choose "Load settings" from the File menu.

Menus

These drop-down menus are located on the top bar of the screen (on Mac), or on the top bar of each window (on Windows).

File menu

New: does nothing.
Open: opens any text file for editing; or opens another Max patch.
Close: closes the active window. Closing the main Setup window closes *Ere's Fokker organ console*, except for the utilities, which need to be closed separately. (Selecting Quit from the Max menu closes all the windows, including the utilities.)
Save and **Save As...:** these are operational only when a text editor window (for example, the registration list editing window) is active, and save only the text file.

Load settings..., Save settings..., Default settings: these are equivalent to the corresponding buttons on the Setup window and "Advanced settings" window.

The following three items give access to functions found on the "Registration list editor" window.
Load registration list...: is equivalent to the button "Load a registration list" (also found on the Setup and Console windows).
Save registration list: is equivalent to the button "Save current registration list, replacing the older version". You will not be asked for a filename, except if the registration list has not yet been saved at all.
Save registration list as...: is equivalent to the button "Save current registration list as...". You will be asked for a filename before saving.

The following three items give access to functions found on the "Crescendo pedal" window.
Load crescendo list...: is equivalent to the button "Load a crescendo list...".
Preset cresc list (man+ped): is equivalent to the button "Preset crescendo list for manuals and pedal".
Preset cresc list (man only): is equivalent to the button "Preset crescendo list for manuals only".

Load custom keymap...: is equivalent to the button "Load a custom keymap" on the Console window.
Preset keymaps...: opens the "Preset keymaps" window.

The following three items give access to functions on the *VMK-88 controls editor* utility, without the need to open the *VMK-88 controls editor* utility itself.
Load VMK-88 settings...: is equivalent to the button "Load VKM-88 settings from a file...".
Default VMK-88 settings: is equivalent to the button "Default settings".
Clear VMK-88 settings: is equivalent to the button "Clear settings".

The following three items give access to functions on the *Fokkerboard controls editor* utility, without the need to open the *Fokkerboard controls editor* utility itself.
Load Fokkerboard settings...: is equivalent to the button "Load FB settings from a file...".
Default Fokkerboard settings: is equivalent to the button "Default settings".
Clear Fokkerboard settings: is equivalent to the button "Clear settings".

Load custom console...: loads and opens a custom (user-defined) console window, for controlling a virtual organ with a different register disposition than the standard Fokker organ one.

Edit menu

Undo, Redo, Cut, Copy, Paste, Delete: these are operational when editing a text file in a text editor window (for example, when editing a registration list in the registration list editing window).

Overdrive: toggles Max's Overdrive function. You may leave it on, but it probably makes no difference when using *Ere's Fokker organ console*.
Note: If you experience slow response to key presses on the computer keyboard when using *Ere's*, toggling the Overdrive function once may help. (It seems to make no difference whether this setting is left on or off afterwards.)

Store current registration in registration list: stores the current registration in the registration list (at the current event #, or any other registration event # you enter). Is equivalent to pressing **shift-S**.
Store current registration at previous registration change: stores the current registration in the registration list, at the latest event number with a registration change (or any other registration event # you enter). Is equivalent to pressing **shift-D**.

"Ere's Fokker organ console" menu

This "central" menu gives direct access to the various (sub-)windows of *Ere's Fokker organ console* (excepting the Utilities), as well as to these Instructions. The same windows are also accessible via the corresponding buttons on the Setup window (for Instructions, Console window, Advanced settings) and on the Console window (for Setup window, Registration list editor, Virtual couplers, Crescendo pedal, Transposer, Preset keymaps, Meantone modulator, Note length control).

This menu also has on/off switches for two of the functions in *Ere's Fokker organ console*. A check mark in front of the menu item shows when the function is on.
Crescendo pedal on/off: is equivalent to the on/off switch on the Crescendo pedal window. Default is **off**.
Meantone modulator on/off: is equivalent to the on/off switch on the Meantone modulator window. Default is **on**.

Commands menu

All the commands in this menu are also accessible via the corresponding buttons on various windows and/or the corresponding keyboard shortcuts.

All registers off, Tutti (registers 1–9), All virtual couplers off, **Next registration, Previous registration, Go to first registration, Restore current registration:** are equivalent to the corresponding buttons on the Console window and the corresponding keyboard shortcuts.
Cancel sync hold / delay: cancels registration hold (without releasing the registration). Is equivalent to clicking the "next on hold" button on the Console window, or pressing **Backspace**.
Meantone Eb–G#, Modulate sharpwards, Modulate flatwards, Reactivate last used meantone: are equivalent to the corresponding keyboard commands of the Meantone modulator.
Transposer off: switches off the Transposer function (sets the transposition to 0).

All notes off: turns off all sounding MIDI notes by sending noteoff messages to all the notes that have been sounded through *Ere's Fokker organ console* and are still sounding. If you have hanging notes, and all MIDI traffic to the Fokker organ is being routed through *Ere's Fokker organ console* software, this should be sufficient to silence the organ. Is equivalent to pressing **Spacebar**.
Force all notes off: turns off all MIDI notes by actually sending noteoff messages to every note existing in the Fokker organ's MIDI specification. If you have hanging notes, and there are several independent devices or applications controlling the Fokker organ at the same time, this command may become necessary. Is equivalent to pressing **shift-Space bar** (on a Mac, also **alt-Space bar**).
Note: If the above does not silence the hanging notes, there is probably a MIDI connection failure between your computer and the Fokker organ. Check all MIDI plugs, then try **Force all notes off** again. If this still does not help, silence the organ by manually pushing all the register knobs in, and quit Max. Then restart *Ere's Fokker organ console* software, re-establish a MIDI connection, and give one more **Force all notes off** command. (Switching off the Fokker organ itself should not be necessary.)

Options menu

This menu is used to switch the various options on and off. A check mark in front of the menu item shows when the option is active. See chapter "Options" below for detailed information on each option.

Utilities menu

From this menu you can launch all of the Utilities. Utilities currently available are *MIDI recorder*, *Method 1 translator*, *Keymap editor*, *Filename utility*, *VMK-88 controls editor*, *Fokkerboard controls editor* and *Command keys editor*. The utilities can also be opened by clicking on the corresponding buttons on the Setup window.

Options

There are several *options* available which can be helpful in specific situations. The options can be switched on and off using the Options menu (at the top bar of the screen/window). A check mark in front of the menu item shows when the option is active.
When you save your settings, the current states of all the options are also saved in the settings file, and will be recalled when you load the settings file.
It is also possible to specify any of these options in a registration list. For more information, see the chapter "Registration list initial commands".

Transfer notes below normal manual range automatically to pedal
When this option is on, notes played on any manual that would sound lower than the low C (i.e. lower than the lowest pipe on the manual registers) will be transferred to the lowest octave (C to B) of the pedal. In order for this to work, the pedal needs to have a suitable registration, including the Subbas 16'. This can be useful in at least these situations:
– when transposing downwards, but needing to have the lowest notes (of the octave C–B) still playable, or
– as a means to "extend" the manual range downwards by one octave.
But keep in mind that on the Fokker organ, it is not always possible to find a pedal registration that matches the timbre and dynamics of the manual(s).

Use highest keys on 12-tone keyboards as meantone modulator
You can use the 15 highest keys on 12-tone MIDI keyboards as meantone modulator:
bb3 = modulate flatwards
cb4 – b4 = Meantone -7...5
c5 = modulate sharpwards
For this to function, the meantone modulator has also to be on.

Use 2 lowest keys on 12-tone manual I for registration changes
With this option enabled, registration changes can be done with the 2 lowest keys of 12-tone manual I:
– AAA = next registration
– BBBb = previous registration
This can be especially useful if an assistant is doing the registration changes.
Note: If you are using keymaps that employ the 2 lowest keys, make sure to disable this option!

Use top row of keys on 31-tone pedalboard as volume pedal
This option may be useful when you do not have a MIDI volume pedal, but would still like to employ the crescendo pedal function. With this option active, the 9 keys on the top row of the Fokker organ's 31-tone pedalboard will correspond to positions of the crescendo pedal. The crescendo position given by each key depends on the size of the current crescendo list, as follows:

# of steps in	Crescendo positions given by top row of pedal keys								
crescendo list	D-	E-	F+	G+	A+	B+	db	eb	f
1	1	1	1	1	1	1	1	1	-
2	1	1	1	1	2	2	2	2	-
3	1	1	1	2	3	3	3	3	-
4	1	1	1	2	3	4	4	4	-
5	1	1	2	3	4	5	5	5	-
6	1	1	2	3	4	5	6	6	-
7	1	2	3	4	5	6	7	7	-
8	1	2	3	4	5	6	7	8	-
9 or more	1	2	3	4	5	6	7	8	9

The highest pedal key (f) will be part of the volume pedal function only when the crescendo list has 9 or more steps.
Note that activating this option does not silence any of the pedal keys for normal playing. Because of this,

this option is most useful when playing on manuals only, and there are no pedal registers used in the crescendo list.
For this option to work, the crescendo pedal function has to be on as well.

Synchronize note and registration messages from external MIDI source
This option is designed to help with the following problem: When a register message is sent to the real Fokker organ via MIDI, the register change does not take place immediately, but only after the register knob has actually moved. This causes a latency of about 157 ms for the operation of any of the 9 registers. If you have a MIDI file (or other MIDI input) where registration messages are placed exactly where they should happen (i.e. not taking the latency of the registers into account), the register changes will inevitably sound too late. This can be remedied by playing the MIDI file through *Ere's Fokker organ console* while activating this option. This option works by relaying all registration messages (program change messages / registration shortcuts) immediately, but delaying note messages by 157 ms. This way, the registration changes will sound properly synchronized with the notes.
This option affects only data received through "input from external MIDI source", and is not needed when playing through a virtual Fokker organ.
If necessary, the delay time can be adjusted in the "Advanced settings" window.

Block program change (register) messages from external MIDI source
When *Ere's Fokker organ console* receives MIDI input through "input from external MIDI source", program change messages that serve as register messages are normally relayed to the MIDI output. If you would like to suppress these program change messages (being register messages or otherwise), use this option.
(However, other program change messages from external MIDI source than 0–17 on channel 1 are always ignored and never forwarded by *Ere's Fokker organ console* anyway.)

Ignore duplicated note messages from external MIDI source
In its normal operating mode, *Ere's Fokker organ console* counts all incoming noteon messages, and only sends out a noteoff message when an equal amount of noteon and noteoff messages for a given pitch have been received. The purpose of this behavior is to prevent a noteoff message arriving from one MIDI keyboard/device from stopping the note in case the same note is simultaneously being played by another MIDI keyboard/device.
However, some MIDI software produce extraneous or duplicated noteon/noteoff messages by themselves (for example, multiple successive noteon messages for one and the same note in a MIDI file, without any intervening noteoff message for that note). Such duplicate messages are normally ignored by MIDI applications, and the first noteoff message received will stop that note regardless of the number of noteon messages received previously.
The option "Ignore duplicated note messages from external MIDI source" exists in order to imitate this commonly expected behavior, and it is on by default. It filters out all extraneous noteon/noteoff messages received through "Input from external MIDI source". Doing this also prevents the input from external MIDI source from interfering with notes played simultaneously by other MIDI keyboards/devices (connected to *Ere's* via other inputs).
If this option is off, and the amounts of noteon and noteoff messages received from the external MIDI device do not match exactly, either there will be hanging notes (if there were too many noteon messages), or same notes played simultaneously via other MIDI inputs might get cut off prematurely (if there are too many noteoff messages).
(This option only affects MIDI input received through "Input from external MIDI source". This is because the other MIDI inputs in *Ere's* are meant primarily for connecting MIDI keyboards, and MIDI keyboards normally don't create extraneous noteon/noteoff messages.)
This option should be switched on, for example, in the following situations:
– If a MIDI file plays correctly without *Ere's Fokker organ console*, but causes hanging notes when played through *Ere's Fokker organ console*. There may be extra noteon messages in the file without a corresponding amount of noteoff messages. (Of course, the MIDI data should be sent through "Input from external MIDI source".)
– When sending MIDI data over an unreliable network connection, any lost noteoff messages will cause hanging notes. You might try to counter the effect of messages lost over the network by sending many

redundant MIDI messages; in that case you'd need to enable this option at the receiving end.
This option should be switched off if you want to connect several MIDI keyboards/devices to "Input from external MIDI source", in order to prevent the different MIDI devices from interfering with each other's held notes. (However, this may only work properly if all the MIDI devices in question send balanced amounts of noteon and noteoff messages.)

Use 31-tone practice keyboard instead of Fokker organ console
The 31-tone practice Fokker keyboard ("Fokkerboard") may be used as an alternative to the full Fokker organ console, and its output is primarily intended to be received via "input from Fokker organ". In this case, it is beneficial to enable this option, which changes the behavior of *Ere's Fokker organ console* in the following ways:
– 8' off couplers 40 and 55 become functional (as well as couplers 14 and 24)
– Split manual function for 31-tone manual I is enabled
– The 5 unused keys and the 5 buttons on the back of the 31-tone practice keyboard can be used as freely programmable controls for *Ere's Fokker organ console* (to edit their settings, use the *Fokkerboard controls editor* utility)
– The functions "Flash buttons on Fokker organ console at registration changes" and "Fokker organ button blink" currently require different implementations for the real Fokker organ and for the Fokker Emulator; when this option is selected, these features work correctly with the Fokker Emulator. (In fact, it is the real Fokker organ where the lights on the buttons do not currently behave according to their MIDI specifications.)

Use 12-tone pedal input for 31-tone practice pedal
When this option is selected, input from 12-tone pedal is treated in a special way, acting as a replacement for a true 31-tone pedalboard for practice purposes.

Flash buttons on Fokker organ console at registration changes
When this option is on, the ten small buttons on the left-hand side of Fokker organ console will flash once at every "Next registration" command and registration shortcut. This can be useful if you wish to have additional visual feedback on the functioning of (automatized) registration changes.

Beep at registration changes
When this option is on, a short beep sound will be played at every "Next registration" command and registration shortcut. The beep is also sounded when a registration "on hold" is released, and at the end of a registration delay. Obviously intended for practice purposes only, this can be useful in providing additional auditory feedback on the functioning of (automatized) registration changes.
The beep is produced by playing the two highest notes of manual I via the regular MIDI output, using the registers that are on at the moment. Because of this, note the following:
– The beep will not sound if manual I has no registers on.
– If the two highest notes of manual I are being played simultaneously with the beep, the beep may interfere with them, or remain unnoticeable.

Ignore sync hold and delay commands in registration list
With this option, you can temporarily disable all registration automation (i.e. sync hold and delay) commands in a registration list. This can be useful when you're practicing sections of a piece and don't want any automatized registration changes to take place. You can still use the regular "Next registration" buttons/controls to step through the registration list.
Note: If you just want to temporarily suspend an automatized registration change, you can more easily press **Backspace** (or click on the "next on hold" button on the Console window while it is lit): this momentarily cancels all active sync hold and delay functions. Normal operation is resumed as soon as you give a "Restore current registration" or a "Next registration" command.

Meantone modulator

When using regular 12-tone MIDI keyboard(s) with the Fokker organ, the keys of a 12-tone keyboard are, by default, assigned to the traditional meantone tuning scheme. This tuning scheme includes all the notes between Eb and G# on the circle of fifths (i.e. all the naturals, plus C#, Eb, F#, G#, Bb). The *meantone modulator* function makes it very easy to transpose (modulate) this scheme, in order to be able to play in farther away tonalities.

Click on the button "Meantone modulator" on the Console window to open the Meantone modulator window. (The meantone modulator actually works equally well regardless of whether the Meantone modulator window is open or not, unless it has been disabled by the on/off switch).

To modulate, you can use the computer keyboard:
comma = modulate sharpwards
M = Meantone Eb–G# (Meantone # 0, the default)
N = modulate flatwards
You can also type a number directly in the "Meantone #" box. (Note that you must click again outside the number box in order to use the computer keyboard for other commands.)

The "Usable notes" and "Usable tonalities" displays remind you of which notes are usable and in which tonalities you can play in with the current tuning (meantone transposition). These displays can be particularly handy when improvising on the Fokker organ.

You can also use the 15 highest keys on (88-key) 12-tone MIDI keyboards as meantone modulator:
bb3 = modulate flatwards
cb4 – b4 = Meantone -7...5
c5 = modulate sharpwards
Activate this function by marking the checkbox at the bottom of the Meantone modulator window, or by selecting it from the Options menu. This setting will be saved when you save all your settings.

You can equally well program MIDI controllers or pedals and/or the buttons on the VMK-88 MIDI keyboard to send meantone modulator commands. For more details, see the chapter "Control changes" and/or use the *VMK-88 controls editor* utility.

Whenever a non-meantone keymap is in use, the "Meantone #" box is colored gray. To reactivate the last used meantone tuning without changing it, press **shift-M**.

If you would prefer to use other keys (than comma and N) for the "modulate sharpwards" and "modulate flatwards" commands – for example, if the comma key on your computer keyboard is not located next to the N and M keys – you can change these command keys. To do this, use the *Command keys editor* utility.

The meantone modulator in fact allows you to modulate far beyond the first 31 transpositions – all the way to ±372. Transpositions beyond ±15 contain, in a cyclical fashion, the same pitches as the first 31 transpositions, but the pitches will fall on different keys of a 12-tone keyboard. This "drift" (amounting to a half step, or one 12-tone key, per 31 transpositions) is caused by the inherent incompatibility of the 31-tone and 12-tone systems.

Keymaps

With the help of *Ere's Fokker organ console*, you can play the 31-tone Fokker organ using regular 12-tone MIDI keyboard(s). A *keymap* tells which 31-tone pitch is assigned to each key of a 12-tone keyboard. There are two kinds of keymaps: *preset keymaps* and *custom keymaps*.

Preset keymaps

The following preset keymaps are available:
– The eight so-called *Euler-Fokker genera*, that were chosen by Adriaan Fokker as preset genera on the Fokker organ's original 12-tone console.
– The traditional *meantone tuning* scheme, in all the 31 possible transpositions.

All the preset keymaps are 12-tone keymaps, i.e. every octave contains the same 12 note names.
To see all the preset keymaps, click "Preset keymaps" on the Console window.

The eight Euler-Fokker genera that are available as preset keymaps contain the following pitches:
– Euler-Fokker genus 1 [33355] {Eb, G#}: C, C#, D, Eb, E, F, F#, G, G#, A, Bb, B
– Euler-Fokker genus 2 [33555] {Ab, A#}: C, D, D#, Eb, E, F#, G, G#, Ab, A#, Bb, B
– Euler-Fokker genus 3 [55577] {D+, A+}: C, D-, D, D+, E, F+, Gb, G#, A+, Bb- (A#), Bb, B# (C-)
– Euler-Fokker genus 4 [55777] {B-, B+}: C, D-, D+, E-, E, F#, Gb, G+, Ab, A#, B-, B+
– Euler-Fokker genus 5 [33377] {G+, C+}: C, D, D+, Eb- (D#), E+, F-, F, G, G+, A+, Bb- (A#), C-
– Euler-Fokker genus 6 [3357] {F, A-}: C, D-, Eb- (D#), E, F-, F, G-, G, A-, A, Bb- (A#), B
– Euler-Fokker genus 7 [3557] {Ab, A-}: C, C#, D-, Eb, E, F-, F#, G, Ab, A-, Bb- (A#), B
– Euler-Fokker genus 8 [3577] {D+, A-}: C, Db, D-, D+, E, F-, Gb, G, A-, A+, Bb- (A#), B

Euler-Fokker genus 1 is identical to the traditional meantone tuning scheme ("Meantone 0").
For a better representation of the pitch contents of the Euler-Fokker genera, see the file *EF pitches* (in folder "Documents").

The Euler-Fokker genera can be activated by clicking on the buttons in the "Preset keymaps" window, or with the following keyboard shortcuts:
– Euler-Fokker genus 1: **shift-Q**
– Euler-Fokker genus 2: **shift-W**
– Euler-Fokker genus 3: **shift-E**
– Euler-Fokker genus 4: **shift-R**
– Euler-Fokker genus 5: **shift-T**
– Euler-Fokker genus 6: **shift-Y**
– Euler-Fokker genus 7: **shift-U**
– Euler-Fokker genus 8: **shift-I**

The regular meantone tuning scheme contains 12 consecutive notes on the circle of fifths. The 31 transpositions of the meantone tuning scheme are:
– Meantone +15 (B# – Dx#, or C- – E+)
– Meantone +14 (E# – Gx#, or F- – A+)
– Meantone +13 (A# – Cx#, or A# – D+)
– Meantone +12 (D# – Fx#, or D# – G+)
– Meantone +11 (G# – Bx, or G# – C+)
– Meantone +10 (C# – Ex, or C# – F+)
– Meantone +9 (F# – Ax, or F# – B-)
– Meantone +8 (B – Dx, or B – E-)
– Meantone +7 (E – Gx, or E – A-)
– Meantone +6 (A – Cx, or A – D-)
– Meantone +5 (D – Fx, or D – G-)

– Meantone +4 (G – B#, or G – C-)
– Meantone +3 (C – E#, or C – F-)
– Meantone +2 (F – A#)
– Meantone +1 (Bb – D#)
– Meantone ±0 (Eb – G#; the default keymap, keyboard shortcut **M**)
– Meantone -1 (Ab – C#)
– Meantone -2 (Db – F#)
– Meantone -3 (Gb – B)
– Meantone -4 (Cb – E, or B+ – E)
– Meantone -5 (Fb – A, or E+ – A)
– Meantone -6 (Bbb – D, or A+ – D)
– Meantone -7 (Ebb – G, or D+ – G)
– Meantone -8 (Abb – C, or G+ – C)
– Meantone -9 (Dbb – F, or C+ – F)
– Meantone -10 (Gbb – Bb, or F+ – Bb)
– Meantone -11 (Cbb – Eb, or B- – Eb)
– Meantone -12 (Fbb – Ab, or E- – Ab)
– Meantone -13 (Bbbb – Db, or A- – Db)
– Meantone -14 (Ebbb – Gb, or D- – Gb)
– Meantone -15 (Abbb – Cb, or G- – B+)

To find out in which meantone keymaps a particular pitch is contained, you can also use the table *MT pitches* (in folder "Documents").

The transpositions of the meantone tuning can also be accessed via the "Meantone modulator" function. With the Meantone modulator it is in fact possible to modulate beyond the above-mentioned 31 transpositions (all the way to ±372). See the chapter "Meantone modulator" for more information.

Custom keymaps

You can also create your own custom keymaps. Hereby any note(s) of the 31-tone scale may be mapped onto any key(s) of 12-tone MIDI keyboard(s). There are several different possibilities for creating custom keymaps:
– you can specify a selection of 12 notes, and have the same 12 notes assigned to every octave of all 12-tone MIDI keyboards.
– you can assign any key of a 12-tone MIDI keyboard independently to any pitch of the 31-tone scale; even in another octave, or on another pipe division than what the MIDI keyboard in question normally plays.
– you can have all the 12-tone keyboards (manual I, II and pedal) mapped identically, or differently from each other.
– you can mix all of these methods, even within one keymap.

Editing custom keymaps

A custom keymap is a plain text file, which can be edited in any text editor. The file must be saved in "plain text" format (with or without the extension ".txt").
The easiest way to create your own keymaps is to choose and open one of the many keymap templates in a text editor, and to edit it. (Remember to save the keymap under a different name, so that you will not overwrite the template!)
The keymap templates can be found in the folder "Keymap templates". You may also use the *Keymap editor* utility, which includes the four most basic templates.

It is currently not possible to edit the active keymap directly. This means that after editing a keymap, you must first save it, and then (re)activate it – by loading it (either select "Load custom keymap" from the File

menu, or click the "Load custom keymap" button on the Console window), or by inserting an appropriate keymap command in the current registration list (see "Commands in registration lists"). If the keymap you just saved is being used by the current registration event, the keymap will be updated to the newly saved version as soon as you press the Enter key (i.e. give the "Restore current registration" command).
You could also assign keymaps to the buttons of the VMK-88 MIDI keyboard – you could then load a keymap by pressing the appropriate button. To do this, use the *VMK-88 controls editor* utility.

Keymap templates

There are many different templates in the "Keymap templates" folder.
The 4 basic ones are:
– *12tone* : a 12-tone keymap, for assigning a selection of 12 notes (pitch classes) to all octaves on all keyboards.
– *88keys* : an 88-key keymap for 1 manual, where all the keyboards will get their pitch data from manual I.
– *88keys-2man* : an 88-key keymap for 2 manuals, whereby both manuals may be independently mapped.
– *12tone+88keys* : a combined 12-tone / 88-key keymap template.
These 4 basic keymap templates can also be found within the *Keymap editor* utility.
The keymap templates in the "MT" subfolders contain keymaps where one meantone tuning is already given, which you can easily use as a basis for your own modifications.

One line in a keymap template may look something like this:

```
1 C_, ;
```

To specify, for example, that you want the C key to sound as C semi-sharp, type the note you want to hear *inbetween the comma and the semicolon*:

```
1 C_, C+;
```

This is the basic procedure for every line in a keymap. More detailed instructions will be found in the next section ("Structure of a keymap").

The overall format of the templates is not binding, so you can copy and paste lines freely from any of them, as long as no key# appears twice. (If a key# is duplicated, only the first occurrence of it will be read.) The key#s don't even need to be in numerical order. You may (but don't have to) delete unnecessary lines which have no pitch data.

Structure of a keymap

Below is an example of a simple keymap that assigns the traditional meantone tuning scheme on all 12-tone MIDI keyboards:

```
1, C;
2, C#;
3, D;
4, Eb;
5, E;
6, F;
7, F#;
8, G;
9, G#;
10, A;
```

11, Bb;
12, B;

The syntax for every line in a keymap file is the same:

[key#], [pitch data];

Note that there must be a comma between the key # and the pitch data, and the line must end with a semicolon.

[key#] defines the key of a 12-tone keyboard onto which any [pitch data] is to be mapped.
There are six different key# domains, each with a different function.

There are three *12-tone keymapping domains*:
– key#s 1–12 refer to the keys C, C#, D, D#, E, F, F#, G, G#, A, A#, B in all octaves of 12-tone manual I, or of any 12-tone MIDI keyboard.
– key#s 201–212 refer to the keys C, C#, D, D#, E, F, F#, G, G#, A, A#, B in all octaves of 12-tone manual II only.
– key#s 401–412 refer to the keys C, C#, D, D#, E, F, F#, G, G#, A, A#, B in all octaves of 12-tone pedal only.

There also are three *88-key keymapping domains*:
– key#s 21–108 refer to the 88 keys of a MIDI keyboard assigned as 12-tone manual I, or of any 12-tone MIDI keyboard (key# = MIDI note number).
– key#s 221–308 refer to the 88 keys of a MIDI keyboard assigned as 12-tone manual II (key# = MIDI note number + 200).
– key#s 421–508 refer to the 88 (potential) keys of a MIDI keyboard assigned as 12-tone pedal (key# = MIDI note number + 400).

In the keymap templates, [key#] is followed by the note/manual name of the key (*before* the comma). For example:

2 C#, ;
or
61 Man1_c#1, ;

The name of the key is added there just for your convenience, and is not obligatory in a keymap; only the key# is significant.

[pitch data] specifies the pitch(es) to be mapped onto the key#, using note names or note numbers.
(This "pitch data" may actually contain anything, even some remarks to yourself, since only valid note names or numbers will be read and everything else will be ignored. Extraneous commas or semicolons are not allowed.)

Note names for the 12-tone keymapping domains (key#s 1–12, 201–212 and 401–412) define only the pitch class, not the octave. Valid note names are:
Cbb Cb- Cb C- C C+ C# C#+ Cx Dbb Db- Db D- D D+ D# Dx ... and so on, until Bx.

Any note name can thus be accompanied by the 31-tone accidentals bb, b-, b, -, +, #, #+ or x. The note names (C, D, E, F, G, A, B) for the 12-tone keymapping domains are not case-sensitive, so you may also type them in lower case, as cbb cb- cb c- c c+ c# c#+ cx etc.

Note names for the 88-key keymapping domains (key#s 21–108, 221–308 and 421–508) define both the pitch and the octave. Valid note names for the entire 4½ octave range of the Fokker organ are:

C C+ C# C#+ Cx Dbb Db- Db D- D D+ D# Dx ... and so on (until Bx) for the lowest octave; then
cbb cb- cb c- c c+ c# c#+ cx etc. for the next octave, then
cbb1 cb-1 cb1 c-1 c1 (= middle C) c+1 c#1 c#+1 cx1 etc.,
c2 c+2 c#2 etc., and
c3 c+3 c#3 etc. until g3.

A complete table of the 31-tone pitches of the Fokker organ can be found in the file called "HFnote#s.pdf", which can be found in the "Documents" folder. For planning complex custom keymaps, you may find it helpful to print this table (or several copies of it) out on paper, mark out which 31-tone pitches are required in a given section of music, and then decide how to best assign them on the keys of a 12-tone keyboard.

To assign a pitch to a key, type in the 31-tone note name you want *between the comma and the semicolon*. For example:

```
2 C#, d-;
```

Or, simply:

```
2, d-;
```

This assigns the pitch class D- to all C# keys on all 12-tone keyboards (unless there is something else assigned to some C# keys in some of the other 12-tone or 88-key keymapping domains).

In the 12-tone keymapping domains (key#s 1–12, 201–212 and 401–412), if you need to assign to a key a pitch that is not within the same octave (octaves are defined as starting on C and ending on B, disregarding any eventual accidentals), you must add +octave or -octave after the note name (with no space inbetween). These can also be abbreviated as +oct and -oct, or +8 and -8. For example, to assign a C, half a tone higher, to a B key, you should type c+oct (or c+8). If you just typed c, the result would be the C in the same octave, i.e. a seventh lower.

Another example, using the 88-key keymapping domain:

```
61 Man1_c#1, c#2;
```

This assigns the note C#2 (and that note only) to the C#1 key on manual I. If there is nothing assigned to the C#1 key on manual II (=key# 261), then this same assignment is in effect also for the C#1 key of manual II.

The pitch data may contain more than one pitch, separated by spaces. For example:

```
60 Man1_c_1, cb c- c c+ c#;
```

This assigns a 5-tone microtonal cluster to the middle C key of manual I.

Any number of pitches from any octaves may be assigned to any key. Any pitch may also be assigned to any number of different keys.

In the 88-key keymapping domains, a notename may be preceded by Man1-, Man2- or Ped- (for example: Man2-c#3).
This specifies that the pitch will be played on that specific pipe division only, even if the key onto which it is being mapped does not normally play that pipe division.
Virtual couplers will have no effect on notes which are assigned to a specific pipe division in this manner.

In the manual I 88-key domain (key#s 21–108), pitches preceded by Man1-, Man2- or Ped- will apply only to keys played on manual I – and *not* to any manual II or pedal keys, which may otherwise be using pitch data from the manual I domain if they have no pitch assignment in the dedicated manual II or pedal domains.

The pitch data can also be expressed using note numbers instead of note names. Valid note numbers in the 12-tone keymapping domains (key#s 1–12, 201–212 and 401–412) are:
[1...31] = C-B# (in the same octave as the key#)
[32...62] = C-B# in the next octave
[-31...-1] = C-B# in the previous octave
[1...31]+oct(ave) or [1...31]+8 = C-B# in the next octave
[1...31]-oct(ave) or [1...31]-8 = C-B# in the previous octave
Note that the number 0 is not used as a note number; thus the number 0 may be used to represent a silent key.

For note numbers in the 88-key keymapping domains (key#s 21–108, 221–308 and 421–508), you would use the so-called "HFnote#s" (101–519; or their keyboard-specific versions: 1101–1519 for manual I, 2101–2519 for manual II, and 3101–3214 for pedal). A complete table of HFnote#s can be found in the file "HFnote#s.pdf", which can be found in the "Documents" folder.

When key#s from different key# domains are present in one and the same keymap, they will be read in a hierarchical order. In short:
– Pitch data in the 88-key keymapping domains supersede the data in 12-tone keymapping domains.
– Notes played on manual II and pedal will use pitch data from the dedicated keymapping domains of manual II and pedal (88-key and 12-tone domains), if available. If no data is found there, data from the "manual I" (88-key and 12-tone) domains is used instead.

In other words:
– When a key on 12-tone manual I is played, and the corresponding key# (i.e. in the range 21–108) is found in the keymap, the pitch data from that line is mapped onto the key. If the key# (in the range 21–108) is not found, the program looks for the appropriate key# in the 12-tone domain of manual I (key#s 1–12), and uses the pitch data found there, transposed to the appropriate octave. If no pitch data is found there either, the key remains silent.
– When a key on 12-tone manual II or pedal is played, the program first looks in the 88-key domain of manual II/pedal (key#s 221–308 or 421–508), and uses the pitch data found there. If no data is found there, the program looks for the appropriate key# in the 12-tone domain of manual II/pedal (key#s 201–212 or 401–412), and uses the pitch data found there, transposed to the appropriate octave. If no data is found there, the program next looks for the corresponding key# in the manual I 88-key domain (key#s 21–108), and uses pitch data found there (but ignoring all pitches preceded by "Man1-", "Man2-" or "Ped-"). If no (other) data is found in the manual I 88-key domain, the program finally looks in the general (manual I) 12-tone domain (key#s 1–12). If no pitch data is found there either, the key remains silent.

Virtual couplers

The original disposition of the Fokker organ includes the standard pedal and manual couplers (P+I, P+II and I+II). Next to these, *Ere's Fokker organ console* provides additional (artificial) couplers, called *virtual couplers*. They can be used instead and/or in addition to the normal couplers. The virtual couplers allow coupling between and within the manual and pedal keyboards in all possible combinations, both at unison pitch and at different octave transpositions (superoctave and suboctave couplers). They enhance the registration possibilities of the organ, and by using octave couplers you can also increase the total sound volume of the organ. In addition, it is possible to employ different virtual couplers for inputs from different MIDI sources (31-tone input, 12-tone input, and external MIDI source).

On the Console window, on the right-hand side, you see some of the virtual couplers. These are the most often used ones.

To see all the available virtual couplers, click on "Show all virtual couplers" on the Console window, or press **V**.
On the Virtual couplers window, you will find dedicated couplers for each of the three types of MIDI input: 31-tone input, 12-tone input, and input from external MIDI source.
– The couplers effecting 31-tone input (pedal, manual I and manual II) are marked with 31-P, 31-I and 31-II.
– The couplers effecting 12-tone input (pedal, manual I and manual II) are marked with 12-P, 12-I and 12-II.
– The couplers effecting input from external MIDI source (pedal, manual I and manual II) are marked with Ext-P, Ext-I and Ext-II.

The number shown on a virtual coupler's button is its register number. This number is used to represent the virtual coupler in registration lists.

To switch virtual couplers on and off, you can either click on them onscreen, or you can use the computer keyboard as follows:
– To control couplers 10–99, hold down Shift (or use caps lock) and type the coupler number.
– To control couplers 100–119, hold down Shift (or use caps lock) and type the coupler number, but with **A** instead of the initial 1 (i.e. 100–119 become A00–A19). On the numeric keypad, you can also use the decimal separator key (comma or dot) instead of the initial 1.
The Shift key may not work for the above-mentioned commands on some operating systems. On Mac, you can also use the ctrl key instead of Shift.
Key presses with shift / caps lock / ctrl need to occur within 1.5 seconds of each other in order to be recognized.

If any virtual couplers are on, the red indicator on the Console window (to the left of the words "Virtual couplers") will be lit. This is particularly useful when some of the couplers 30–119 are on, but the Virtual couplers window itself is not visible.
To quickly switch off all virtual couplers, you can click on the red indicator light, or on the "All virtual couplers off" button, or you can press **C** (as for "**C**lear **C**ouplers").

The virtual couplers on the Console window (couplers 10–28) are in fact "group switches" that control the individual virtual couplers for all of the three MIDI input sources simultaneously. For example, switching on virtual coupler 18 (I + II 4') switches on the I + II 4' couplers for all the input sources (i.e. couplers 44, 74 and 104). In registration lists, the presence of any coupler number 10–28 also indicates that the corresponding couplers in the range 30–119 are on.
Some of the less used virtual couplers are not available amongst the "group switches" (couplers 10–28) on the Console window; they are found only on the Virtual couplers window.

When dealing with only one MIDI input source (for example, input from 31-tone keyboards), you can simply use the "group" couplers 10–28, without having to worry about couplers 30–119.
On the other hand, if you have simultaneous input from several MIDI sources, it is possible to employ different virtual couplers for each of them, even when they are "playing on the same manual". For example:

imagine that you are playing on 31-tone manual I, and simultaneously receiving input from an external MIDI source, also on (the channel of) manual I. If you want to couple the 31-tone manual I to the Fokker organ's manual II but you don't want the external manual I to be coupled, then you should not use the normal I+II coupler, but use virtual coupler 43 (31-I + II) instead. This way, only 31-tone manual I will be coupled to manual II, and the external manual I remains uncoupled. If you used the normal I+II coupler, manual I from both inputs would be coupled to manual II; the same would also happen if you used the equivalent virtual "group" coupler I+II (coupler 17).
For this reason, all the normal couplers (P+I, P+II and I+II) are also available as virtual couplers. When working with complex combinations of virtual couplers, it is often more beneficial to entirely refrain from using the normal couplers, and use their virtual counterparts instead.

There is another important difference between the regular couplers and the virtual couplers. The regular couplers, like all the stop knobs on the real Fokker organ, have a slight delay between the moment a register change message is sent (for example, at a synchronized registration change) and the moment that the register or coupler is actually engaged or disengaged. This is because the stop knob on the real organ needs to physically move for the change to happen. *Virtual couplers do not have this delay* – they are activated or deactivated immediately, as they involve no moving parts. This makes a difference in how they can be used in connection with synchronized registration changes. In other words, if only virtual couplers are changing at a registration event, you don't need to take into account the delay that is usually associated with registration changes. (Read more about this in the section "Synchronizing registration events with MIDI input".)

One particularly useful function for virtual couplers is when you want to play a 4' register at 8' pitch. You can do this by using a 16' coupler (in order to sound the 4' register one octave lower). For example, to couple the Roerfluit 4' to manual I, but at 8' pitch (as if it was a Roerfluit 8'), you can use virtual coupler 16 (I + II 16'). Or, to play the Prestant 4' alone on manual I, but at 8' pitch (as if it was a Prestant 8'), you can use virtual couplers 13 (I 16') and 14 (I 8' off) together (however, consider the limitations mentioned below).

Note: The 8' off couplers do not work on the 31-tone keyboards of the real Fokker organ. This is because the organ console sends its regular note messages always directly to the organ pipes, and there is no way of turning this behavior off.

The 8' off couplers can be successfully used with 12-tone MIDI keyboards and with input from an external MIDI source. For these inputs, it is even possible to completely reverse manuals I and II using virtual couplers (for example, by using virtual couplers 14, 17, 24 and 27 all together).
The 8' off couplers are also functional when using the 31-tone practice Fokker keyboard, with the option "Use 31-tone practice keyboard instead of Fokker organ console" selected.

However, it is very important to keep in mind that **virtual couplers do not add any pipes to the organ**.
Virtual couplers can only use the pipes that physically exist within the normal range of the instrument.
For example: if you're using a virtual coupler at 4' pitch (one octave higher), the highest manual key that can be played with this coupler is g2 (and *not* the normal upper limit g3). This is because every note is transposed an octave higher – the g2 key plays the g3 pipe – and there is no higher pipe than that available. In other words: with a 4' coupler, the highest octave of the normal range will be silent; and similarly with a 16' coupler, the lowest octave is silent.
Also, virtual couplers can not double any existing pipes. For example, if you play the octave c1+c2 on one register on manual I (sounding two pipes), and then add the virtual coupler I 4', the result will not be four pipes sounding, but just three: c1+c2+c3. The note c2 will not be doubled, because there is only one pipe for that pitch (in each register).

The pedal-on-manual couplers (which allow the pedal registers to be played from a manual keyboard), of course, only function in the range of the pedal registers, i.e. C to f – or in the case of the 16' pedal-on-manual couplers, c to f1.
The *bass couplers* are a special variety of pedal-on-manual couplers. With a bass coupler, only the lowest currently sounding note of the manual texture is coupled to the pedal (even if there were several notes being played in the range of the pedal registers).

Note that the textures from each of the different MIDI inputs are considered separately by the bass couplers. The bass couplers function as follows: Whenever a new lowest note enters the texture, the bass coupler will take that note (and drop the previous lowest note, if one was sounding). If the note currently being played by the bass coupler is released but no new bass note is played, the bass coupler will be silent until a new lowest note is played; i.e. the bass coupler will not revert to sounding a previously attacked note.
However, for practical reasons, the bass couplers have been designed with some tolerance for overlegato. This means that if a bass note is played somewhat before the previous one is released, the bass coupler will still take the new bass note – even if it is a higher note – if the release of the previously sounding note occurs within a specified tolerance time. By default, the tolerance time is set at 150 milliseconds. If you wish, this can be modified in the "Advanced settings" window (at "Bass coupler overlegato tolerance"). (Setting the tolerance time to a very high value would effectively change the behavior of the bass couplers to such that they would always be playing the current lowest note of the texture, regardless of when this note had been attacked.)
Since the most common use for any pedal-on-manual coupler would be the one where just a bass line needs to be coupled to the pedal, only bass couplers are included among the most often used virtual couplers on the Console window. For most uses, i.e. whenever no chords need to be coupled to the pedal, the bass couplers will indeed function exactly the same as the other pedal-on-manual couplers.

Transposer

The MIDI input from 12-tone MIDI keyboards or an external MIDI source can be easily transposed up or down by 1/5 tones using the *Transposer* function.
The range of the transposer is 31 dieses (= one octave) up and down. (Diesis = 1/5 tone, the smallest interval of the 31-tone system.)
The transposer also works in combination with preset and custom keymaps.

You can simply enter a number in the "Transposition" number box at the lower left corner of the Console window, or click the box and use the up/down arrow keys to change the value. (Note that you must click again outside the number box in order to use the computer keyboard for other commands.)
Or, you can click the "Transposer" button to open the Transposer window. In the Transposer window there is a pop-up menu with more details about the possible transpositions.

When the transposer is off (i.e. transposition = 0), a1 will sound at (about) 443 Hz. This is the Fokker organ's standard pitch. Using the transposer, you can change the pitch in 1/5 tone increments – for example to play together with instruments tuned at a lower (Baroque) pitch. Some of the most useful transpositions for such purposes are:
+2 dieses, a1 = 463 Hz
+1 dieses, a1 = 453 Hz
-1 dieses, a1 = 433 Hz
-2 dieses, a1 = 424 Hz
-3 dieses, a1 = 414 Hz
-4 dieses, a1 = 405 Hz
-5 dieses, a1 = 396 Hz

If you wish the lowest keys on the manuals to remain playable when transposing downwards, keep the option "Transfer notes below normal manual range automatically to pedal" selected. Note that for this to work, the pedal needs to have a suitable registration with 16'.

Note: Transposer does not work properly when playing on the 31-tone keyboards of the real Fokker organ.
This is because the organ console always sends note messages directly to the organ pipes, and there is no way to transpose them before that happens.
The only way to play transposed on the 31-tone keyboards of the real Fokker organ is to play on one manual with no registers on, and use a virtual coupler to couple this manual to the other manual which does have registers on.

Note length control

The *note length* function lets you add extra length to notes played through *Ere's*. This could be used to help with legato playing, or to lengthen notes in MIDI files that have too short notes (i.e. notes that are so short that the organ pipes do not sound properly), as well as for special effects. Another practical example: adding 30–35 ms of extra length to notes played on the VMK-88 keyboard can make the playing experience on that keyboard significantly more pleasant.

The note length function works by delaying received noteoff messages by the specified amount of time. (Noteon messages are not affected in any way.) The delaying of noteoff messages takes place immediately after the MIDI messages have been received by *Ere's*. Even though the MIDI in monitors on the Setup and Console windows do show the real timing of the received noteoff messages (= key releases), in all other respects (including the MIDI out monitor on the Setup window) *Ere's* behaves as if the noteoff messages were only received after the delay time specified by Note length controls. For example, if a sync command is waiting for a specific noteoff message, it will be triggered not when the player releases the key, but only after the specified delay time has elapsed.

If a new noteon message arrives while a noteoff message for the same key is still being delayed by the note length function, *Ere's* will ignore the new noteon message, and act as if the notes were tied together instead of being repeated. In other words: repeated notes will become tied notes if the time between them is less than the extra note length set in Note length controls. This will also affect the sync function's ability to recognize a repeated note.

To open the Note length control window, select it from the "Ere's Fokker organ console" menu, or click on the "Note length control" button on the Console window. By using the number boxes in the Note length control window, you can specify the extra length (in milliseconds) that is to be added to notes played.

Notes from each of the nine different MIDI inputs (Fokker organ manual I, II and pedal; 12-tone manual I, II and pedal; and External manual I, II and pedal) can be treated differently and independently. There are separate number boxes for each of the nine inputs; these are the nine boxes on the lower right-hand side. Above them, there are three number boxes with which you can change the values for all keyboards of the Fokker organ, all 12-tone keyboards and all External inputs, respectively. To the left, there are three number boxes by which you can change the values for all the Manual I, Manual II and Pedal inputs, respectively. To change the value for all MIDI inputs at once, use the first number box at the top left corner.

The "Reset all" button sets the values in all the number boxes to 0, i.e. deactivates the note length function.

Since it is possible that the values shown in (some of) the seven "collective" number boxes don't agree with the values shown in (some of) the nine number boxes for the individual inputs, the values shown in the individual number boxes always take precedence. (Whenever a value in a "collective" number box does not apply to all its related individual number boxes, it is shown in italics.)

Note length values can be included in a registration list, in order for them to be applied automatically. For the commands to use, see the "Note length" section under the chapter "Commands in registration lists".
The note length values are automatically set to 0 when loading a registration list, and they will remain inactive until the point in the registration list where some note length commands are given (if any), or until note lengths are specified manually using the number boxes in the Note length control window.
Currently it is not yet possible to store the note length values (as specified in the Note length control window) automatically in a registration list. You must enter the relevant note length commands into the registration list manually.

Registration lists

Registration lists are the most powerful feature of *Ere's Fokker organ console*. In spite of their name, they can be used not only for storing and recalling register combinations, but they can actually control almost all of the available functions at the push of a button, and even automatically following MIDI input from a live performer. Registration lists may thus contain many different kinds of commands, and they might in fact better be called *event lists*. You could even have a registration list with no registrations at all, for example a list that contains only keymap changes. All the settings, too, may be (indirectly) modified through registration lists, using the so-called registration list initial commands (see chapter "Registration list initial commands").

Editing registration lists

Registration lists are plain text files (just like the keymaps), and can be edited in any text editor, as well as using the "Registration list editor". To open the registration list editor window, click on the "Edit registration list" button on the Console window, or select "Registration list editor" from the "Ere's" menu.

The current registration, as shown on the Console / Virtual couplers windows, can be stored in the registration list by simply pressing **shift-S**. This opens a dialog window where you can enter any registration event # where you wish the current registration to be stored in. The current event number is offered by default, so if you want to store the registration at the current event number, simply press Enter.
If you would rather store the registration at the latest event number with a registration change (which is not necessarily same as the current event number), you can press **shift-D** instead, and this event number will be offered by default.
Storing the registration in this way replaces all the register numbers previously stored at that registration event number with the currently active ones, but leaves all other commands (keymap, sync etc.) intact.
The range of valid registration event numbers is 1–9999. The event numbers do not need not be consecutive.

In order to add other commands into a registration list, you need to edit it directly using either a text editor or the "Registration list editor" (click on the "Edit registration list" button on the Console window, or select it from the "Ere's" menu). For this you need to know the structure of a registration list, which will be explained further below (in the section "Structure of a registration list").
With the Registration list editor, you are editing the currently active registration list. However, edits made in the list editing window will not be applied until you click on the "Registration list editor" window again. Please note that as long as the list editing window is the active window, the computer keyboard will only be typing into the editing window; other computer keyboard commands become active again only when you close the editing window, or click on another window.

Navigating a registration list

To move forwards and backwards in a registration list, thus recalling the registrations and executing the commands stored therein, you can:
– click on the "Next registration" and "Previous registration" buttons on the Console window.
– press the **left** and **right arrow** keys (or **+** and **-** keys) on the computer keyboard.
– click on the "registration event #" box on the Console window, type in a number and press Enter. If the number you typed doesn't exist in the registration list, the next smaller event number will be chosen. (You must click again outside the number box in order to free the computer keyboard for other commands.)
– send MIDI control change messages from a MIDI keyboard, pedal switch or other controller (for more information, see the chapter "Control changes" and/or open the *VMK-88 controls editor* utility).

If you want to move quickly forward in a registration list, without executing any of the eventual sync hold or delay commands, you can press **shift-right arrow**. (On a Mac, **alt-right arrow** works as well.)

To return to the beginning of the registration list, you can:
– click on the "Go to first registration" button on the Console window.
– press the shortcut key for "Go to first registration": by default, this is the **apostrophe** key, or the **Home** key. (It is possible to assign some other key instead of the apostrophe key for the "Go to first registration" command; to do this, use the *Command keys editor* utility.)

When the current registration has been altered manually, and thereby does not correspond any more to the registration stored at the displayed registration event number, the registration event # display turns grey to remind you of this. If you wish to revert to the registration stored in the registration list, click on the "Restore current registration" button, or press its keyboard shortcut, the **Enter** key.
The "Restore current registration" command also causes the whole registration list to be reread, including all the keymaps required in the registration list, and updates the program's memory in case there have been any changes.

Structure of a registration list

Each line in a registration list represents one registration *event*, and the correct syntax for each line is as follows:

[event#], [any data];

Note that the event # must be followed by a comma, and the line must end with a semicolon.

[event#] can be any integer between 1 and 9999. The event numbers in a registration list need not be consecutive.
[any data] can be register numbers, other commands, or even any remarks to yourself. All commands and other items are separated by spaces, and they may be written in any order. Anything that is not understood as a command will simply be ignored. (If you wish to write a longer remark to yourself, containing several words and/or numbers, you could enclose it in quotes, to ensure that no individual words/numbers within it will be interpreted as commands.)

The elements of any one command must in principle be typed in without any spaces inbetween. If the command includes a filename or another expression that contains spaces, the whole command must be enclosed in quotes.
Additional commas or semicolons are not allowed in a registration line, except if they are within quotes.

The registration list parser understands the following commands:
– numbers 1 to 9: registers 1–9
– numbers 10 to 119: virtual couplers
– negative numbers: custom registers
– number 0: all registers off
– keymap
– transposition
– split manual I
– note length
– name
– sync hold (these commands start with the "@" sign)
– delay
– regsync
– halt
– goto
– skip

The precise syntax of each command will be given in the next subchapter ("Commands in registration lists").

First, here is an example of a simple registration list:

```
1, 1 3 5 8;
12, 1 3 5 7 8;
35, 1 5;
```

Explanation:
– At registration event # 1, the registers 1, 3, 5 and 8 will be on.
– At event # 12, register 7 is added.
– At event # 35, registers 3, 7 and 8 are disengaged; registers 1 and 5 remain.
These event numbers could represent bar numbers 1, 12 and 35 of a piece of music.

Another example:

```
1, 1 2 5 6 8 km=MyKeymap1;
2, 1 2 3 4 5 6 7 8 9;
3, km=MyKeymap2;
4, 6 8 @c1;
5, km=MT-1 1 6 8;
6, 1 2 3 4 5 6 7 8 9 18 @off-c2 @off-e2 @Man2-C delay=1000;
```

Explanation:
– At registration event # 1, registers 1, 2, 5, 6 and 8 are on, and a keymap named "MyKeymap1" is used.
– At event # 2, all registers (1–9) are on.
– At event # 3, keymap is changed to "MyKeymap2". The registration remains unchanged.
– Immediately after event # 3, event # 4 is automatically put on hold, because line # 4 has the sync hold command "@c1". The program starts monitoring incoming MIDI messages, and when middle C (c1) is played, event # 4 is activated, whereby the registration changes to 6 + 8.
– When a "Next registration" command is given to go to event # 5, the keymap changes to Meantone-1 (meantone tuning, transposed a fifth down), and register 1 is added.
(Event # 5 will not be triggered automatically since it has no sync hold or delay command.)
– Immediately after event # 5, event # 6 is automatically put on hold – because of sync hold and delay commands present – and the program starts monitoring incoming MIDI messages. When c2 or e2 is released (on any keyboard), or when low C on manual II is played, event # 6 is activated, and all registers plus virtual coupler 18 will go on – but only after an additional delay of 1 second (1000 milliseconds).

If there are no register numbers, or no keymap / transposition / split / length command at a given registration event, the registers / keymap / transposition / split / length status from the previous event(s) in the list are considered to remain in force. This rule is observed also when moving backwards in the list, or when jumping directly to an event number.

Commands in registration lists

In this section, where the possible syntax variations of a command are schematically given, the following conventions apply:
– Items in [brackets] are to be replaced by a number or an expression as described afterwards (but without the brackets). These items are obligatory, unless stated otherwise.
– Characters enclosed in parentheses may be omitted at will. (The parentheses themselves are not to be typed in, of course.)

Registers

– numbers 1 to 9: registers 1–9
– numbers 10 to 119: virtual couplers
– negative numbers: custom registers
– number 0: all registers off

To turn all registers off, use the number 0 by itself. (In the presence of other numbers, zeros will be ignored.) If no registers (or 0) are specified at the beginning of a registration list, it is interpreted as "no change".

Keymaps

Syntax:
keymap=[keymap] or
km=[keymap] or
k=[keymap]

[keymap] can be the filename of a saved custom keymap, or the abbreviated name of a preset keymap. For more information on keymaps, read the main chapter "Keymaps".

The abbreviated names of the preset keymaps are:
– EF1 to EF8 for the eight Euler-Fokker genera.
– MT[#] for the traditional meantone tuning scheme, where [#] indicates the transposition. MT0 = meantone Eb–G#, MT-1 = meantone Ab–C#, MT+1 = meantone Bb–D#, etc. [#] can be between -372 and +372.
– Meantone tunings can also be specified in the formats MT[notename]-[notename], MT[notename](-) or MT-[notename], specifying one or both of the limit notes. Besides natural notes, note names with double flats, flats, sharps and double sharps may be used, and these may be written in upper or lower case characters. (Microtonal accidentals are not allowed here.) If both limit notes are given, but they don't describe a proper 12-tone segment on the circle of fifths, only the the lower limit (flat side) is considered, and the other limit is ignored.

The default keymap at startup is Meantone 0 (Eb–G#). If no keymap is specified at the beginning of a registration list, any keymap used previously will remain in effect.

If the keymap name contains spaces, the name must be enclosed in quotes (in fact the whole command – but this will be automatically corrected for you).
The default extension for custom keymap files is ".txt". The ".txt" extension may be omitted from filenames in registration lists. Exception: if a folder with the exact same name exists within Max's search path, the ".txt" extension may not be omitted.

So that a keymap file can be found when it is specified by its filename only, the file must be located within Max's search path. By default, the search path includes:
– the folder where "Ere's" is located

– the folders "Keymaps", "Registrations", "Settings" and "Patches" (if found in the same folder as "Ere's") and all their sub-folders
– the Max application folder and all its sub-folders.
The folder "Keymaps" is thus the obvious place to store your custom keymaps in.
If a keymap file is located outside the search path, it must be specified in the registration list using its full filepath. Use the *Filename utility* to find out the correct filepath of a keymap file. If the filepath contains any spaces, the entire filepath has to be enclosed in quotes.

Transposition

Syntax:
transposition=[-31...31] or
transp=[-31...31] or
tr=[-31...31] or
t=[-31...31]

The number indicates the amount of transposition in dieses (one diesis = 31th part of an octave, the smallest interval of the 31-tone system). Maximum transposition is one octave (31 dieses) up or down. More details can be found in the pop-up menu inside the Transposer window (to get there, select "Transposer" from the "Ere's" menu, or click on "Transposer" on the Console window).

If there is no transposition command at the beginning of a registration list, transposition is automatically set to 0 (i.e. no transposition) when the registration list is loaded. In subsequent use, however, the absence of a transposition command at the beginning of the registration list is interpreted as "no change".

Split manual I

A 31-tone or a 12-tone keyboard that is normally assigned as manual I can be split between manuals I and II. The split command defines which part of manual I becomes manual II, and at which pitch it should play manual II.

Syntax:
split=[lower limit]-[upper limit] or
split[pitch]=[lower limit]-[upper limit]

split may be abbreviated to s.
[pitch] can be 64, 32, 16, 8, 4, 2 or 1 (as in regular register pitch indications); if omitted, default is 8 (= unison pitch).
[lower limit] and [upper limit] define the range on manual I which becomes manual II. Either one (or both) of the limits may be omitted, meaning "until the end of the keyboard". The "-" must not be omitted, though. If both lower and upper limits are specified, they may be given in any order.
For 31-tone manual I, the limits can be expressed in notenames (C to g3, with any accidentals from double flats to double sharps) or HFnote#s (101–519 or 1101–1519).
For 12-tone manual I, the limits can be expressed in 12-tone notenames (AAA to c5, naturals, sharps or flats) or MIDI note numbers (21–108).
Whenever applicable, the split command applies simultaneously to both 31-tone and 12-tone manual I. If you want the split command to apply only to 31-tone manual I or only to 12-tone manual I, express at least one of the limits in a manner that does not apply to both.
The command split=0 deactivates the split function (in fact, any invalid expression on the right side of the equal sign will do the same).

The split function can be activated only through a command in a registration list. The split function can be

(temporarily) deactivated with an "All registers off" command (shortcut key **0** or **shift-0**).
If there is no split command at the beginning of a registration list, the default is "no split".

Note: Split manual I does not work on the 31-tone keyboards of the real Fokker organ, because there is no way to prevent the organ console from sending note messages from manual I directly to the manual I pipes. Consequently, split manual I can apply to a 31-tone manual I only when playing on the 31-tone practice Fokker keyboard. Also the option "Use 31-tone practice keyboard instead of Fokker organ console" must be selected.

Note length

The note length commands have the same effect as the number boxes in the Note length control window. They control the amount of extra length added to notes played through *Ere's*. For more information, see the main chapter "Note length control".
Notes received through different MIDI inputs can have different values of extra length (or none). Similarly to the 16 number boxes in the Note length control window, there are 16 different note length commands available.

Syntax of possible note length commands:

length=[#](ms)	affects all notes from all MIDI inputs
length-FOrg=[#](ms)	affects all notes from Fokker organ
length-12t=[#](ms)	affects all notes from 12-tone keyboards
length-Ext=[#](ms)	affects all notes from External MIDI source
length-Man1=[#](ms)	affects all Manual 1 notes
length-Man2=[#](ms)	affects all Manual 2 notes
length-Ped=[#](ms)	affects all Pedal notes
length-FOrgMan1=[#](ms)	affects notes from Fokker organ manual I
length-FOrgMan2=[#](ms)	affects notes from Fokker organ manual II
length-FOrgPed=[#](ms)	affects notes from Fokker organ pedal
length-12tMan1=[#](ms)	affects notes from 12-tone manual I
length-12tMan2=[#](ms)	affects notes from 12-tone manual II
length-12tPed=[#](ms)	affects notes from 12-tone pedal
length-ExtMan1=[#](ms)	affects notes from External manual I
length-ExtMan2=[#](ms)	affects notes from External manual II
length-ExtPed=[#](ms)	affects notes from External pedal

length may be abbreviated to len or l.
[#] is the extra note length in milliseconds.
To turn off the note length function, use the command length=0.

If there are several note length commands at one registration event, they will be parsed in the following hierarchical order:
– Values given by the general "length=" command are superseded by the device-specific values given by the "-FOrg", "-12t" and "-Ext" commands.
– Values given by any of the above-mentioned commands are superseded by the keyboard-specific values given by the "-Man1", "-Man2" and "-Ped" commands.
– Values given by the "-FOrgMan1", "-FOrgMan2", "-FOrgPed", "-12tMan1", "-12tMan2", "-12tPed", "-ExtMan1", "-ExtMan2" and "-ExtPed" commands supersede all other length commands.

If there are no note length commands at the beginning of a registration list (or no note length commands for some MIDI inputs), all the missing note length values are automatically set to 0 (i.e. no extra length) when the registration list is loaded. In subsequent use, however, the absence of a note length value at the beginning of the registration list is interpreted as "no change" (instead of being set to 0).

If a particular MIDI input has no note length specified at a registration event, the previously valid note length value for that MIDI input remains in force (even if values for other MIDI inputs were changed). If there was no previous value given for that MIDI input at all in the registration list, it is interpreted as "no change".

Name

Syntax:
name=[name] or
n=[name]

This command displays a name for the registration event on the Console window, just below the registration event number. [name] can be anything. If the name contains spaces, it must be enclosed in quotes.

Sync hold

A registration change can be automatized and synchronized with specified MIDI note events. These MIDI events can be either keys played on the keyboards, or note messages received from other MIDI sources. To activate this function, insert one or more sync hold commands at the registration event.

If the next registration event in the registration list contains a sync hold command, the sync hold function will be automatically activated (unless countered by a "halt" command). When a registration change is "on hold", and waiting for the specified MIDI event(s) to occur, the "next on hold" button on the Console window is lit red.
The registration "on hold" can be released prematurely by giving a "Next registration" command.
The sync hold can be canceled (without release) by clicking the "next on hold" button, by selecting "Cancel sync hold / delay" from the Commands menu, or by pressing the **Backspace** key.
If a sync hold has been canceled, it can be reactivated by giving a "Next registration" or a "Restore current registration" command.
Any number of sync hold commands can coexist at one registration event. The registration change is triggered when any one of the specified MIDI events occurs.

Syntax of possible sync hold commands:

@(next)mess(age)	@mess(age)(*)[#]
@(next)noteon	@noteon(*)[#]
@(next)noteoff	@noteoff(*)[#]
@(next)pedmess(age)	@pedmess(age)(*)[#]
@(next)ped(note)on	@ped(note)on(*)[#]
@(next)ped(note)off	@ped(note)off(*)[#]
@(next)(note)(on-)[notename]	@(note)(on-)[notename]*[#]
@(next)(note)off-[notename]	@(note)off-[notename]*[#]

@(next)(note)(on-)[Man1/Man2/Ped](-)[notename]
@(note)(on-)[Man1/Man2/Ped](-)[notename]*[#]
@(next)(note)off-[Man1/Man2/Ped](-)[notename]
@(note)off-[Man1/Man2/Ped](-)[notename]*[#]

Characters in parentheses may be omitted.
[#] stands for occurrence number (how many times the MIDI event should occur before the registration event is triggered); if absent, default is 1 (= trigger at first occurrence of the specified MIDI event). An occurrence number of 0 makes the command inactive.

The commands without a note name are triggered by any MIDI event of its kind (any MIDI message, any

noteon message, any noteoff message, any pedal message, any pedal noteon message, any pedal noteoff message).
The commands with a note name are triggered when a MIDI message (noteon / noteoff) with the specified pitch is received.

For [notename], valid note names for the entire range of the Fokker organ's 31-tone keyboards are:
C C+ C# C#+ Cx Dbb Db- Db D- D D+ D# Dx ... and so on (until Bx) for the lowest octave; then
cbb cb- cb c- c c+ c# c#+ cx etc. for the next octave, then
cbb1 cb-1 cb1 c-1 c1 (= middle C) c+1 c#1 c#+1 cx1 etc.,
c2 c+2 c#2 etc., and
c3 c+3 c#3 etc. until g3.

Valid note names for the full 88-key range of a 12-tone MIDI keyboard start with
AAA AAA# BBBb BBB; then
CC CC# etc.; then
C C# etc.,
c c# etc.,
c1 c#1 etc., until c5.

By adding Man1, Man2 or Ped before the note name, you can specify a key on that keyboard only.

All note names that are natural, sharp or flat refer simultaneously to both 31-tone and 12-tone keyboards. Note names with any other accidentals refer to 31-tone keyboards only.

Delay

Delays the execution of a registration event by a specified amount of time.

The delay command is one of the two registration automation commands (the other one being sync hold) which will cause the program to automatically advance to the next registration event. If the next registration event in the registration list contains a delay command (without any sync hold commands), the delay will become active immediately after the execution of the previous registration event. After the specified delay time, the next registration event will be executed. This behavior could be used, for example, to create an automatic chain of pre-timed registration changes. But if you don't want the delay in the next registration event to be activated automatically, this can be countered with a "halt" command (see below).
The delay command is also very useful in combination with sync hold command(s) at the same registration event. The delay will always take place *after* the sync hold.
When a delay is active, the "next on hold" button on the Console window is lit yellow.
The delay can be truncated by giving a "Next registration" command.
The delay can be canceled (without advancing to the next registration) by clicking the "next on hold" button, by selecting "Cancel sync hold / delay" from the Commands menu, or by pressing the **Backspace** key.
If a delay has been canceled, it can be reactivated by giving a "Next registration" or a "Restore current registration" command.

The delay can be expressed either in milliseconds or as a metronomic tempo value, with an eventual factor (an integer or a floating point value, to express several and/or fractional beats of the tempo), and/or an eventual extra amount of time to add or subtract (in milliseconds).

Syntax of possible delay commands:
delay=[#](ms)
delay=MM[#]
delay=MM[#]*[#.#]
delay=MM[#]+[#](ms)

delay=MM[#]*[#.#]+[#](ms)
delay=MM[#]-[#](ms)
delay=MM[#]*[#.#]-[#](ms)

delay may be abbreviated to d.
Characters in parentheses may be omitted.
[#](ms) (= time in milliseconds) can be any integer >=0.
MM[#] (= metronome number) can be any integer >=1.
[#.#] (= factor) can be any float or integer >= 0.

Note: In case there is a delay command but no sync hold command, the setting "Delay time before activating sync hold" (in the Advanced settings window) imposes a lower limit to the shortest possible delay achievable with the delay command. (The default value is 150 ms.) This preset delay time is observed even when there is just a delay command without sync hold commands. The time spent on this preset delay will be subtracted from the time specified in the delay command, so that the total delay time will be exactly as specified, but the total delay time can never be shorter than that. If you need shorter delays, you must set a shorter value for "Delay time before activating sync hold" in the Advanced settings window.

Halt

Syntax:
halt

The sync hold and delay commands normally cause the program to advance automatically to the next registration event, and to immediately activate the sync hold or delay. If you don't want this to happen automatically, the halt command will prevent it. You will then have to give a "Next registration" command to go to the next registation event. The halt command is placed on the line preceding the line with the sync hold or delay command(s).
The halt command is not indispensable, however: the same result can be achieved by inserting an empty registration line (with no commands) before the one with the sync hold or delay command(s).

Go to

Syntax:
goto[#] or
goto=[#]

[#] defines the registration event number where to go to next when a "Next registration" command is given. If the registration event at the goto address has sync hold or delay commands, the program will advance there automatically just as if the registration events were consecutive.

Skip

Syntax:
skip

Inserting this command in a registration line causes the program to ignore the line and skip over it both at "Next registration" and "Previous registration" commands. Any register, keymap, transposition or split commands on a skipped line will, however, be read as usual if the following line(s) lack any of those elements.
The skipped registration event can still be accessed by typing its number directly into the registration event

number box.

Regsync

Syntax:
regsync or
sync

On the real Fokker organ, the register knobs must physically move to a new position before a registration change is heard. This causes a small delay (averaging about 157 ms) between the moment a register message is sent and the moment the change is heard. This can cause some pipes in registers that are about to be disengaged to make unwanted sounds before the register knobs actually have had time to move.
The regsync command's function is to synchronize played notes with the movement of the Fokker organ's register knobs.
When a registration event that includes a regsync command is executed, the program first sends register change messages for registers 1 to 9 as usual, but simultaneously holds on to all incoming note messages for a period of 157 ms, thus giving the register knobs of the Fokker organ enough time to move to their new positions before releasing the held note messages and restoring the normal flow of note messages.
Similarly, regsync will synchronize virtual couplers with the real registers by delaying any changes to the virtual couplers by the same amount of time.
However, consider that in many cases a better solution to this problem would be to synchronize the registration change with a preceding noteoff message.

Note: regsync command has no effect when playing on the 31-tone keyboards of the real Fokker organ, because the organ console always sends note messages directly to the organ pipes, and there is no way of delaying them.
However, even when playing on the Fokker organ's 31-tone keyboards, the regsync command can be effectively employed in instances where a register change doesn't need to be exactly simultaneous with a particular key press or release, but where it is desirable to precisely synchronize regular register changes and virtual coupler changes (for example, if such a register change is to occur during a held note).

Registration list initial commands

Registration list initial commands are commands that are executed when the registration list is loaded. Commands are available for the following functions:
– Load settings
– Change individual settings
– Load VMK-88 settings
– Load Fokkerboard settings
– Load crescendo list
– Load custom console
– Load Max patch

Each command takes up one line in a registration list, and can appear only once in a registration list (with the exception of the loadMaxpat command, which may occur multiple times). The command lines may be placed anywhere in the registration list, and in any order, although it is most practical to place them at the beginning of the list.
The command lines do not have a line number, but start with the command itself. The basic syntax for all registration list initial commands is (the brackets are not to be typed in):

[command], [filename or parameter];

Filenames that contain spaces must be inside quotes.
The default extension for filenames in registration lists is ".txt". The ".txt" extension may be omitted from the filenames. Exception: if a folder with the exact same name exists within Max's search path, the ".txt" extension may not be omitted.

So that a file can be found when it is specified by its filename only, it must be located within Max's search path. By default, the search path includes:
– the folder where "Ere's" is located
– the folders "Keymaps", "Registrations", "Settings" and "Patches" (if found in the same folder as "Ere's") and all their sub-folders
– the Max application folder and all its sub-folders.
If a file is located outside the search path, it must be specified using its full filepath. Use the *Filename utility* to find out the correct filepath of a file. If the filepath contains any spaces, the entire filepath has to be enclosed in quotes.

To re-execute the initial commands in a registration list, reload the registration list.

Load settings

Loads settings from a file, or recalls the default settings.

Syntax:
loadSettings, [filename];
loadSettings, default;

Filenames that contain spaces must be inside quotes.
Use default instead of a filename to recall the default settings.

Change individual settings

Any of the settings that can be saved in a settings file can also be individually changed through registration list initial commands. This can be useful if you want a registration list to change only some settings, without

loading a whole settings file. Or, you can also load a settings file (or recall the default settings) *and* additionally modify certain individual settings. This is possible because the "load a settings file" command is always executed *before* changing any individual settings – regardless of the order in which the commands appear in the registration list.

Syntax:
[setting name], [parameter];

The setting names used here are the same as can be found in a saved settings file. These are, with their required parameters:

midiInputFOrg	name of MIDI device
midiInput12tMan1	name of MIDI device
midiInput12tMan2	name of MIDI device
midiInput12tPed	name of MIDI device
midiInputCtrlPed	name of MIDI device
midiInputExt	name of MIDI device
midiInputExtMode	31-tone mode = 0; 12-tone mode = 1; off = 2
midiOutput	name of MIDI device
midiOutput2	name of MIDI device
LowestOctToPed	0 = no; 1 = yes
12tManAsMTMod	0 = no; 1 = yes
RegCtrlMan1	0 = no; 1 = yes
31tPedAsVolPed	0 = no; 1 = yes
ExtRegSync	0 = no; 1 = yes
BlockExtReg	0 = no; 1 = yes
IgnoreExtDuplicates	0 = no; 1 = yes
PracticeSetup	0 = no; 1 = yes
PracticePedal	0 = no; 1 = yes
Flash@Reg	0 = no; 1 = yes
Beep@Reg	0 = no; 1 = yes
IgnoreAutoadvance	0 = no; 1 = yes
disableFOrgMan1	0 = no; 1 = yes
disableFOrgMan2	0 = no; 1 = yes
disableFOrgPed	0 = no; 1 = yes
ch12tMan1	MIDI channel 1–16
ch12tMan2	MIDI channel 1–16
ch12tPed	MIDI channel 1–16
chExt31tMan1	MIDI channel 1–16
chExt31tMan2	MIDI channel 1–16
chExt31tPed	MIDI channel 1–16
chExt12tMan1	MIDI channel 1–16
chExt12tMan2	MIDI channel 1–16
chExt12tPed	MIDI channel 1–16
chRegShortcuts	MIDI channel 1–16
chFOrgButtonBlink	MIDI channel 1–16
chCustomReg	MIDI channel 1–16
ctrlAllNotesOff	MIDI control change 0–127
ctrlNextReg	MIDI control change 0–127
ctrlPrevReg	MIDI control change 0–127
ctrlModSharp	MIDI control change 0–127
ctrlModFlat	MIDI control change 0–127
ctrlCrescPed	MIDI control change 0–127

BCouplerTolerance	time in milliseconds
autoadvanceDelay	time in milliseconds
regsyncDelay	time in milliseconds (max = 500 ms)
keyNextReg	key code
keyPrevReg	key code
keyFirstReg	key code
keyModSharp	key code
keyModFlat	key code

If the name of a MIDI device contains spaces, it must be enclosed inside quotes.
To find out key codes, use the *Command keys editor* utility.

Load VMK-88 settings

Loads settings for the VMK-88 MIDI keyboard's buttons, knobs and sliders from a file, or recalls the default VMK-88 settings, or clears the settings.

Syntax:
loadVMK88settings, [filename];
loadVMK88settings, default;
loadVMK88settings, clear;

Filenames that contain spaces must be inside quotes.
Use default instead of a filename to recall the default VMK-88 settings.
Use clear instead of a filename to clear the settings (i.e. to make the buttons/knobs/sliders do nothing).

Load Fokkerboard settings

Loads settings for the 31-tone practice keyboard's unused keys and buttons on the back from a file, or recalls the default Fokkerboard settings, or clears the settings.

Syntax:
loadFBsettings, [filename];
loadFBsettings, default;
loadFBsettings, clear;

Filenames that contain spaces must be inside quotes.
Use default instead of a filename to recall the default Fokkerboard settings.
Use clear instead of a filename to clear the settings (i.e. to make the unused keys and buttons do nothing).

Load crescendo list

Loads a crescendo list from a file, or activates one of the preset crescendo lists.

Syntax:
loadCrescList, [filename];
loadCrescList, preset1;
loadCrescList, preset2;

Filenames that contain spaces must be inside quotes.
preset1 is the preset crescendo list for manuals and pedal (the default crescendo list).

preset2 is the preset crescendo list for manuals only.

Load custom console

Loads a custom console script file and activates the custom registers specified therein.

Syntax:
loadCustomConsole, [filename];

Filenames that contain spaces must be inside quotes.

Load Max patch

This command can be used to automatically load other Max patches (including your own ones) that you want to use together with *Ere's Fokker organ console*.

Syntax:
loadMaxpat, [filename(s)];

You can specify one or more Max patches to be loaded at the same time as the registration list. One loadMaxpat command may contain multiple filenames – type the filenames after each other, separated by spaces. You may also include multiple loadMaxpat commands in the registration list. All files specified on all loadMaxpat command lines will be loaded.
If a filename contains spaces, it must be enclosed in quotes.

It is also possible to automatically activate and deactivate loaded Max patches depending on which registration list is currently active, by using the following method.
– Include the following chain of objects in your Max patch:
receive RegListName – zl compare [name of registration list without quotes] – toggle
– The toggle object will receive "1" after loading the specified registration list, and "0" when loading any other registration list.
– The toggle object can thus be used to activate and deactivate the patch, depending on whether the specified registration list is the current registration list or not.

Control changes

If you have programmable MIDI pedals or other suitable MIDI controllers, you can use them to send commands to *Ere's Fokker organ console* through MIDI *control change* messages.

The following functions can be controlled in this manner:
– All notes off default control change # = 100
– Next registration default control change # = 99
– Previous registration default control change # = 98
– Modulate sharpwards default control change # = 97
– Modulate flatwards default control change # = 96
– Crescendo pedal position default control change # = 95

To use these controls, assign your pedals / other controllers to send the appropriate MIDI control change messages. The control change numbers given above are the default values, and if necessary, they can be modified in the "Advanced settings" window.

Possible input devices to receive control changes from are: 12-tone manuals I and II, Control pedals and External MIDI source. The control changes can be sent thru any channel.

– To send an All Notes Off message, send the appropriate control change # (100 by default) with the value 127.

– Next registration, Previous registration, Modulate sharpwards, Modulate flatwards: These commands are activated when the control change # is received with the value 127. After this, the value 0 has to be received before the command can be activated again (this is to prevent any jitter from control pedals). Normally, these values correspond to the down and up positions of a standard pedal switch.

– For Crescendo pedal position, send the control change # (95 by default) with values ranging from 0 to 127. This function is primarily intended for use with a standard MIDI volume pedal, and the values 0–127 from the pedal will automatically be translated into crescendo pedal positions.

Specific instructions for programming Next registration, Previous registration, Modulate sharpwards or Modulate flatwards commands to MIDI pedals that are connected to the VMK-88 MIDI keyboard:

1. Press the EDIT button on the VMK-88 keyboard
2. Press the pedal you want to program
3. Use the Page up/down buttons to select which value to edit, and the Data Entry wheel to change the value on the display. Enter the following values:
– MIDI Channel: any channel #
– CTRL Change: 99, 98, 97 or 96
– Value Min: 0
– Value Max: 127
– Polarity: UP>DN
4. When finished editing all values, press STORAGE – ENTER – STORAGE – ENTER
5. To program another pedal, repeat the procedure.

It would be possible to program the buttons on the VMK-88 keyboard in this manner as well, but it is easier and preferable to use the *VMK-88 controls editor* utility for this purpose.

Registration shortcuts

Registration event numbers, as well as "Next registration" and "Previous registration" commands, can be sent to *Ere's Fokker organ console* simply as MIDI notes. These are called *registration shortcuts*.

To send a registration shortcut message, send a MIDI note (noteon) message as follows:
– Pitch = registration shortcut #
– Velocity = any non-zero value
– Channel = the default channel is 10, but this can be changed in the "Advanced settings" window

Registration shortcut # (MIDI pitch value) can be:
– 1 ... 126 = go to registration #
– 127 = Next registration
– 0 = Previous registration

Possible input devices to receive registration shortcuts from are: 12-tone manuals I and II, and External MIDI source.

Registration shortcuts are the easiest method for an external MIDI source, such as a MIDI sequencer, to communicate with a registration list in *Ere's Fokker organ console*.
If you're using MIDI sequencing software, simply create an extra track that plays MIDI notes (with or without noteoff messages) on channel 10 at the appropriate moments. Each MIDI note played on this channel will trigger the corresponding registration event in the registration list.
If you're using notation software, you would just create one extra staff in your score, and assign this staff to play on channel 10. On this staff you would place notes whose MIDI note numbers correspond to the desired registration event numbers. Timing-wise, only the beginning of the note will be significant. The length of the note is irrelevant, since noteoff messages will be ignored as registration shortcuts.

Concerning the timing of registration events, please read also the discussion about the option "Synchronize note and registration messages from external MIDI source" (in chapter "Options").

The obvious drawback of the registration shortcuts is that you can directly access only registration event numbers up to 126. You can, however, use the "Next registration" shortcut (note # 127) to advance past event number 126, if needed.

Even if you prefer to step through your registration list using only "Next registration" shortcuts, do include a MIDI note with pitch value 1 at the beginning of the piece, in order to always start at registration event # 1.

Registration shortcuts could also be created on the buttons of the VMK-88 MIDI keyboard by programming the buttons to send MIDI notes, but it is easier and preferable to use the *VMK-88 controls editor* utility for this purpose.

For Max programmers

It is possible to use *Ere's Fokker organ console*, and the many functions it provides, in combination with Max patches of your own for controlling the Fokker organ. One of the benefits of doing this is that you won't need to worry about the details of the Fokker organ's MIDI protocol; *Ere's* will take care of sending the correctly formatted MIDI messages to the Fokker organ. You could also make full use of the power of the registration lists, and the automated features they offer.

In the folder "Patches", there is a Max patch named "**Max patching with Ere's.maxpat**". Inside this patch, you will find more information and examples on how to make your own Max patches work together with *Ere's Fokker organ console*.

For those Max programmers who prefer to create their own self-sufficient Max patches, without using *Ere's Fokker organ console*, there is another Max patch in the "Patches" folder, named "**MIDI to Fokker organ examples.maxpat**". Inside this patch, you will find useful examples and subroutines for formatting and sending MIDI messages to the Fokker organ. Feel free to use these and develop them further in your own patches.

Known issues

– If you edited a registration list in the editing window, and closed the editing window without clicking on any button in the registration list editor window, the edits will not get updated in the program's memory unless the registration list editor window becomes the top window again. In a situation where this didn't happen, give a "Restore registration" command (press the Enter key) to update the program's memory.

– If you're using Max 5 or 6, you must open *Ere's* by clicking on it, or by selecting "Open With Max" from the context menu. If you first start Max and then use the menu command File > Open to start *Ere's*, the file search paths may not be set up properly, due to a bug in Max.

– Key commands involving number keys with Shift (such as for the virtual couplers) may not work correctly on some operating systems. Using caps lock (instead of Shift) does work; and on Mac only, the ctrl key.

– The default preset crescendo list (with pedals) might not work in *Ere's* 3.10.2. This will be fixed in a future update.

Version history

3.1.0
The first version to be publicly downloadable.

3.1.1
Added separate input indicators for each of the MIDI inputs on the Setup window.

3.2.0
Method 1 translator utility added.
Bass couplers are now functional.
The MIDI monitor on the Setup window now reacts only to MIDI input from the selected input sources.
Several new functions and improvements regarding the use of the 31-tone practice Fokker keyboard.
Other smaller improvements and bug fixes.

3.2.1
Fixed problems with virtual couplers.
Fixed problem with regsync command.
Fixed problem with shortcut keys for Euler-Fokker genera.

3.2.2
Some bug fixes.

3.3.0
Fixed problem that could cause hanging notes at registration changes when Split manual I was used in combination with virtual couplers.
Sync hold commands at noteoffs no longer produce audible artefacts when using the Fokker Emulator.
Opening multiple instances of *Ere's Fokker organ console* simultaneously is now prevented.
Other smaller improvements and bug fixes.

3.4.0
Created drop-down menus for easy access to the different functions of *Ere's Fokker organ console*.
The "Instructions for use" button now opens the "instructions.pdf" file.
Other smaller improvements.

3.5.0
Folders "Keymaps", "Registrations" and "Settings" (if present in the same folder as "Ere's"), and all their sub-folders, are now automatically included in Max's search path.
The files and folders in the downloadable .zip file have been rearranged accordingly.
Any number of custom keymaps required in a registration list can now be loaded in memory (the former maximum of 100 does not apply any more).
New alternative "All registers off" command (only as key command shift-0), which does not change the displayed registration event number.
New keyboard shortcut (Backspace key) for Cancel sync hold.
Registers can now be controlled also via numeric keypad.

3.7.1
Added virtual couplers for 12-tone pedal.
In 12-tone keymaps, "+8" and "-8" may now be used instead of "+oct(ave)" and "-oct(ave)".
12-tone manual II and pedal can now have their own independent 12-tone keymaps (key# domains 201–212 and 401–412).
Option "Ignore autoadvance commands in registration list" has been expanded to ignore sync hold and delay commands as well.
New alternative "Next registration" command (only as shortcut key shift-→), which ignores any sync hold and delay.

Improvements to the user interface.
Other smaller improvements and bug fixes.

3.8.0
New Custom console feature, to control a virtual organ with a different (user-specified) disposition.
Added a second MIDI output, for use with a (custom) virtual organ.
Folder "Patches", if present in the same folder as "Ere's", and all its sub-folders, are now also included in Max's search path.

3.8.1
Fixed an issue whereby some delay, split and sync hold commands did not work in Max 6.

3.8.6
Option "Ignore duplicated note messages from external MIDI source" implemented.
New "Force all notes off" command that sends noteoff messages for all possible notes even if they had not been played through "Ere's".
Fixed a bug whereby the MIDI recorder utility recorded unnecessary custom console registers when no custom console was loaded.
The default settings for VMK-88 keyboard have been altered to not use Knob 1, because that knob is currently malfunctioning and not usable on the Huygens-Fokker Foundation's VMK-88 keyboard.

3.9.0
Added a MIDI in indicator on the Console window.
Autoadvance command deprecated. From now on, if a sync hold or delay command is present at the next registration event, "Ere's" will automatically activate the sync hold / delay, without need to place an "auto" command. For the rare occasions where you'd wish this not to happen, there is a new "halt" command to prevent automatic advancing.
At startup, "Ere's" will now look in the text file "Ere'sStartupSettingsFilename", and if there is a filename as the first line of text, automatically load a settings file with that name.

3.9.3
When loading a registration list, if there is no transposition command at the beginning of the registration list, transposition is set to 0 by default.
Fixed an issue whereby Max would crash if there was MIDI input during startup of "Ere's".
Created a new, user-modifiable MIDI player patch for playing back MIDI files at specified registration events. It can be found, with instructions for use, in the "Patches" subfolder.

3.9.6
New Note length feature, including the Note length control window, and note length commands for use in a registration list.
Added a new registration list initial command: loadCustomConsole.
New function "Custom register #" added to VMK-88 controls editor (on buttons) and to Fokkerboard controls editor.
It is now possible to have several loadMaxpat initial commands in a registration list.

3.10.0
The file "Max patching with Ere's.maxpat", in the Patches folder, provides information on how to make your own Max patches work together with Ere's Fokker organ console.
Key commands 0 and shift-0 have been swapped: 0 = All registers off without changing registration event #; shift-0 = All registers off, reset registration event display to 0. Other All registers off commands now work similarly to key command 0.
If the registration has been changed manually, and you move forwards or backwards in the registration list using Next registration and Previous registration commands, the registration will only change when an explicit new registration is encountered in the registration list. This allows manual registration changes to be kept intact within sections where there are no registration changes in the registration list.

Fixed an issue that prevented Ere's home folder and its subfolders from being automatically included in Max's search path.
Fixed an issue with Note length control, whereby notes would hang when pressing the "Reset all" button.
On Mac, key commands involving number keys with Shift (which may not work correctly) can now be typed using the ctrl key (instead of Shift or the alt/option key). Using caps lock works, too.
Corrected the display color of three number boxes in Max 7 (on Console window and Meantone modulator window).
Fixed an issue in Max 7 whereby the text boxes for the names of current registration list, registration event name and keymap name might display too wide.
Fixed an issue in Max 7 that caused register buttons on a custom console to not change color when switched on.
Improved the positioning and appearance of some user interface objects in Max 7.
Ere's is now compatible also with Max 8.

3.10.1.
Improvements on the visual appearance of the user interface in Max 7 and later.
The option "Ignore duplicated note messages from external MIDI source" is now on by default.
Giving a Restore current registration command from a button on the VMK-88 keyboard, while Overdrive is on, no longer causes an incorrect "Keymaps not found" message to be shown.
Fixed an issue in Max 7 that caused unnecessary virtual coupler numbers to be stored in registration list.
The file "MIDI to Fokker organ examples.maxpat" (in the Patches folder) provides useful information and example patches for Max programmers who want to create their own Max patches for controlling the Fokker organ without using *Ere's Fokker organ console*. The file "Max patching with Ere's.maxpat" has also been revised.

3.10.2.
Restored functionality of regsync command and virtual couplers in registration lists (broken since version 3.10.0).
Fixed an issue preventing a custom keymap from activating upon loading a registration list (broken since version 3.10.1).

Instructions version 22.12.2023