GrandOrgue Help

COLLABORATORS

	TITLE : GrandOrgue Help		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 20, 2016	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

List of Figures

Chapter 1

Welcome

Welcome to the manual for GrandOrgue.

- Click the Contents tab to display a list of important topics about the program.
- Click the Index tab to display an alphabetical listing of topics.
- Click the Search tab to search for a specific word in the help system.

Chapter 2

Getting Started

2.1 Introduction

GrandOrgue allows the user to load and play virtual pipe organs. Using sample sets, GrandOrgue is able to combine the individual pipe sounds and organ behaviors to simulate a variety of pipe organs.

2.2 Sample Sets

GrandOrgue is capable of loading and faithfully reproducing a variety of sample sets. The sample set format is described later in this manual. The format is backward compatible to Hauptwerk™ version 1 samplesets. As such, you won't be able to do anything at all with GrandOrgue without at least one sample set. Sample sets generally come packaged or with installers. By default, the software loads organs from "My Organs" in your documents folder, but you may store sample sets wherever is convenient and space permits. Please look at https://sourceforge.net/p/ourorgan/samplesets/ for more information.

Note
Some Hauptwerk™ Organ Definition File keywords are not used by GrandOrgue and are reported in a pop-up window. These warning messages can be safely ignored

2.3 Features

- High definition output with supporting hardware
- High precision internal processing for optimal sound quality
- Integrated software reverberation
- Polyphony management to avoid CPU overload
- Low latency output with proper hardware support
- Quick and easy MIDI setup, including "Listen for Event" and "Complex MIDI Event detection"
- Lossless compression reduces RAM requirements by up to 40%
- Release sample enhancements for realistic sound
- All features and enhancements configurable
- High speed loading and caching of sample sets

- Save settings per sample set or export to settings files
- Shipped with a small demo organ which is automatically loaded on the very first run of GrandOrgue after installation

2.4 Requirements

2.4.1 All Platforms

- SSE3 capable processor

  NOTE: A newer, faster processor will permit a higher polyphony
- Sufficient RAM to fully load the largest sample set you wish to use. A 64 bit OS is recommended to support larger sample set sizes.[1]
- A hard disk (or other storage) with enough free space to store sample sets. The use of the cache increases the necessary disk space.

[1] A 32 bit GrandOrgue is limited to less than 3 GB of loaded sample data and depending on your operation system, loading more than 1.5 to 2 GB will inhibt some features.

Chapter 3

User Interface

The user interface has a menu bar, a tool bar and a console section where the organ console is displayed.

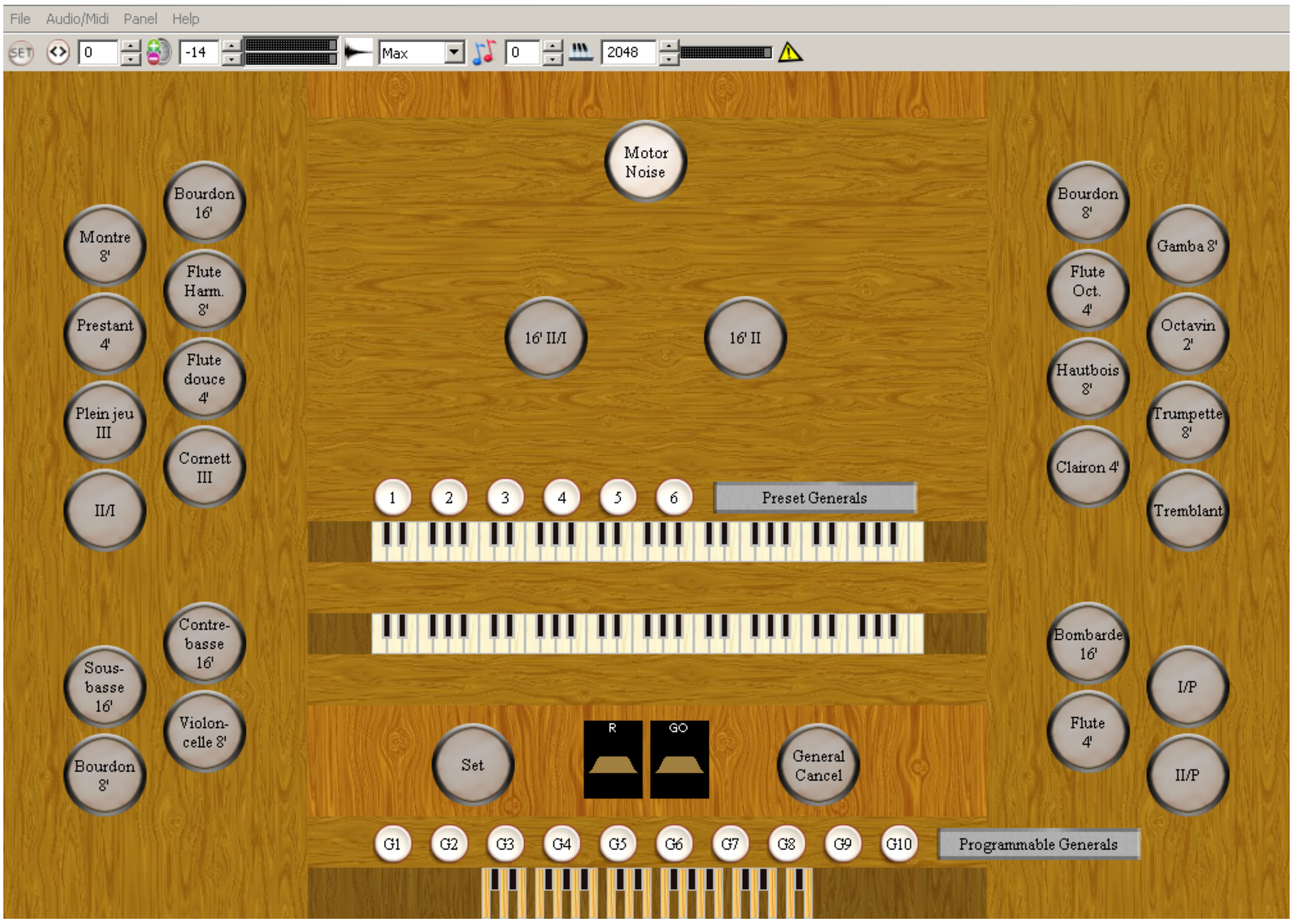


Figure 3.1: GrandOrgue User Interface

3.1 Menu bar

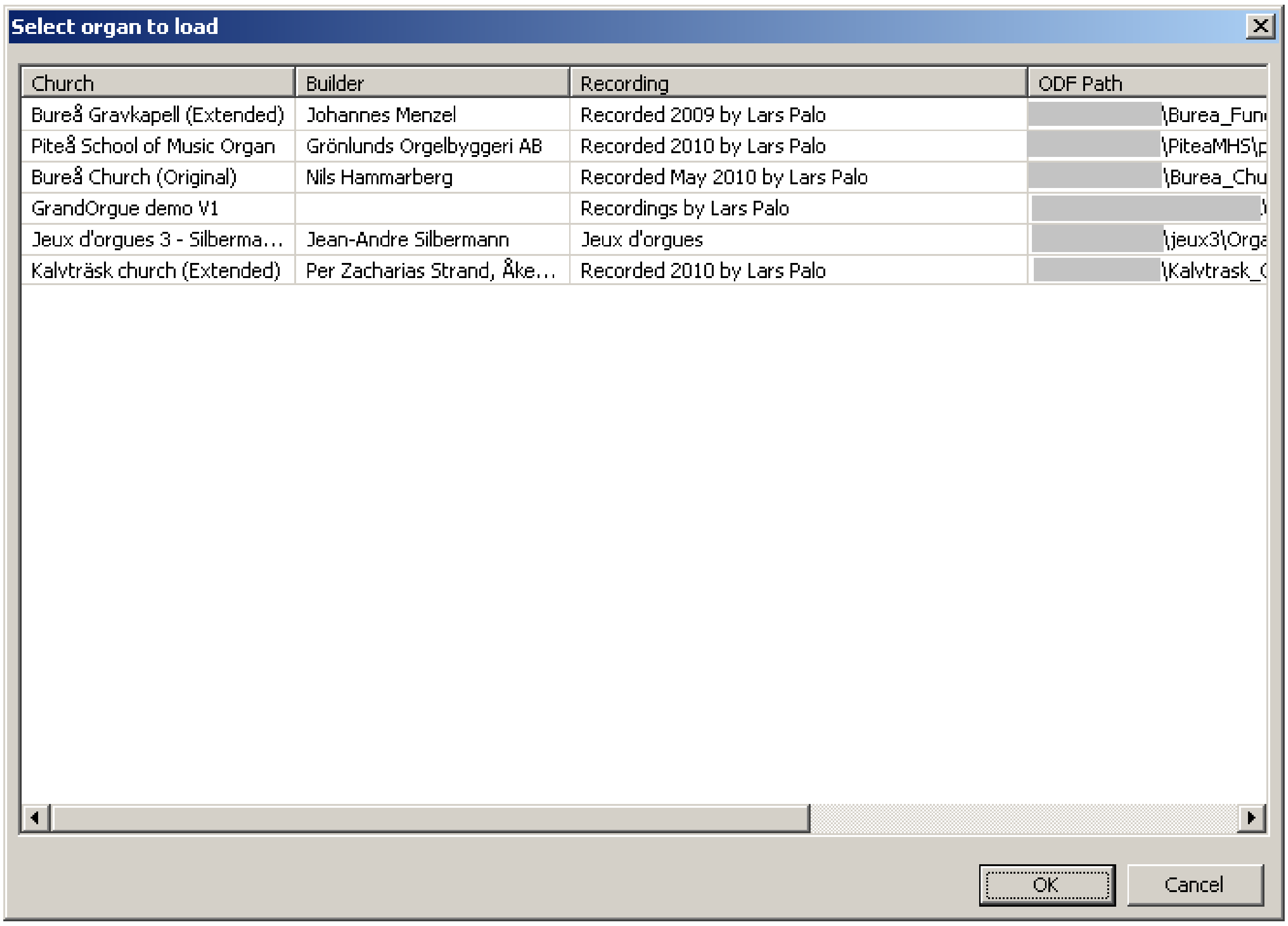
The menu bar has 4 menus:

- File: file operations, exit
- Audio/Midi: manage settings
- Panel: manage additional panels
- Help: display on-line help

3.1.1 File Menu

The File menu contains commands for dealing with the loading and saving of sample sets.

3.1.1.1 Load



Church	Builder	Recording	ODF Path
Bureå Gravkapell (Extended)	Johannes Menzel	Recorded 2009 by Lars Palo	[redacted]\Burea_Fun
Piteå School of Music Organ	Grönlunds Orgelbyggeri AB	Recorded 2010 by Lars Palo	[redacted]\PiteaMHS\p
Bureå Church (Original)	Nils Hammarberg	Recorded May 2010 by Lars Palo	[redacted]\Burea_Chu
GrandOrgue demo V1		Recordings by Lars Palo	[redacted]\
Jeux d'orgues 3 - Silberma...	Jean-Andre Silbermann	Jeux d'orgues	[redacted]\jeux3\Orga
Kalvträsk church (Extended)	Per Zacharias Strand, Åke...	Recorded 2010 by Lars Palo	[redacted]\Kalvtrask_(

Figure 3.2: Known organs list

Open and load a sampleset from the known organs list which is updated whenever a new sample set is successfully loaded.

Select an organ in the list, then click **OK**.

Note
The known organs list is managed in the Organs Settings tab.

3.1.1.2 Favorites

This submenu displays the top ten organs in the known organs list. Organs display order is managed in the Organs Settings tab.

Choosing an organ in this menu opens and loads its sampleset.

3.1.1.3 Open

Open and load a sample set stored in a definition file. The location will be remembered for the next open operation. The most recently used sample sets are also remembered and available in the "Load", "Open Recent" and "Favorites" menu items. Loading sample sets takes some time; a progress window will display indicating estimated time remaining and other indicators.

When the sampleset loads successfully, it registers in the configuration file, thus automatically feeding the known organs list.

3.1.1.4 Open Recent

This submenu displays the ten last used organs from the known organs list. Organs in this menu are ordered by last used date from newest down to oldest.

Choosing an organ in this menu opens and loads its sampleset.

3.1.1.5 Organ Properties

Open a window which displays various characteristics of the loaded sample set as provided by the sample set producer. If available, there is also a link to "More Information" which will display additional information in another application.

3.1.1.6 Preset

Select the saved settings number. A single sample set can create and use up to 10 different saved settings.

Selecting the preset number triggers a reload of the sample set, using the presets stored at the selected number. If no saved settings exist for that number, the sample set is reloaded from the organ definition file as provided by the sample set creator.

A new saved settings set is created whenever the Save menu item is selected. The file is stored in the directory currently designated by the Settings Store parameter in the *Options* tab, and the file name is meaningless (as in: not related to the sample set/organ/church name) for the user.

3.1.1.7 Save

Save to the preset number currently in use; create a new file if none in use. The settings consist of the stop and piston MIDI messages; stop engaged states; divisional, general, and program combinations; volume spinner value; temperament setting; fine tuning data. Note that this is completely reversible using the Reset to Defaults option, and is the recommended method for most users.

3.1.1.8 Update Cache

Save the loaded and processed samples to hard disk. Doing so decreases loading time at the expense of hard disk space.

The cache file is stored in the directory currently designated by the Cache Store parameter in the *Options* tab, and the file name is meaningless (as in: not related to the sample set/organ/church name) for the user.

Creating or updating the cache takes some time; a progress window will display indicating estimated time remaining and other indicators.

Note that each different preset uses its own cache. The Manage cache system option manages automatic creation/update of the current cache.

3.1.1.9 Delete Cache

Delete the current cache from hard disk.

3.1.1.10 Reload

Reload the currently loaded sample set from disk. This may be used to quickly restore a sample set to its saved status. The currently selected preset file is reapplied if it exists on disk.

3.1.1.11 Reset to Defaults

Reset the currently loaded sample set to the settings as originally supplied by the sample set provider. You will be asked to verify that you wish to discard your modifications. Note that the file containing saved settings currently in use is *deleted*.

3.1.1.12 Import Settings

Apply a previously exported settings file to the currently loaded sample set. All current settings and combinations are overwritten.

Note that settings may only be applied to a matching sample set, and you will receive a warning if you try to do otherwise. The location where the settings were imported from will be remembered for the next time.

A settings file matches a sample set when the *ChurchName* property of the settings file matches the *ChurchName* property of the definition file.

3.1.1.13 Import Combinations

Apply only the combination settings from a previously exported settings file to the currently loaded sample set. All other settings that might have previously been changed are preserved.

Note that settings may only be applied to a matching sample set, and you will receive a warning if you try to do otherwise. The location where the combinations were imported from will be remembered for the next time.

A settings file matches a sample set when the *ChurchName* property of the settings file matches the *ChurchName* property of the definition file.

3.1.1.14 Export Settings/Combinations

Save a settings file including the stop and piston MIDI messages; stop engaged states; divisional, general, and program combinations; volume spinner value; temperament setting; fine tuning data.

Export Settings/Combinations allows the user to choose a specific file name and folder. The latest location where settings were exported to will be remembered for the next time.

3.1.1.15 Close

Close the currently loaded sample set. If you have made any settings changes, you will be asked if you would like to save them before the sample set closes. Note that this will also unload the sample set and free the RAM it required.

3.1.2 Audio/Midi Menu

The Audio/Midi menu contains commands for dealing with loaded organs and associated settings.

3.1.2.1 Temperament

Opens a bunch of menus and sub menus allowing to select a new tuning temperament.

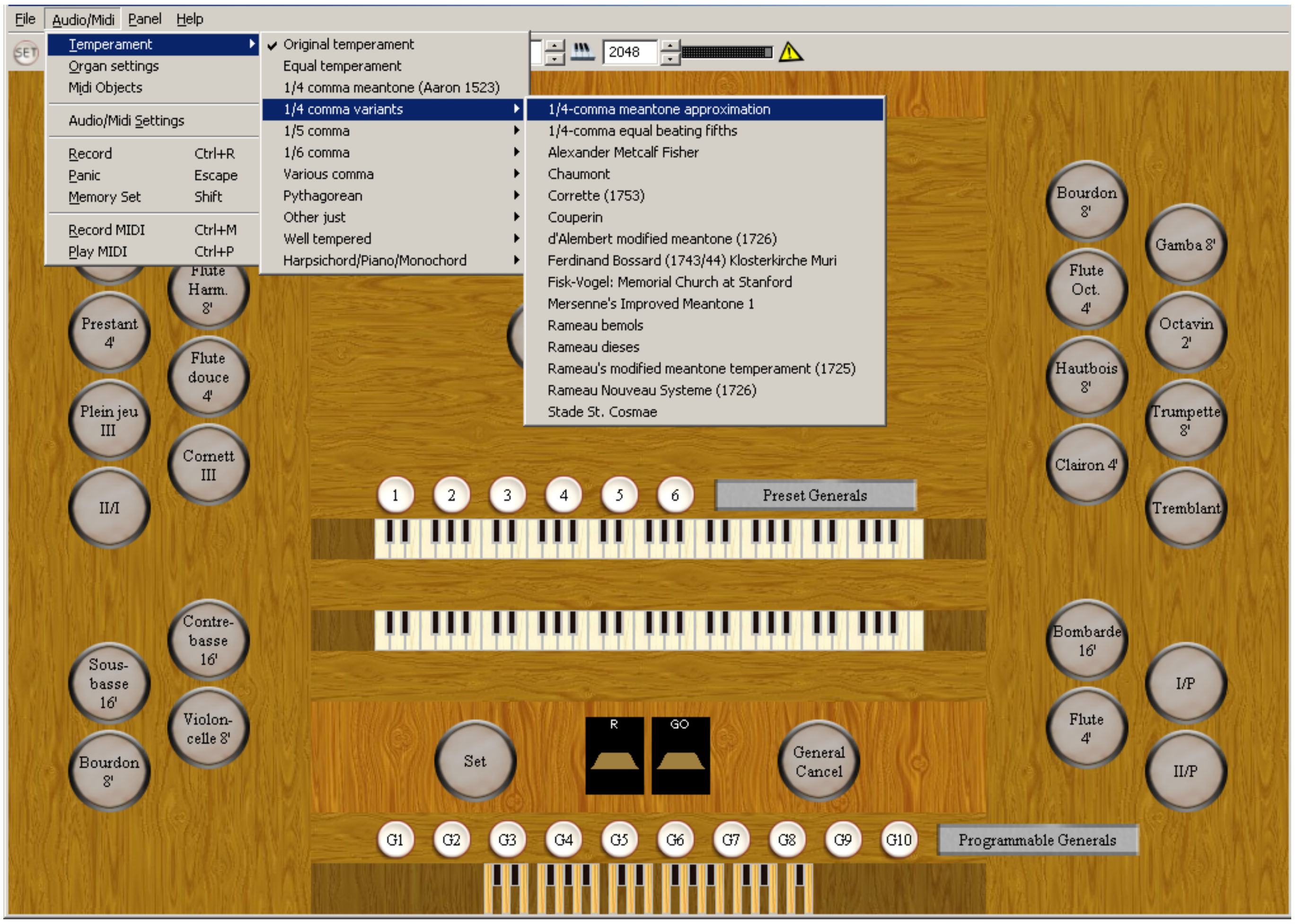
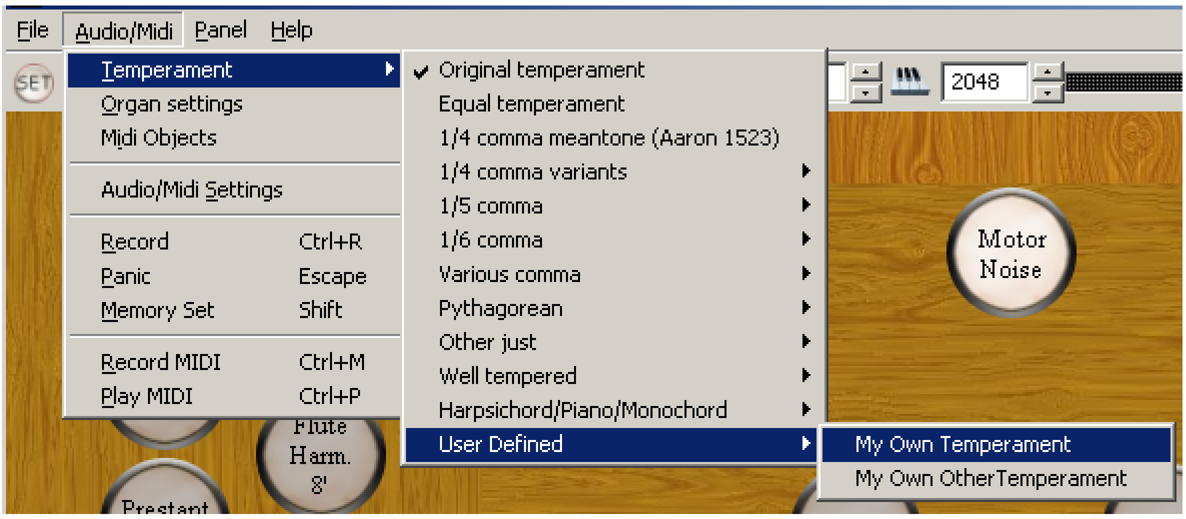


Figure 3.3: Temperaments

The first menu item is always *Original temperament*, which allows the user to return to the tuning provided by the sample set creator.

The samples are retuned on the fly when playing. No additional disk storage is required.

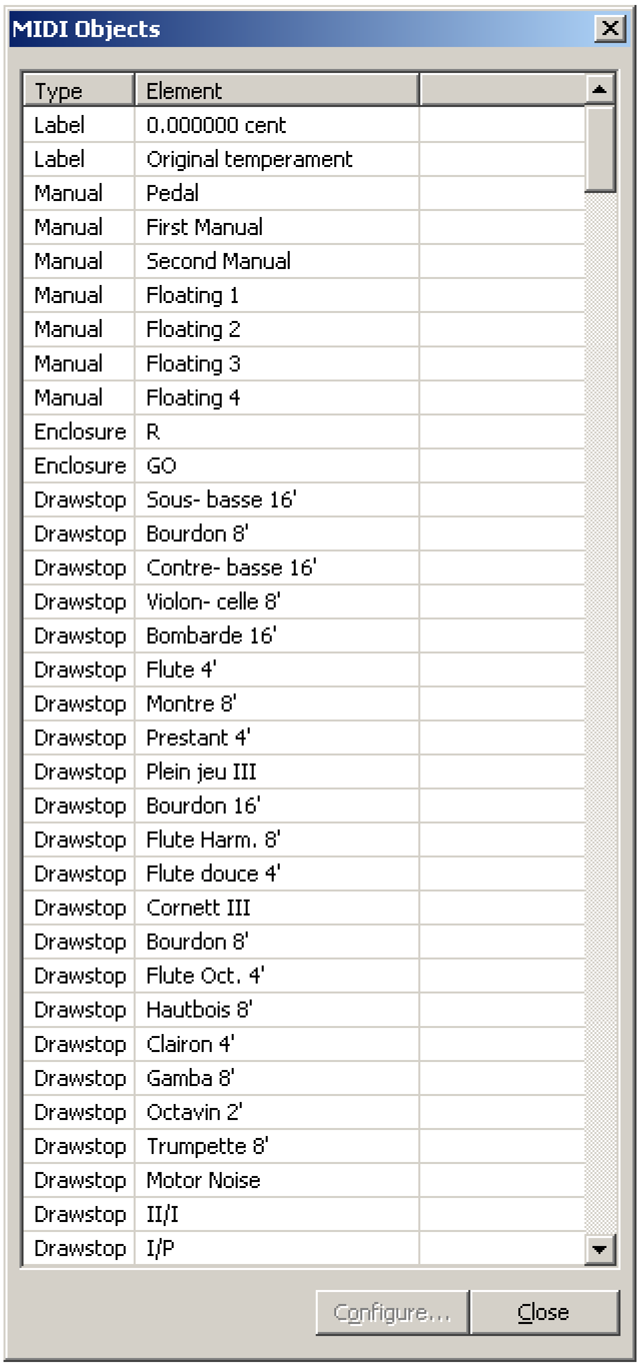
User temperaments that are added in the Temperaments tab are dynamically displayed at the end of this menu



3.1.2.2 Organ Settings

Displays a dialog which allows to set audio settings (amplitude, gain, tuning, etc.) at every level of the sample set from the whole organ level down to the individual pipe. See Organ Settings for detailed information on this topic.

3.1.2.3 Midi Objects



Type	Element	
Label	0.000000 cent	
Label	Original temperament	
Manual	Pedal	
Manual	First Manual	
Manual	Second Manual	
Manual	Floating 1	
Manual	Floating 2	
Manual	Floating 3	
Manual	Floating 4	
Enclosure	R	
Enclosure	GO	
Drawstop	Sous- basse 16'	
Drawstop	Bourdon 8'	
Drawstop	Contre- basse 16'	
Drawstop	Violon- celle 8'	
Drawstop	Bombarde 16'	
Drawstop	Flute 4'	
Drawstop	Montre 8'	
Drawstop	Prestant 4'	
Drawstop	Plein jeu III	
Drawstop	Bourdon 16'	
Drawstop	Flute Harm. 8'	
Drawstop	Flute douce 4'	
Drawstop	Cornett III	
Drawstop	Bourdon 8'	
Drawstop	Flute Oct. 4'	
Drawstop	Hautbois 8'	
Drawstop	Clairon 4'	
Drawstop	Gamba 8'	
Drawstop	Octavin 2'	
Drawstop	Trumpette 8'	
Drawstop	Motor Noise	
Drawstop	II/I	
Drawstop	I/P	

Figure 3.4: Midi Objects

Opens a window displaying all GrandOrgue objects which can have a MIDI event assigned to them.

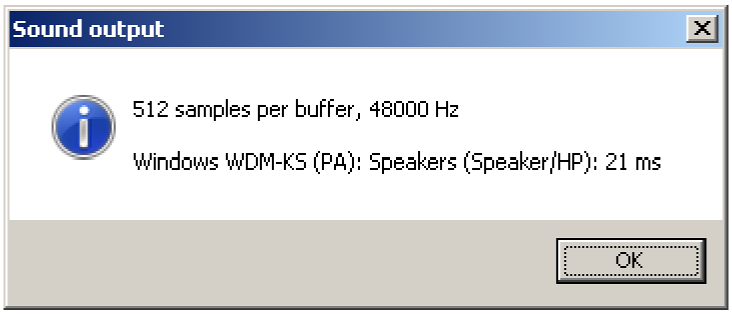
The **Configure** button opens the midi event editor dialog for this element.

Double-clicking on the element in the list opens the midi event editor as well.

3.1.2.4 Audio/Midi Settings

Displays a tabbed dialog containing the various settings. See Audio & Midi Settings for detailed information on this topic.

3.1.2.5 Sound Output State


Sound output

512 samples per buffer, 48000 Hz

Windows WDM-KS (PA): Speakers (Speaker/HP): 21 ms

This popup tries to display *actual* latency. Note that the estimate can be based on numbers provided by the audio hardware and drivers and that it may vary. The RtAudio backend is more likely to display too low numbers.

3.1.2.6 Record

Opens a dialog prompting where to save the recorded WAV file. The location chosen in this dialog will be remembered for the next time. Upon clicking to select a file name, recording commences and continues until the Record menu option is selected again. A check mark appears next to the menu item to indicate when recording is active.

The **Record** function can also be invoked by the computer keyboard shortcut *Ctrl-R*.

3.1.2.7 Panic

The effect of the panic menu item is to reset the sound, stopping all sound and turning off all notes. It may also be quickly accessed on the toolbar or via the *Escape* key on the computer keyboard. This should be generally unnecessary, although it may be useful if you need to quickly stop the sound output, or if the sound is breaking up due to CPU overload.

3.1.2.8 Memory Set

Enters memory set mode until the **Memory Set** menu option or toolbar button is selected again. While the memory set is engaged, whenever you push a divisional or general piston or activate a program change, the organ will store the current settings to that piston or program change rather than recalling them. For example, engaging a divisional piston 1 during memory set will store the state of stops for that manual into piston 1, which could then be pushed later (after memory set has been disengaged) to recall those stops. A checkmark appears next to the menu item, and the toolbar button remains pushed, to indicate when memory set is active.

3.1.2.9 Record Midi

Opens a dialog prompting where to save the recorded MIDI file. The location chosen in this dialog will be remembered for the next time. Upon clicking to select a file name, recording commences and continues until the *Record Midi*menu option is selected again. A check mark appears next to the menu item to indicate when recording is active.

Caution
GrandOrgue stores Midi data using a specific internal format that makes the file unsuitable for usage with a regular Midi sequencer.
Playing a recorded Midi file in another sampleset than the sampleset used for recording will yield unexpected results.

3.1.2.10 Play Midi

Opens a file chooser to select a recorded MIDI file. The location chosen in this dialog will be remembered for the next time. The player starts as soon as a file is selected. See above for caveats.

3.1.3 Panel Menu

GrandOrgue provides additional panels. Some are predefined and contain fixed functions (crescendo, couplers, generals and divisionals). The organ definition can also ship organ specific panels.

3.1.4 Help Menu

3.1.4.1 Index

Opens this help document. Pressing F1 at other locations in the program will jump directly to the appropriate topic.

3.1.4.2 About

Displays the splash screen.

3.2 Tool bar

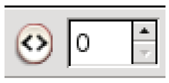
The tool bar provides a number of controls which can be accessed easily while playing the organ.

3.2.1 Memory Set

This "button" is a shortcut to Memory Set in the *Audio/Midi menu*.

The *Shift* key on the computer keyboard is a fugitive shortcut to this button: when the user clicks on a piston while holding the *Shift* key, the currently pulled drawstops are stored in the piston memory.

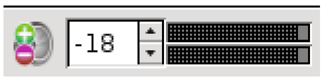
3.2.2 Program Changes



The spinner sets the current program number. This might be termed a "sequencer" on modern computerized organs. It is possible to store up to 1000 organ states and access them sequentially via the predefined MIDI events, or access any setting directly through the spinner. You may also use the left and right keys on the computer keyboard to scroll through the programs, and down to reactivate the current program (if you have changed the stops since last activating it, for example). Note also that when memory set is engaged, the target program will be stored to, i.e. if you push Shift + Right then the program would be advanced and the current settings would be saved to this same next program.

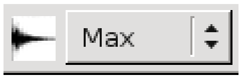
More options are available on the *Combination Setter* panel.

3.2.3 Volume Control



The volume control sets the volume control *in dB*. Left and right audio monitors are displayed next to the spinner. The clip indicator at the far right will turn red if clipping occurs, and will stay red until the volume is adjusted.

3.2.4 Release Tail Length

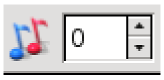


The release tail length control sets the duration the sound engine uses to play the release section of the sample. The values are: Max, 2.8 s, 1.4 s, 700 ms, 350 ms, 175 ms.

The *Max* value makes the sound engine play the complete release section as provided by the sample set creator.

The other values are used by the sound engine as the fade out duration of the sample release section.

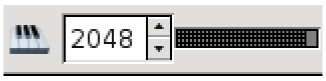
3.2.5 Transposer



The transposer control shifts the notes up or down by semitone steps. The allowed range is {-11, +11} semitones.

Note that the samples are *not* retuned. This transposer works like the mechanical transposer on genuine pipe organs or reed organs: the *keyboard* is shifted up or down, and if the targeted note is missing (i.e. no sample) there is no sound.

3.2.6 Polyphony Control



The polyphony control sets the maximum polyphony before GrandOrgue will refuse to play any additional samples. To over-simplify, polyphony is the number of pipes that may sound simultaneously; but polyphony may also be consumed when a key is released. It is recommended to experiment with this setting initially, but leave it once you find an acceptable setting. Your setting should balance between too little polyphony (where you might not be able to play many notes on large sample sets) and too much polyphony (where the CPU will overload and artifacts will appear in the sound). Note that "wet" sample sets (those with considerable reverberation) will consume polyphony many seconds after the note is actually released. The default setting of 2048 should provide a reasonable starting point on a 1GHz CPU. The polyphony monitor is displayed next to the spinner. The clip indicator at the far right will turn red if the polyphony limit has been reached, and will stay red until the polyphony is adjusted.

3.2.7 Panic Button

The effect of the panic button is to reset the sound, stopping all sound and turning off all notes. It may also be quickly accessed on the toolbar or via the *Escape* key. This should be generally unnecessary, although it may be useful if you need to quickly stop the sound output, or if the sound is breaking up due to CPU overload.

3.3 Console Section

The console section displays the organ console and all its bells and whistles as provided by the sample set creator.

Note: the figure above shows the "classical" church organ disposition. It is possible to design any kind of disposition, since all display elements are freely placeable anywhere on the console panel.

3.3.1 Manuals

Each manual as well as the pedalboard is displayed in the center of the screen. The keys are outlined when messages are received, and can be engaged by clicking the mouse. Right clicking on a manual will open its corresponding MIDI Event Editor.

3.3.2 Expression Shoes

Expression shoes are displayed directly above the pedalboard, sometimes referred to as "swell pedals", although they may control any of the manuals (Great, Swell, Wonderful, etc). On a pipe organ, swell shutters dampen the sound, and this is simulated in GrandOrgue by adjusting the volume of the enclosed pipes. While the mouse is over a pedal, you may

- use the mouse wheel to adjust its position,
- left click and drag the shoe to the desired position
- right click to open its corresponding MIDI Event Editor.

The sample set creator can also provide a crescendo shoe.

3.3.3 Drawstops

The drawstops engage or disengage various components of the organ when clicked - generally new ranks of pipes which are applied to each manual. Other drawstops may perform actions such as coupling one manual to another. This deals with the functionality of pipe organs and is beyond the scope of this documentation. Right clicking on a drawstop will open its corresponding MIDI Event Editor. Any changes made there may be saved or exported into a settings file.

Note: Keep the left button pressed and sweep the mouse over the stop jamb to pull or push multiple drawstops in a single movement.

Note: The "sweep" feature is usable on all clickable display elements, and should be very useful with touch screens.

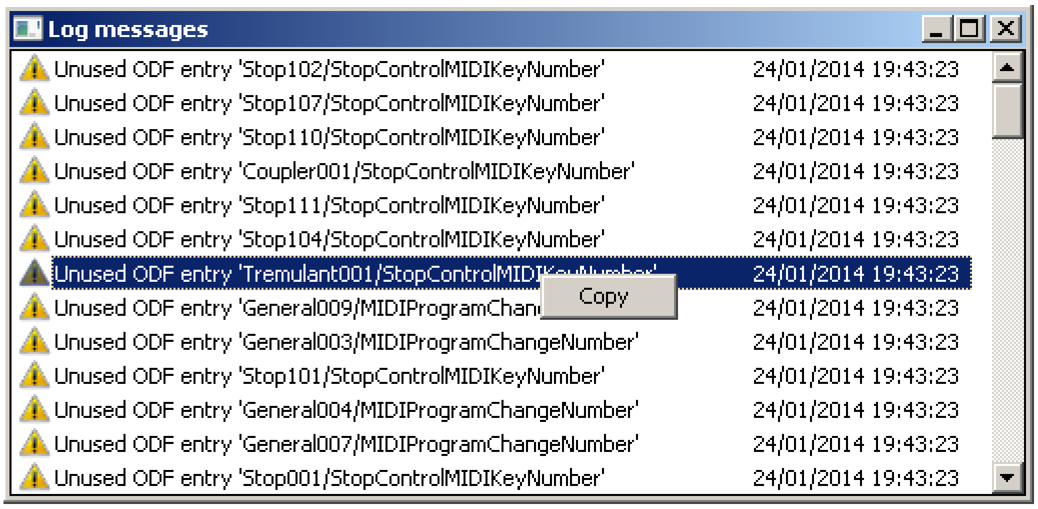
3.3.4 Pistons

Pistons generally perform an action when clicked and do not have an on/off toggle, although some pistons may toggle in this way. Pistons often recall drawstop combinations, and when used in combination with the Memory Set, you can store your own combinations in the pistons. Right clicking on a piston will open its corresponding MIDI Event Editor. Any changes made there may be saved or exported into a settings file.

3.3.5 Panels

GrandOrgue provides additional panels. These panels work as console extensions and are not displayed by default. They can be accessed via the Panels menu.

3.4 Message log



```
Unused ODF entry 'Stop102/StopControlMIDIKeyNumber'          24/01/2014 19:43:23
Unused ODF entry 'Stop107/StopControlMIDIKeyNumber'          24/01/2014 19:43:23
Unused ODF entry 'Stop110/StopControlMIDIKeyNumber'          24/01/2014 19:43:23
Unused ODF entry 'Coupler001/StopControlMIDIKeyNumber'       24/01/2014 19:43:23
Unused ODF entry 'Stop111/StopControlMIDIKeyNumber'          24/01/2014 19:43:23
Unused ODF entry 'Stop104/StopControlMIDIKeyNumber'          24/01/2014 19:43:23
Unused ODF entry 'Tremulant001/StopControlMIDI               24/01/2014 19:43:23
Unused ODF entry 'General009/MIDIProgramChan                 24/01/2014 19:43:23
Unused ODF entry 'General003/MIDIProgramChangeNumber'        24/01/2014 19:43:23
Unused ODF entry 'Stop101/StopControlMIDIKeyNumber'          24/01/2014 19:43:23
Unused ODF entry 'General004/MIDIProgramChangeNumber'        24/01/2014 19:43:23
Unused ODF entry 'General007/MIDIProgramChangeNumber'        24/01/2014 19:43:23
Unused ODF entry 'Stop001/StopControlMIDIKeyNumber'          24/01/2014 19:43:23
```

Figure 3.5: Message Log

Whenever GrandOrgue needs to display a message, it opens a pop-up window. Messages are displayed in reverse chronological order : newest on top of the window.

Messages are displayed with a rating icon.

⚠ Warning message

⊗ Serious message

Right-clicking anywhere in this window displays a small menu which copies the window's content to the clipboard. When pasted to a file, the messages are displayed in chronological order: oldest on the first line.

Chapter 4

Midi Event Editor

Every console element, including those defined on panels, is configurable with this editor. The Midi Event Editor is invoked by right-clicking on the element.

4.1 Receive

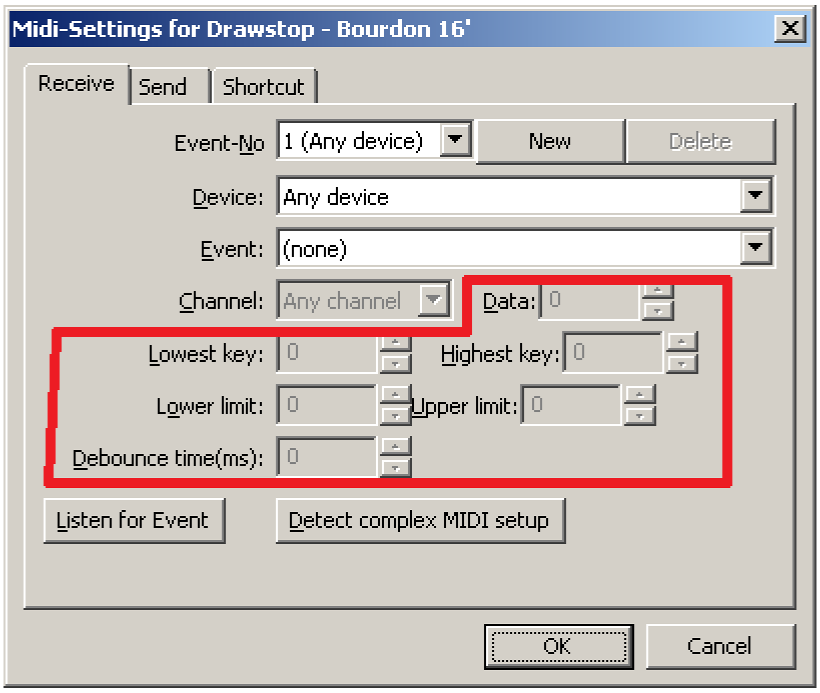


Figure 4.1: Midi received event

This tab configures which MIDI events will drive the console element.

The overall layout is the same for all elements. Depending on the element being configured, the fields circled in red will be disabled or relabeled.

It is possible to define several received events for the same GUI control. Any of them received at any time will trigger the control.

Event-No Event identifier. If the event is associated to a specific MIDI device, the device name is part of the identifier.

Device The dropdown displays every MIDI input device that was ever known to GrandOrgue. If a device is selected in the dropdown, GrandOrgue listens only for that event coming from that device. If no device is selected, GrandOrgue listens for that event coming from any selected input device (in the Midi Devices Tab) connected to the computer.

Event The dropdown displays the list of allowed events for the control.

Channel MIDI channel for this event (1-16 or Any). If set to "Any channel", GrandOrgue listens for that event on any channel.

New This button allows to add a new event to the list of received events. A blank form is displayed with an incremented event number.

Delete This button allows to delete the currently selected event. *It is enabled only when there are at least 2 events in the list.*

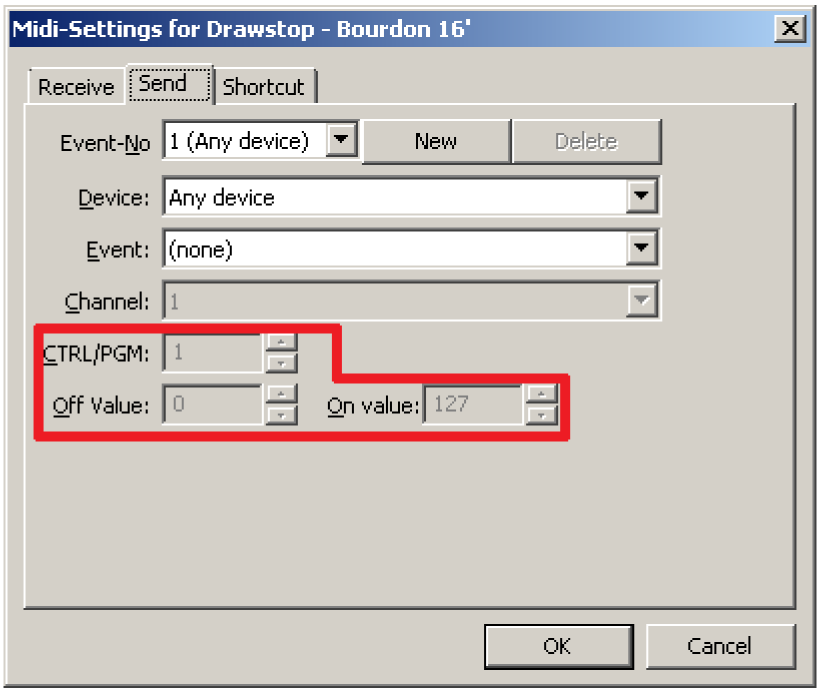
Note
An event can also be marked for deletion by setting its *Event* dropdown to ***None***. It is effectively deleted when the OK button is pressed.
It is the only way to delete the last event and blank out the list

Listen for Event Triggers *listening mode*. GrandOrgue waits for a MIDI event, then fills the dialog with the event's data.

Note
This legacy configuration technique is useful only for very simple settings. The preferred way to configure MIDI is to use the *Detect Complex Midi Setup* functionality.

Detect complex MIDI Setup Triggers *listening mode*. GrandOrgue asks for two MIDI event sequences which define the operating range for the control. More information is available for each console element in the next section. Suitable instructions are displayed in the space above the OK / Cancel buttons.

4.2 Send


Midi-Settings for Drawstop - Bourdon 16'

Event-No: 1 (Any device)
Device: Any device
Event: (none)
Channel: 1
CTRL/PGM: 1
Off Value: 0
On value: 127

Figure 4.2: Midi sent event

This tab configures which MIDI events are sent from the console element.

The overall layout is the same for all elements. Depending on the element being configured, the fields circled in red will be disabled or relabeled.

It is possible to define several sent events for the same GUI control. All of them will be sent whenever the control is triggered.

Event-No Event identifier. If the event is associated to a specific MIDI device, the device name is part of the identifier.

Device The dropdown displays every MIDI output device that was ever known to GrandOrgue. If a device is selected in the dropdown, GrandOrgue outputs that event only to that device. If no device is selected, GrandOrgue outputs that event to all selected output devices (in the Midi Devices Tab) connected to the computer.

Event The dropdown displays the list of allowed events for the control.

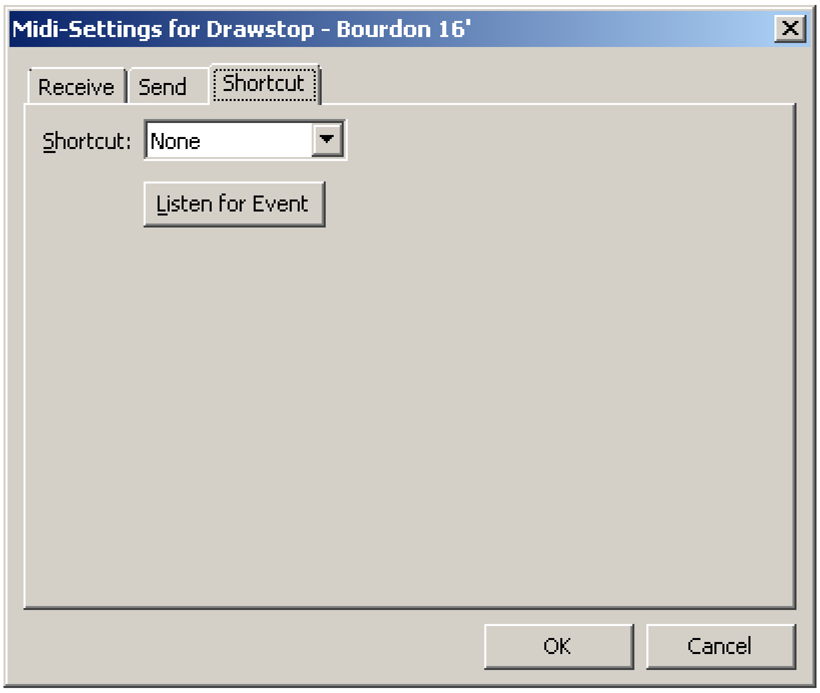
Channel MIDI channel for this event (1-16, defaults to 1).

New This buttow allows to add a new event to the list of sent events. A blank form is displayed with an incremented event number.

Delete This button allows to delete the currently selected event. *It is enabled only when there are at least 2 events in the list.*

Note
An event can also be marked for deletion by setting its *Event* dropdown to ***None***. It is effectively deleted when the OK button is pressed.
It is the only way to delete the last event and blank out the list

4.3 Keyboard shortcut


Midi-Settings for Drawstop - Bourdon 16'

Shortcut: None

Figure 4.3: Keyboard shortcut

This tab is displayed only for drawstops and buttons.

Shortcut This dropdown lists all usable key values.

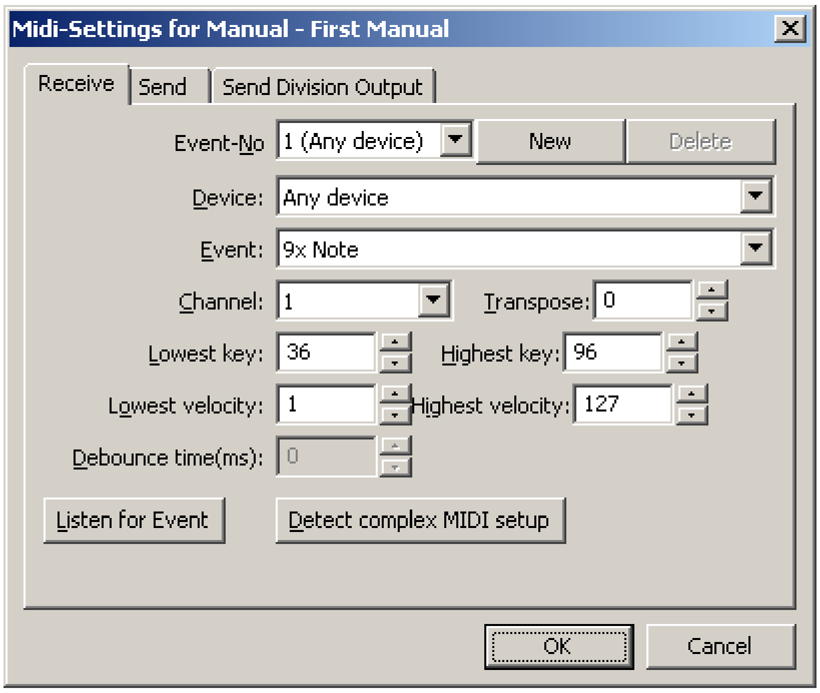
Note
According to the platform running GrandOrgue, some keys will be unusable.

Listen for Event Triggers *listening mode*. GrandOrgue waits for a key to be depressed, then sets the dropdown to that key.

4.4 Editor by console element

4.4.1 Keyboard

4.4.1.1 Receive


Midi-Settings for Manual - First Manual

Receive | Send | Send Division Output

Event-No: 1 (Any device) [New] [Delete]
Device: Any device
Event: 9x Note
Channel: 1 Transpose: 0
Lowest key: 36 Highest key: 96
Lowest velocity: 1 Highest velocity: 127
Debounce time(ms): 0

[Listen for Event] [Detect complex MIDI setup]

[OK] [Cancel]

Event The available events are:

9x Note Receive NoteOn / NoteOff with velocity. The velocity value is used to select the "best" attack sample and alter sound rendering (e.g. piano plays loud/soft according to velocity).
This setting honors the key mapping that can be defined in the Organ Definition File (Manual999, optional MIDIKey999 entries).

9x Note without Velocity Same as 9x Note. The velocity value, if any, is always translated to On = 127, Off = 0.

9x Note short octave at low key Same as 9x Note plus hardcoded short octave processing in the lowest octave.
The lowest octave is that of the lowest C on a full compass keyboard. *Lowest key* field value needs to match the actual lowest C MIDI key number.

Note
The hardcoded processing matches E key to C note, F key to F note, # key to D note, G key to G note, G# key to E note, and from A upwards, the key on the keyboard matches the same note.

The main usage of this setting is to quickly convert a full compass keyboard to short octave. See appendix A.

9x Note without map Ignore the key mapping defined in the Organ Definition File (Manual999, optional MIDIKey999 entries).

The main usage of this setting is to quickly convert a short octave keyboard to full compass. See appendix A.

Transpose This spinner sets a permanent transposition in semitones. Positive values transpose up, negative values transpose down.

Lowest Key, Highest Key These spinners set the key number range which GrandOrgue will acknowledge when listening for MIDI events. Any key number outside the range will be ignored. The lowest key value designates the lowest octave (where short octave processing takes place).

Lowest velocity, Highest velocity These spinners set the velocity range which GrandOrgue will acknowledge when listening for MIDI events.

- All events with velocity > "highest velocity" are ignored.
- All events with velocity < "lowest velocity" are translated to *NoteOff*.
- Velocity values between "lowest velocity" and "highest velocity" are rescaled to 0-127

Warning
Lowest velocity must be at least 1, otherwise all key releases will be missed.

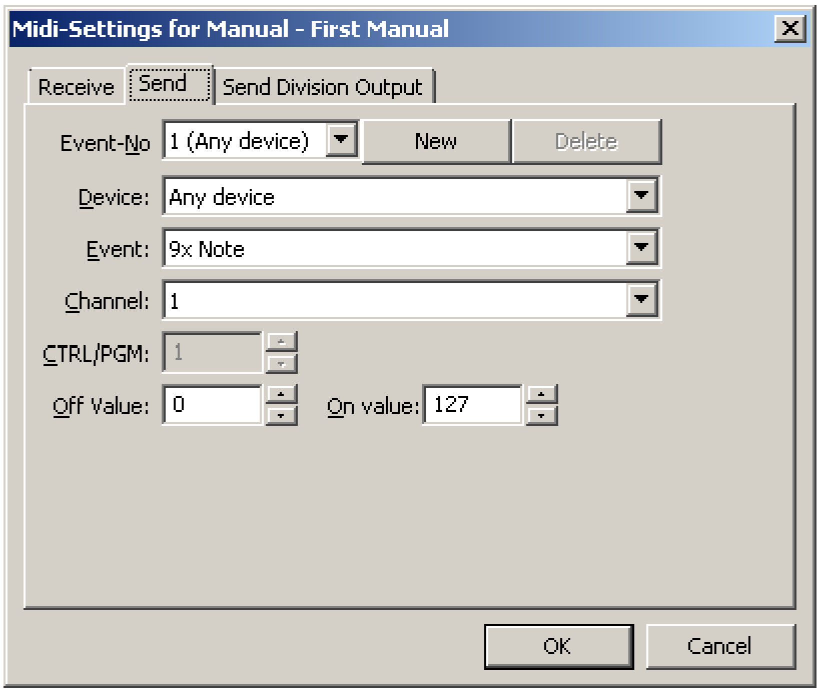
Debounce time This spinner is always disabled for a keyboard.

Detect complex MIDI Setup Asks to press the lowest and highest keys available on the keyboard. GrandOrgue uses the values it reads to determine the channel, lowest key number, highest key number, lowest velocity value. Highest velocity value is always set to 127. Event is set to

- **9x Note short octave at low key** if the lowest E is hit
- **9x Note** if any other key in the lowest octave is hit

4.4.1.2 Send

This tab controls how the input state of the manual keys is mirrored to an external MIDI device


Midi-Settings for Manual - First Manual

Receive | Send | Send Division Output

Event-No: 1 (Any device)
Device: Any device
Event: 9x Note
Channel: 1
CTRL/PGM: 1
Off Value: 0
On value: 127

Event The available events are:

> **9x Note** Send NoteOn / NoteOff. Velocity is sent, rescaled to the range defined in the *Off Value* and *On Value* fields.
>
> **9x Note without Velocity** Send NoteOn / NoteOff. The velocity value is always set to On = value of *On Value* field, Off = value of *Off Value* field.

CTRL/PGM This spinner is always disabled for a keyboard.

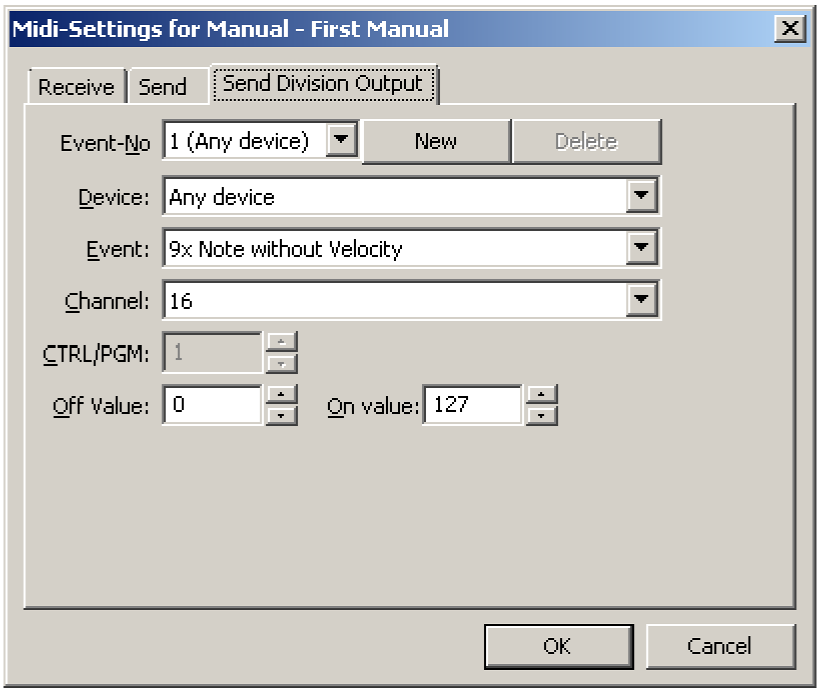
Off Value This spinner sets the velocity value sent when a key is released on the keyboard.

On Value This spinner sets the velocity value sent when a key is depressed on the keyboard.

Note
Off value **must** be set to zero (0) if NoteOff is wished.

4.4.1.3 Send Division Output

This tab controls how the state **after coupler processing** of the division is sent to an external MIDI device.



Event-No: 1 (Any device)
Device: Any device
Event: 9x Note without Velocity
Channel: 16
CTRL/PGM: 1
Off Value: 0
On value: 127

Event The available events are:

> **9x Note** Send NoteOn / NoteOff. Velocity is sent, rescaled to the range defined in the *Off Value* and *On Value* fields.
>
> **9x Note without Velocity** Send NoteOn / NoteOff. The velocity value is always set to On = value of *On Value* field, Off = value of *Off Value* field.

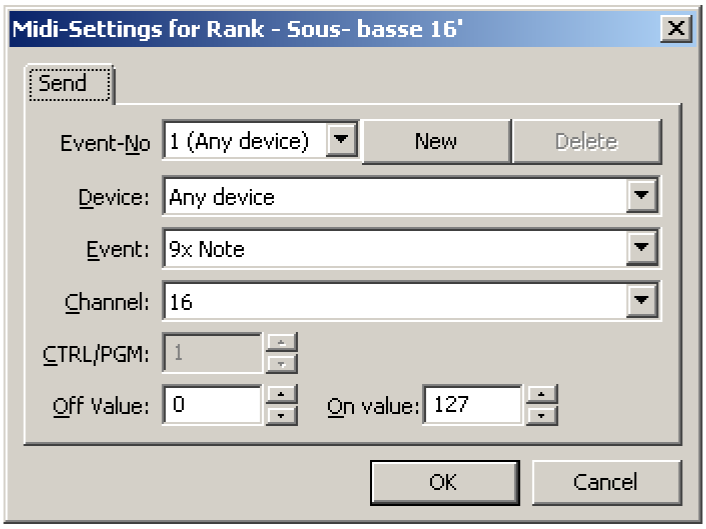
CTRL/PGM This spinner is always disabled for a keyboard.

Off Value This spinner sets the velocity value sent when a key is released on the keyboard.

On Value This spinner sets the velocity value sent when a key is depressed on the keyboard.

Note
Off value **must** be set to zero (0) if NoteOff is wished.

4.4.2 Rank


Midi-Settings for Rank - Sous- basse 16'

Send

Event-No: 1 (Any device)

Device: Any device

Event: 9x Note

Channel: 16

CTRL/PGM: 1

Off Value: 0

On value: 127

This setting allows to send NoteOn / NoteOff data only when a rank has been activated by pulling the stop it belongs to. This setting is loosely related to the *Manual's Send* settings. The following table summarizes the main use cases for each setting.

Manual: Send	Mirrors the keyboard input state. The configured Midi event is sent whenever the key is physically depressed. Couplers defined in GrandOrgue are **ignored**	Can be used for recording keyboard action to an external Midi device
Manual: Send Division Output	Sends the division state after all couplers have been processed. Even if the key is displayed as depressed in GrandOrgue GUI, the event tied to the manual is **not** sent	Can be used to control an external windchest having a single valve per note
Rank: Send	Sends the individual pipe state	Can be used to control an external windchest having a valve per pipe

Event The available events are:

9x Note Send NoteOn / NoteOff. Velocity is sent, rescaled to the range defined in the *Off Value* and *On Value* fields.

9x Note without Velocity Send NoteOn / NoteOff. The velocity value is always set to On = value of *On Value* field, Off = value of *Off Value* field.

CTRL/PGM This spinner is always disabled for a rank.

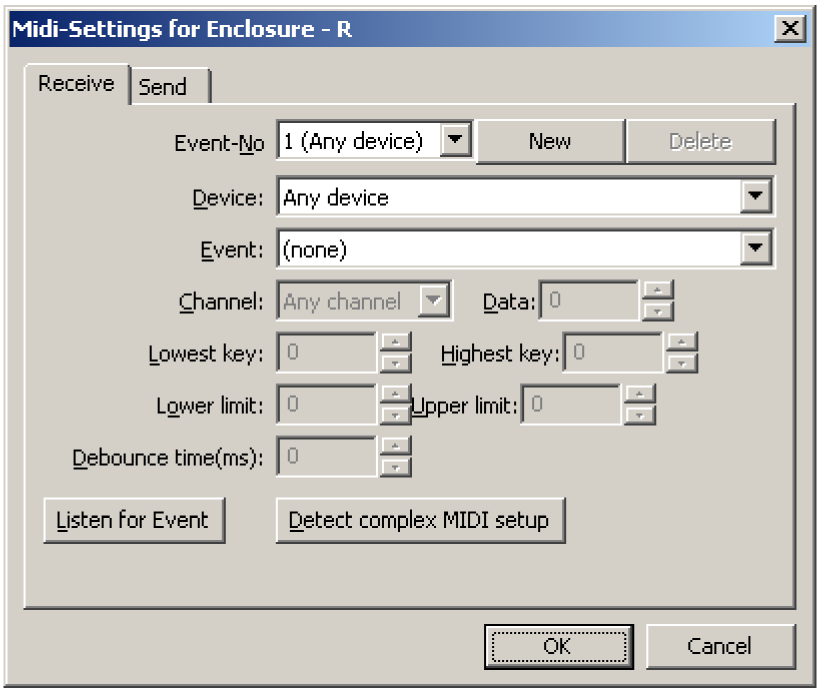
Off Value This spinner sets the velocity value sent when a pipe is muted.

On Value This spinner sets the velocity value sent when a pipe starts playing.

Note
Rank Midi configuration can be accessed only via the Midi Objects list.

4.4.3 Enclosure

4.4.3.1 Receive



Midi-Settings for Enclosure - R

Event-No: 1 (Any device)
Device: Any device
Event: (none)
Channel: Any channel
Data: 0
Lowest key: 0
Highest key: 0
Lower limit: 0
Upper limit: 0
Debounce time(ms): 0

Event The available events are:

Bx controller Receive Control Change messages.

GrandOrgue expects this message sequence: Bn controller# MSB; controller value.

RPN Receive RPN (Registered Parameter Number) messages.

GrandOrgue supports: Parameter number MSByte, Parameter number LSByte and only Data Entry MSByte (0x06).

GrandOrgue expects this message sequence: Bn 0x65 parameter# MSB; [Bn 0x64 parameter# LSB;] Bn 0x06 parameter value

NRPN Receive NRPN (Non Registered Parameter Number) messages.

GrandOrgue supports: Parameter number MSByte, Parameter number LSByte and only Data Entry MSByte (0x06).

GrandOrgue expects this message sequence: Bn 0x63 parameter# MSB; [Bn 0x62 parameter# LSB;] Bn 0x06 parameter value

Data Sets the controller (Bx) or parameter (RPN/NRPN) number.

When receiving RPN, if the MIDI device sends Bn 0x65 <data1>; Bn 0x64 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2*.

E.g.: Bn 0x65 1; Bn 0x64 2 will display 130.

The behavior is identical for NRPN.

Note
The label of this field changes according to the event type:

Bx Controller	=>	Controller-No
RPN or NRPN	=>	Parameter-No

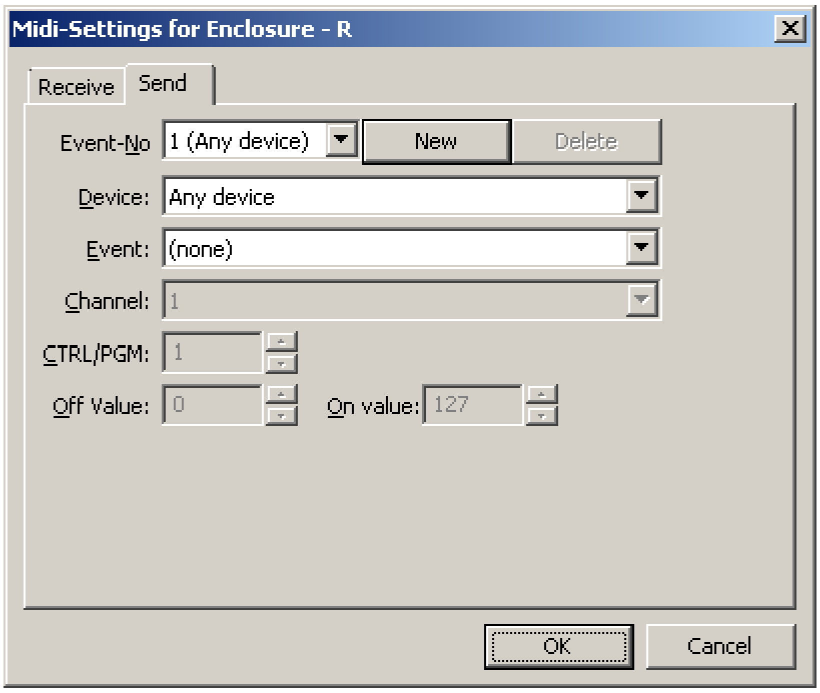
Lowest Key, Highest Key These spinners are always disabled for an enclosure.

Lower limit, Upper limit These spinners set the controller values range which defines a full open/closed enclosure. The value is rescaled to the full range (0-127) for that controller. Values outside the range are translated to their limit values: value < lower limit yields 0, value > upper limit yields 127.

Debounce time This spinner is always disabled for an enclosure.

Detect complex MIDI Setup Asks to move the physical device (usually a shoe) to fully open and close the enclosure. GrandOrgue uses the values it reads to determine the channel, controller type and number and the limits of the usable range.

4.4.3.2 Send



Event-No: 1 (Any device)
Device: Any device
Event: (none)
Channel: 1
CTRL/PGM: 1
Off Value: 0
On value: 127

Event The available events are:

Bx controller Send Control Change messages.

GrandOrgue sends this message sequence: Bn controller# MSB; controller value.

RPN Send RPN (Registered Parameter Number) messages.

GrandOrgue supports: Parameter number MSByte, Parameter number LSByte and only Data Entry MSByte (0x06).

GrandOrgue sends this message sequence: Bn 0x65 parameter# MSB; Bn 0x64 parameter# LSB; Bn 0x06 parameter value

NRPN Send NRPN (Non Registered Parameter Number) messages.

GrandOrgue supports: Parameter number MSByte, Parameter number LSByte and only Data Entry MSByte (0x06).

GrandOrgue sends this message sequence: Bn 0x63 parameter# MSB; Bn 0x62 parameter# LSB; Bn 0x06 parameter value

CTRL/PGM For a Bx controller, this spinner sets the controller number (range 0-127).

For RPN or NRPN, this value is in the range 0-16383. GrandOrgue will automatically split this parameter number into its most significant byte and least significant byte to send Bn 0x65 <data1>; Bn 0x64 <data2> where data1 is computed as "ParameterNo right shifted 7 bits AND 0x7F" and data2 is computed as "ParameterNo AND 0x7F"

Note

The label of this field changes according to the event type:

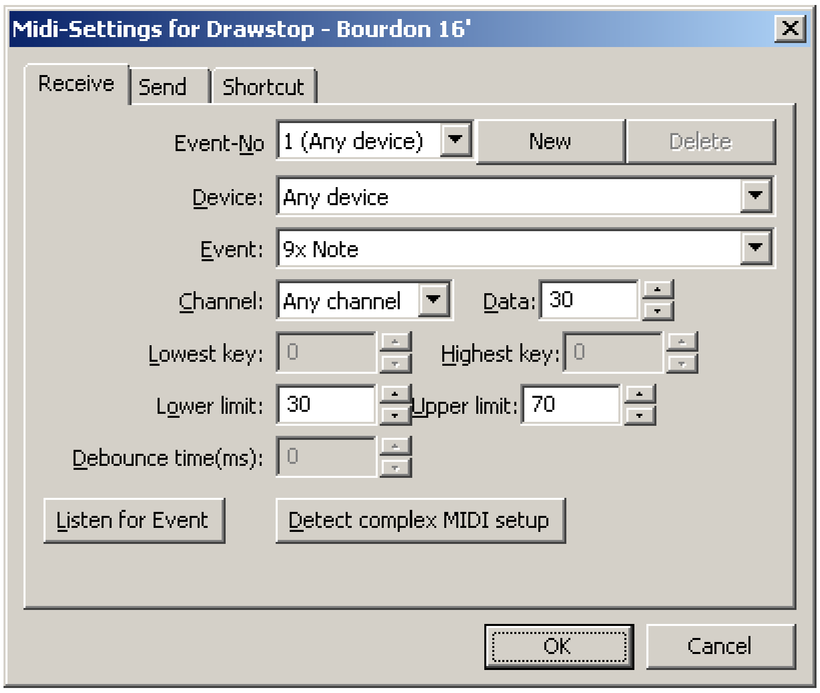
Bx Controller	=>	Controller-No
RPN or NRPN	=>	Parameter-No

Off Value, On Value Define the values for sending a fully open/closed enclosure. Values will be rescaled to the range defined by these limits.

4.4.4 Drawstop and Pushbutton

4.4.4.1 Receive

4.4.4.1.1 9x Note


Midi-Settings for Drawstop - Bourdon 16'

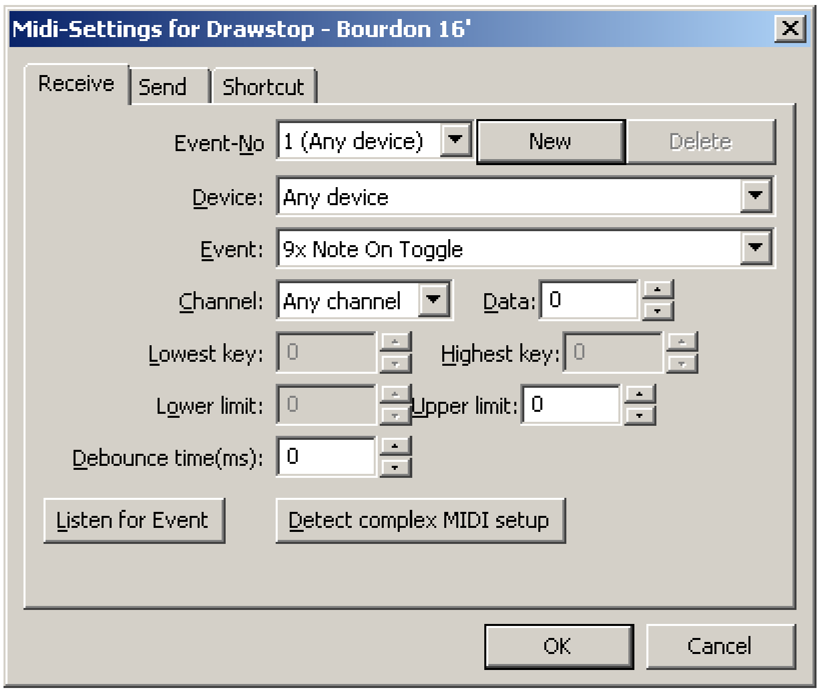
Receive | Send | Shortcut

Event-No: 1 (Any device)
Device: Any device
Event: 9x Note
Channel: Any channel
Data: 30
Lowest key: 0
Highest key: 0
Lower limit: 30
Upper limit: 70
Debounce time(ms): 0

Receive NoteOn / NoteOff.

Data	Key number
Lower limit	Maximum velocity value that toggles the button or drawstop off
Upper limit	Minimum velocity value that toggles the button or drawstop on
	All velocity values between lower and upper limit are ignored
Lowest key	Unused by this setting
Highest key	Unused by this setting
Debounce time	Unused by this setting

4.4.4.1.2 9x Note On Toggle


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)

Device: Any device

Event: 9x Note On Toggle

Channel: Any channel

Data: 0

Lowest key: 0

Highest key: 0

Lower limit: 0

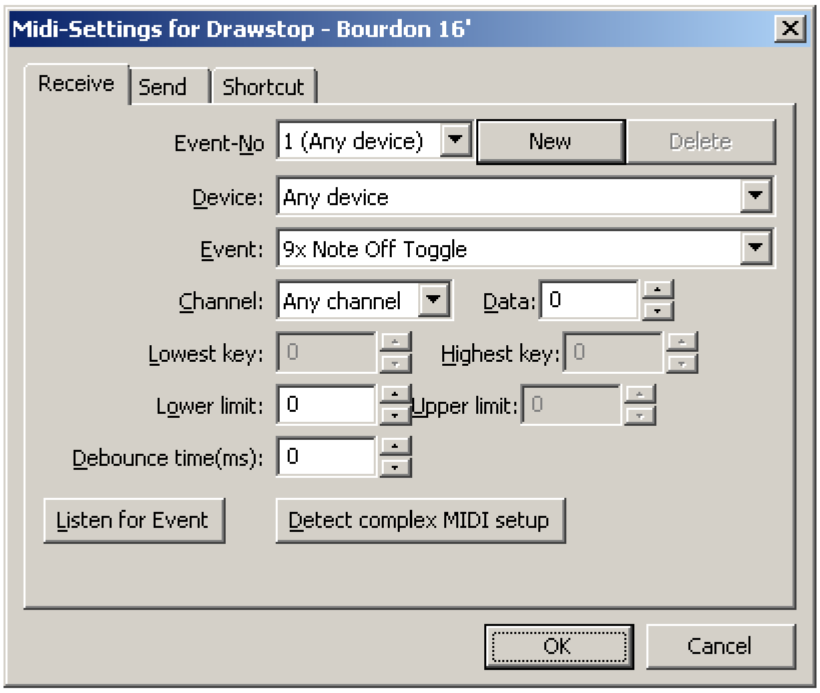
Upper limit: 0

Debounce time(ms): 0

Receive NoteOn messages only. Two successive NoteOn messages on the same key number will toggle the state of the button/-drawstop.

Data	Key number
Upper limit	Minimum velocity value that toggles the button or drawstop
Debounce time	Minimum time in milliseconds that must elapse between two successive NoteOn messages for the drawstop/button to toggle its state.
Lower limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.3 9x Note Off Toggle


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)

Device: Any device

Event: 9x Note Off Toggle

Channel: Any channel

Data: 0

Lowest key: 0

Highest key: 0

Lower limit: 0

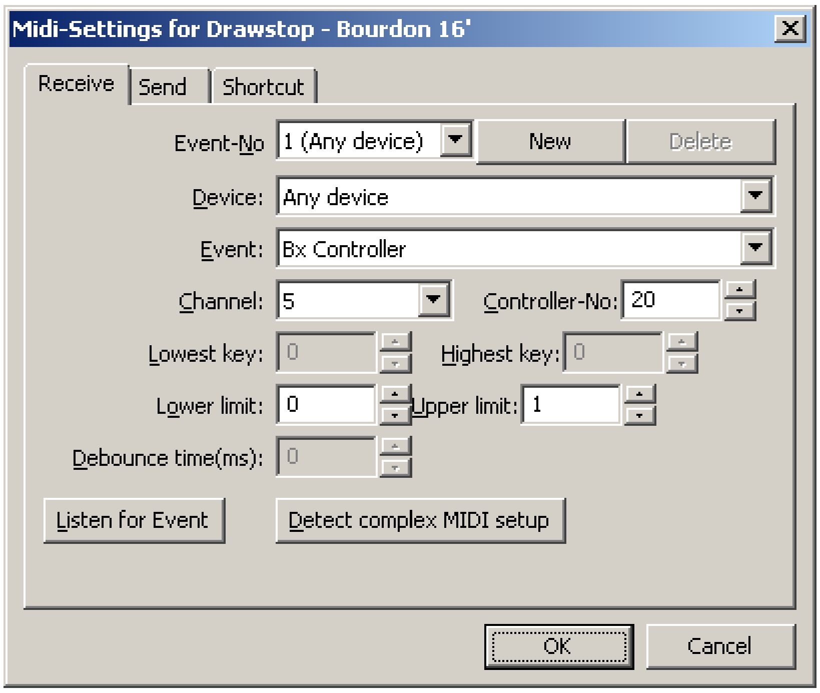
Upper limit: 0

Debounce time(ms): 0

Receive NoteOff messages only. Two successive NoteOff messages on the same key number will toggle the state of the button/-drawstop.

Data	Key number
Lower limit	Maximum velocity value that toggles the button or drawstop
Debounce time	Minimum time in milliseconds that must elapse between two successive NoteOff messages for the drawstop/button to toggle its state.
Upper limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.4 Bx Controller


Midi-Settings for Drawstop - Bourdon 16'
Receive
Send
Shortcut
New
Delete
Listen for Event
Detect complex MIDI setup
OK
Cancel

Event-No: 1 (Any device)
Device: Any device
Event: Bx Controller
Channel: 5
Controller-No: 20
Lowest key: 0
Highest key: 0
Lower limit: 0
Upper limit: 1
Debounce time(ms): 0

Receive Control Change messages. GrandOrgue expects this message sequence: Bn controller# MSB; controller value.

Controller-No Controller number. Range is [0, 127]

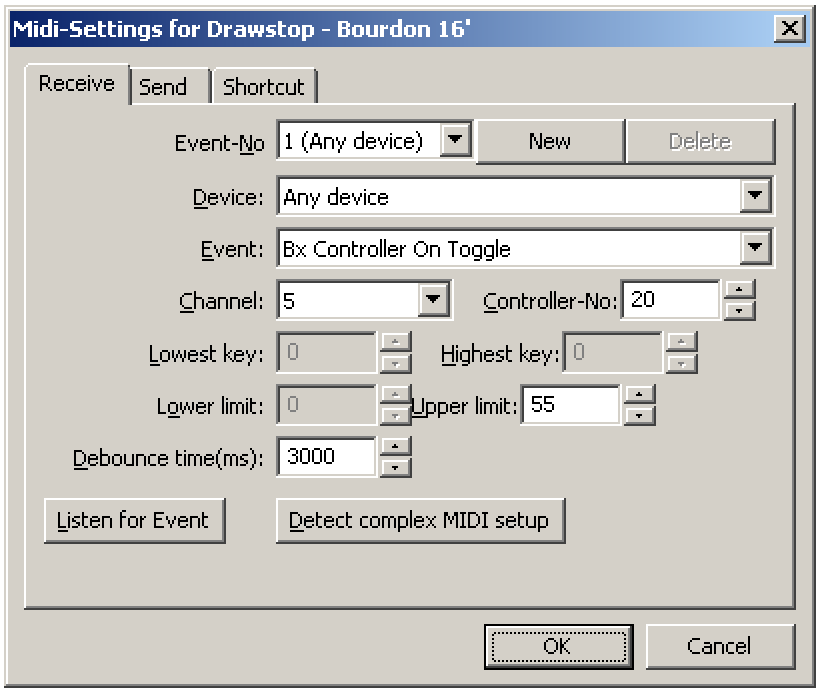
Note
Bank Select, RPN, NRPN and Data Entry controllers are reserved by GrandOrgue and can NOT be used for normal matching. Namely:

0x00 (0)	Bank Select MSB
0x20 (32)	Bank Select LSB
0x06 (6)	Data entry LSB
0x62 (98)	NRPN LSB
0x63 (99)	NRPN MSB
0x64 (100)	RPN LSB
0x65 (101)	RPN MSB

Lower Limit Maximum data value that toggles the button or drawstop off. Range is [0, 127]
Upper Limit Minimum data value that toggles the button or drawstop on. Range is [0, 127]
All data values between lower and upper limit are ignored

Lowest key Unused by this setting
Highest key Unused by this setting
Debounce time Unused by this setting

4.4.4.1.5 Bx Controller On Toggle


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

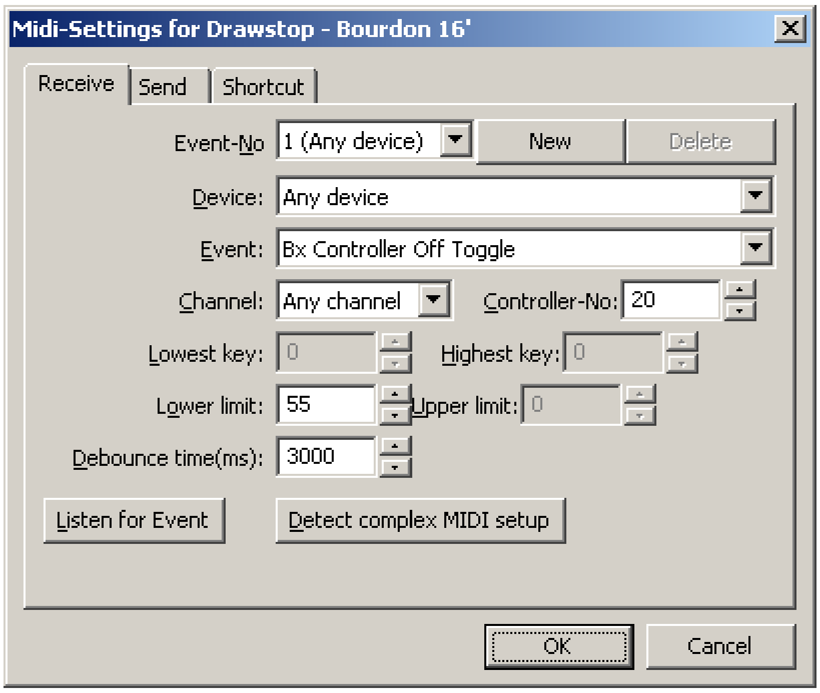
Event-No: 1 (Any device)
Device: Any device
Event: Bx Controller On Toggle
Channel: 5
Controller-No: 20
Lowest key: 0
Highest key: 0
Lower limit: 0
Upper limit: 55
Debounce time(ms): 3000

Receive Control Change messages. GrandOrgue expects this message sequence: Bn controller# MSB; controller value.

The button or drawstop toggles its state whenever two successive "identical" messages are received.

Controller-No	Controller number. Range is [0, 127]. See above for exclusions.
Upper Limit	Minimum data value that toggles the button or drawstop state. Range is [0, 127]. "Identical" means that two successive controller values are above this limit.
Debounce time	Minimum time in milliseconds that must elapse between two successive "identical" messages for the drawstop/button to toggle its state.
Lower Limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.6 Bx Controller Off Toggle


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device) | New | Delete

Device: Any device

Event: Bx Controller Off Toggle

Channel: Any channel | Controller-No: 20

Lowest key: 0 | Highest key: 0

Lower limit: 55 | Upper limit: 0

Debounce time(ms): 3000

Listen for Event | Detect complex MIDI setup

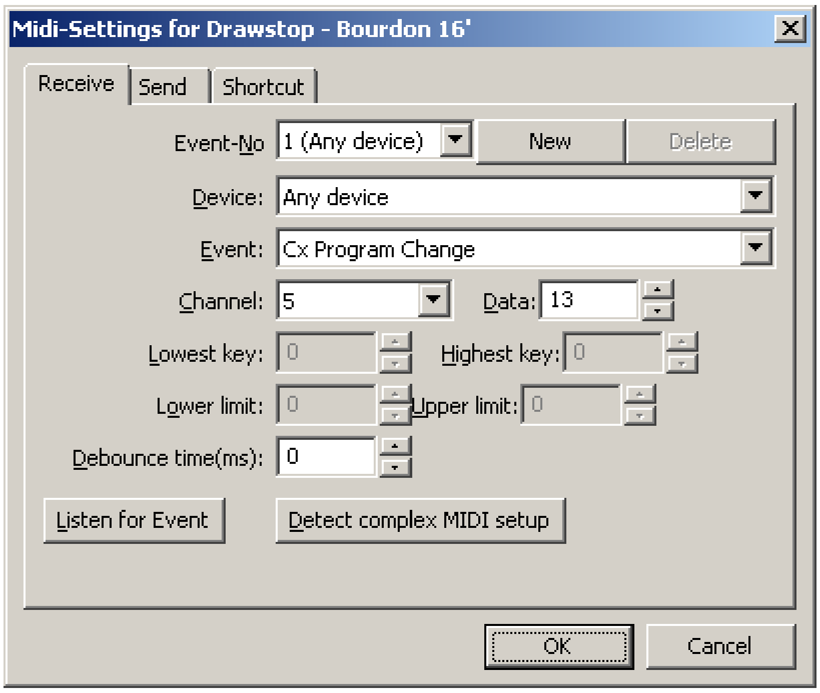
OK | Cancel

Receive Control Change messages. GrandOrgue expects this message sequence: Bn controller# MSB; controller value.

The button or drawstop toggles its state whenever two successive "identical" messages are received.

Controller-No	Controller number. Range is [0, 127]. See above for exclusions.
Lower Limit	Maximum data value that toggles the button or drawstop state. Range is [0, 127]. "Identical" means that two successive controller values are below this limit.
Debounce time	Minimum time in milliseconds that must elapse between two successive "identical" messages for the drawstop/button to toggle its state.
Upper Limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.7 Cx Program Change


Midi-Settings for Drawstop - Bourdon 16'

Event-No: 1 (Any device)
Device: Any device
Event: Cx Program Change
Channel: 5
Data: 13
Lowest key: 0
Highest key: 0
Lower limit: 0
Upper limit: 0
Debounce time(ms): 0

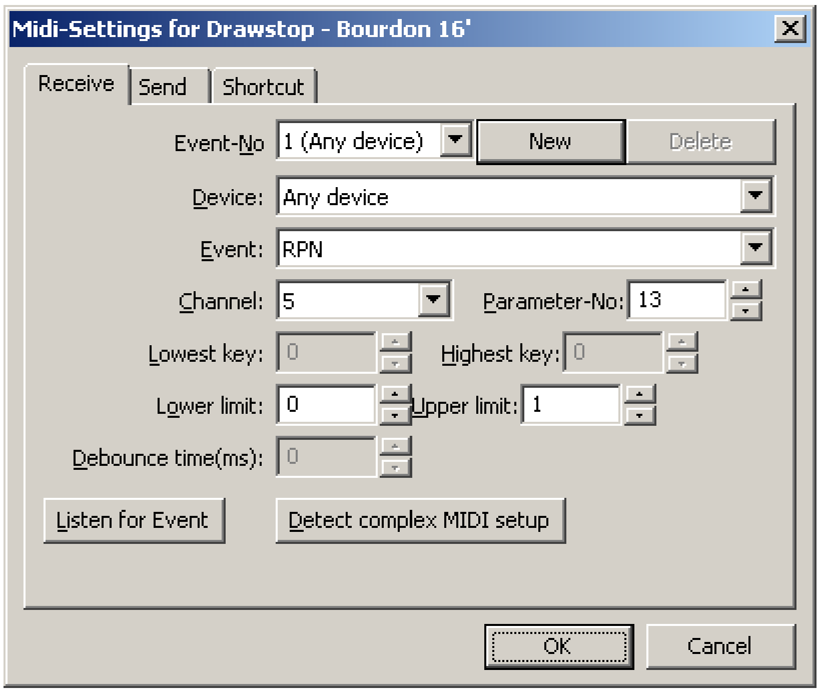
Receive Program Change messages. GrandOrgue expects this message sequence: [Bn 0x00 bank# MSB;][Bn 0x20 bank# LSB;] Cn prog#.

The optional bank select controllers enable GrandOrgue to manage more than 128 drawstops or buttons on a single channel with Program Change messages.

This control is a toggle: two successive messages changing the same program will toggle the state of the drawstop/button.

Data	Program number. Range is [0, 127] when bank select messages are not used. When receiving Bank Select + Program Change, if the MIDI device sends Bn 0x00 <data1>; Bn 0x20 <data2>; Cn prog#, the program number will be displayed as the value of *data1* shifted left 14 bits ORed (bitwise) with *data2* shifted left 7 bits ORed (bitwise) with *prog#* + 1 [math: ((data1 << 14) | (data2 << 7) | prog#) + 1].
Debounce time	Minimum time in milliseconds that must elapse between two successive messages changing to the same program for the drawstop/button to toggle its state.
Lower Limit	Unused by this setting
Upper Limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.8 RPN



Midi-Settings for Drawstop - Bourdon 16'

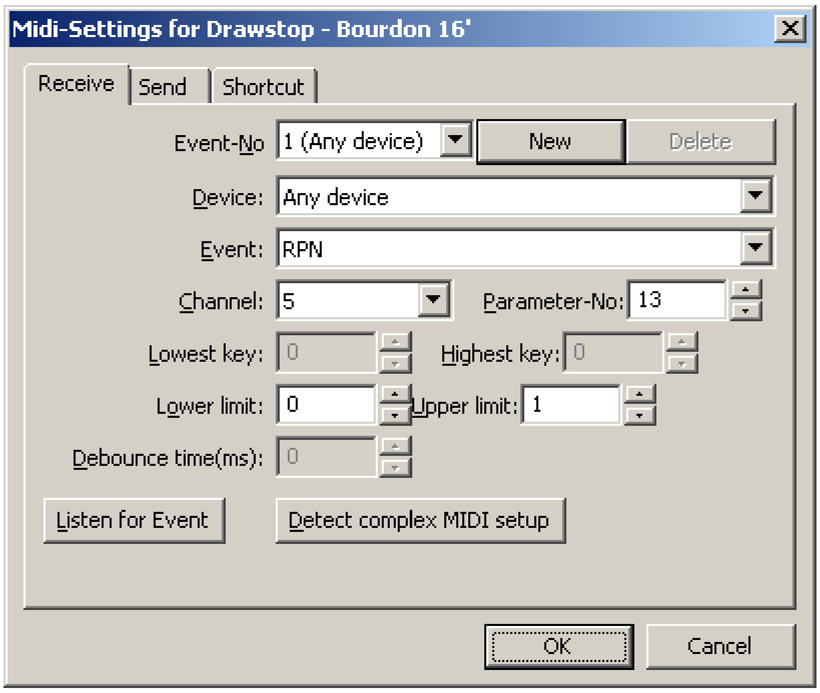
Receive | Send | Shortcut

Event-No: 1 (Any device)
Device: Any device
Event: RPN
Channel: 5
Parameter-No: 13
Lowest key: 0
Highest key: 0
Lower limit: 0
Upper limit: 1
Debounce time(ms): 0

Receive RPN (Registered Parameter Number) messages. GrandOrgue expects this message sequence: Bn 0x65 parameter# MSB; [Bn 0x64 parameter# LSB;] Bn 0x06 parameter value.

Parameter-No	Parameter number. If the MIDI device sends Bn 0x65 <data1>; Bn 0x64 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
Lower Limit	Maximum parameter value that toggles the button or drawstop off. Range is [0, 127]
Upper Limit	Minimum parameter value that toggles the button or drawstop on. Range is [0, 127]
Lowest key	Unused by this setting
Highest key	Unused by this setting
Debounce time	Unused by this setting

4.4.4.1.9 NRPN

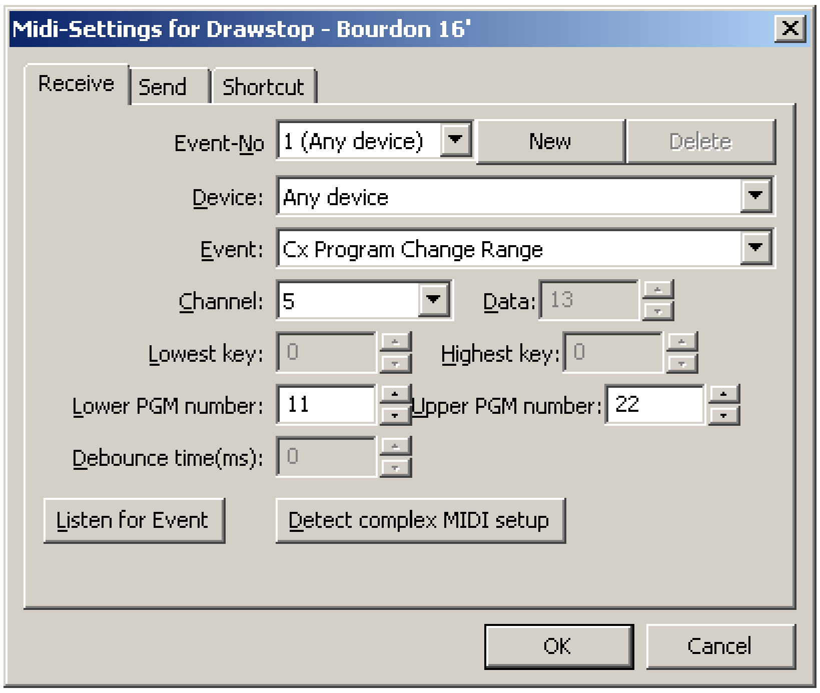

Midi-Settings for Drawstop - Bourdon 16'
Receive
Send
Shortcut
New
Delete
Listen for Event
Detect complex MIDI setup
OK
Cancel

Event-No: 1 (Any device)
Device: Any device
Event: RPN
Channel: 5
Parameter-No: 13
Lowest key: 0
Highest key: 0
Lower limit: 0
Upper limit: 1
Debounce time(ms): 0

Receive NRPN (Non Registered Parameter Number) messages. GrandOrgue expects this message sequence: Bn 0x63 parameter# MSB; [Bn 0x62 parameter# LSB;] Bn 0x06 parameter value.

Parameter-No	Parameter number. If the MIDI device sends Bn 0x63 <data1>; Bn 0x62 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
Lower Limit	Maximum parameter value that toggles the button or drawstop off. Range is [0, 127]
Upper Limit	Minimum parameter value that toggles the button or drawstop on. Range is [0, 127]
Lowest key	Unused by this setting
Highest key	Unused by this setting
Debounce time	Unused by this setting

4.4.4.1.10 Cx Program Change Range


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)
Device: Any device
Event: Cx Program Change Range
Channel: 5
Data: 13
Lowest key: 0
Highest key: 0
Lower PGM number: 11
Upper PGM number: 22
Debounce time(ms): 0

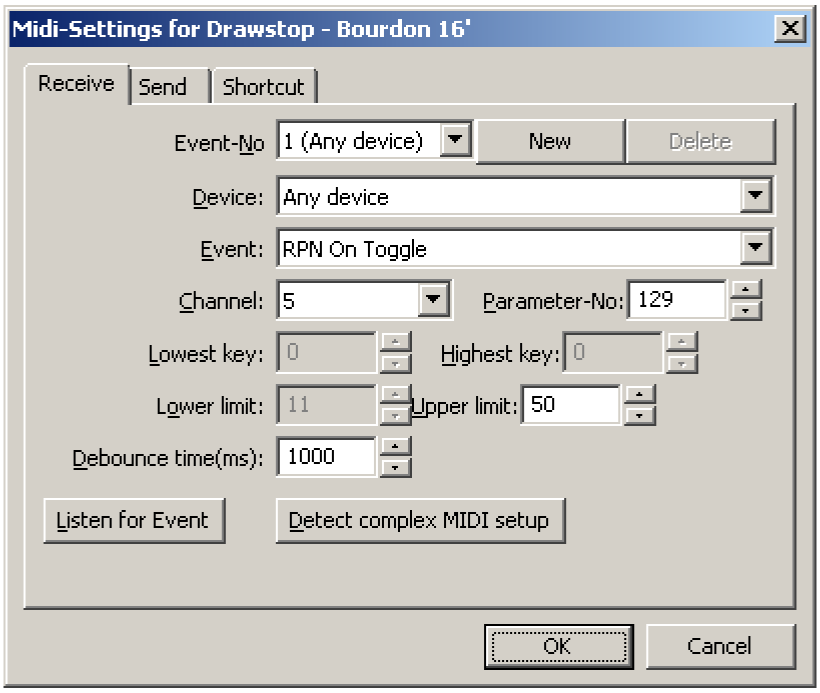
Receive Program Change messages. GrandOrgue expects this message sequence: [Bn 0x00 bank# MSB;][Bn 0x20 bank# LSB;] Cn prog#.

This control uses 2 **different** program numbers: one to pull and the other to push the drawstop or button.

The optional bank select controllers enable GrandOrgue to manage more than 64 drawstops or buttons on a single channel with Program Change messages.

Lower PGM number	Program number to push the drawstop or button. Range is [1, 128] when bank select messages are not used. When receiving Bank Select + Program Change, if the MIDI device sends Bn 0x00 <data1>; Bn 0x20 <data2>; Cn prog#, the program number will be displayed as the value of *data1* shifted left 14 bits ORed (bitwise) with *data2* shifted left 7 bits ORed (bitwise) with *prog#* + 1 [math: ((data1 << 14) | (data2 << 7) | prog#) + 1].
Upper PGM number	Program number to pull the drawstop or button. Range is [1, 128] when bank select messages are not used. When receiving Bank Select + Program Change, if the MIDI device sends Bn 0x00 <data1>; Bn 0x20 <data2>; Cn prog#, the program number will be displayed as the value of *data1* shifted left 14 bits ORed (bitwise) with *data2* shifted left 7 bits ORed (bitwise) with *prog#* + 1 [math: ((data1 << 14) | (data2 << 7) | prog#) + 1].
Data	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting
Debounce time	Minimum time in milliseconds that must elapse between two successive messages changing to the same program for the drawstop/button to toggle its state.

4.4.4.1.11 RPN On Toggle


Midi-Settings for Drawstop - Bourdon 16'

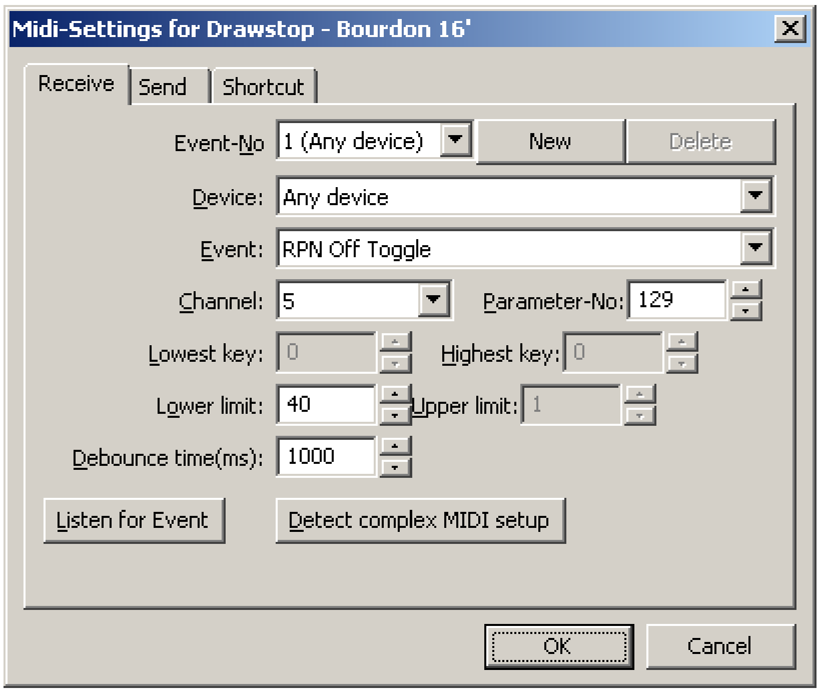
Event-No: 1 (Any device)
Device: Any device
Event: RPN On Toggle
Channel: 5
Parameter-No: 129
Lowest key: 0
Highest key: 0
Lower limit: 11
Upper limit: 50
Debounce time(ms): 1000

Receive RPN (Registered Parameter Number) messages. GrandOrgue expects this message sequence: Bn 0x65 parameter# MSB; [Bn 0x64 parameter# LSB;] Bn 0x06 parameter value.

The button or drawstop toggles its state whenever two successive "identical" messages are received.

Parameter-No	Parameter number. If the MIDI device sends Bn 0x65 <data1>; Bn 0x64 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
Upper Limit	Minimum parameter value that toggles the button or drawstop state. Range is [0, 127]. "Identical" means that two successive parameter values are above this limit.
Debounce time	Minimum time in milliseconds that must elapse between two successive RPN messages for the drawstop/button to toggle its state.
Lower Limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.12 RPN Off Toggle


Midi-Settings for Drawstop - Bourdon 16'
Receive
Send
Shortcut
New
Delete
Listen for Event
Detect complex MIDI setup
OK
Cancel

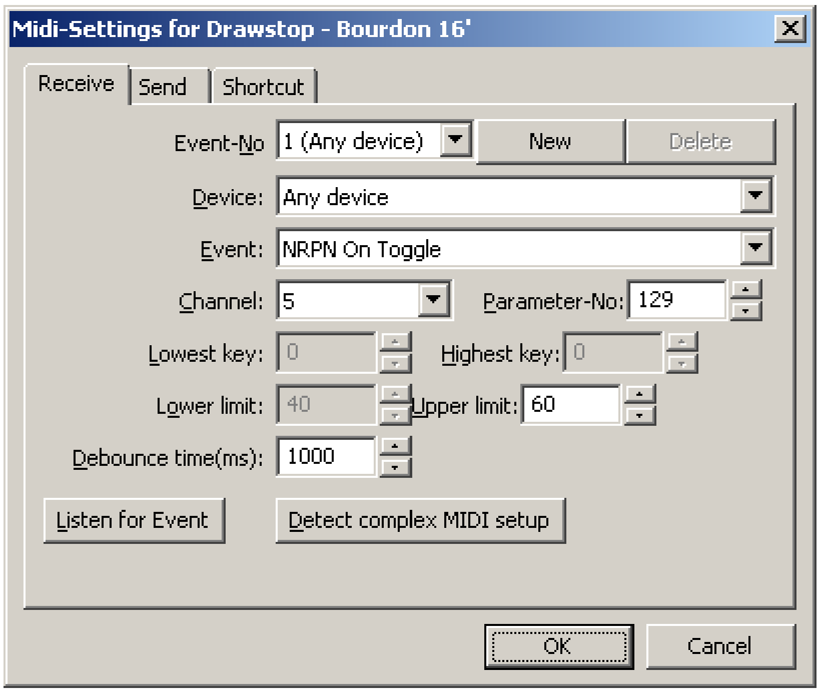
Event-No: 1 (Any device)
Device: Any device
Event: RPN Off Toggle
Channel: 5
Parameter-No: 129
Lowest key: 0
Highest key: 0
Lower limit: 40
Upper limit: 1
Debounce time(ms): 1000

Receive RPN (Registered Parameter Number) messages. GrandOrgue expects this message sequence: Bn 0x65 parameter# MSB; [Bn 0x64 parameter# LSB;] Bn 0x06 parameter value.

The button or drawstop toggles its state whenever two successive "identical" messages are received.

Parameter-No	Parameter number. If the MIDI device sends Bn 0x65 <data1>; Bn 0x64 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
Lower Limit	Maximum parameter value that toggles the button or drawstop state. Range is [0, 127]. "Identical" means that two successive parameter values are below this limit.
Debounce time	Minimum time in milliseconds that must elapse between two successive RPN messages for the drawstop/button to toggle its state.
Upper Limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.13 NRPN On Toggle


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

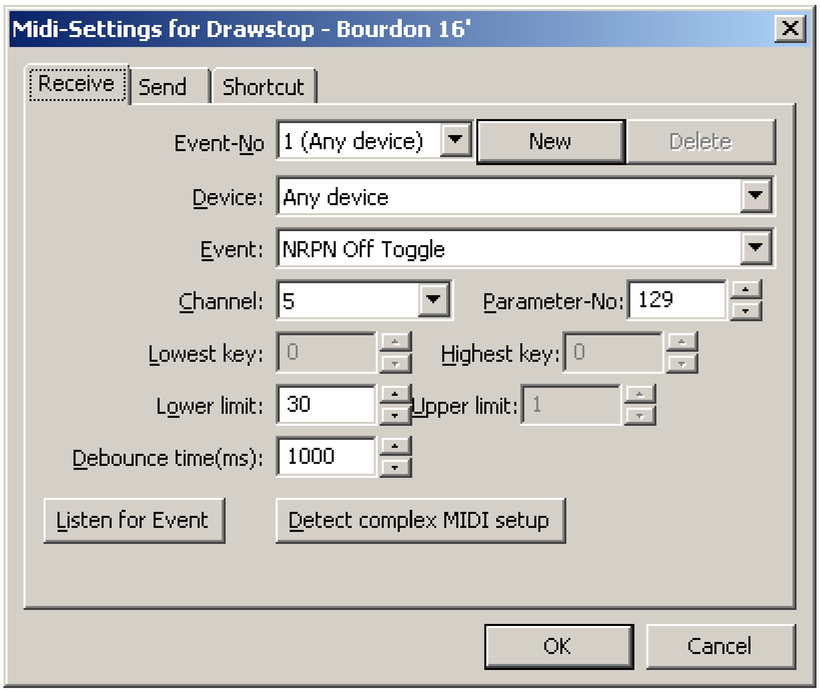
Event-No: 1 (Any device)
Device: Any device
Event: NRPN On Toggle
Channel: 5
Parameter-No: 129
Lowest key: 0
Highest key: 0
Lower limit: 40
Upper limit: 60
Debounce time(ms): 1000

Receive NRPN (Non Registered Parameter Number) messages. GrandOrgue expects this message sequence: Bn 0x63 parameter# MSB; [Bn 0x62 parameter# LSB;] Bn 0x06 parameter value.

The button or drawstop toggles its state whenever two successive "identical" messages are received.

Parameter-No	Parameter number. If the MIDI device sends Bn 0x65 <data1>; Bn 0x64 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
Upper Limit	Minimum parameter value that toggles the button or drawstop state. Range is [0, 127]. "Identical" means that two successive parameter values are above this limit.
Debounce time	Minimum time in milliseconds that must elapse between two successive NRPN messages for the drawstop/button to toggle its state.
Lower Limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.14 NRPN Off Toggle


Midi-Settings for Drawstop - Bourdon 16'
Receive
Send
Shortcut
New
Delete
Listen for Event
Detect complex MIDI setup
OK
Cancel

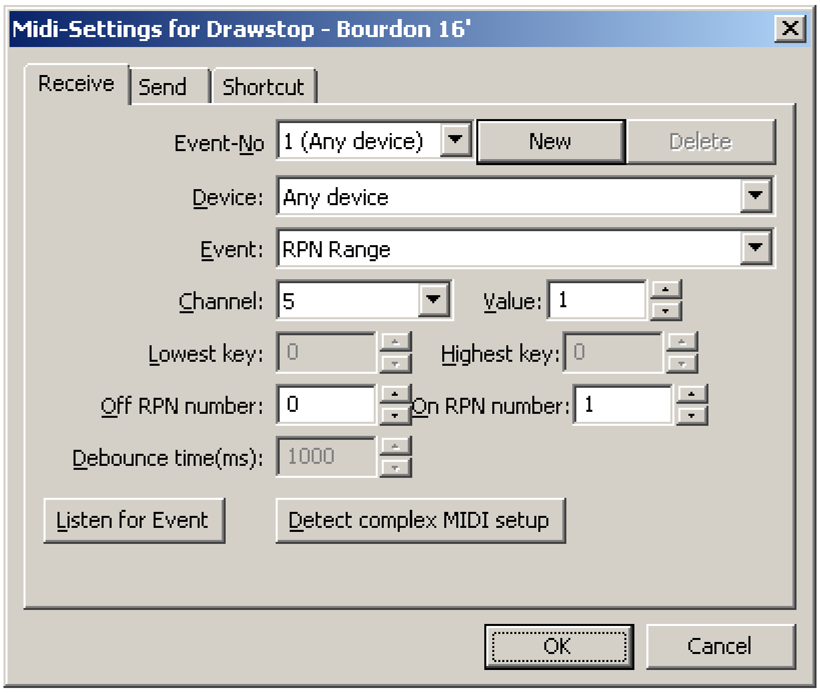
Event-No: 1 (Any device)
Device: Any device
Event: NRPN Off Toggle
Channel: 5
Parameter-No: 129
Lowest key: 0
Highest key: 0
Lower limit: 30
Upper limit: 1
Debounce time(ms): 1000

Receive RPN (Registered Parameter Number) messages. GrandOrgue expects this message sequence: Bn 0x65 parameter# MSB; [Bn 0x64 parameter# LSB;] Bn 0x06 parameter value.

The button or drawstop toggles its state whenever two successive "identical" messages are received.

Parameter-No	Parameter number. If the MIDI device sends Bn 0x65 <data1>; Bn 0x64 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
Lower Limit	Maximum parameter value that toggles the button or drawstop state. Range is [0, 127]. "Identical" means that two successive parameter values are below this limit.
Debounce time	Minimum time in milliseconds that must elapse between two successive NRPN messages for the drawstop/button to toggle its state.
Upper Limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.15 RPN Range


Midi-Settings for Drawstop - Bourdon 16'

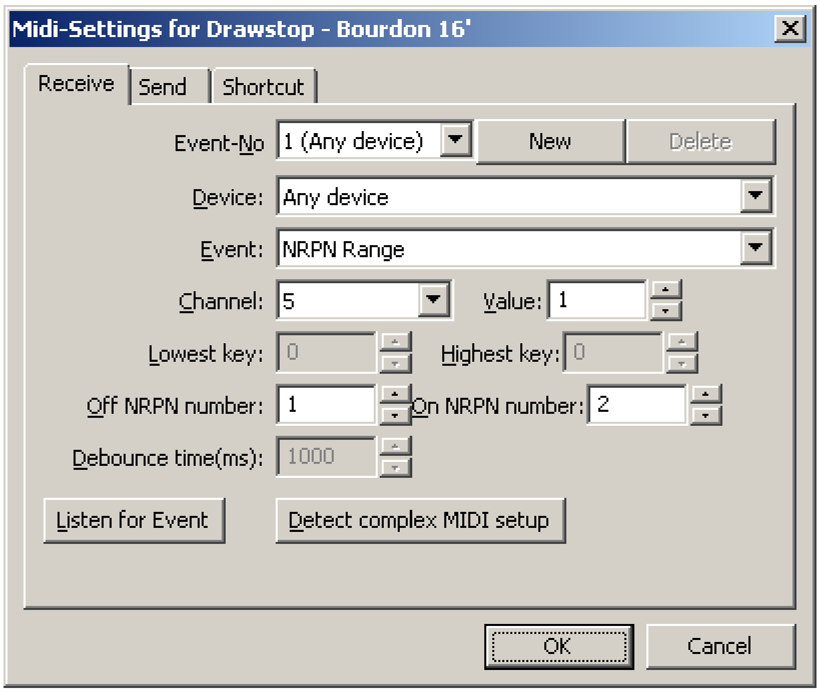
Event-No	1 (Any device)
Device:	Any device
Event:	RPN Range
Channel:	5
Value:	1
Lowest key:	0
Highest key:	0
Off RPN number:	0
On RPN number:	1
Debounce time(ms):	1000

Receive RPN (Registered Parameter Number) messages. GrandOrgue expects this message sequence: Bn 0x65 parameter# MSB; [Bn 0x64 parameter# LSB;] Bn 0x06 parameter value.

This control uses 2 **different** parameter numbers: one to pull and the other to push the drawstop or button. The parameter value **MUST** be the same for both (pull and push) messages.

Value	Parameter value. Range is [0, 127].
Off RPN number	Parameter number to push the drawstop or button. Range is [0, 127]. If the MIDI device sends Bn 0x65 <data1>; Bn 0x64 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
On RPN number	Parameter number to pull the drawstop or button. Range is [0, 127]. If the MIDI device sends Bn 0x65 <data1>; Bn 0x64 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
Lowest key	Unused by this setting
Highest key	Unused by this setting
Debounce time	Unused by this setting

4.4.4.1.16 NRPN Range


Midi-Settings for Drawstop - Bourdon 16'
Receive
Send
Shortcut
New
Delete
Listen for Event
Detect complex MIDI setup
OK
Cancel

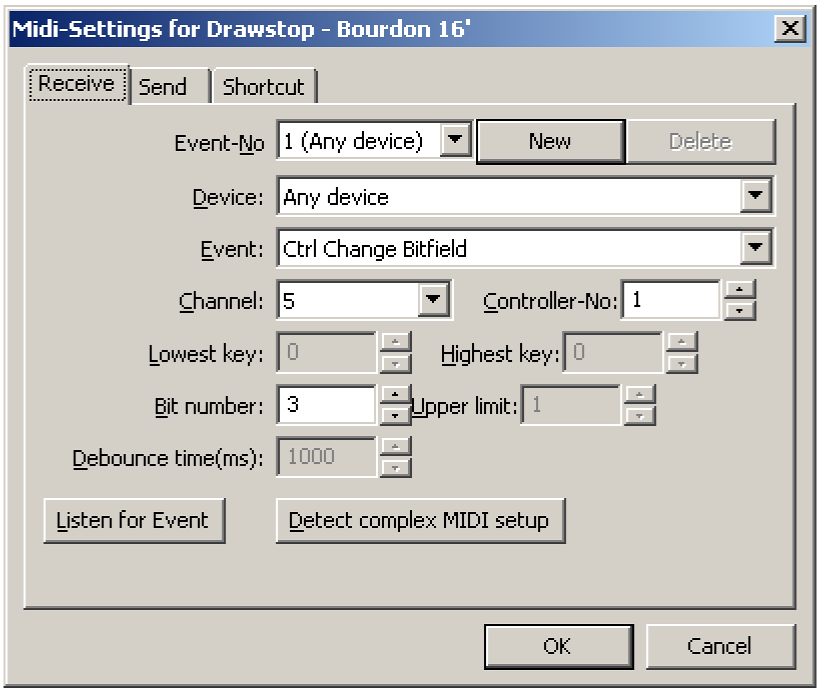
Event-No: 1 (Any device)
Device: Any device
Event: NRPN Range
Channel: 5
Value: 1
Lowest key: 0
Highest key: 0
Off NRPN number: 1
On NRPN number: 2
Debounce time(ms): 1000

Receive NRPN (Registered Parameter Number) messages. GrandOrgue expects this message sequence: Bn 0x63 parameter# MSB; [Bn 0x62 parameter# LSB;] Bn 0x06 parameter value.

This control uses 2 **different** parameter numbers: one to pull and the other to push the drawstop or button. The parameter value **MUST** be the same for both (pull and push) messages.

Value	Parameter value. Range is [0, 127].
Off NRPN number	Parameter number to push the drawstop or button. Range is [0, 127]. If the MIDI device sends Bn 0x63 <data1>; Bn 0x62 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
On NRPN number	Parameter number to pull the drawstop or button. Range is [0, 127]. If the MIDI device sends Bn 0x63 <data1>; Bn 0x62 <data2>, the parameter number will be displayed as the value of *data1* shifted left 7 bits ORed (bitwise) with *data2* [math: (data1 << 7) | data2].
Lowest key	Unused by this setting
Highest key	Unused by this setting
Debounce time	Unused by this setting

4.4.4.1.17 Ctrl Change Bitfield


Midi-Settings for Drawstop - Bourdon 16'

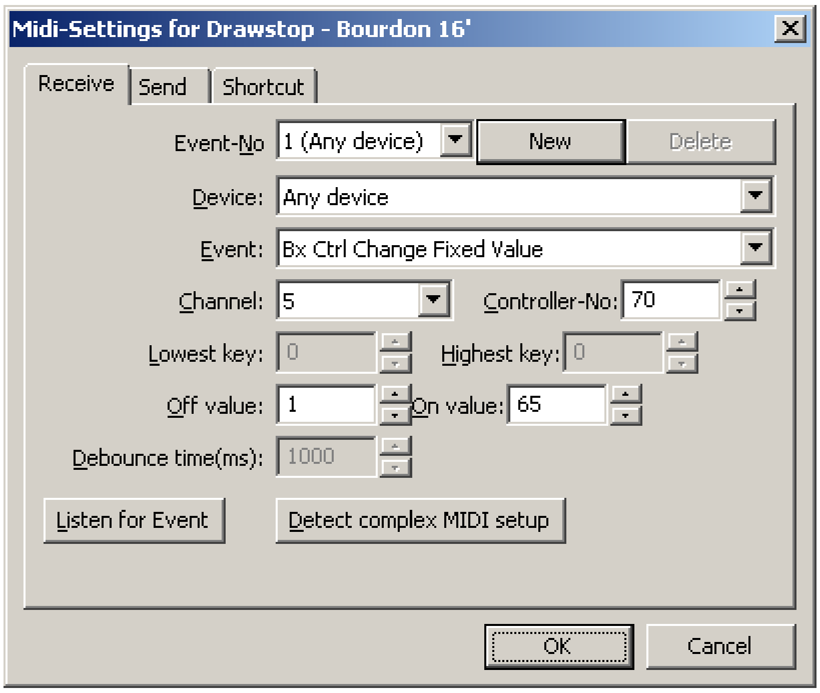
Receive | Send | Shortcut

Event-No: 1 (Any device)
Device: Any device
Event: Ctrl Change Bitfield
Channel: 5
Controller-No: 1
Lowest key: 0
Highest key: 0
Bit number: 3
Upper limit: 1
Debounce time(ms): 1000

This setting is used by some Content™ digital organs. Up to 7 stops are packed in a single controller value where each bit controls a single stop.

Controller-No	Controller number. Range is [0, 127]
Bit number	Bit number which state controls the stop. Range is [0, 6]
Upper Limit	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting
Debounce time	Unused by this setting

4.4.4.1.18 Bx Ctrl Change Fixed Value


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device) [New] [Delete]

Device: Any device

Event: Bx Ctrl Change Fixed Value

Channel: 5 Controller-No: 70

Lowest key: 0 Highest key: 0

Off value: 1 On value: 65

Debounce time(ms): 1000

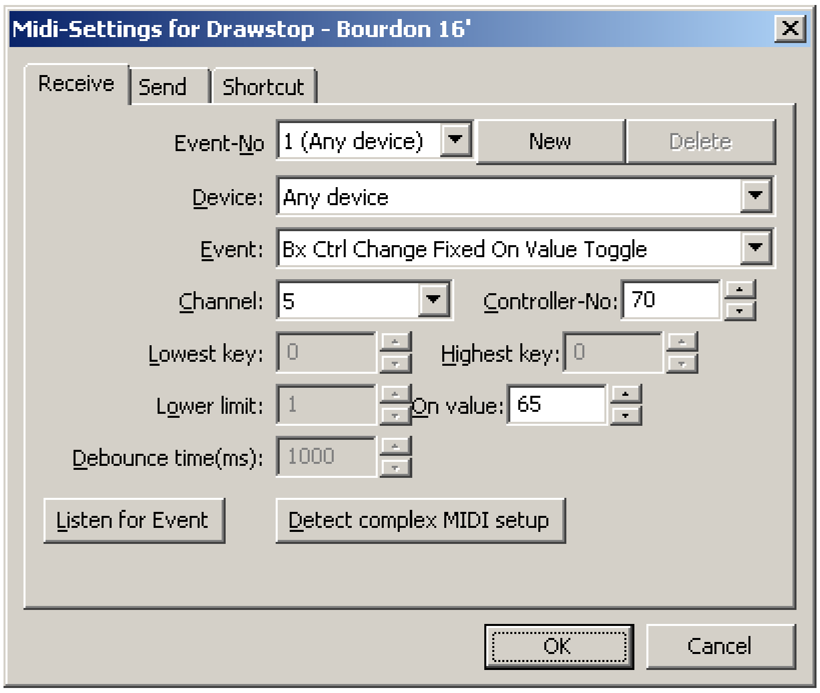
[Listen for Event] [Detect complex MIDI setup]

[OK] [Cancel]

This setting is used by some Alhborn™ digital organs. GrandOrgue detects bit patterns like Stop on = 01xx xxxx ; Stop Off = 00xx xxxx.

Controller-No	Controller number. Range is [0, 127]. Refer to the organ MIDI chart for the actual controller number(s).
Off value	Data value that toggles the button or drawstop off. Range is [0, 63]
On value	Data value that toggles the button or drawstop on. Range is [64, 127]. Always *off value* + 64.
Lowest key	Unused by this setting
Highest key	Unused by this setting
Debounce time	Unused by this setting

4.4.4.1.19 Bx Ctrl Change Fixed On Value Toggle


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)

Device: Any device

Event: Bx Ctrl Change Fixed On Value Toggle

Channel: 5

Controller-No: 70

Lowest key: 0

Highest key: 0

Lower limit: 1

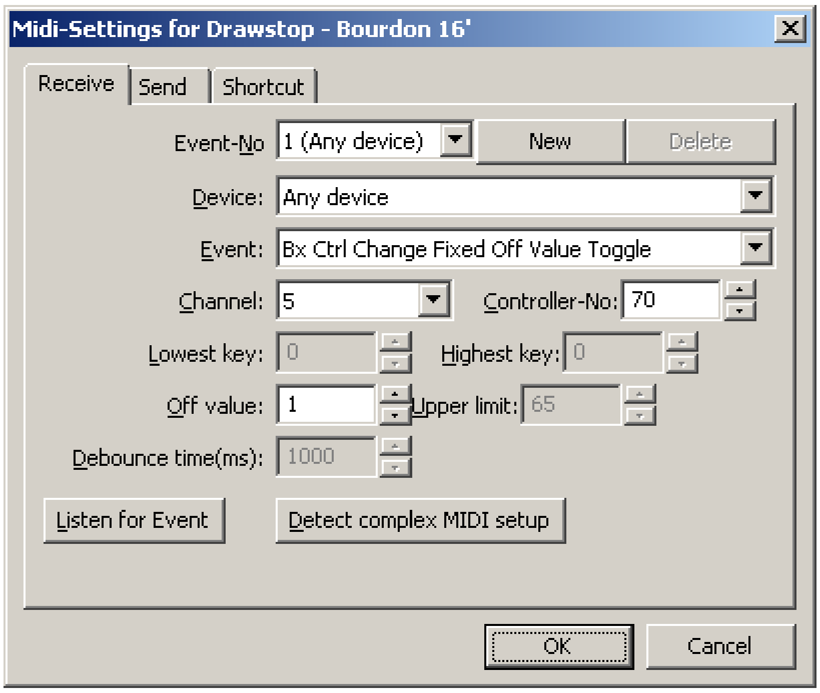
On value: 65

Debounce time(ms): 1000

This setting is used by some Alhborn™ digital organs. GrandOrgue toggles the drawstop or button state when detecting bit patterns like 01xx xxxx.

Controller-No	Controller number. Range is [0, 127]. Refer to the organ MIDI chart for the actual controller number.
On value	Data value that toggles the button or drawstop state. Range is [64, 127].
Debounce time	Minimum time in milliseconds that must elapse between two successive identical control change messages for the drawstop/button to toggle its state.
Off value	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.20 Bx Ctrl Change Fixed Off Value Toggle


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)

Device: Any device

Event: Bx Ctrl Change Fixed Off Value Toggle

Channel: 5

Controller-No: 70

Lowest key: 0

Highest key: 0

Off value: 1

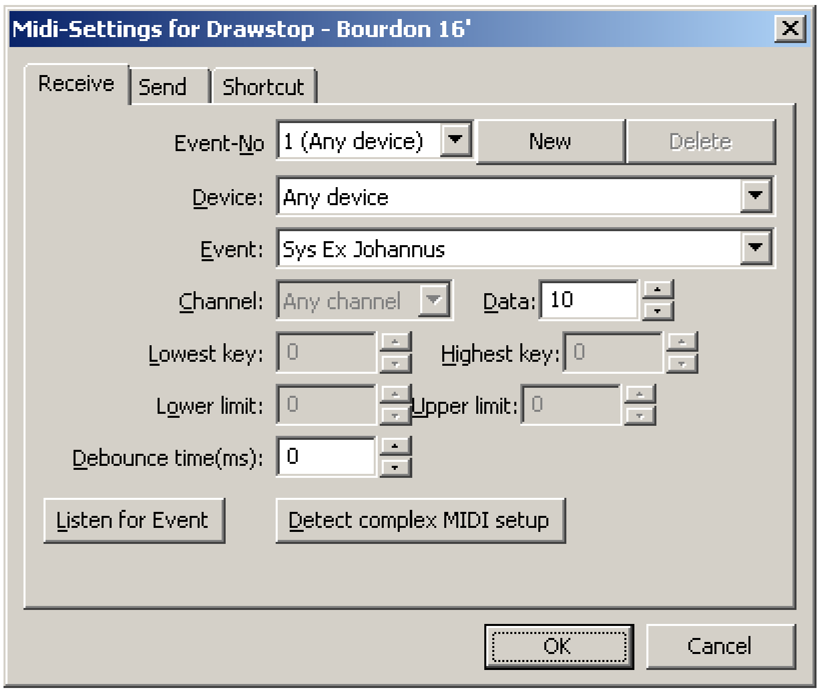
Upper limit: 65

Debounce time(ms): 1000

This setting is used by some Alhborn™ digital organs. GrandOrgue toggles the drawstop or button state when detecting bit patterns like 00xx xxxx.

Controller-No	Controller number. Range is [0, 127]. Refer to the organ MIDI chart for the actual controller number.
Off value	Data value that toggles the button or drawstop state. Range is [0, 63]
Debounce time	Minimum time in milliseconds that must elapse between two successive identical control change messages for the drawstop/button to toggle its state.
On value	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting

4.4.4.1.21 Sys Ex Johannus


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)

Device: Any device

Event: Sys Ex Johannus

Channel: Any channel

Data: 10

Lowest key: 0

Highest key: 0

Lower limit: 0

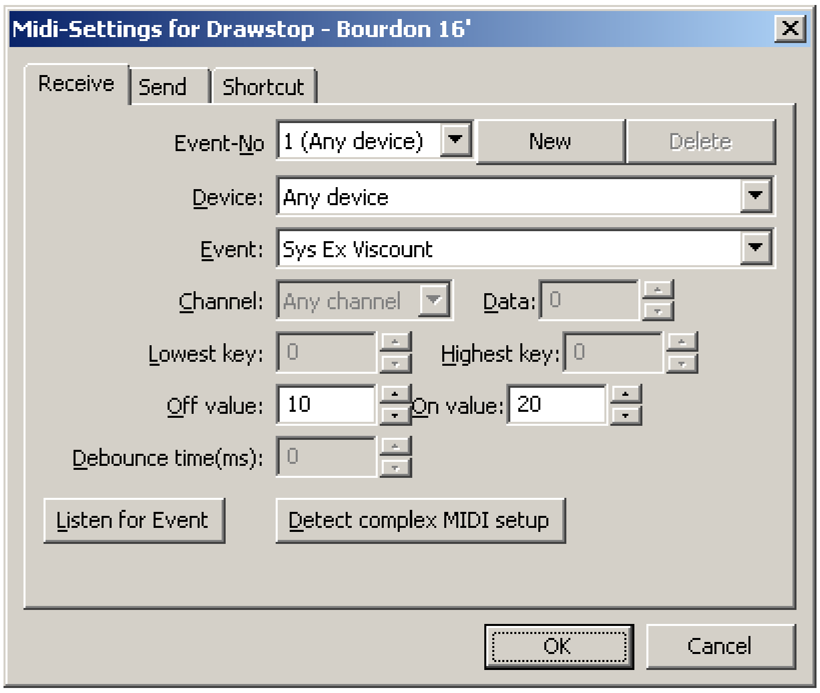
Upper limit: 0

Debounce time(ms): 0

This setting is used by Johannus™ digital organs. The button or drawstop toggles its state whenever two successive identical messages are received.

Data	Data value that toggles the button or drawstop state. Range is [0, 127]
Debounce time	Minimum time in milliseconds that must elapse between two successive identical SysEx messages for the drawstop/button to toggle its state.
Lowest key	Unused by this setting
Highest key	Unused by this setting
Lower limit	Unused by this setting
Upper limit	Unused by this setting

4.4.4.1.22 Sys Ex Viscount

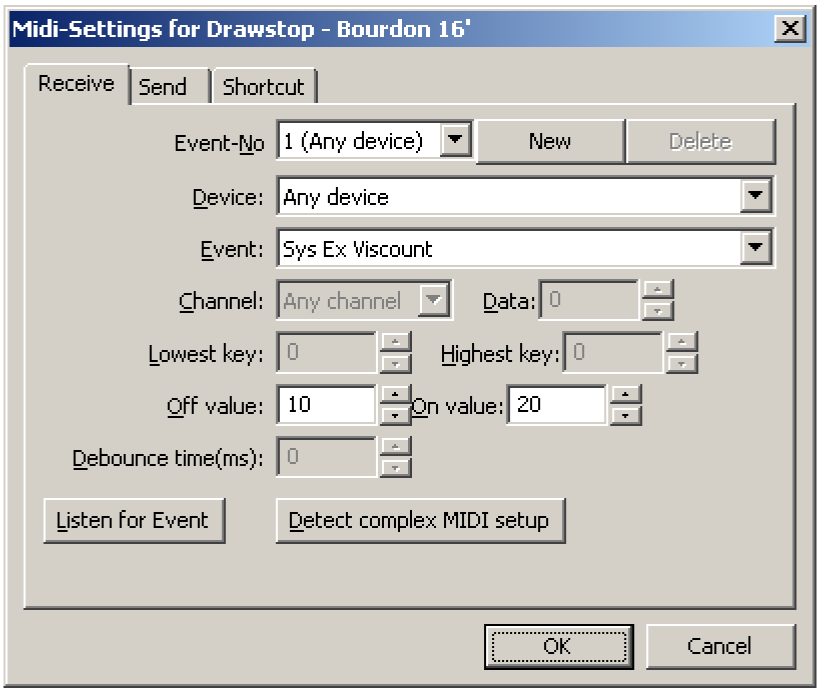

Midi-Settings for Drawstop - Bourdon 16'
Receive
Send
Shortcut
New
Delete
Listen for Event
Detect complex MIDI setup
OK
Cancel

Event-No: 1 (Any device)
Device: Any device
Event: Sys Ex Viscount
Channel: Any channel
Data: 0
Lowest key: 0
Highest key: 0
Off value: 10
On value: 20
Debounce time(ms): 0

This setting is used by Viscount™ digital organs.

Off value	Data value that toggles the button or drawstop state off. Range is [0, 2097151].
On value	Data value that toggles the button or drawstop state on. Range is [0, 2097151].
Data	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting
Debounce time	Unused by this setting

4.4.4.1.23 Sys Ex Viscount Toggle


Midi-Settings for Drawstop - Bourdon 16'
Receive
Send
Shortcut
New
Delete
Listen for Event
Detect complex MIDI setup
OK
Cancel

Event-No: 1 (Any device)
Device: Any device
Event: Sys Ex Viscount
Channel: Any channel
Data: 0
Lowest key: 0
Highest key: 0
Off value: 10
On value: 20
Debounce time(ms): 0

This setting is used by Viscount™ digital organs. The button or drawstop toggles its state whenever two successive identical messages are received.

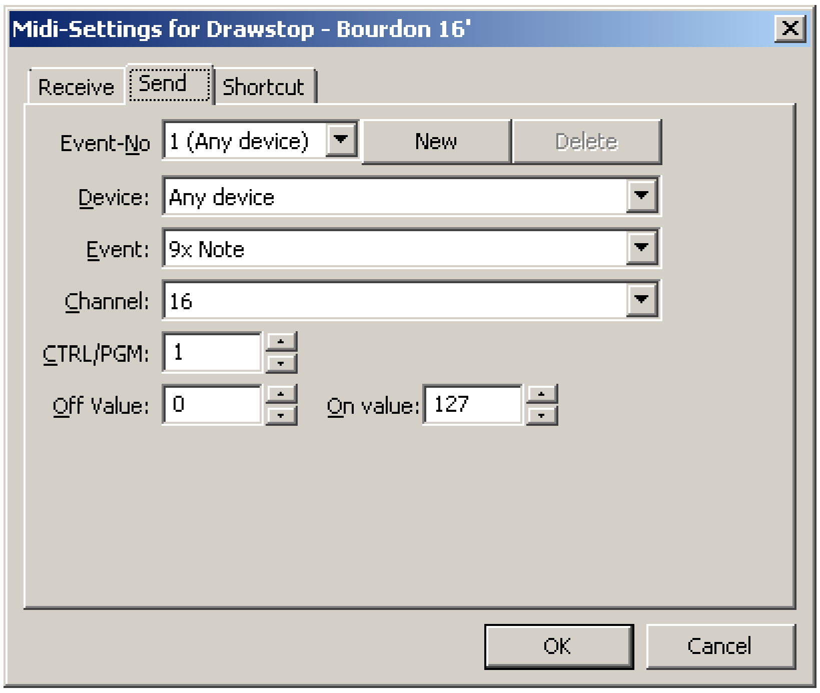
Value	Data value that toggles the button or drawstop state. Range is [0, 2097151].
Debounce time	Minimum time in milliseconds that must elapse between two successive identical SysEx messages for the drawstop/button to toggle its state.
Data	Unused by this setting
Lowest key	Unused by this setting
Highest key	Unused by this setting
Upper limit	Unused by this setting

4.4.4.1.24 Detect complex MIDI Setup

Asks to toggle the drawstop or pushbutton *On* then *Off*. GrandOrgue uses the values it reads to determine the MIDI event and all associated settings.

4.4.4.2 Send

4.4.4.2.1 9x Note


Midi-Settings for Drawstop - Bourdon 16'

Event-No: 1 (Any device)
Device: Any device
Event: 9x Note
Channel: 16
CTRL/PGM: 1
Off Value: 0
On value: 127

Send NoteOn when the drawstop/button toggles to *On* state. Send "NoteOff" when the drawstop/button toggles to *Off* state.

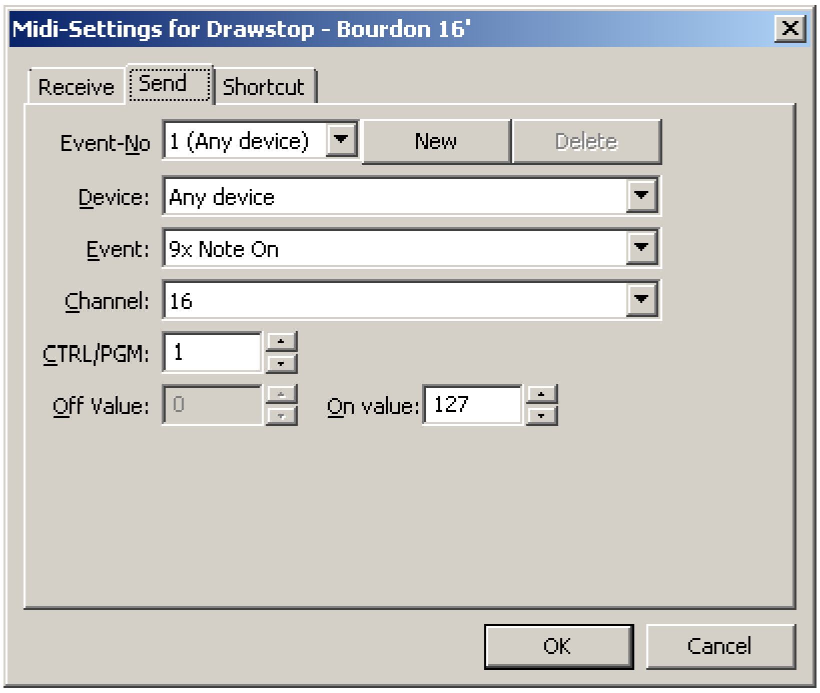
CTRL/PGM Key number.

Off Value Velocity value when drawstop/button goes to Off state.

> **Note**
> Off value **must** be set to zero (0) if a genuine NoteOff is wished.

On Value Velocity value when drawstop/button goes to On state.

4.4.4.2.2 9x Note On


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)

New | Delete

Device: Any device

Event: 9x Note On

Channel: 16

CTRL/PGM: 1

Off Value: 0

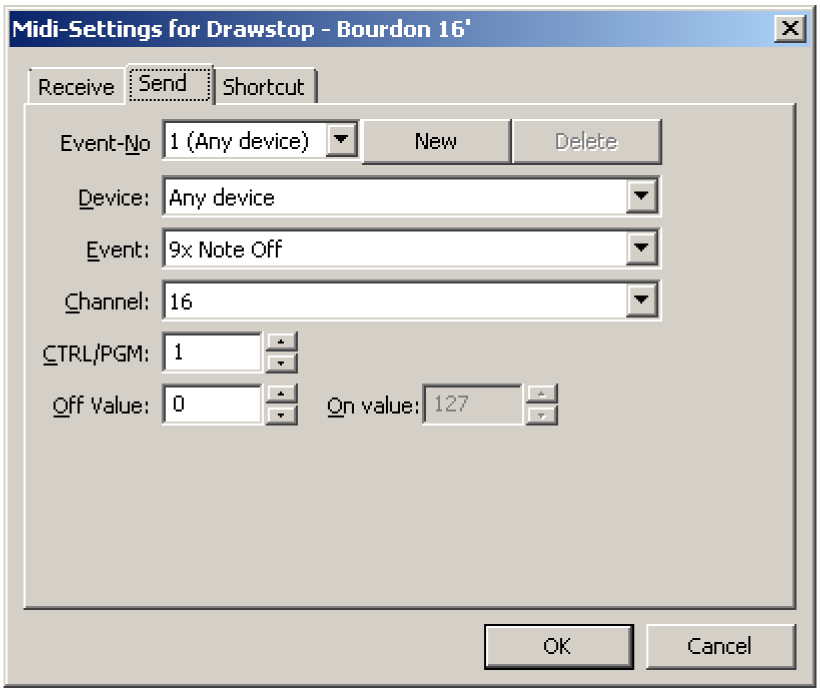
On value: 127

OK | Cancel

Send NoteOn only when the drawstop/button goes to *On* state.

CTRL/PGM	Key number.
Off Value	Unused by this setting.
On Value	Velocity value when drawstop/button goes to On state.

4.4.4.2.3 9x Note Off


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)

New | Delete

Device: Any device

Event: 9x Note Off

Channel: 16

CTRL/PGM: 1

Off Value: 0

On value: 127

OK | Cancel

Send "NoteOff" only when the drawstop/button goes to *Off* state.

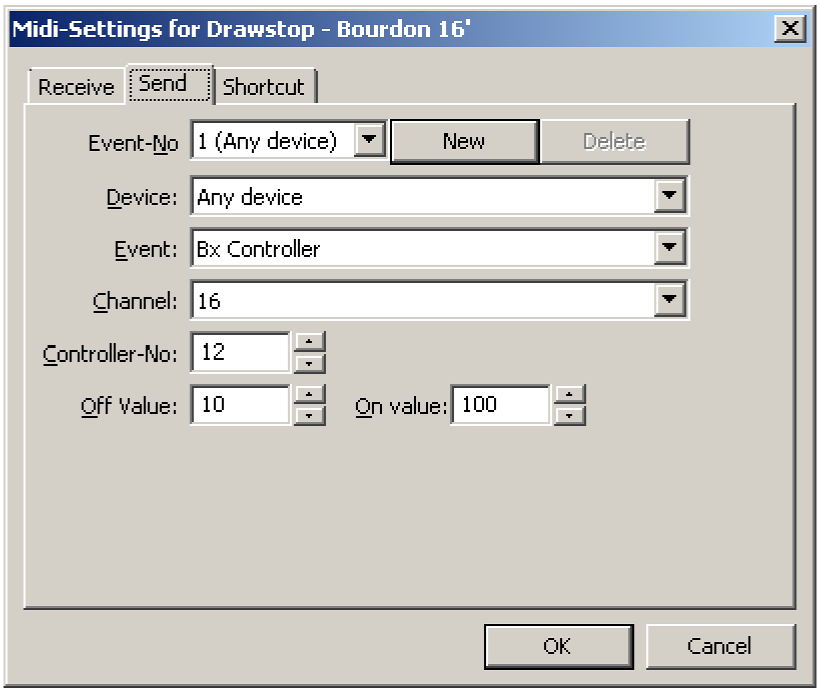
CTRL/PGM	Key number.

Off Value	Velocity value when drawstop/button goes to Off state.

Note
Off value **must** be set to zero (0) if a genuine NoteOff is wished.

On Value	Unused by this setting.

4.4.4.2.4 Bx Controller

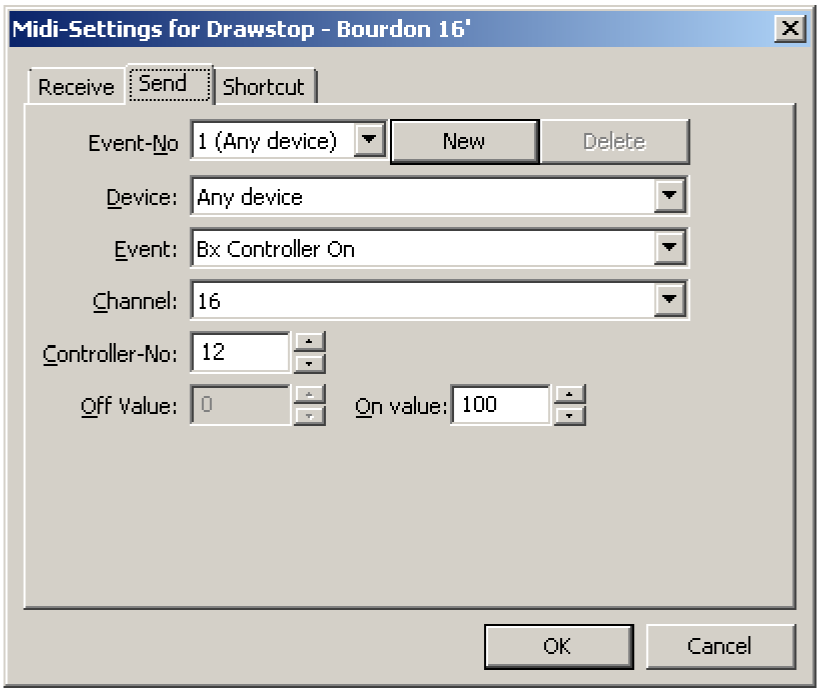


Event-No: 1 (Any device)
Device: Any device
Event: Bx Controller
Channel: 16
Controller-No: 12
Off Value: 10
On value: 100

Send controller values when the drawstop/button changes state. The message sequence is Bn Controller# <value>

Controller-No	Controller number. Range is [0,127].
Off Value	Controller value when drawstop/button goes to Off state.
On Value	Controller value when drawstop/button goes to On state.

4.4.4.2.5 Bx Controller On

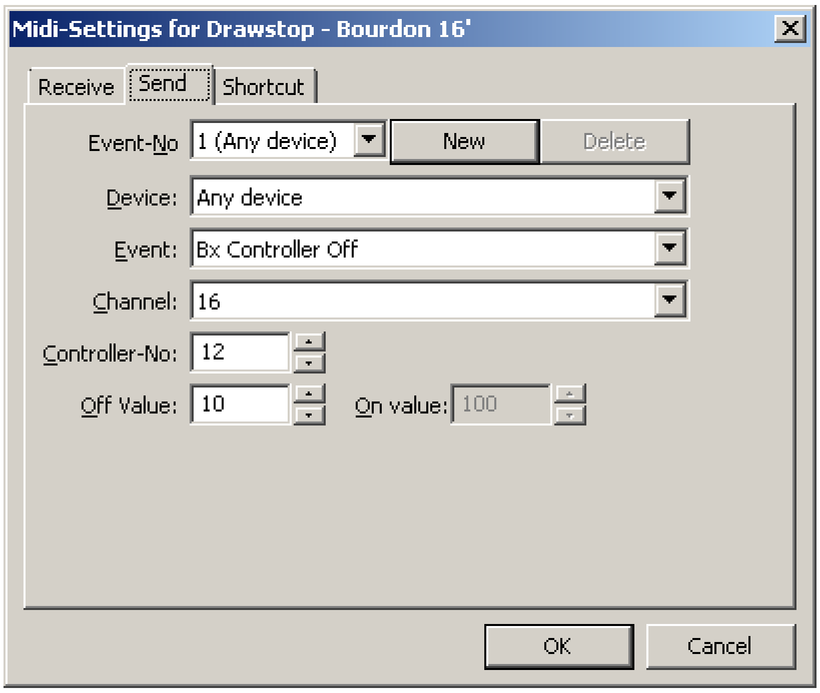


Event-No: 1 (Any device)
Device: Any device
Event: Bx Controller On
Channel: 16
Controller-No: 12
Off Value: 0
On value: 100

Send controller value only when the drawstop/button goes to *On* state.

Controller-No	Controller number. Range is [0,127].
Off Value	Unused by this setting.
On Value	Controller value when drawstop/button goes to On state.

4.4.4.2.6 Bx Controller Off



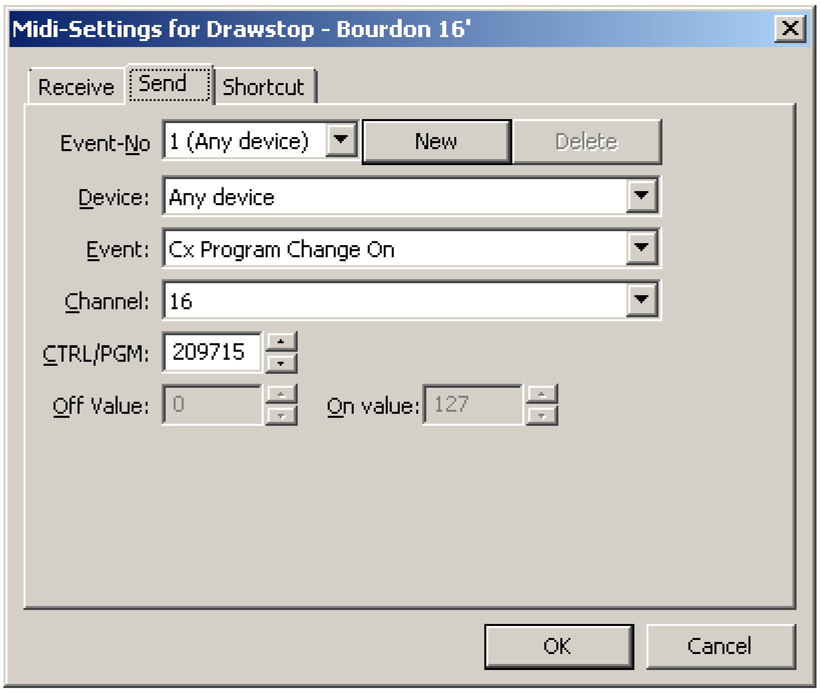
Event-No: 1 (Any device)
Device: Any device
Event: Bx Controller Off
Channel: 16
Controller-No: 12
Off Value: 10
On value: 100

Send controller value only when the drawstop/button goes to *Off* state.

Controller-No	Controller number. Range is [0,127].

Off Value	Controller value when drawstop/button goes to Off state.
On Value	Unused by this setting.

4.4.4.2.7 Cx Program Change On

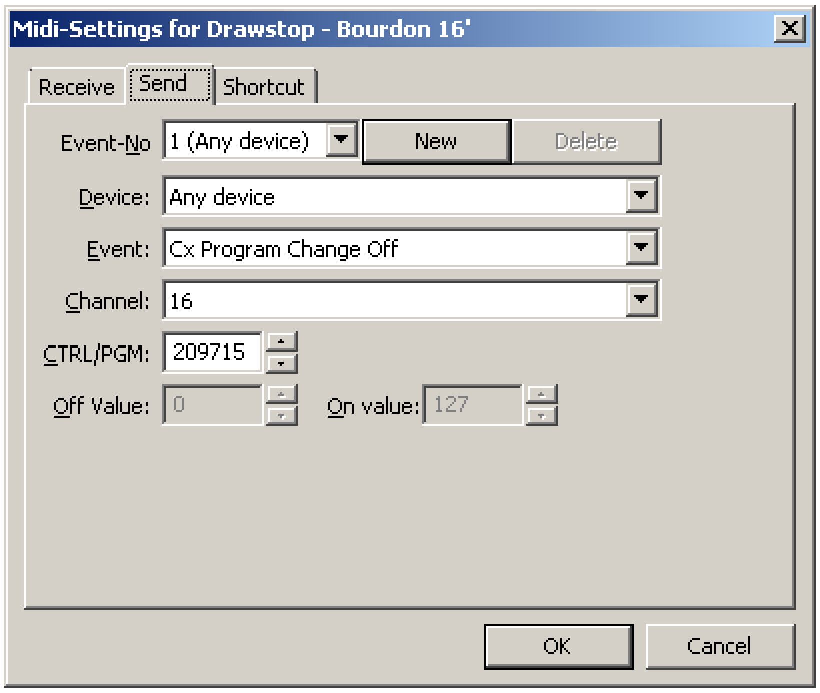

Midi-Settings for Drawstop - Bourdon 16'

Event-No: 1 (Any device)
Device: Any device
Event: Cx Program Change On
Channel: 16
CTRL/PGM: 209715
Off Value: 0
On value: 127

Send Bank Select and Program Change only when the drawstop/button goes to *On* state. The message sequence is: Bn 0x00 bank# MSB ; Bn 0x20 bank# LSB ; Cn prog#.

CTRL/PGM	Program number and optional Bank Select MIDI messages. Range is [0, 2097152] GrandOrgue always sends Bn 0x00 <data1>; Bn 0x20 <data2>; Cn prog#. This field value is the result of *data1* shifted left 14 bits ORed (bitwise) with *data2* shifted left 7 bits ORed (bitwise) with *prog#* + 1 [math: ((data1 << 14) | (data2 << 7) | prog#) + 1].
Off Value	Unused by this setting.
On Value	Unused by this setting.

4.4.4.2.8 Cx Program Change Off

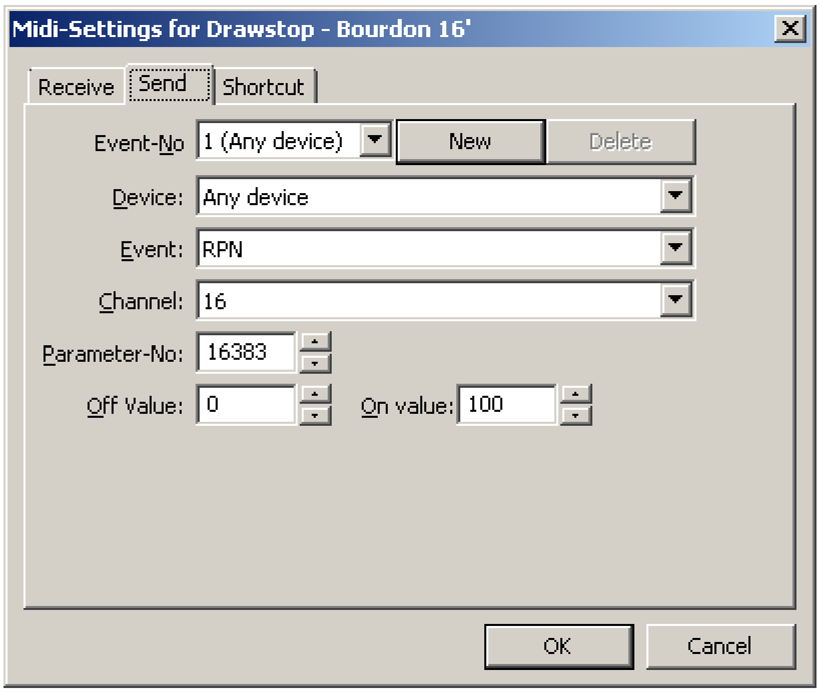

Midi-Settings for Drawstop - Bourdon 16'

Event-No: 1 (Any device)
Device: Any device
Event: Cx Program Change Off
Channel: 16
CTRL/PGM: 209715
Off Value: 0
On value: 127

Send Bank Select and Program Change only when the drawstop/button goes to *Off* state. The message sequence is: Bn 0x00 bank# MSB ; Bn 0x20 bank# LSB ; Cn prog#.

CTRL/PGM	Program number and optional Bank Select MIDI messages. Range is [0, 2097152] GrandOrgue always sends Bn 0x00 <data1>; Bn 0x20 <data2>; Cn prog#. This field value is the result of *data1* shifted left 14 bits ORed (bitwise) with *data2* shifted left 7 bits ORed (bitwise) with *prog#* + 1 [math: ((data1 << 14) | (data2 << 7) | prog#) + 1].
Off Value	Unused by this setting.
On Value	Unused by this setting.

4.4.4.2.9 RPN


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device) New Delete

Device: Any device

Event: RPN

Channel: 16

Parameter-No: 16383

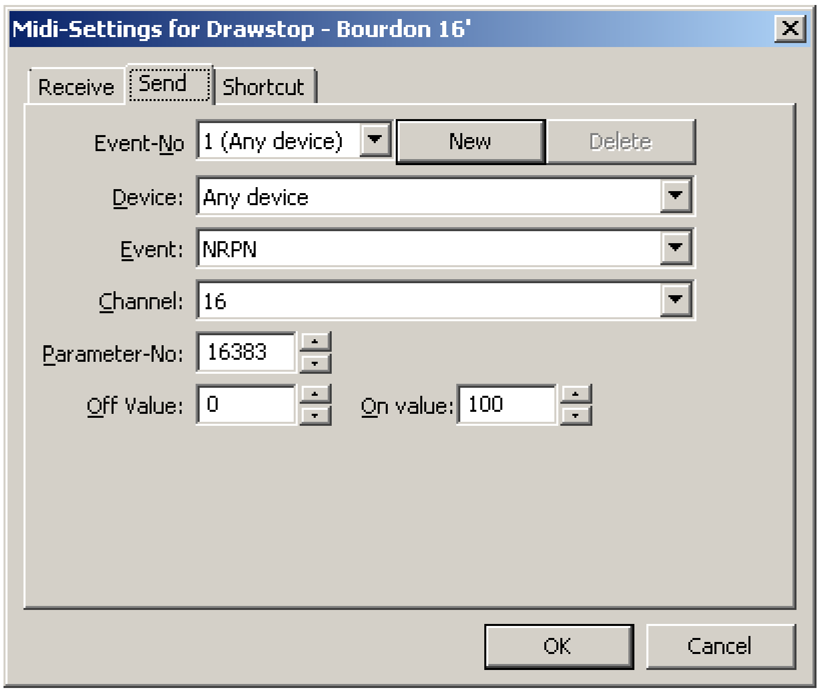
Off Value: 0 On value: 100

OK Cancel

Send RPN values when the drawstop/button changes state. The message sequence is Bn 0x65 parameter# MSB; Bn 0x64 parameter# LSB; Bn 0x06 parameter value.

Parameter-No	Parameter number. Range is [0,16383]. The parameter number displays the result of *parameter# MSB* shifted left 7 bits ORed (bitwise) with *parameter# LSB* [math: (parameter# MSB << 7) | parameter# LSB].
Off Value	Parameter value when drawstop/button goes to Off state.
On Value	Parameter value when drawstop/button goes to On state.

4.4.4.2.10 NRPN


Midi-Settings for Drawstop - Bourdon 16'

Event-No: 1 (Any device)

Device: Any device

Event: NRPN

Channel: 16

Parameter-No: 16383

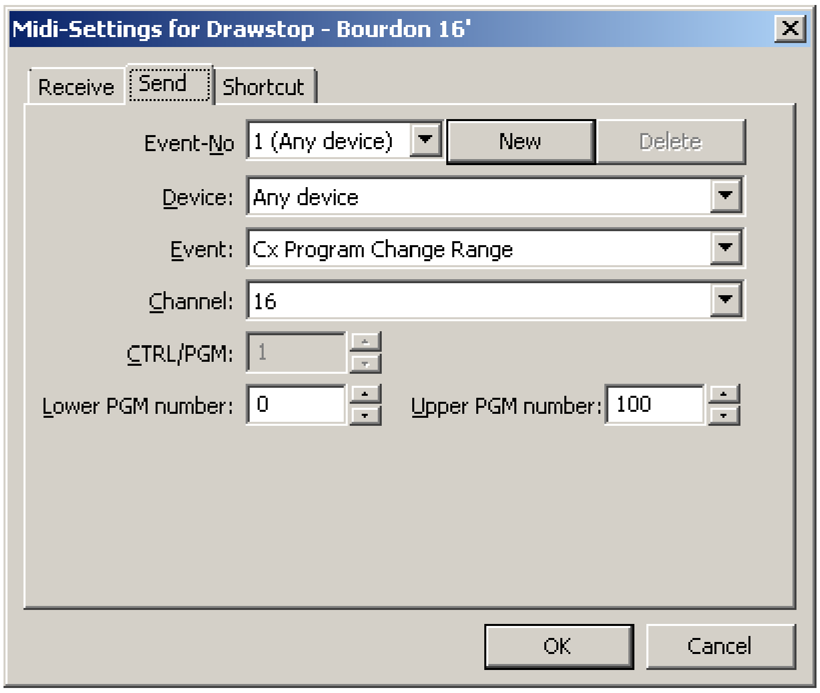
Off Value: 0

On value: 100

Send NRPN values when the drawstop/button changes state. The message sequence is Bn 0x63 parameter# MSB; Bn 0x62 parameter# LSB; Bn 0x06 parameter value.

Parameter-No	Parameter number. Range is [0,16383]. The parameter number displays the result of *parameter# MSB* shifted left 7 bits ORed (bitwise) with *parameter# LSB* [math: (parameter# MSB << 7) | parameter# LSB].
Off Value	Parameter value when drawstop/button goes to Off state.
On Value	Parameter value when drawstop/button goes to On state.

4.4.4.2.11 Cx Program Change Range


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)

Device: Any device

Event: Cx Program Change Range

Channel: 16

CTRL/PGM: 1

Lower PGM number: 0

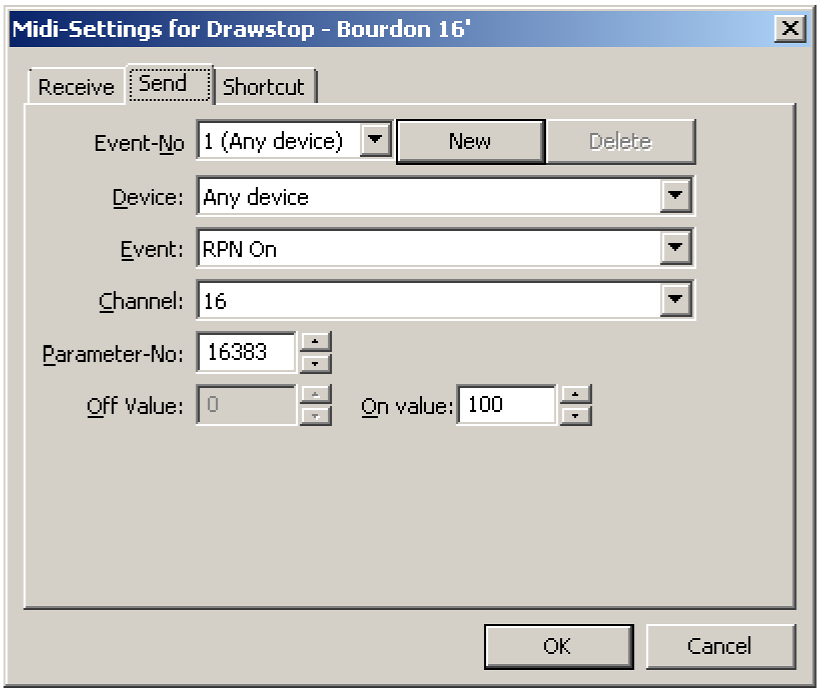
Upper PGM number: 100

Send Bank Select and Program Change when the drawstop/button changes state.

GrandOrgue always sends Bn 0x00 <data1>; Bn 0x20 <data2>; Cn prog#. This field value is the result of *data1* shifted left 14 bits ORed (bitwise) with *data2* shifted left 7 bits ORed (bitwise) with *prog#* + 1 [math: ((data1 << 14) | (data2 << 7) | prog#) + 1].

Lower PGM number	Program number and optional Bank Select MIDI messages sent when the drawstop/button goes to *Off* state. Range is [0, 2097152]
Upper PGM number	Program number and optional Bank Select MIDI messages sent when the drawstop/button goes to *On* state. Range is [0, 2097152]
CTRL/PGM	Unused by this setting.

4.4.4.2.12 RPN On

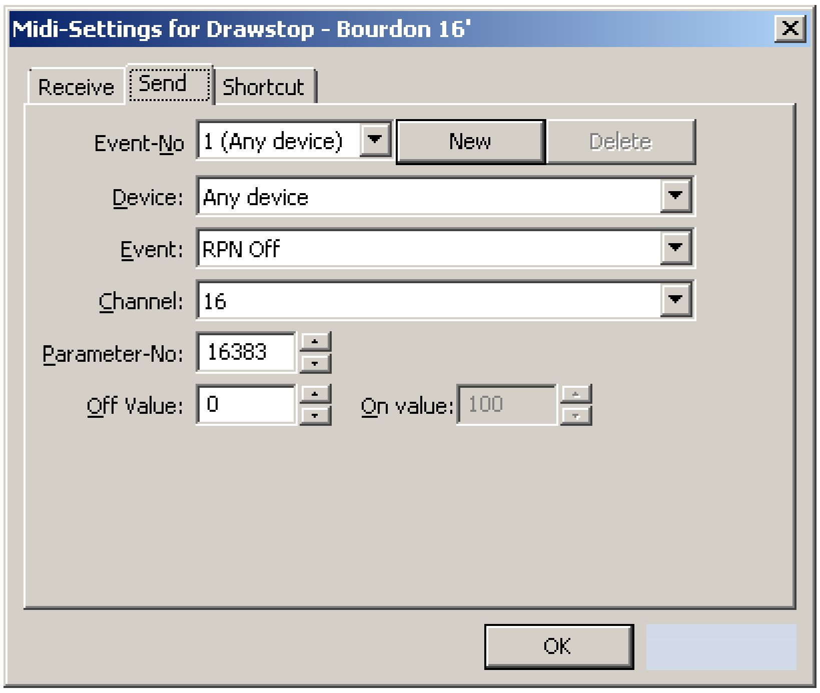


Event-No: 1 (Any device)
Device: Any device
Event: RPN On
Channel: 16
Parameter-No: 16383
Off Value: 0
On value: 100

Send RPN value only when the drawstop/button changes to *On* state. The message sequence is Bn 0x65 parameter# MSB; Bn 0x64 parameter# LSB; Bn 0x06 parameter value.

Parameter-No	Parameter number. Range is [0,16383]. The parameter number displays the result of *parameter# MSB* shifted left 7 bits ORed (bitwise) with *parameter# LSB* [math: (parameter# MSB << 7) | parameter# LSB].
Off Value	Unused by this setting
On Value	Parameter value when drawstop/button goes to On state.

4.4.4.2.13 RPN Off


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device) — New — Delete

Device: Any device

Event: RPN Off

Channel: 16

Parameter-No: 16383

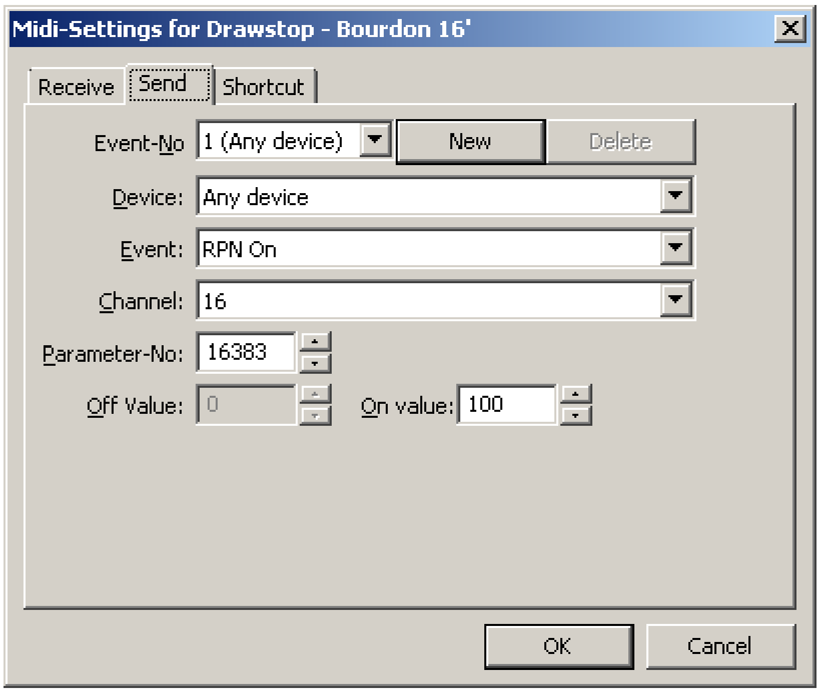
Off Value: 0 — On value: 100

OK

Send RPN value only when the drawstop/button changes to *Off* state. The message sequence is Bn 0x65 parameter# MSB; Bn 0x64 parameter# LSB; Bn 0x06 parameter value.

Parameter-No	Parameter number. Range is [0,16383]. The parameter number displays the result of *parameter# MSB* shifted left 7 bits ORed (bitwise) with *parameter# LSB* [math: (parameter# MSB << 7) | parameter# LSB].
Off Value	Parameter value when drawstop/button goes to Off state.
On Value	Unused by this setting

4.4.4.2.14 NRPN On


Midi-Settings for Drawstop - Bourdon 16'

Receive | Send | Shortcut

Event-No: 1 (Any device)

Device: Any device

Event: RPN On

Channel: 16

Parameter-No: 16383

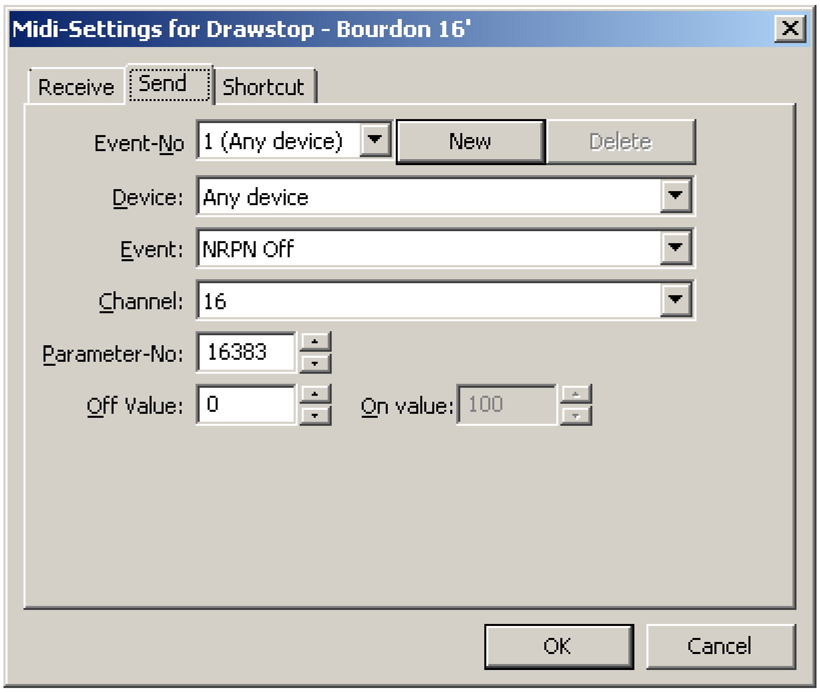
Off Value: 0

On value: 100

Send NRPN value only when the drawstop/button changes to *On* state. The message sequence is Bn 0x63 parameter# MSB; Bn 0x62 parameter# LSB; Bn 0x06 parameter value.

Parameter-No	Parameter number. Range is [0,16383]. The parameter number displays the result of *parameter# MSB* shifted left 7 bits ORed (bitwise) with *parameter# LSB* [math: (parameter# MSB << 7) | parameter# LSB].
Off Value	Unused by this setting
On Value	Parameter value when drawstop/button goes to On state.

4.4.4.2.15 NRPN Off



Event-No: 1 (Any device)
Device: Any device
Event: NRPN Off
Channel: 16
Parameter-No: 16383
Off Value: 0
On value: 100

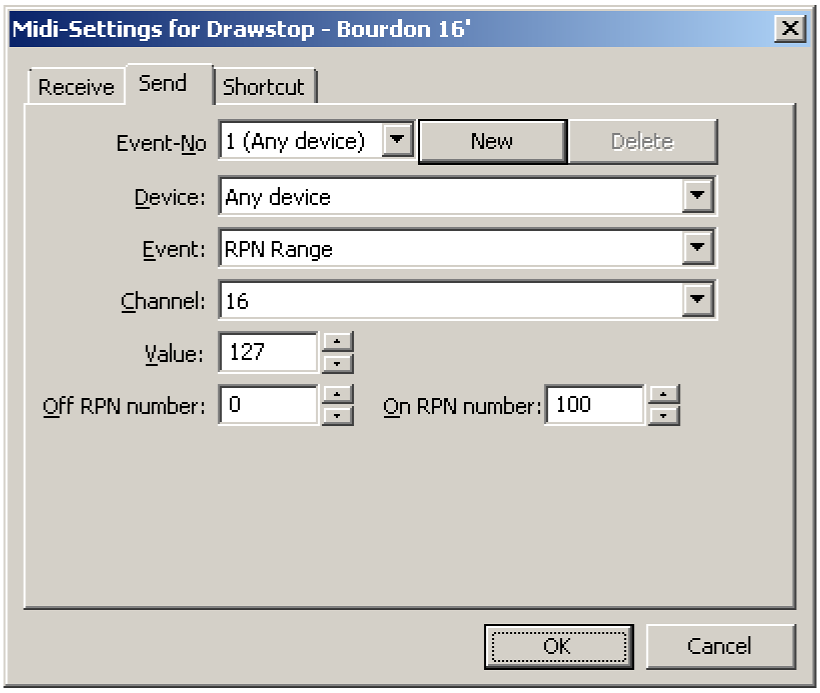
Send NRPN value only when the drawstop/button changes to *Off* state. The message sequence is Bn 0x63 parameter# MSB; Bn 0x62 parameter# LSB; Bn 0x06 parameter value.

Parameter-No Parameter number. Range is [0,16383].
The parameter number displays the result of *parameter# MSB* shifted left 7 bits ORed (bitwise) with *parameter# LSB* [math: (parameter# MSB << 7) | parameter# LSB].

Off Value Parameter value when drawstop/button goes to Off state.

On Value Unused by this setting

4.4.4.2.16 RPN Range


Midi-Settings for Drawstop - Bourdon 16'

Event-No: 1 (Any device)

Device: Any device

Event: RPN Range

Channel: 16

Value: 127

Off RPN number: 0

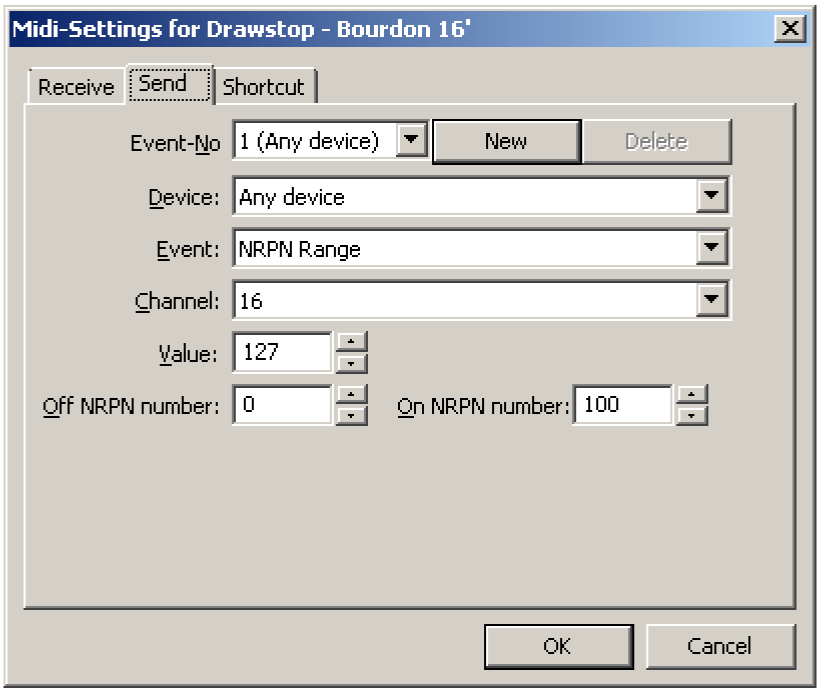
On RPN number: 100

Send RPN when the drawstop/button changes state.

GrandOrgue always sends Bn 0x65 parameter# MSB; Bn 0x64 parameter# LSB; Bn 0x06 parameter value. The parameter number displays the result of *parameter# MSB* shifted left 7 bits ORed (bitwise) with *parameter# LSB* [math: (parameter# MSB << 7) | parameter# LSB].

Off RPN number	RPN number MIDI messages sent when the drawstop/button goes to *Off* state. Range is [0, 16383]
On RPN number	RPN number MIDI messages sent when the drawstop/button goes to *On* state. Range is [0, 16383]
Value	Parameter value. Range is [0, 127]

4.4.4.2.17 NRPN Range



Event-No: 1 (Any device)
Device: Any device
Event: NRPN Range
Channel: 16
Value: 127
Off NRPN number: 0
On NRPN number: 100

Send NRPN when the drawstop/button changes state.

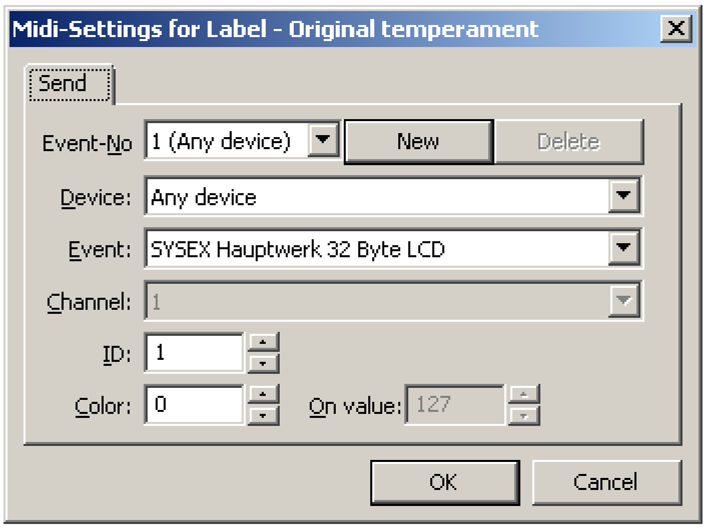
GrandOrgue always sends Bn 0x63 parameter# MSB; Bn 0x62 parameter# LSB; Bn 0x06 parameter value. The parameter number displays the result of *parameter# MSB* shifted left 7 bits ORed (bitwise) with *parameter# LSB* [math: (parameter# MSB << 7) | parameter# LSB].

Off NRPN number	NRPN number MIDI messages sent when the drawstop/button goes to *Off* state. Range is [0, 16383]
On NRPN number	NRPN number MIDI messages sent when the drawstop/button goes to *On* state. Range is [0, 16383]
Value	Parameter value. Range is [0, 127]

4.4.5 Label

GrandOrgue supports "active" labels. These are labels which display variable contents, such as the step number in the Crescendo panel, the temperament name in the Master Controls panel and many others. Labels can be configured only for sending data to a Midi LCD device. GrandOrgue sends the label contents following the Hauptwerk™ standards.

4.4.5.1 Hauptwerk SYSEX 32 byte LCD



Midi-Settings for Label - Original temperament

Send

Event-No: 1 (Any device)

Device: Any device

Event: SYSEX Hauptwerk 32 Byte LCD

Channel: 1

ID: 1

Color: 0

On value: 127

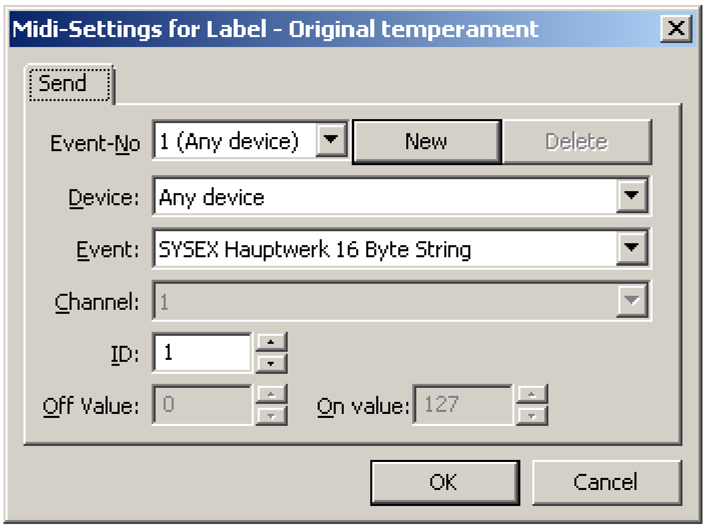
This setting drives a Hauptwerk™ compatible LCD display, including the RGB LED color. Up to 32 characters are sent to the LCD display.

Channel Always disabled, since SYSEX is independent of any channel.

ID Sets the display's ID number. Range is [0, 127].

Color Sets the LED's color. Range is [0, 127].

4.4.5.2 Hauptwerk SYSEX 16 byte string



Midi-Settings for Label - Original temperament

Send

Event-No: 1 (Any device)

Device: Any device

Event: SYSEX Hauptwerk 16 Byte String

Channel: 1

ID: 1

Off Value: 0

On value: 127

This setting sends character strings to a Hauptwerk™ compatible LCD display. The RGB LED (if any) color is ignored. Up to 16 characters are sent to the LCD display.

Channel Always disabled, since SYSEX is independent of any channel.

ID Sets the display's ID number. Range is [0, 127].

Chapter 5

Audio and Organ settings overview

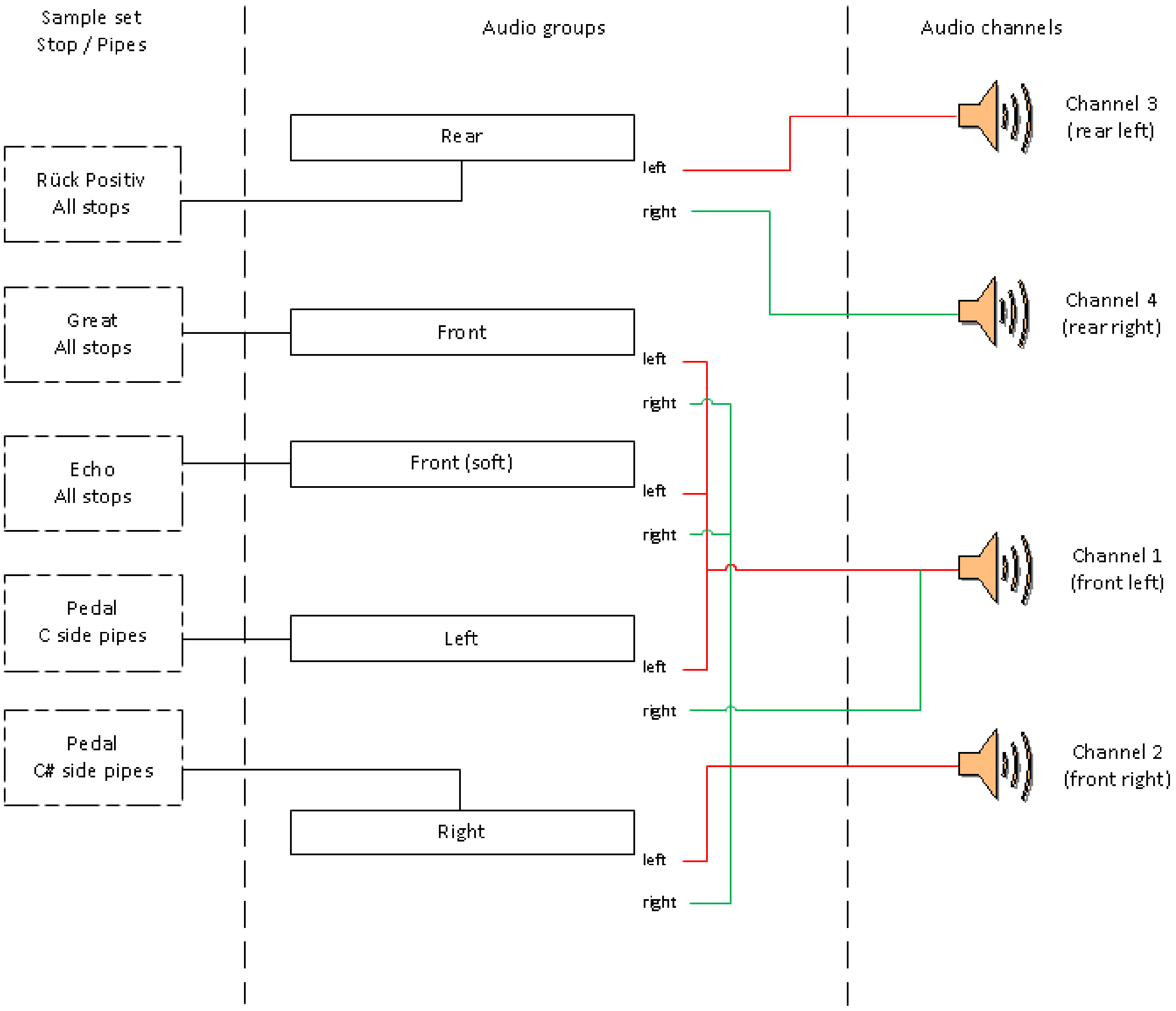


Figure 5.1: Audio & Organ settings overview

This schema describes the close relations that exist between the real-world audio hardware capablities and the sample set disposition as provided by the sample set designer. For illustration, the free sampleset of Bureå Church is customized to build a totally different spatial disposition. Here are the assumptions:

- The organ has a main case and a separate choir (RückPositiv) case in the organist's back
- The console is embedded facing the main case, therefore the main case sounds "front" and the choir case sounds "rear"
- The organ has 2 pedal turrets on the far left and right sides of the main case
- The organ has an echo keyboard in the basement of the main case
- The computer's audio device is 4.0 surround (front left and right speakers, rear left and right speakers)

Settings in the Midi & Audio Settings tab describe the physical environment, while settings in the Organ settings dialog describe how the different parts of the virtual organ map to the real-world audio outputs.

Channel:	a real-world channel of the audio device (e.g. left and right for a stereo board). GrandOrgue channel names are always **Channel 1** to **Channel n**, *n* being the number of channels available on the audio device (stereo:2, surround 4.0:4, and so on). Only trial and error can determine which GrandOrgue channel is linked to which audio device physical channel.
Audio group:	a name holder used to design logical blocks within the audio system (e.g. "front speakers", "rear speakers", "subwoofer", ...). It is possible to create as many audio groups as needed.

Caution

If an audiogroup is not used in the loaded sample set, it does add processing overhead, **thus reducing polyphony**, so adding unnecessary audio groups is not recommended.

In this example, the decision was made to

- sound the Choir (Svällverk) through the rear speakers, full volume
- sound the Great (Huvudverk) through the front speakers, full volume
- sound the Echo (Bröstverk) through the front speakers, softened volume
- sound the C side of the Pedal through the left front speaker
- sound the C# side of the Pedal through the right front speaker

This is done by linking audio groups to channels in the Midi & Audio Settings dialog and spreading pipes over the audio groups in the Organ Settings dialog.

Chapter 6

Midi & Audio settings

This tabbed dialog is displayed when the *Audio/Midi Settings* menu item is chosen in the *Audio/Midi* menu.

It manages "system-wide" GrandOrgue options. In other terms, these options apply to any loaded sampleset.

These options are located at different places according to the platform.

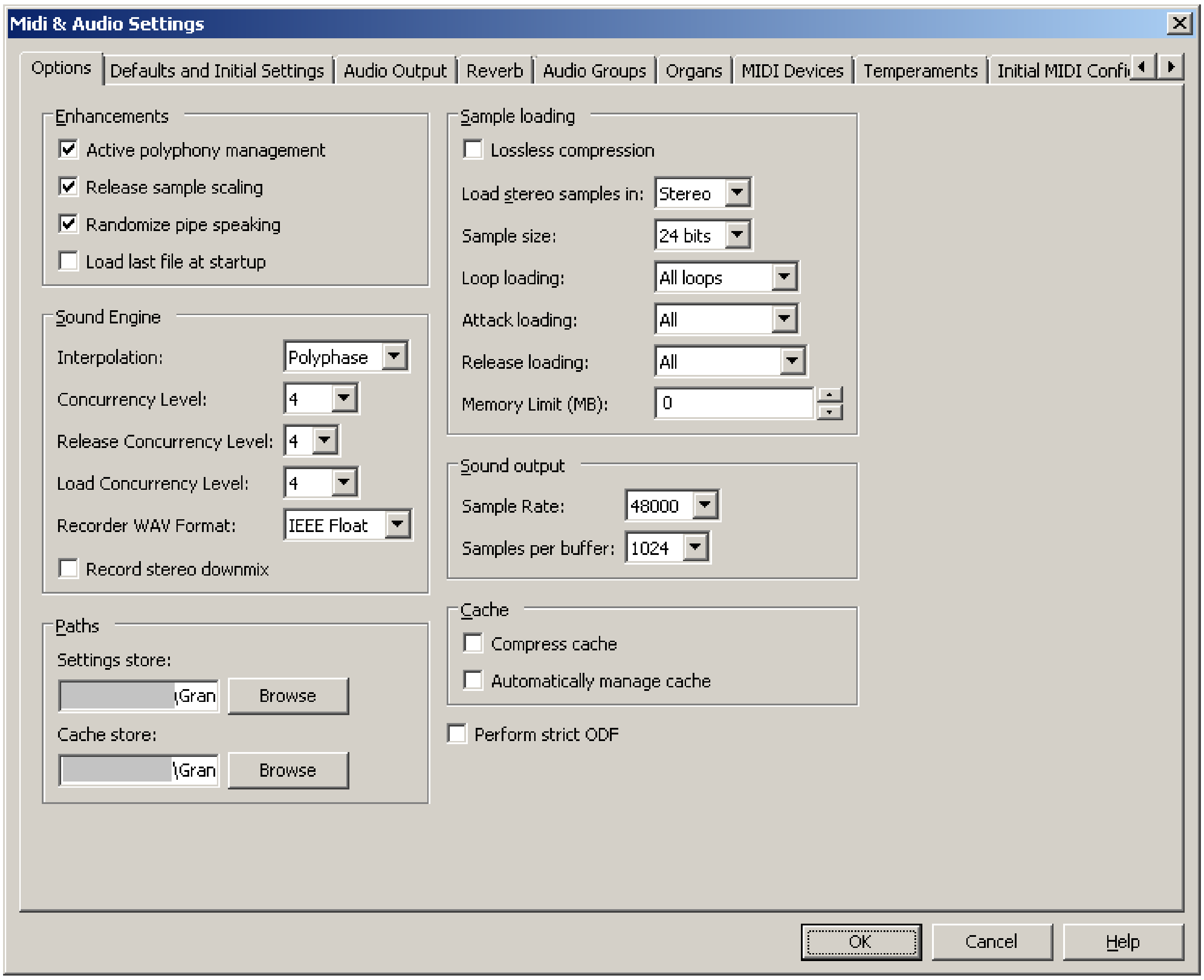
Windows:	registry key HKEY_CURRENT_USER\Software\Our Organ\GrandOrgue and %APPDATA%\GrandOrgueConfig file
Linux:	$HOME/.GrandOrgue and $HOME/GrandOrgueConfig
Mac:	$HOME/Library/Preferences/.GrandOrgue and $HOME/Library/Preferences/GrandOrgue

Note

When launched with an instance name (*GrandOrgue -i **instanceName***), the name *GrandOrgue* becomes *GrandOrgue-**instanceName***, the name *GrandOrgueConfig* becomes ***GrandOrgueConfig-instanceName***

The user can reset settings to default values by deleting the "options files" according to the platform.

6.1 Options tab



Enhancements
- [✓] Active polyphony management
- [✓] Release sample scaling
- [✓] Randomize pipe speaking
- [ ] Load last file at startup

Sound Engine

Interpolation:	Polyphase
Concurrency Level:	4
Release Concurrency Level:	4
Load Concurrency Level:	4
Recorder WAV Format:	IEEE Float

- [ ] Record stereo downmix

Paths

Settings store: \Gran

Cache store: \Gran

Sample loading
- [ ] Lossless compression

Load stereo samples in:	Stereo
Sample size:	24 bits
Loop loading:	All loops
Attack loading:	All
Release loading:	All
Memory Limit (MB):	0

Sound output

Sample Rate:	48000
Samples per buffer:	1024

Cache
- [ ] Compress cache
- [ ] Automatically manage cache

- [ ] Perform strict ODF

Figure 6.1: Midi & Audio Settings / Options

6.1.1 Enhancements frame

6.1.1.1 Active polyphony management

When polyphony reaches 3/4 of the current maximum value, release samples (reverberation tails) are faded out in order to conserve polyphony.

Memory No impact

Polyphony Conserved

Load time No impact

6.1.1.2 Release Sample Scaling

For those sample sets with long reverberation tails ("wet"), enabling this option will fade out the release sample if the note is played staccato. It will have little to no effect on "dry" sample sets.

Memory No impact

Polyphony No impact

Load time No impact

6.1.1.3 Randomize Pipe Speaking

This will apply a very small random detuning before each pipe begins to speak. The detuning is small enough so that the ear can't appreciate the tuning difference and large enough so that it approximates the tiny interferences (due to physical location) that occur when playing multiple real pipes in an organ.

Memory No impact

Polyphony No impact

Load time No impact

6.1.1.4 Load last file at startup

This will trigger the automatic load of the last used sample set.

Memory No impact

Polyphony No impact

Load time No impact

6.1.2 Sound engine frame

6.1.2.1 Interpolation

This dropdown controls the method used to interpolate the waveform when resampling the samples.

Polyphase This method gives better audio results at the expense of higher CPU load.

Linear This method lowers CPU load at he expense of audio quality. It is the only usable option when lossless compression is enabled, as polyphase interpolation is not yet implemented with lossless compression.

The assertions about audio quality should be used as guidelines only. As a rule of thumb, polyphase gets better treble, linear gets better bass, and this is most noticeable in sample sets that use interpolation extensively (e.g. retuning a detuned organ). Audio quality is highly subjective, so the user is strongly encouraged to listen for himself and retain the setting that suits him better.

Memory No impact

Polyphony Linear gives more polyphony

Load time No impact

6.1.2.2 Concurrency Level

This number states how many threads GrandOrgue creates to spread the load. Cores number is the recommended choice for computers without Hyperthreading. Less doesn't use the whole computer. More wastes resources while managing overhead.

With Hyperthreading enabled, the CPU load seems more evenly spread among virtual cores.

Memory No impact

Polyphony Raising the number of cores raises the polyphony attained before the sound starts to break.

Load time No impact

6.1.2.3 Release Concurrency Level

This setting has the same role as Concurrency Level, and is dedicated to release processing. It works around some limitations in the current code, and may be removed in a future release. The same value as concurrency is recommended.

Memory Same as concurrency

Polyphony Same as concurrency

Load time Same as concurrency

6.1.2.4 Load Concurrency Level

This number states how many threads GrandOrgue creates to load samples in memory. It has **NO** effect when loadind samples from cache.

Higher speed-up loading while reducing the available memory for samples. A zero (0) value means classic load.

6.1.2.5 Recorder WAV format

This selects the format used by the recorder invoked by selecting Record. The supported formats are: 8 bit, 16 bit, 24 bit PCM and 32 bit IEEE float.

Memory No impact

Polyphony No impact

Load time No impact

6.1.2.6 Record stereo downmix

This textbox enables GrandOrgue to record multi-channel audio in simple stereo. This option is disabled by default, as it increases CPU usage even if the audio recorder is off.

When stereo downmis is disabled, all audio channels are recorded in the WAV file.

Memory No impact

Polyphony Slightly - like an additional audio interface

Load time No impact

6.1.3 Paths frame

6.1.3.1 Settings store

This selects the directory where preset files are stored. The *Browse* button opens a file browser where the desired directory can be chosen.

6.1.3.2 Cache store

This selects the directory where cache files are stored. The *Browse* button opens a file browser where the desired directory can be chosen.

6.1.4 Sample loading frame

6.1.4.1 Lossless compression

Uses a lossless algorithm (i.e. the original samples can be perfectly reconstructed) so that RAM requirements are lessened. Particularly useful for large sample sets.

Memory Variable savings up to 40%

Polyphony Approx. 10% lower

Load time Approx. 20% slower. Only affects load without cache - as the compressed data is cached.

6.1.4.2 Load stereo samples

Selects whether stereo samples are loaded in Mono or Stereo

If set to Mono, all samples are loaded in Mono. When compression is disabled, this will save 50%, but when compression is enabled, much less will be saved because the compression is much less effective on monoaural inputs.

This setting can be set to don’t load, if you want GrandOrgue to avoid loading any samples. Via Organ Settings, it is still possible to load only specific parts of the organ.

Memory Variable savings up to 50%

Polyphony Slight impact, as loading in Mono saves a few calculations per sample.

Load time No impact

6.1.4.3 Sample size

Selects whether the samples are loaded with 8 or 12 or 16 or 24 bits precision.

Memory High impact

Polyphony The polyphony will increase with lower bit sizes and decrease with higher bit sizes.

Load time Slight impact

6.1.4.4 Loop loading

Selects which loop(s) are loaded if the samples have multiple loops.

First loop Load only the first loop.

Longest loop Load only the longest loop found in the sample. Longer loops usually feel more lively.

All loops Load all loops found in the sample. While playing, the loops are cycled using a round-robin scheme.

Memory *First loop* less than *Longest loop* less than *All loops*

Load time Increases with the amount of data loaded

Polyphony No impact

6.1.4.5 Attack loading

The sample set creator can provide multiple separate attack files.This selects which attack(s) are loaded.

All Load all attack files.

Single attack Load only the "best" provided file.

Memory *Single attack* less than *All*

Polyphony No impact

Load time Increases with the amount of data loaded

6.1.4.6 Release loading

The sample set creator can provide multiple separate release files.This selects which release(s) are loaded.

All Load all release files. The sound engine selects which release to play according to how much time the sustain sample was held. For example, it usually selects the shortest release sample while playing staccato.

Single release Load only the "best" provided release file.

Memory Same as Attack loading

Polyphony No impact

Load time Same as Attack loading

6.1.4.7 Memory limit

Chooses the size of the memory cache where samples are loaded. This is especially useful when running the 32 bits Windows build on 64 bits platforms to avoid a crash while loading a big sample set.

If this parameter is set to zero, it tries to load as much data as possible.

Caution
In this case, GrandOrgue keeps loading until the OS fails to provide memory. It can provide large amounts of swap space with expectable adverse impact on performances.

6.1.5 Sound output frame

6.1.5.1 Sample rate

Selects the output sample rate. Allowed values are 44100, 48000 and 96000 Hz.

The samplerate should match both the configured samplerate in the audio interface and the samplerate of the recorded samples to avoid resampling. If not, resampling can occur in all layers of the audio stack, and audio quality can suffer.

6.1.5.2 Samples per buffer

Allowed values range from 16 to 1024 by increment of 16.

This parameter sets the output buffer size. Larger values usually reduce sound artifacts at the expense of latency.

6.1.6 Cache frame

6.1.6.1 Compress cache

Selects whether the disk cache must be compressed when created or updated.

Memory No impact

Polyphony No impact

Load time Loading from an uncompressed cache is I/O bound, while loading from a compressed cache requires more CPU. With a slow disk + fast CPU, compressed might be better.

This feature is closely related to the hardware capacities, so the user is encouraged to test for himself and retain the best setting for his computer.

6.1.6.2 Automatically manage cache

Selects whether the cache must be automatically created or updated when the sample set is loaded.

6.1.7 Perform strict ODF

Organ Definition File (ODF) syntax has much changed since the days of Hauptwerk™ version 1. Some Hauptwerk™ 1 keywords are not used by GrandOrgue and are reported as warning in the pop-up message window.

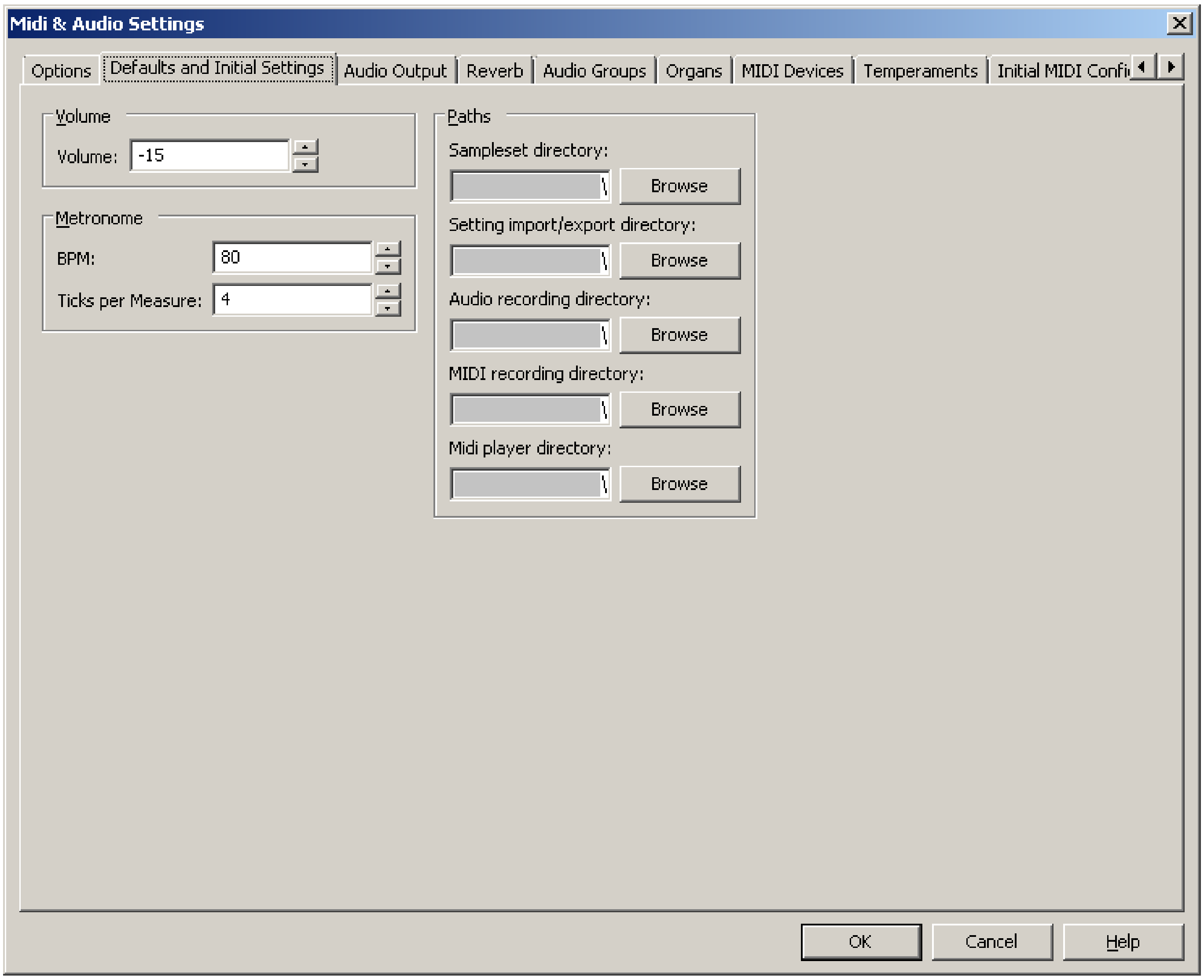
Introducing ODF warnings to GrandOrgue is a two-step development process:

1. A new warning is added to the ODF parsing utilities. It is displayed only in strict ODF mode, to give samplesets designers time to remove the new warning from their Organ Definition Files, without disturbing users.
2. After a suitable number of months has elapsed, the warning is moved to normal mode and is always displayed.

This checkbox enables strict ODF mode.

Note
Sampleset designers are **strongly** encouraged to update their Organ Definition Files in order to remove warnings and improve user experience.
Likewise, GrandOrgue users are **strongly** encouraged to report new warnings to the sampleset designer.

6.2 Defaults and Initial Settings tab


Midi & Audio Settings

Options | Defaults and Initial Settings | Audio Output | Reverb | Audio Groups | Organs | MIDI Devices | Temperaments | Initial MIDI Confi

Volume

Volume: -15

Metronome

BPM: 80

Ticks per Measure: 4

Paths

Sampleset directory: \ Browse

Setting import/export directory: \ Browse

Audio recording directory: \ Browse

MIDI recording directory: \ Browse

Midi player directory: \ Browse

OK Cancel Help

Figure 6.2: Defaults and Initial Settings tab

This dialog manages default and initial values for settings which are controlled in various panels or dialogs. The values defined in this tab are stored in the global configuration file.

6.2.1 Volume

This spinner provides the default volume value. Initialized to -15 on the very first run of a GrandOrgue instance. This value is saved to settings files.

6.2.2 Metronome

These spinners provide default values for the number of beats per minute (BPM) and the number of beats per bar in the metronome panel. Initialized to 80 beats per minute and 4 beats per bar on the very first run of a GrandOrgue instance. They are saved to settings files.

6.2.3 Paths

These text boxes provide default paths for various storage areas. They are initialized to paths relative to the user's home folder, and their names are localized. The *Browse* button located besides each text box allows to change the value.

6.2.3.1 Windows Paths

Sampleset directory	%HOMEDRIVE%\%HOMEDIR%\Documents\GrandOrgue\Organs
Setting inport/export directory	%HOMEDRIVE%\%HOMEDIR%\Documents\GrandOrgue\Settings
Audio recording directory	%HOMEDRIVE%\%HOMEDIR%\Documents\GrandOrgue\Audio recordings
MIDI recording directory	%HOMEDRIVE%\%HOMEDIR%\Documents\GrandOrgue\MIDI recordings
MIDI player directory	%HOMEDRIVE%\%HOMEDIR%\Documents\GrandOrgue\MIDI recordings

6.2.3.2 Linux Paths

Sampleset directory	$HOME/GrandOrgue/Organs
Setting inport/export directory	$HOME/GrandOrgue/Settings
Audio recording directory	$HOME/GrandOrgue/Audio recordings
MIDI recording directory	$HOME/GrandOrgue/MIDI recordings
MIDI player directory	$HOME/GrandOrgue/MIDI recordings

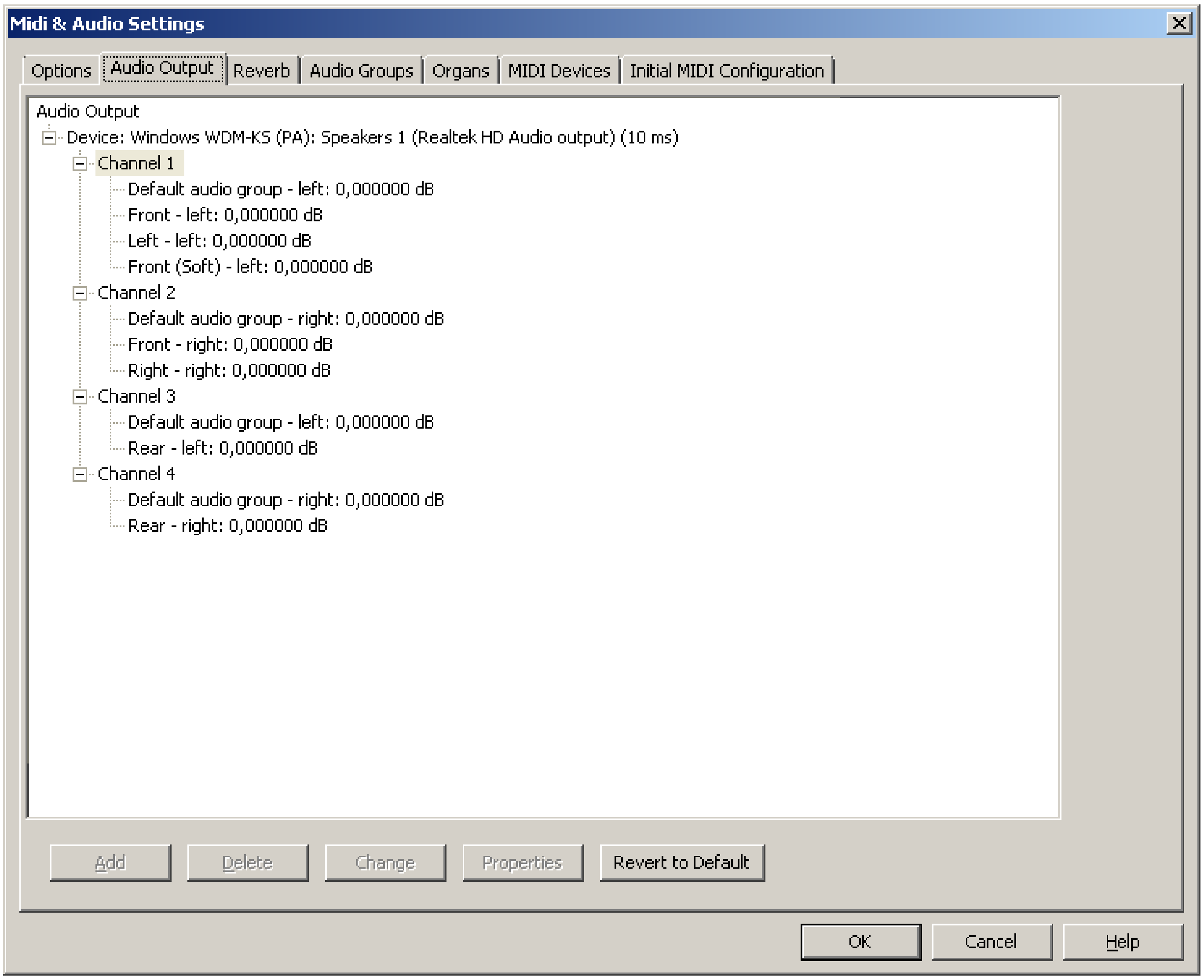
6.2.3.3 Mac Paths

Sampleset directory	$HOME/Library/Preferences/GrandOrgue
Setting inport/export directory	$HOME/Library/Preferences/GrandOrgue/Settings
Audio recording directory	$HOME/Library/Preferences/GrandOrgue/Audio recordings
MIDI recording directory	$HOME/Library/Preferences/GrandOrgue/MIDI recordings
MIDI player directory	$HOME/Library/Preferences/GrandOrgue/MIDI recordings

6.2.3.4 Path usage

Sampleset directory	This is the folder where GrandOrgue looks for samplesets. It is opened when the File > Open menu item is selected.
Setting inport/export directory	This is the folder where GrandOrgue stores or looks for settings files. It is opened when one of the File > Import Settings, File > Import Combinations, File > Export Settings/Combinations menu items is selected. This folder is usually different from the folder where GrandOrgue stores preset files.
Audio recording directory	This is the folder where GrandOrgue stores audio recording files. It is opened when the Audio/Midi > Record menu item is selected.
MIDI recording directory	This is the folder where GrandOrgue stores MIDI recording files. It is opened when the Audio/Midi > Record MIDI menu item is selected.
MIDI player directory	This is the folder where GrandOrgue looks for MIDI recording files. It is opened when the Audio/Midi > Play MIDI menu item is selected.

6.3 Audio Output tab



Audio Output
- Device: Windows WDM-KS (PA): Speakers 1 (Realtek HD Audio output) (10 ms)
  - Channel 1
    - Default audio group - left: 0,000000 dB
    - Front - left: 0,000000 dB
    - Left - left: 0,000000 dB
    - Front (Soft) - left: 0,000000 dB
  - Channel 2
    - Default audio group - right: 0,000000 dB
    - Front - right: 0,000000 dB
    - Right - right: 0,000000 dB
  - Channel 3
    - Default audio group - left: 0,000000 dB
    - Rear - left: 0,000000 dB
  - Channel 4
    - Default audio group - right: 0,000000 dB
    - Rear - right: 0,000000 dB

Figure 6.3: Audio Output tab

This dialog selects the audio output device and defines which channels and audio groups the audio output is dispatched to.

Channel:	a real-world channel of the audio device (e.g. left and right for a stereo board). GrandOrgue channel names are always **Channel 1** to **Channel n**, *n* being the number of channels available on the audio device (stereo:2, surround 4.0: 4, and so on). Only trial and error can determine which GrandOrgue channel is linked to which audio device physical channel.
Audio group:	a name holder used to fine tune the routing of audio output from the individual pipe to a particular channel of the audio device. See Organ Settings for more information.

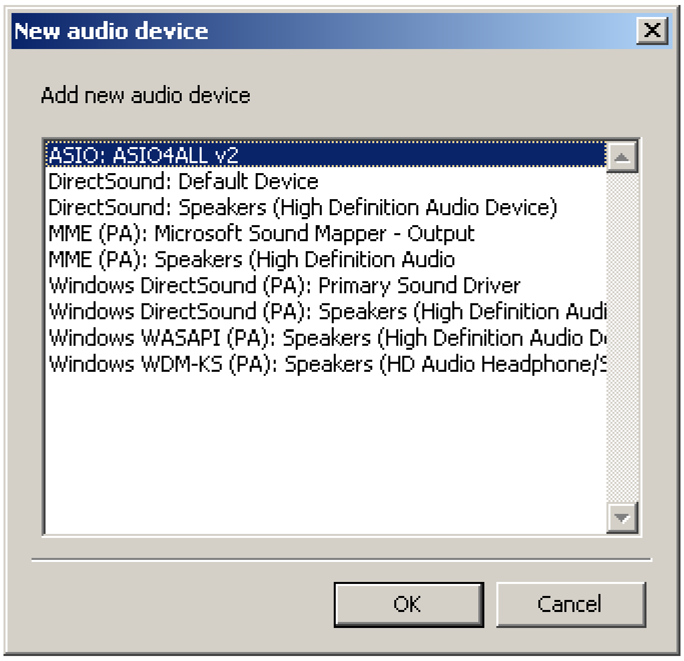
6.3.1 Managing devices

6.3.1.1 Add device

This feature is available when the **<Audio Output>** node is selected. The *Add* button is active. This node is the place where it is technically possible to add multiple audio devices.

Caution
Multiple audio devices are currently **NOT** supported. Setting multiple audio devices leads to unpredictable results. A warning is displayed and if acknowledged, a new audio device is set in the list.

The *Add* button opens a pick list with all audio devices found in the system.


New audio device

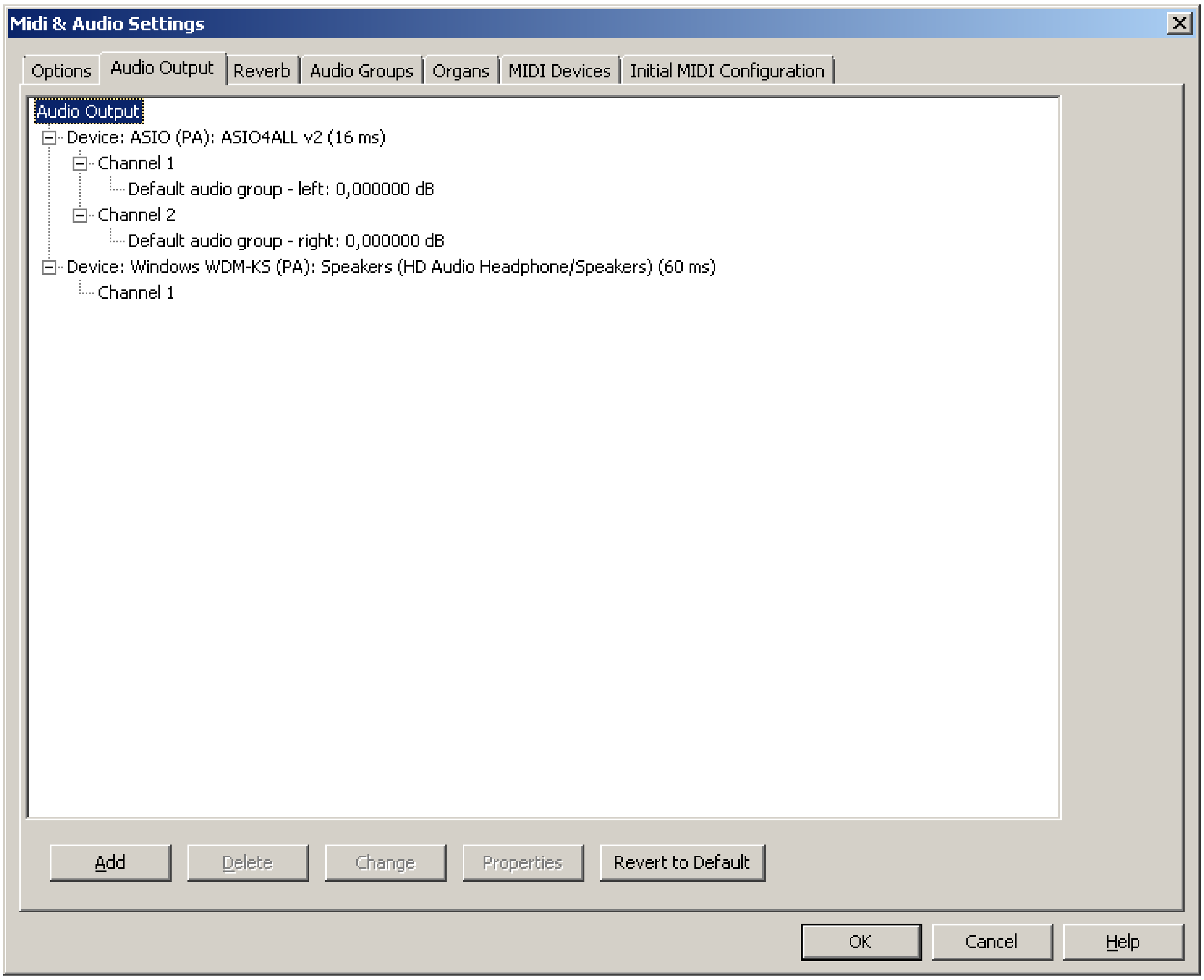
Add new audio device

ASIO: ASIO4ALL v2
DirectSound: Default Device
DirectSound: Speakers (High Definition Audio Device)
MME (PA): Microsoft Sound Mapper - Output
MME (PA): Speakers (High Definition Audio
Windows DirectSound (PA): Primary Sound Driver
Windows DirectSound (PA): Speakers (High Definition Audi
Windows WASAPI (PA): Speakers (High Definition Audio D
Windows WDM-KS (PA): Speakers (HD Audio Headphone/S

Each audio device is prefixed by its type (ASIO, DirectSound, ALSA, JACK, etc.)

The **(PA)** indicator shows which devices are driven using the *Portaudio* backend. All other devices are driven using the legacy *RtAudio* backend.

Select the desired audio device. The devices list is updated with the new device which is initialized with a single channel.



Audio Output
Device: ASIO (PA): ASIO4ALL v2 (16 ms)
Channel 1
Default audio group - left: 0,000000 dB
Channel 2
Default audio group - right: 0,000000 dB
Device: Windows WDM-KS (PA): Speakers (HD Audio Headphone/Speakers) (60 ms)
Channel 1

The new device needs further configuration to be usable. See Managing Channels for more information.

Reminder
Multi-device configuration is **NOT** supported.

6.3.1.2 Change device

This feature is available when a **Device:** node is selected in the tree. The *Change* button is active. It opens the same pick list as for adding a device.

Select the desired device. The devices list is updated with the new device, while the channel configuration is retained.

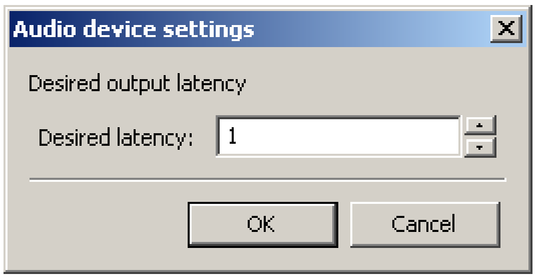
6.3.1.3 Delete device

This feature is available when a **Device:** node is selected in the tree. The *Delete* button is active *only if there are at least two devices in the tree*.

Select the device to delete. Push the *Delete* button.The device is removed from the tree.

6.3.1.4 Device properties

The *Properties* button becomes active when an audio device is selected in the tree. It opens a dialog which allows to set a *requested* latency.


Audio device settings

Desired output latency

Desired latency: 1

Note
This feature is closely related to the samples per buffer setting in the *Options* tab. Its behavior differs according to the chosen backend.

6.3.1.4.1 Audio driver selection

GrandOrgue feature two audio backends:

RtAudio	selects the number of buffers based on the desired latency setting. The "samples per buffer" and sample rate setting **must be supported by the soundcard/driver**, otherwise sound rendering will break.
PortAudio	uses the desired latency setting, the "samples per buffer" setting and the capabilities of the soundcard/driver to determine the operational buffering it will use. Such drivers are marked with *(PA)* in its name.

Audio/Sound Output State tries to displays *actual* latency. Note that the estimate can be based on numbers provided by the hardware and the drivers and may vary. RtAudio is more likely to display to low numbers.

6.3.1.4.1.1 Windows

WDM/KS This driver allows direct access to the Windows kernel driver[1]. This is the recommended driver if there is no other software which blocks access to the kernel audio streams. It is supported only via PortAudio.

WASAPI WASAPI support started with Window Vista. Use it if WDM/KS does not work for you. It is supported only via PortAudio.

Note
The samplerate of GrandOrgue must match the samplerate of the audio subsystem. So if WASAPI does not work, try a different samplerate in GrandOrgue.

DirectSound Use it if WDM/KS and WASAPI does not work for you. It is supported via PortAudio and RtAudio.

WMME This a classic, not low-latency audio API. Use it as last fallback. It is supported only via PortAudio.

ASIO GrandOrgue can be compiled with ASIO support. It is supported via PortAudio and RtAudio.

ASIO loads third party code into the GO process, so a bad ASIO driver can cause GO hangs.

[1] ASIO4ALL also uses that interface

Note
There is no reason to use ASIO4ALL, because it acts as a pass-through to the native kernel streaming APIs (e.g. WDM/KS) that GrandOrgue can use directly.

Jack GrandOrgue can also be compiled with Jack Support on Windows. See the Linux Jack usage notes.

6.3.1.4.1.2 OS X

CoreAudio You can use the OS X native audio API CoreAudio via RtAudio as well as PulseAudio.

Jack GrandOrgue can also be compiled with Jack Support on OS X. See the Linux Jack usage notes.

6.3.1.4.1.3 Linux

ALSA You can use the Linux native sound output either via RtAudio as well as via PortAudio. To allow direct hardware access, your Linux user account needs access to the sound card[2].

ALSA also provides "virtual" sound card entries. "default" routes the audio to the current default sound card and allows shared access. If the active samplerate on the hardware does not match the GrandOrgue samplerate, ALSA will start resampling, which will increase the CPU usage.

Note
Distributions often route the default audio via the pulseaudio daemon. If you don't want to use pulseaudio, you can start GrandOrgue via via pasuspender(1).

Jack You can connect to a Jack audio server either via RtAudio as well as PulseAudio. GrandOrgue also includes jack MIDI support, so will likely see MIDI devices twice (via jack as well as via the native OS APIs).

GrandOrgue has support for the native audio APIs and jack adds complexity to the sound output stack. Therefore avoid running jack, unless you want to use GrandOrgue together with other jack applications.

6.3.1.5 Revert to default

This button discards all audio configuration settings and creates a stereo configuration for the current device:

Channel 1:	one 'left' entry for each defined audio group, gain set to 0 db
Channel 2:	one 'right' entry for each defined audio group, gain set to 0 db

6.3.2 Managing Channels

6.3.2.1 Add channel

This feature is available when a **Device:** node is selected in the tree. The *Add* button becomes active only if the number of channels currently defined for that device is less than the device's maximum number of channels.

The *Add* button adds a new channel which is numbered with the next channel number. The new channel needs further configuration (audio groups) to be usable. See Managing Audio Groups for further information on that topic.

[2] that means being member of the *audio* group on many Linux distributions

Note
Only trial and error can determine which GrandOrgue channel is linked to each real-world audio channel.

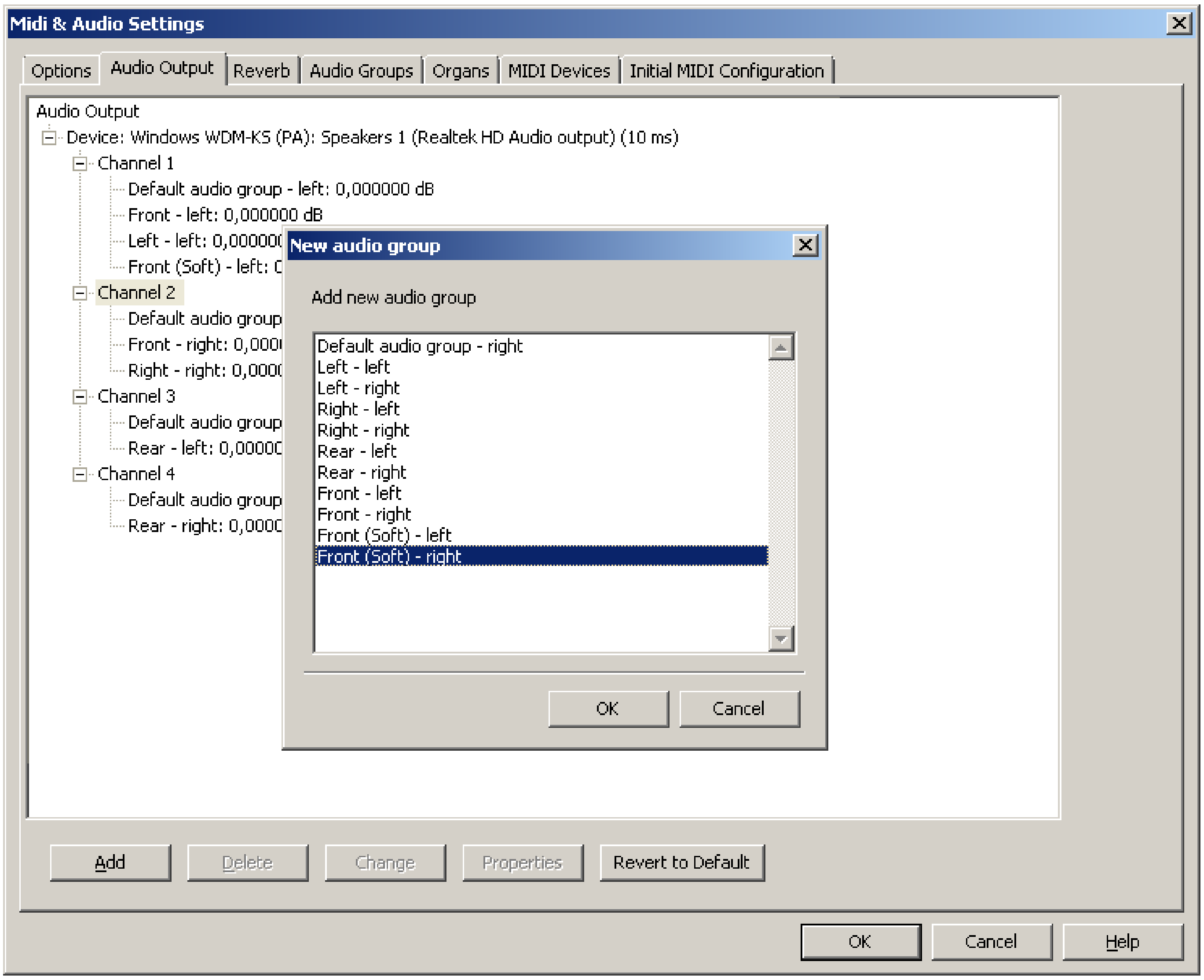
6.3.2.2 Delete channel

This feature is available when a **Channel n** node is selected in the tree. The *Delete* button is active only if there are at least two channels defined for the device. Push the *Delete* button. The selected channel is deleted with its configuration.

6.3.3 Managing audio groups

6.3.3.1 Add audio group

This feature is available when a **Channel n** node is selected in the tree. The *Add* button opens a pick list which allows to choose one or more audio group channels.



Audio Output
Device: Windows WDM-KS (PA): Speakers 1 (Realtek HD Audio output) (10 ms)
Channel 1
Default audio group - left: 0,000000 dB
Front - left: 0,000000 dB
Left - left: 0,00000
Front (Soft) - left: 0
Channel 2
Default audio group
Front - right: 0,000
Right - right: 0,000
Channel 3
Default audio group
Rear - left: 0,0000
Channel 4
Default audio group
Rear - right: 0,0000

New audio group
Add new audio group
Default audio group - right
Left - left
Left - right
Right - left
Right - right
Rear - left
Rear - right
Front - left
Front - right
Front (Soft) - left
Front (Soft) - right
OK Cancel

The *Add* button becomes unavailable when all audio group channels are used and the pick list is empty.

Note
An audio group always contains a left an a right channel. These channel names can't be changed.

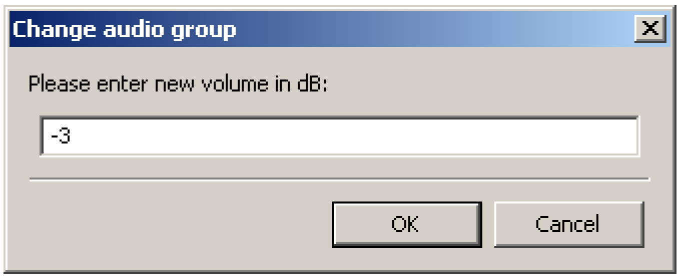
6.3.3.2 Delete audio group

This feature is available when an **audio group** node is selected in the tree. Every audio group channel (left/right) can be deleted from its parent GrandOrgue channel. Push the *Delete* button. The selected audio group channel is deleted.

Caution
When an audio group is not assigned to at least one audio channel (i.e. it is not used), all pipes assigned to this audio group in the Organ Settings dialog are **muted**

6.3.3.3 Group properties

This feature is available when an **audio group** node is selected in the tree. The *Properties* button opens the property dialog which allows to set a gain value (in db). 0 means "original volume". A positive value makes the audio group louder than original, a negative value makes the audio group softer.

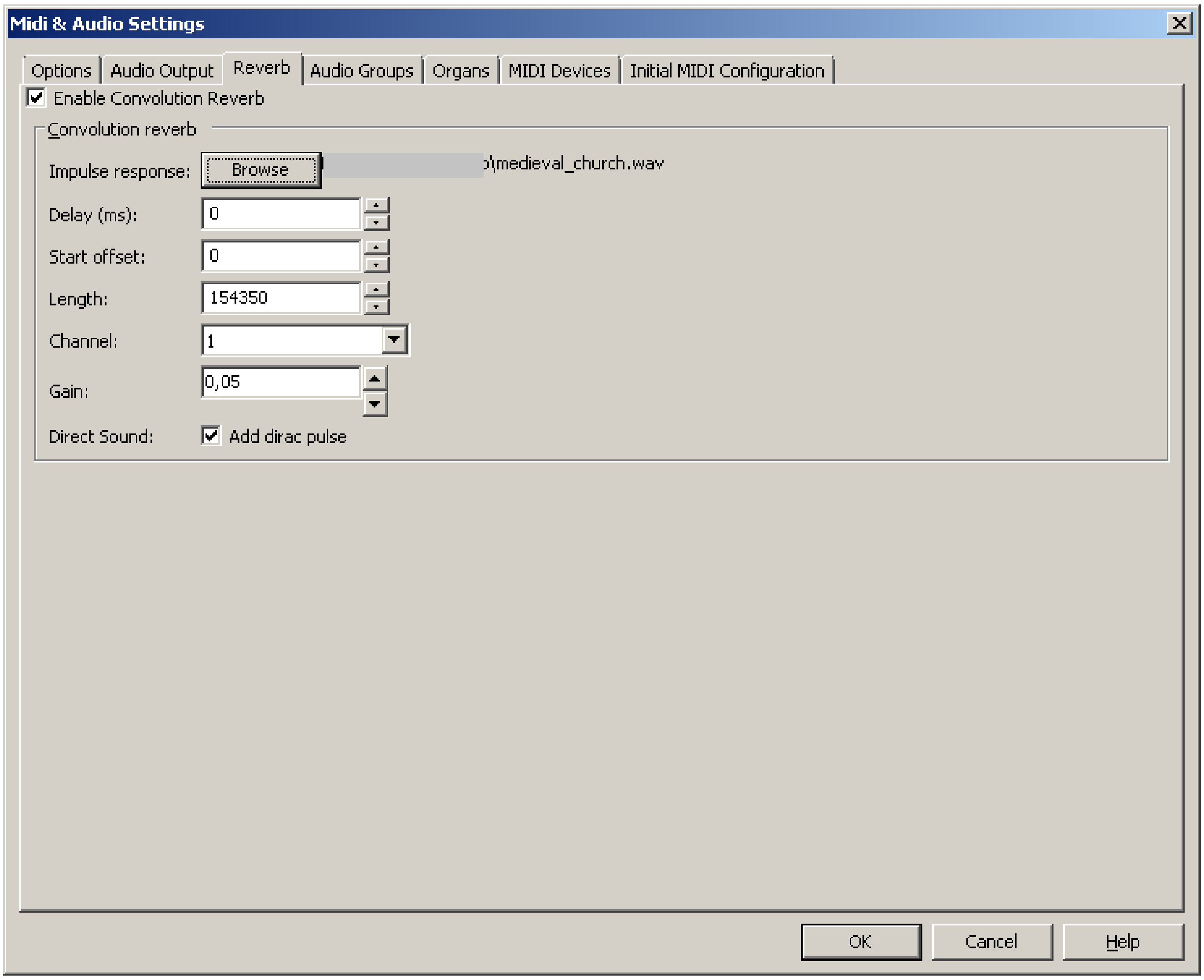


Change audio group

Please enter new volume in dB:

-3

6.4 Reverb tab


Midi & Audio Settings

☑ Enable Convolution Reverb

Convolution reverb

Impulse response:	Browse \medieval_church.wav
Delay (ms):	0
Start offset:	0
Length:	154350
Channel:	1
Gain:	0,05
Direct Sound:	☑ Add dirac pulse

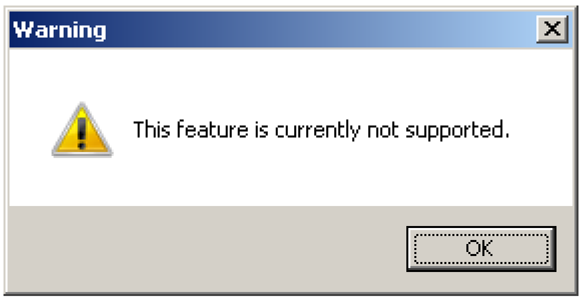
Figure 6.4: Reverberation options

This tab manages settings for an embedded **Convolution Reverberation engine**.

Convolution reverberation uses the mathematical model of a room response to alter an audio wave so it sounds as if it had been produced in that room. The room response is triggered by a bang (starter pistol, balloon pop, hands clap, ...) which is closest to a pulse covering all frequencies. The room response is made of the volume and decay of sound reflection on walls, floor, ceiling, and the delay introduced by the room size. The bang record is fed to the convolution engine to produce the reverberated wave.

The reverberated wave is usually mixed with the direct dry sound to better simulate what the auditor would hear if he were in the physical room.

Enable Convolution Reverb This checkbox enables the feature. A pop-up is displayed which can be safely ignored.



This feature is currently not supported.

Browse Impulse Response This button opens a file chooser which allows to choose and load an impulse response file. GrandOrgue currently supports only Wave (.wav) or WavPack (.wpk) files.

> **Note**
> A response file is ***required*** when the reverberation is enabled.
> GrandOrgue wiki displays a page linking to sites where IR files can be downloaded.

Delay This spinner sets the delay in ms before the reverberated signal quicks in. This setting is additive to the delay that can be generated by the IR file itself.

Start Offset This spinner sets the number of samples to skip from the beginning of the impulse response file.

Length This spinner sets the number of samples to use in the impulse response file (starting from the offset).

Channel This dropdown sets the channel to use in an impulse response file. Depending on the recording method, an IR file can have many channels. GrandOrgue current reverberation engine uses only one channel.

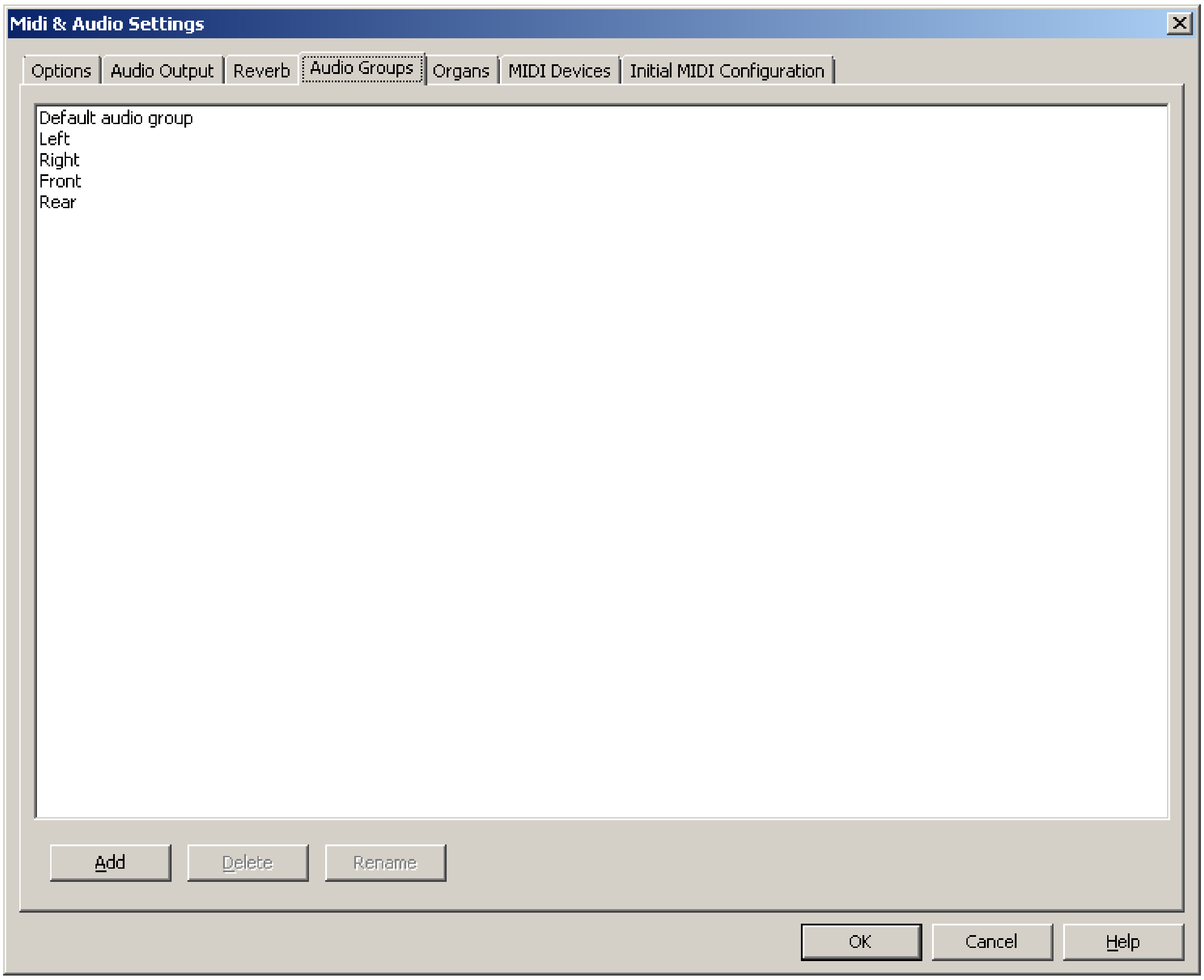
Gain This spinner sets the gain to apply to the reverberated signal. *Use low values (< 0.1) to avoid artifacts by overloading the sound system.*

Direct Sound This checkbox enables mixing the direct and reverberated signals.

The reverberated sound, when fed alone to the audio system, usually sounds a bit muffled and a bit garbled by the alterations introduced by the room response (delays, multiple echos, etc.). Adding the direct sound is the only way to get a clear sound, so this box should be checked at all times.

> **Note**
> Set the release tail length to a low value if you want to use the convolution reverberation alone, ie. without the recorded release tail of wet samples.

6.5 Audio Groups tab



Default audio group
Left
Right
Front
Rear

Figure 6.5: Audio Groups tab

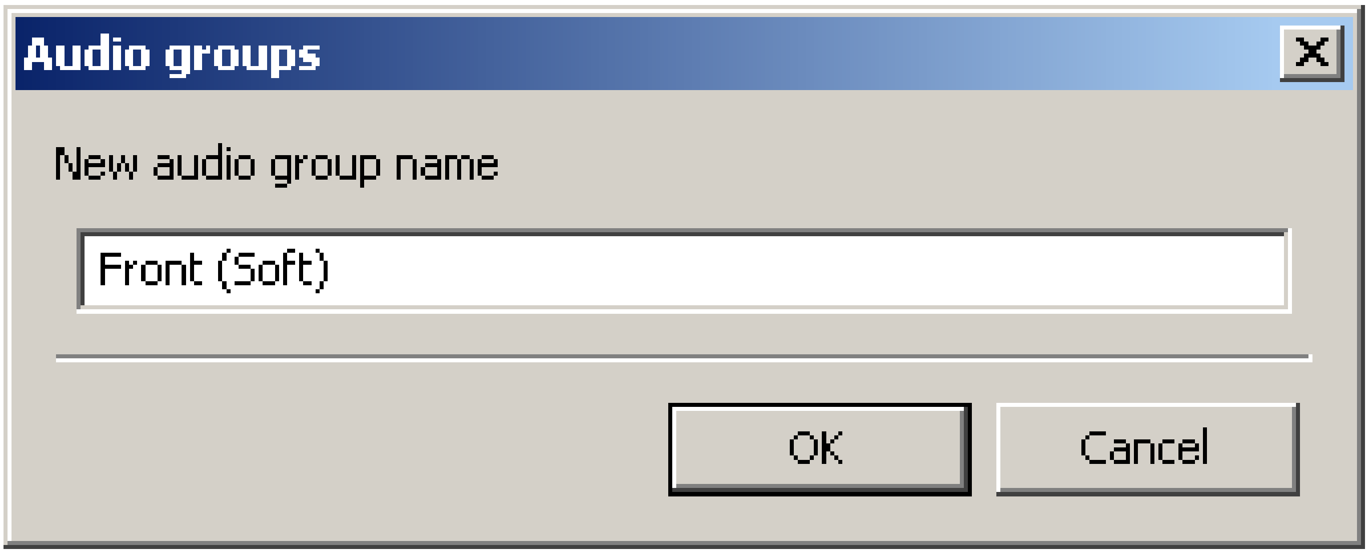
This dialog allows to manage audio groups. As said above, audio groups are name holders that are used to map the audio output of the individual pipe to definite parts of the audio system. See Organ Settings for more information.

The system requires at least one audio group, and is initialized with a default audio group named *Default audio group*.

The *Default audio group* is **always** the first audio group in the list, so the *Default audio group* name can be changd at any time.

6.5.1 Add

The *Add* button opens a dialog where one can type the new audio group name.



Caution
If an audiogroup is not used in the loaded sample set, it does add processing overhead, **thus reducing polyphony**, so adding unnecessary audio groups is not recommended.

6.5.2 Delete

This feature is available when an audio group name is selected in the list. The *Delete* button is active. Push the button, the selected audio group is deleted.

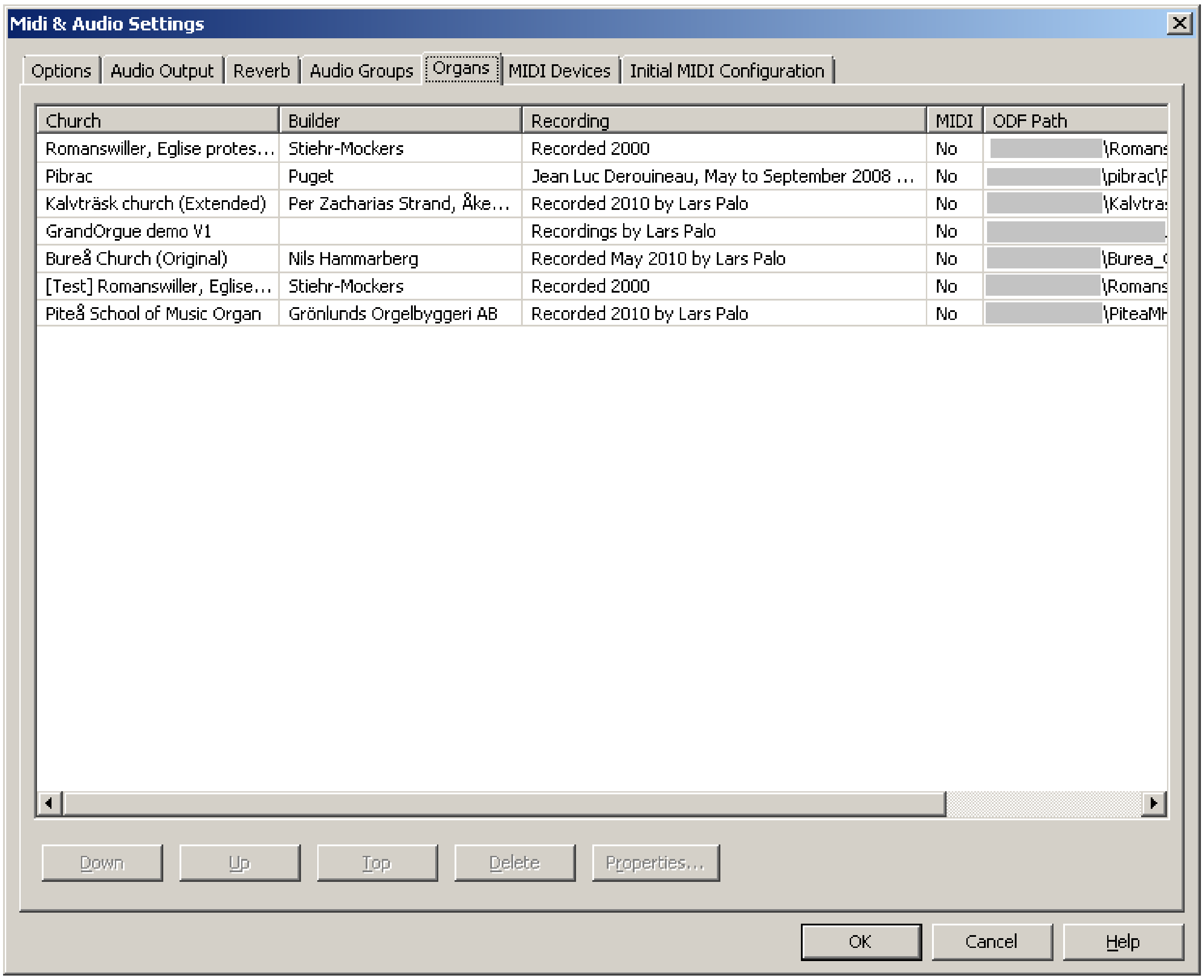
Caution
If the deleted audio group is assigned to an organ element (Organ Settings dialog), all pipes routed to this audio group will be automatically routed to the default audio group.

6.5.3 Rename

This feature is available when an audio group name is selected in the list. The *Rename* button is active. It opens the same dialog as in Add, where the audio group's name can be changed.

Caution
If the renamed audio group is assigned to an organ element (Organ Settings dialog), all pipes routed to this audio group will be automatically routed to the default audio group.

6.6 Organs tab



Midi & Audio Settings

Options | Audio Output | Reverb | Audio Groups | Organs | MIDI Devices | Initial MIDI Configuration

Church	Builder	Recording	MIDI	ODF Path
Romanswiller, Eglise protes...	Stiehr-Mockers	Recorded 2000	No	[redacted]\Romans
Pibrac	Puget	Jean Luc Derouineau, May to September 2008 ...	No	[redacted]\pibrac\P
Kalvträsk church (Extended)	Per Zacharias Strand, Åke...	Recorded 2010 by Lars Palo	No	[redacted]\Kalvtra
GrandOrgue demo V1		Recordings by Lars Palo	No	[redacted]
Bureå Church (Original)	Nils Hammarberg	Recorded May 2010 by Lars Palo	No	[redacted]\Burea_
[Test] Romanswiller, Eglise...	Stiehr-Mockers	Recorded 2000	No	[redacted]\Romans
Piteå School of Music Organ	Grönlunds Orgelbyggeri AB	Recorded 2010 by Lars Palo	No	[redacted]\PiteaMH

Figure 6.6: Organs tab

This tab manages the *known organs list*.

All organs successfully loaded via the File/Open menu item are automatically registered in this list.

This list provides for the lists displayed in the File/Load, File/Open Recent and File/Favorites menu items.

The buttons are grayed out until a line is selected in the list.

The organs order on this tab is reproduced as is in the File/Load menu whereas only the 10 first entries are displayed in the File/Favorites menu.

6.6.1 Down

This button moves the selected entry down one position. It is grayed out if the selected line is last in the list.

6.6.2 Up

This button moves the selected entry up one position. It is grayed out if the selected line is first in the list.

6.6.3 Top

This button moves the selected entry to the first position in the list. It is grayed out if the selected line is first in the list.

6.6.4 Delete

This button removes a line from the list. GrandOrgue deregisters this organ.

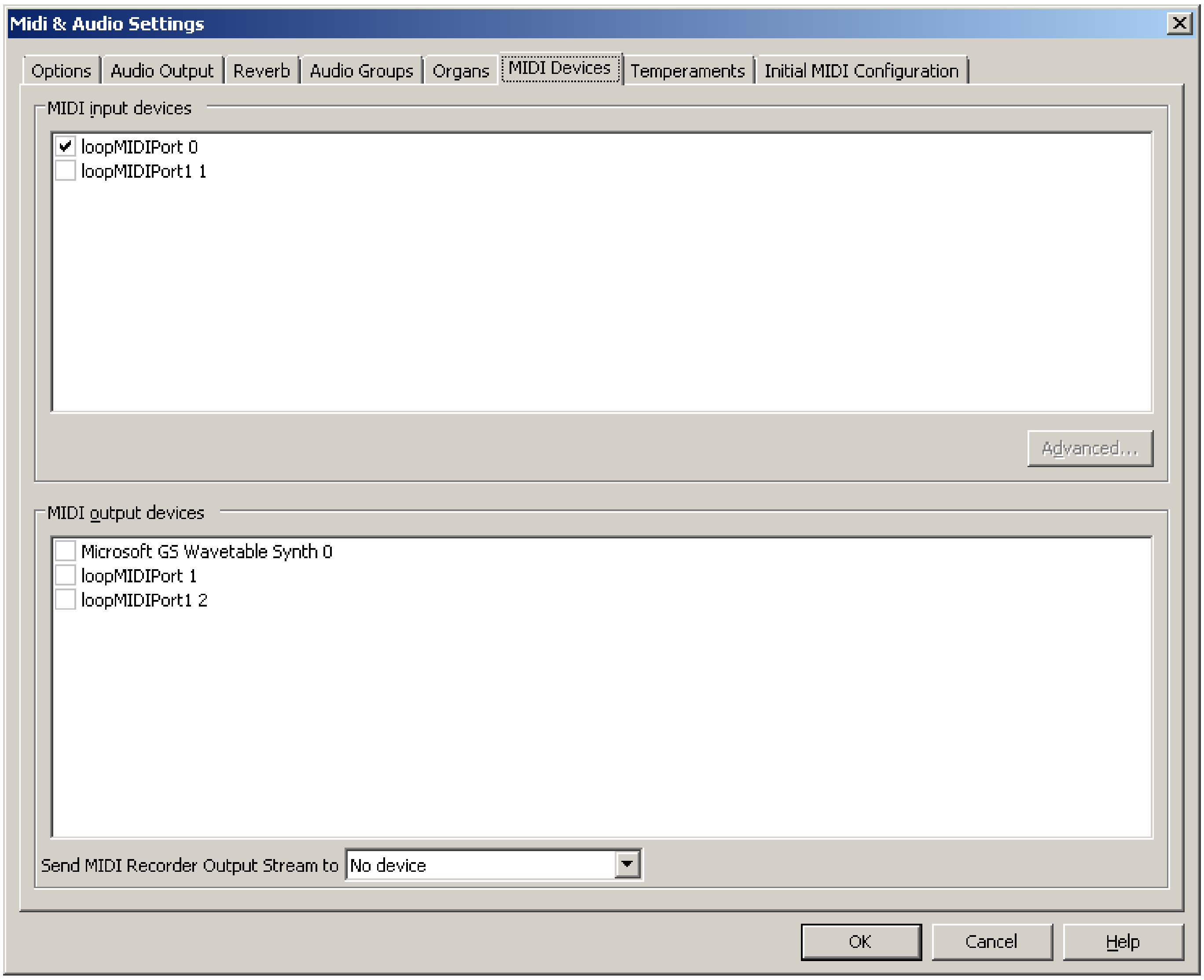
Note
If presets were saved and/or the samples were cached, the files are **NOT** deleted from disk.

6.6.5 Properties

This button opens the Midi Event Editor to manage the MIDI event(s) which enable GrandOrgue to load a sampleset when received.

A use case of this feature can be: use a rotary selector that sends MIDI **Program Change** messages to easily switch between samplesets.

6.7 MIDI Devices tab



MIDI input devices

- [✓] loopMIDIPort 0
- [ ] loopMIDIPort1 1

MIDI output devices

- [ ] Microsoft GS Wavetable Synth 0
- [ ] loopMIDIPort 1
- [ ] loopMIDIPort1 2

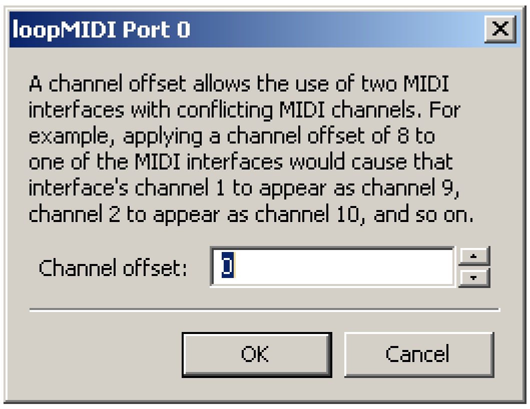
Send MIDI Recorder Output Stream to: No device

Figure 6.7: MIDI devices options

This tab shows all available MIDI devices.

Upper frame The upper frame shows a list of all available MIDI input interfaces. Selecting a device means that GrandOrgue will listen for incoming MIDI messages through that interface. This allows to listen to more than one device at a time; for example, to listen to both a controller and a sequencer or loopback device.

The *Advanced* button is grayed out unless an input channel is selected. It opens a dialog to set advanced properties of the selected input channel.



A channel offset allows the use of two MIDI interfaces with conflicting MIDI channels. For example, applying a channel offset of 8 to one of the MIDI interfaces would cause that interface's channel 1 to appear as channel 9, channel 2 to appear as channel 10, and so on.

Channel offset: 0

Figure 6.8: MIDI input channel advanced properties

This allows the setting of a channel offset. A channel offset allows the use of two MIDI interfaces with conflicting MIDI channels. For example, applying a channel offset of 8 to one of the MIDI interfaces would cause that interface's channel 1 to appear as channel 9, channel 2 to appear as channel 10, and so on. This may be useful if you have multiple keyboards that are configured to use the same channel.

Note
GrandOrgue is capable of matching the MIDI interface and this is configured by the MIDI detection by default. Therefore it is possible to use the same MIDI channel on multiple interfaces without configuring any channel offset.

Lower frame The lower frame shows a list of all available MIDI output interfaces. Selecting a device means that GrandOrgue will write MIDI information to external hardware. Possible uses are SAM or LED drivers for stop control feedback, or physical pipes drivers when digitally expanding a genuine pipe organ.

Send MIDI Recorder Output Stream The dropdown labeled *Send MIDI Recorder Output Stream to* allows the user to pick a target output MIDI device to receive all Recorder-generated output. The selected device must be activeted in the *MIDI output devices* frame.

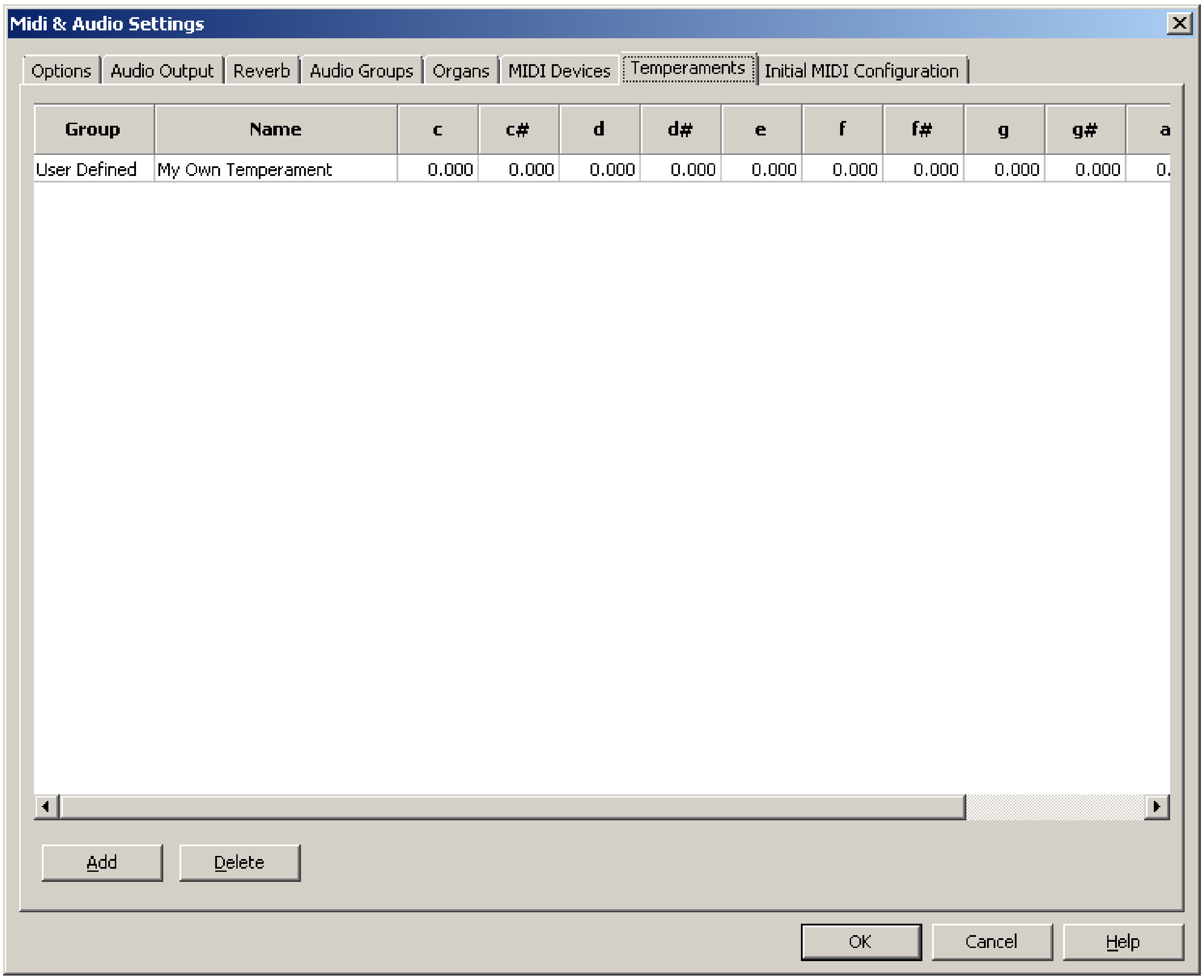
When an output device is selected in that dropdown, the recorder outpout is generated even if no MIDI file is being recorded.

Note
The devices list is refreshed whenever the *Audio & Midi Settings* dialog is displayed.

Note
GrandOrgue requires a MIDI input/output port to connect to. MIDI (sequencer) software, that just tries to connect to existing MIDI ports, is not compatible with GrandOrgue. If you want to use such software, create a virtual MIDI port[a] and let GrandOrgue as well as the other MIDI software connect to it.

[a] For linux, use *sudo modprobe snd-seq-dummy ports=9*

6.8 Temperaments



Group	Name	c	c#	d	d#	e	f	f#	g	g#	a
User Defined	My Own Temperament	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.

Figure 6.9: User-defined Temperaments

This tab enables the user to add his own custom temperaments. User-defined temperaments are displayed in dynamic submenus in the *Audio/Midi* > *Temperaments* menu.

The **Add** button adds a new "empty" line.

The **Delete** button deletes the selected line.

Group Defines the submenu label in the *Audio/Midi* > *Temperaments* menu. The submenus are dynamically added as soon as this screen has entries. Entries having the same **Group** attribute are grouped in the same submenu.

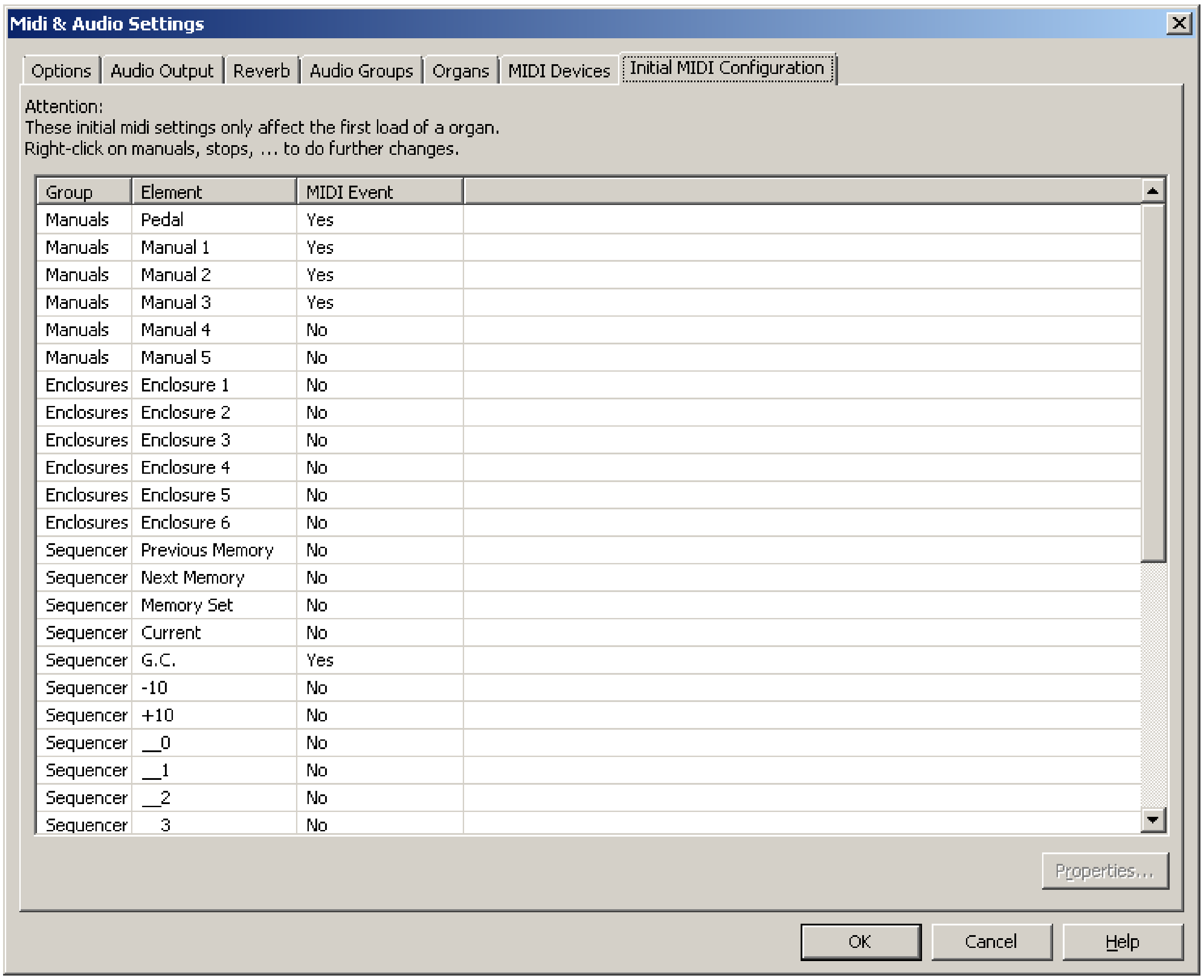
Name Defines the menu item label.

c, c# ... b A column for each note of the scale defines the deviation in cents from equal temperament for that note.

To modify any value in the grid, double-click in the cell. Updated values are stored as soon as the **OK** button is depressed.

Temperaments are stored in the system-wide configuration file.

6.9 Initial Midi Configuration tab



Attention:
These initial midi settings only affect the first load of a organ.
Right-click on manuals, stops, ... to do further changes.

Group	Element	MIDI Event
Manuals	Pedal	Yes
Manuals	Manual 1	Yes
Manuals	Manual 2	Yes
Manuals	Manual 3	Yes
Manuals	Manual 4	No
Manuals	Manual 5	No
Enclosures	Enclosure 1	No
Enclosures	Enclosure 2	No
Enclosures	Enclosure 3	No
Enclosures	Enclosure 4	No
Enclosures	Enclosure 5	No
Enclosures	Enclosure 6	No
Sequencer	Previous Memory	No
Sequencer	Next Memory	No
Sequencer	Memory Set	No
Sequencer	Current	No
Sequencer	G.C.	Yes
Sequencer	-10	No
Sequencer	+10	No
Sequencer	__0	No
Sequencer	__1	No
Sequencer	__2	No
Sequencer	__3	No

Figure 6.10: Initial Midi Configuration options

This screen displays default MIDI settings which enable the user to store some physical console MIDI settings.

When a sampleset is loaded for the first time **OR** without customization, these default settings are used to initialize the MIDI Event dialogs that are accessible by right-clicking a control in the GUI (e.g. manual, stop, enclosure, piston, etc.).

Customizations done by the user always supersede the matching initial setting. If customizations are saved to a preset number, all subsequent loads will ignore the initial MIDI settings and all futher updates to the sampleset's MIDI configuration will be done by right-clicking on the elements in the user interface.

When a loaded sampleset is reset to default, the preset file is deleted and the initial MIDI settings are used again to initialize the MIDI configuration.

Note

The sampleset creator must define proper ***MIDIInputNumber*** values in the Organ Definition File to make manual mapping and enclosure mapping to the initial MIDI settings work properly.

The "Group" column shows which group the element is filed under; the "Element" column identifies the element; The "MIDI Event" column shows if the element is associated to MIDI event(s).

The MIDI Event Editor can be displayed by selecting an element and pushing the "Properties" button, or by double-clicking on the element's entry.

The available initial setting are those that are tied to the physical hardware and have no reason to change across dispositions:

Manuals Group Midi setting for pedal (always Manual 0) and up to 5 manuals. Manuals are ordered bottom (Manual 1) to top (Manual 5).

Enclosures Group Midi setting for up to 6 enclosures. Enclosures are ordered left (Enclosure 1) to right (Enclosure 6).

Sequencer Group Midi setting for various useful controls from the combination setter.

- **Previous Memory, Next Memory** Navigate the memory banks.
- **Memory Set** Store a combination in a memory slot.
- **Current** Recalls the current combination.
- **G.C.** General Cancel.
- **-10, +10** Navigate 10 slots backwards or forward in the current memory bank without recalling the combination (prepare it).
- **__0 .. __9** Navigate to slot xx0 to xx9 and recall the combination, the first two digits being set by the +/-10 controls.

Master Controls Group Midi setting for various useful controls from the main control panel.

- **-1 Cent, +1 Cent, -100 Cent, +100 Cent** These controls drive the amount of tuning for the whole organ.
- **Prev Temperament, Next Temperament** Navigate the temperaments list.
- **Transpose -, Transpose +** Transpose down, transpose up, one semitone at a time.

Chapter 7

Organ settings

This dialog allows to customize settings for the currently loaded sample set.

Initial values come from the current preset number or imported settings file or from the organ definition file if the settings file does not exist.

Modified settings are saved in the current preset number. They also can be exported to a settings file.

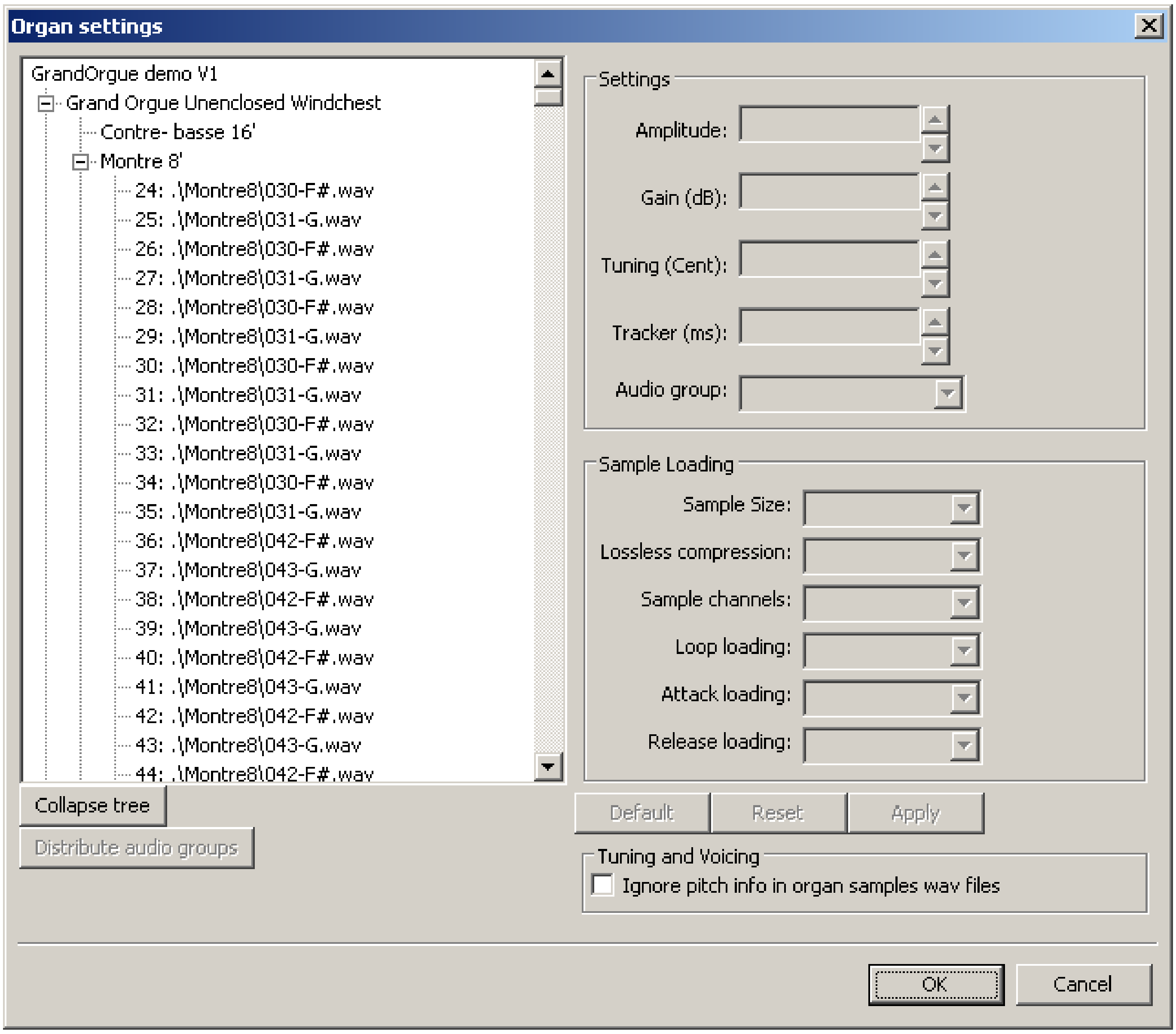


Figure 7.1: Organ Settings dialog

In its left box, this dialog displays all ranks found in the definition file grouped by windchest. The tree can unfold down to the individual pipe level. Within windchests, the ranks are ordered as found in the definition file.

When the tree is unfolded down to the pipe level, the dialog displays the MIDI number assigned to the sample file, along with the file's path.

The right frames display controls which allow to fine set features at the rank or pipe level.

Settings can be modified at the organ, rank and individual pipe level. The windchest level is only used as logical grouping, and any settings modified at that level are ignored.

Some setting are cumulative from the organ level down to the pipe level: the effective setting value applied to the individual pipe is calculated from values at all levels.

For the other (non cumulative) settings, the effective setting value is applied to the individual pipe using these precedence rules:

- If a pipe-level value is defined, use it

- Else if a rank-level value is defined, use it
- Else if an organ-level value is defined, use it
- Else use the value defined in the Midi & Audio Settings dialog Options tab if any.

Note
Select the element you wish to customize. The right frames are populated with values picked from the organ definition file or the current preset number if the latter was saved.
It is possible to select multiple elements at once, to e.g. update one setting of multiple elements to the same value in a single operation.
To the sample set designer: Fine customizations to the sample set, when made in the *Organ Settings* dialog, are **NOT** copied to the organ definition file. If they were designed to be integral part of the sample set, these customizations need to be manually merged from the settings file to the organ definition file.

7.1 Settings frame

7.1.1 Amplitude

Linear amplitude scale factor applied to the element. The allowed range is {0, +1000}. Amplitude is cumulative from organ level down to pipe level.

Note: A 100 value means that this setting value has no effect on the effective amplitude.

7.1.2 Gain

Amplitude scale factor in dB applied to the selected element. The allowed range is {-120, +40}. Gain is cumulative from organ level down to pipe level.

Note: A 0 value means that this setting value has no effect on the effective gain.

7.1.3 Tuning

This spinner allows to fine tune the pitch of the element. The unit is 1/100th of a semitone, and the allowed range is {-1200, +1200}. Tuning is cumulative from organ level down to pipe level.

Note: A 0 value means that this setting value has no effect on the effective tuning.

7.1.4 Tracker

This spinner allows to define a delay in ms. This delay simulates the time spent by the tracker system to open the valve after a key is depressed.

Tracker delay is cumulative from organ level down to pipe level.

7.1.5 Audio Group

This dropdown selects which audio group the sound from the selected element is routed to

Note: A blank value means that this setting value is fetched from the upper level. If the effective audio group is still blank, the pipe is routed to the default audio group.

7.2 Sample loading frame

7.2.1 Sample size

This dropdown list allows to overwrite its counterpart at the parent level.

The *Parent default* value means that the value is fetched from the parent level.

See Sample size in the *Midi & Audio Settings* dialog for further detail on this topic.

7.2.2 Lossless Compression

This dropdown list allows to overwrite its counterpart at the parent level. The allowed values are:

Parent default Means that the value is fetched from the parent level.

Enabled Enables lossless compression at that level

Disabled Disables lossless compression at that level

See Lossless compression in the *Midi & Audio Settings* dialog for further detail on this topic.

7.2.3 Sample channels

This dropdown list allows to overwrite its counterpart at the parent level. The allowed values are:

Parent default Means that the value is fetched from the parent level.

Don't load Don't load the samples at that level.

Mono Load stereo samples in mono.

Stereo Load stereo samples in stereo.

See Load stereo samples in the *Midi & Audio Settings* dialog for further detail on this topic.

7.2.4 Loop loading

This dropdown list allows to overwrite its counterpart at the parent level. The allowed values are:

Parent default Means that the value is fetched from the parent level.

First loop Load the first loop found in the sample file.

Longest loop Load the longest loop found in the sample file.

All loops Load all loops found in the sample file.

See Loop loading in the *Midi & Audio Settings* dialog for further detail on this topic.

7.2.5 Attack loading

This dropdown list allows to overwrite its counterpart at the parent level. The allowed values are:

Parent default Means that the value is fetched from the parent level.

Single attack Loads only the "best" provided attack sample file.

All Loads all attack sample files

See Attack loading in the Midi & Audio Settings dialog for further detail on this topic.

7.2.6 Release loading

This dropdown list allows to overwrite its counterpart at the parent level. The allowed values are:

Parent default Means that the value is fetched from the parent level.

Single release Loads only the "best" provided release sample file.

All Loads all release sample files

See Release loading in the Midi & Audio Settings dialog for further detail on this topic.

7.3 Buttons row

These buttons are used to drive the dialog and always act on the selected element.

Default This button restores the setting values for the selected element from the organ definition file.

Reset This button is grayed out until a value is modified. It restores all previously applied values.

Apply This button applies all modified values to the selected element. Changes **MUST** be applied before proceeding to another element modification or closing the dialog with the **OK** button.

7.4 Tuning and Voicing frame

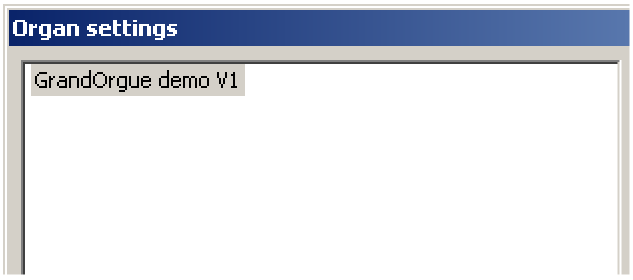
This frame displays a single check box labeled *Ignore pitch info in organ samples wav files*.

This feature is closely related to the temperament feature which assumes, when the samples are dynamically retuned after a new temperament is applied, that pitch information was embedded in each sample file *or in the pipe entry within the Organ Definition File*. If the sample set designer has **NOT** embedded pitch information, the result is weird and unpredictable.

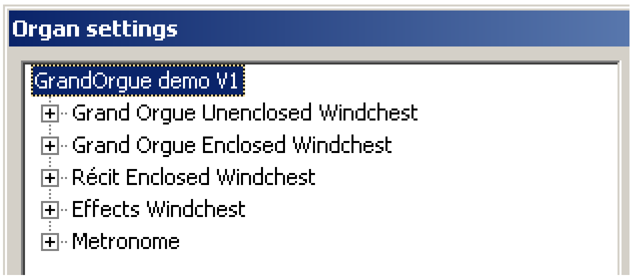
When checked, this box instructs GrandOrgue to ignore any pitch information found is sample files while retuning the organ to another temperament than the original. This way the retuned sample set behaves as expected. In this case, GrandOrgue assumes that the samples are tuned to equal temperament.

7.5 Collapse tree button

When the dialog opens, the tree is displayed unfolded to the pipe level. This button is a shortcut for displaying only the root node:



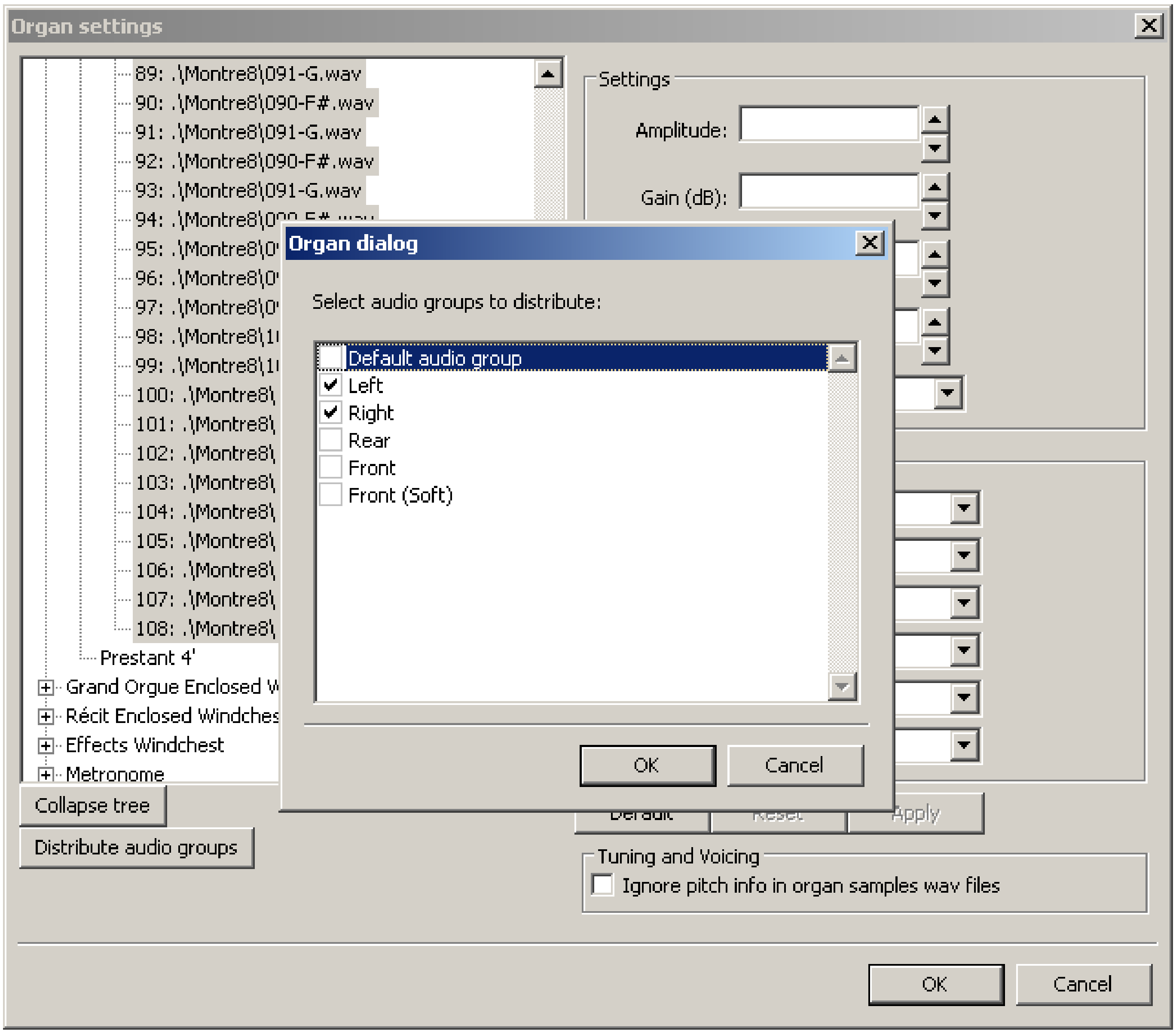
Double-click the organ name to unfold the tree to the windchest level:



7.6 Distribute audio groups button

This button allows to spread the selected elements over selected audio groups. The elements are spread over audio groups using a round-robin scheme.

Select the elements to spread, click the button. A dialog opens where the desired audio groups can be selected by checking their box.



Organ dialog

Select audio groups to distribute:

- [ ] Default audio group
- [✓] Left
- [✓] Right
- [ ] Rear
- [ ] Front
- [ ] Front (Soft)

OK Cancel

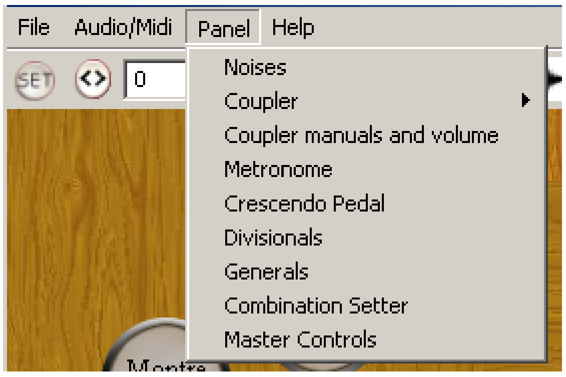
- [ ] Ignore pitch info in organ samples wav files

Push the **OK** button. The modification is applied.

Caution
The round-robin scheme is applied to the audio groups in their display order in the selector dialog. They appear in that dialog in the same order they were created. Therefore, when elements need to be spread over a list of audio groups, these audio groups **must** be created in the order of the round-robin scheme.

Chapter 8

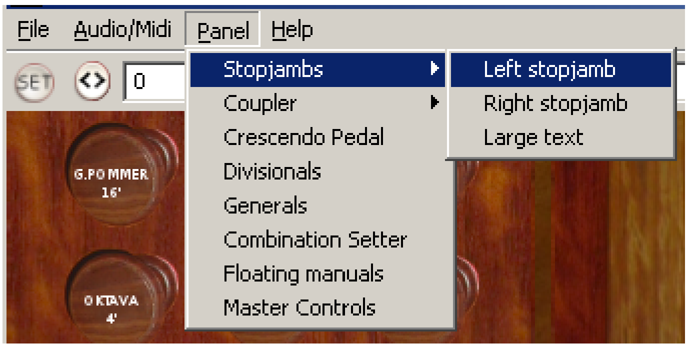
Panels



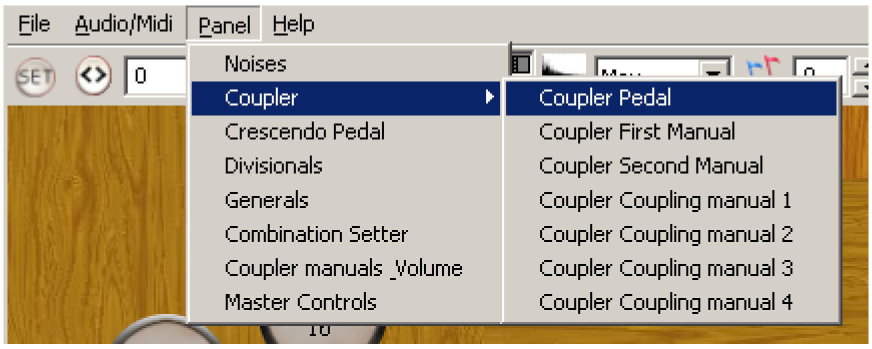
This menu controls control panels. Menu items starting at *Coupler* down to *Master Controls* are standard menu items (i.e. present whatever sampleset is loaded).

All other menu items (above *Coupler*) are defined within the loaded sampleset and should be documented in the sampleset's specific user manual (if any). They usually are referred to as *Custom Panels*.

Custom Panel menu items can be organized in submenus as shown in this screen shot taken from Piteå Music School sampleset.



8.1 Coupler



GrandOrgue provides a default set of couplers for each keyboard or pedal board found in the Organ Definition File, plus a coupler set for each of the default coupler manuals. This enables the user to take advantage of "modern" accessories even when the sampleset doesn't provide them.

The menu items display the *master* keyboards, i.e. the keyboard that "pulls" the coupled keyboards.

The coupler panel displays a line of controls for every *target* keyboard or pedal board found in the Organ Definition file.

Note
The coupling manuals are not designed to be the target of a coupler.

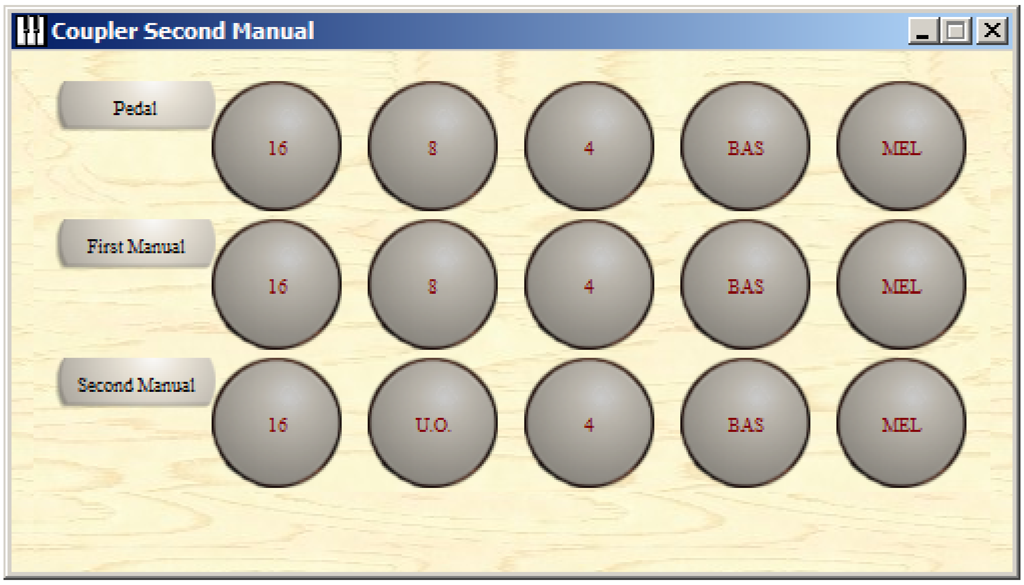


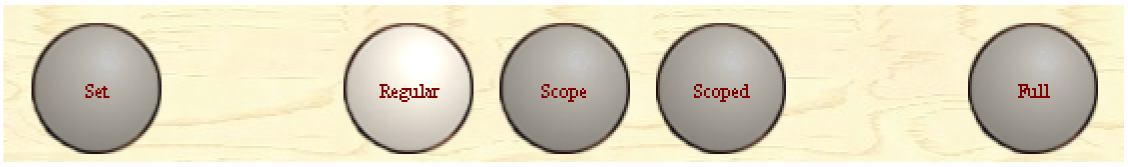
Figure 8.1: Coupler <keyboard name> Layout

16	Sub-octave coupler. The target key/pedal board is "pulled" 1 octave lower than the note played on the "master" key/pedal board.
8	"Normal" coupler. The target key/pedal board is "pulled" at the same octave as the note played on the "master" key/pedal board.
U.O.	Unison Off. When engaged, this coupler mutes the pipes at the played note pitch on the "master" key/pedal board. There are many use cases like using the octave/sub-octave couplers alone, turning the keyboard into a coupling keyboard, etc.
4	Octave coupler. The target key/pedal board is "pulled" 1 octave over the note played on the "master" key/pedal board.

BAS	"Bass" coupler. The lowest note in any chord played on the "master" keyboard is "pulled" on the target key/pedal board.
MEL	"Melody" coupler. The upper note in any chord played on the "master" keyboard is "pulled" on the target key/pedal board.

Note
In multi-part polyphony, the BASS coupler tries to enforce the rests found in the lowest line. This is not 100% safe, so unwanted jumps to the next upper line may occur. The same applies for the MEL coupler and the highest line.

8.2 Regular, Scope and Scoped, Full



These features are shared by the *Crescendo*, *Divisionals*, *Generals* and *Combination Setter* panels and are documented once here.

8.2.1 Regular

Regular enables to control all stops and pistons as allowed by the Organ Definition file (see *Full* below). When *Regular* is on while setting, the current state of all stops and couplers is stored. The scope is set to "all elements". This is the default behaviour.

8.2.2 Scope

When *Scope* is on while setting, the combination data store is configured for its storing only the currently turned on elements. Never forget to push all buttons intended to drive that scope.

8.2.3 Scoped

When *Scoped* is on while setting, the current state of the elements stored with scope is saved.

Note
TYPICAL WORKING SEQUENCE OF OPERATIONS:

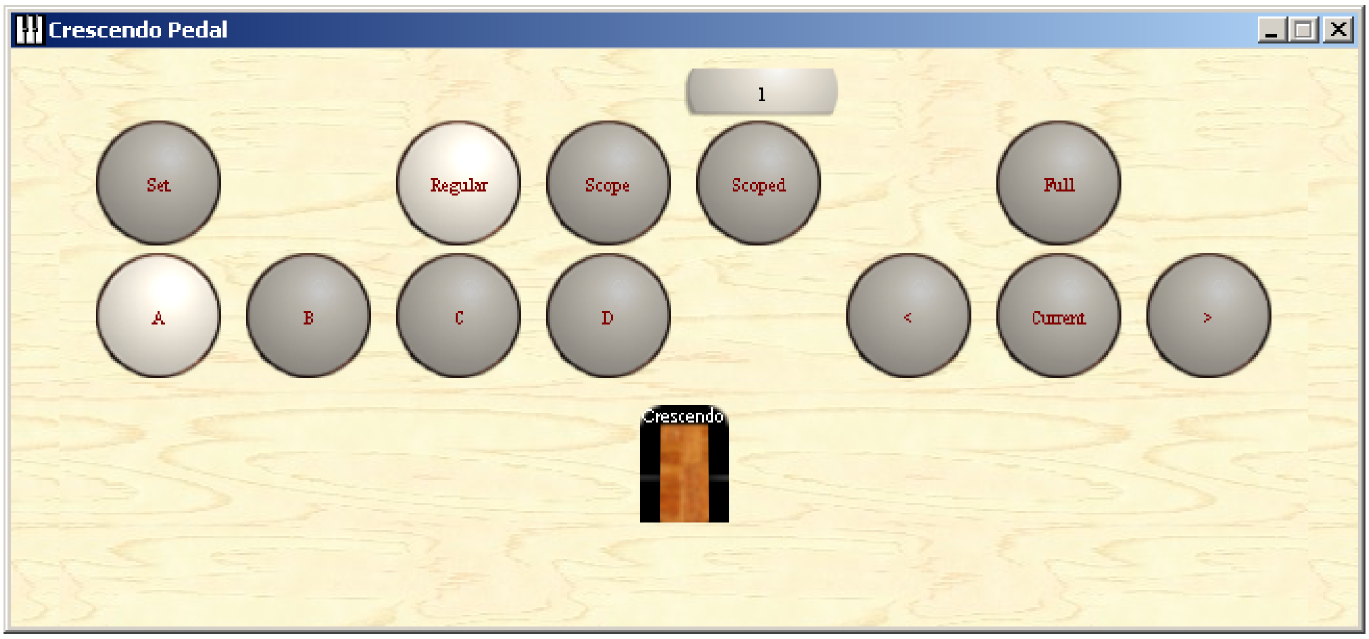
1. Turn on some stops and couplers
2. Turn on **Scope**
3. Turn on **Set**
4. Push all buttons intended to use that scope
5. Turn on **Scoped**
6. Turn on the stops and couplers wished for the fist button
7. Push the button (make sure that **Set** is turned on)
8. Repeat 6-7 for all buttons

Once the scope is defined and assigned to a piston, there is no need to redefine it (steps 1-4) if the saved state of the elements in scope is changed (steps 5-6-7).

8.2.4 Full

Full allows to store all stops and couplers in a combination, regardless of restrictions that may have been defined in the Organ Definition File. See StoreInDivisional, StoreInGeneral, DivisionalsStoreIntermanualCouplers, DivisionalsStoreIntramanualCouplers, DivisionalsStoreTremulants, GeneralsStoreDivisionalCouplers and CombinationsStoreNonDisplayedDrawstops for more information.

8.3 Crescendo Pedal



GrandOrgue automatically adds 4 crescendo banks of 32 steps each. This enables the user to take advantage of "modern" accessories even when the sampleset doesn't provide them.

Label on top Displays the current step. It is an "active" label, and as such supports a Midi send configuration

Set Shortcut to Memory Set

Regular, Scope, Scoped, Full See Regular, Scope and Scoped, Full

A, B, C, D Select the crescendo bank to use. The current bank is highlighted

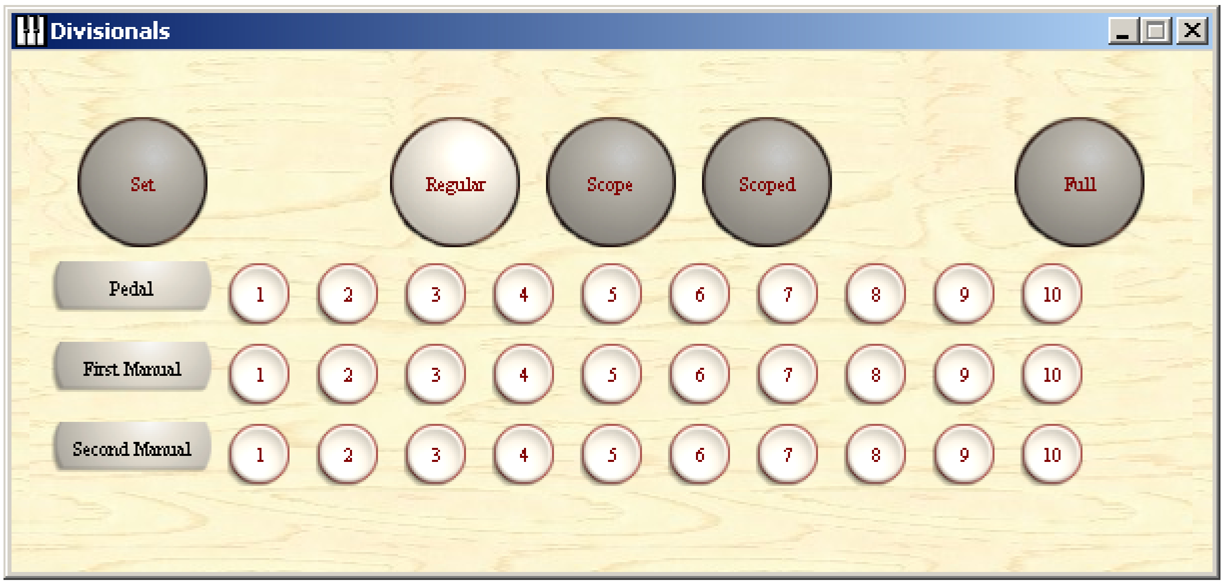
< Move to the previous step. While setting, store the state of drawstops and couplers (according to regular or scope/scoped) into the previous step.

Current While setting, store the state of drawstops and couplers (according to regular or scope/scoped) into the current crescendo step, otherwise recall the current step's state.

> Move to the next step. While setting, store the state of drawstops and couplers (according to regular or scope/scoped) into the next step.

Crescendo shoe This shoe controls the crescendo. It behaves like an expression shoe. If the physical console has a dedicated MIDI controller, the 32 steps are evenly spread across the range of values set in the MIDI event editor

8.4 Divisionals



GrandOrgue automatically adds a set of 10 divisional combination stores to each keyboard or pedal board found in the Organ Definition file. This enables the user to take advantage of "modern" accessories even when the sampleset doesn't provide them.

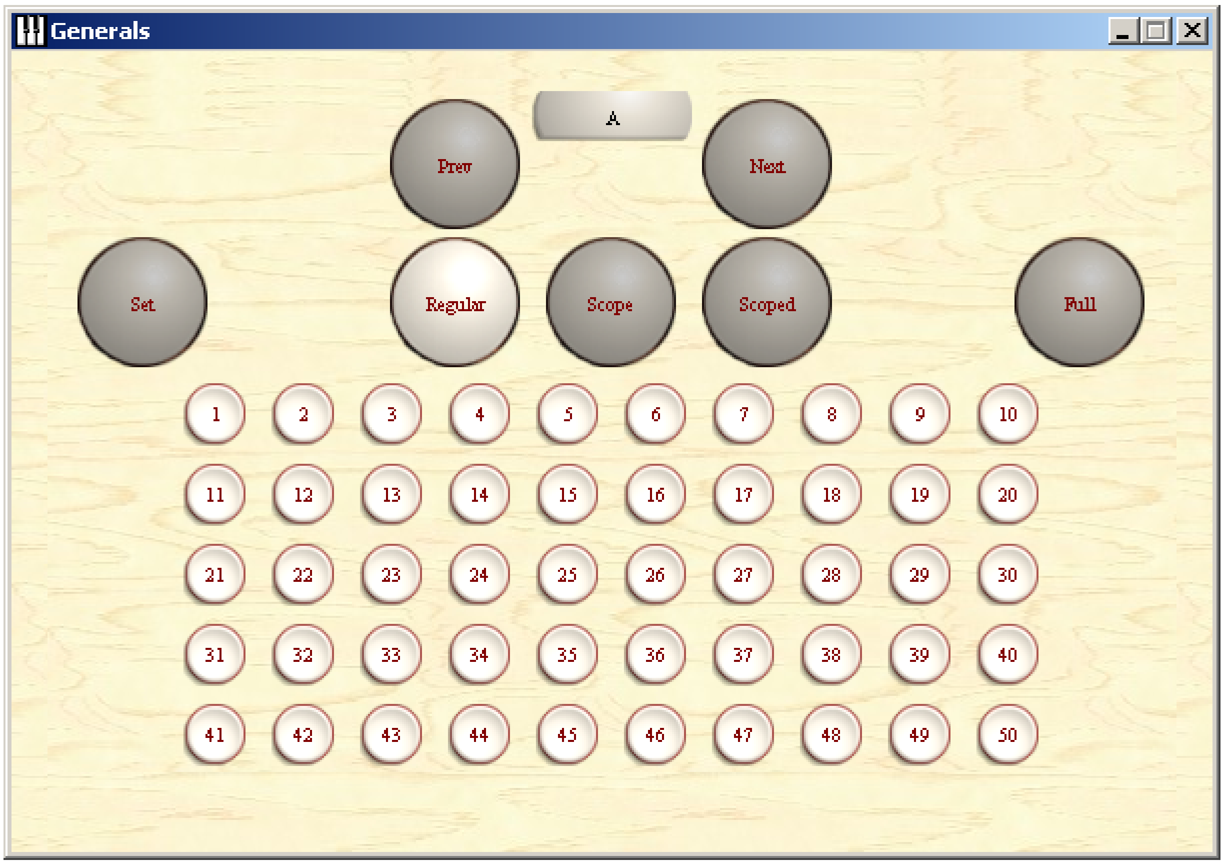
Set Shortcut to Memory Set

Regular, Scope, Scoped, Full See Regular, Scope and Scoped, Full

Notes

- All buttons in this panel are enabled for a Midi Event configuration.
- All configuration done on this panel can be saved to the current preset number. If not, GrandOrgue always asks to save on sampleset unloading.

8.5 Generals



GrandOrgue automatically adds 4 banks of 50 general combination stores to any loaded sampleset. This enables the user to take advantage of "modern" accessories even when the sampleset doesn't provide them.

Prev Switch to the previous generals bank

Label in between Displays the current bank. Values are A, B, C, D. It is an "active" label, and as such supports a Midi send configuration

Next Switch to the next generals bank

Set Shortcut to Memory Set

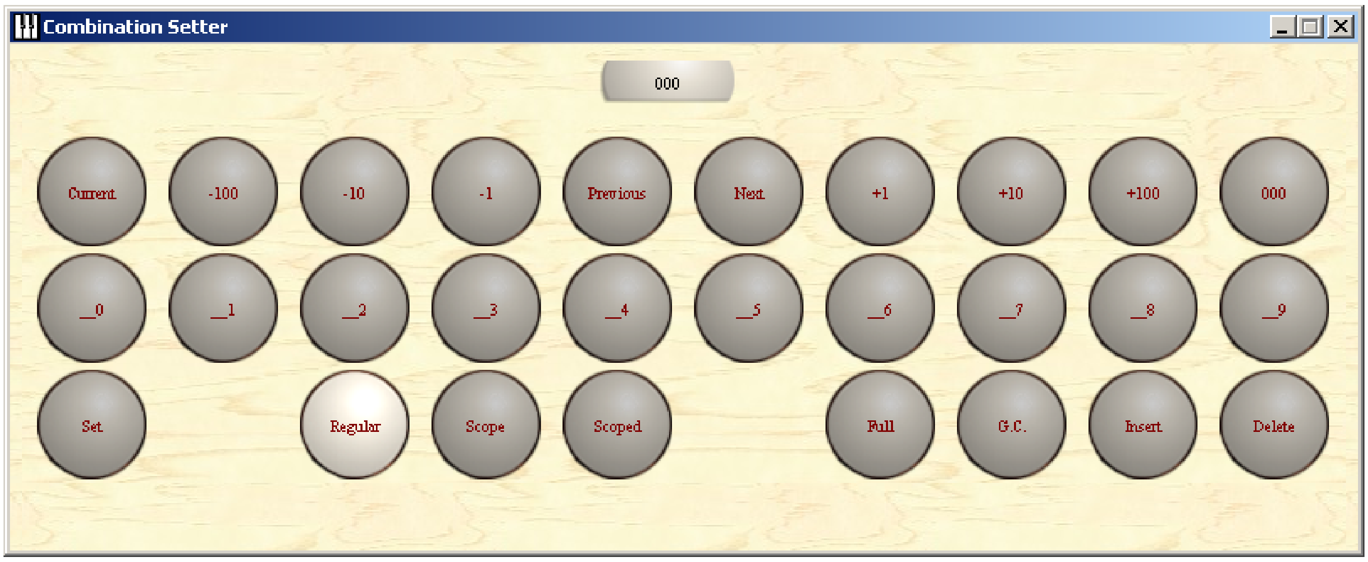
Regular, Scope, Scoped, Full See Regular, Scope and Scoped, Full

---

Notes

- All buttons in this panel are enabled for a Midi Event configuration.
- All configuration done on this panel can be saved to the current preset number. If not, GrandOrgue always asks to save on sampleset unloading.

8.6 Combination Setter



GrandOrgue automatically adds a 1000-slot combination setter to any loaded sampleset. This enables the user to take advantage of "modern" accessories even when the sampleset doesn't provide them.

Label on top Displays the current slot. It is an "active" label, and as such supports a Midi send configuration

Current While setting, store the state of drawstops and couplers (according to regular or scope/scoped) into the current slot, otherwise recall the current slot's state.

-100, +100 Navigate 100 slots backwards or forward without recalling the combination (prepare it)

-10, +10 Navigate 10 slots backwards or forward without recalling the combination (prepare it)

-1, +1 Navigate 1 slot backwards or forward without recalling the combination (prepare it)

Next While setting, store the state of drawstops and couplers (according to regular or scope/scoped) into the next slot, otherwise recall the next slot's state.

Previous While setting, store the state of drawstops and couplers (according to regular or scope/scoped) into the previous slot, otherwise recall the previous slot's state.

__0 .. __9 While setting, navigate to slot xx0 to xx9 and store the state of drawstops and couplers (according to regular or scope/scoped), otherwise navigate to slot xx0 to xx9 and recall the combination. The first two digits were previously set by the +/-100 and +/-10 controls.

Set Shortcut to Memory Set

Regular, Scope, Scoped, Full See Regular, Scope and Scoped, Full

G.C. General cancel. Resets the state to "all pushed"

Insert Shifts right all slots following the current slot. The behavior changes depending on the state of the Set button.

Set ON the state of drawstops and couplers (according to regular or scope/scoped) is stored in the current slot

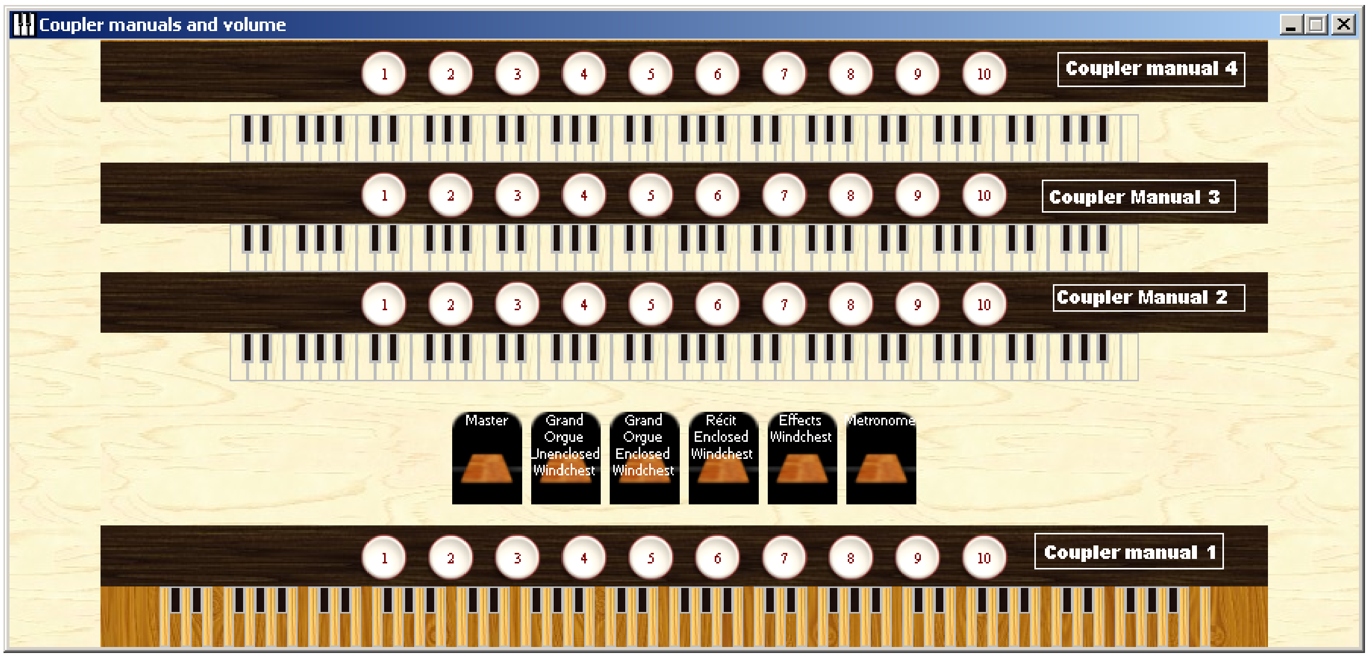
Set OFF the state of the current slot (including scope) is kept unmodified

Note
The combination stored in the last slot (#999) is **lost**

Delete Shifts left all slots following the current slot. The state of the Set button does not change the behavior.

Note
The combination stored in the current slot is lost. The combination stored in slot #999 is reset to "empty".

8.7 Coupler Manuals and Volume



This panel displays a III/P fictional console with swell shoes. The keyboards are coupler manuals. This fictional console is independent of the actual layout of the sample set.

8.7.1 Coupler manuals

A coupler manual is mute, as it is not connected to a division. By design, any keyboard (including pedal) can be coupled to a coupler manual (see Couplers>).

A set of 4 coupler manuals is added to each loaded sample set, whatever the actual number of manuals and/or pedal boards. The coupler manuals are numbered bottom to top. The coupler manual number 0 is displayed as pedal.

The displayed compass is derived from the lowest and highest MIDI key number found in the organ definition file. Each coupler manual is fit with 10 divisional pistons.

8.7.2 Expression shoes

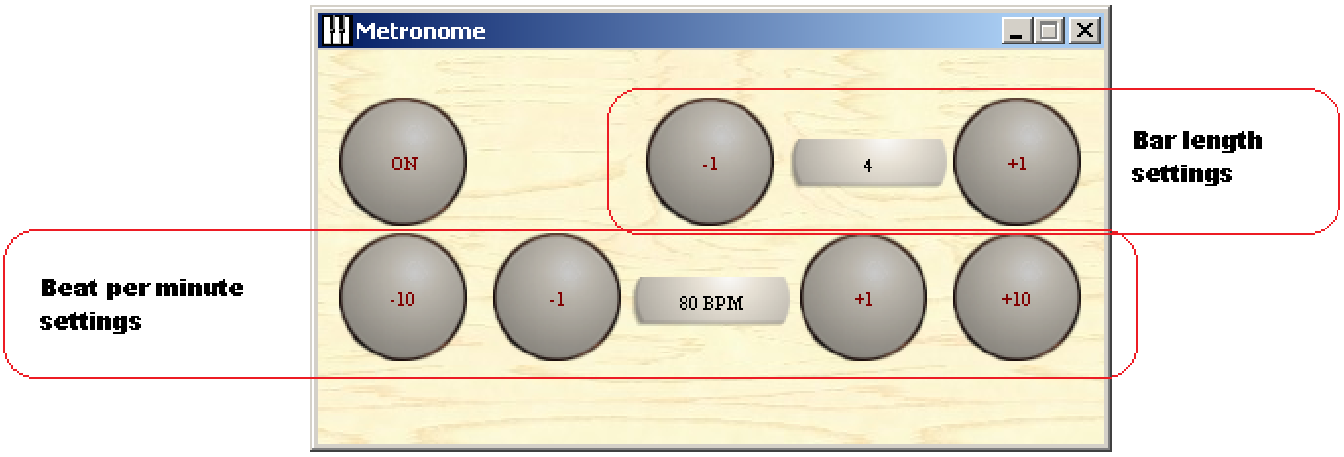
An enclosure expression shoe is added to each windchest found in the Organ Definition File, plus another designed to control the whole virtual organ volume.

Note
The embedded metronome is managed using its own separate windchest.

These enclosure settings are NOT replicated anywhere in the user interface.

These enclosure settings are saved in the current settings file only when the File > Save menu item is used.

8.8 Metronome



This panel controls the embedded metronome.

8.8.1 ON

This is the On / Off button. Starts or stops the metronome.

8.8.2 Bar length settings

This section manages the number of metronome beats per bar. The first beat of each bar is emphasized using a different sound. The initial value when the panel is first opened is 4.

If the number of beats per bar is set to 0, the "other beat" tick sound is used.

If the number of beats per bar is set to 1, the "first beat" tick sound is used.

-1	Decreases the number of beats per bar by 1.
+1	Increases the number of beats per bar by 1.
Label in between	This label displays the current beats per bar number. It is an *active* label, meaning that it can be configured to send MIDI messages.

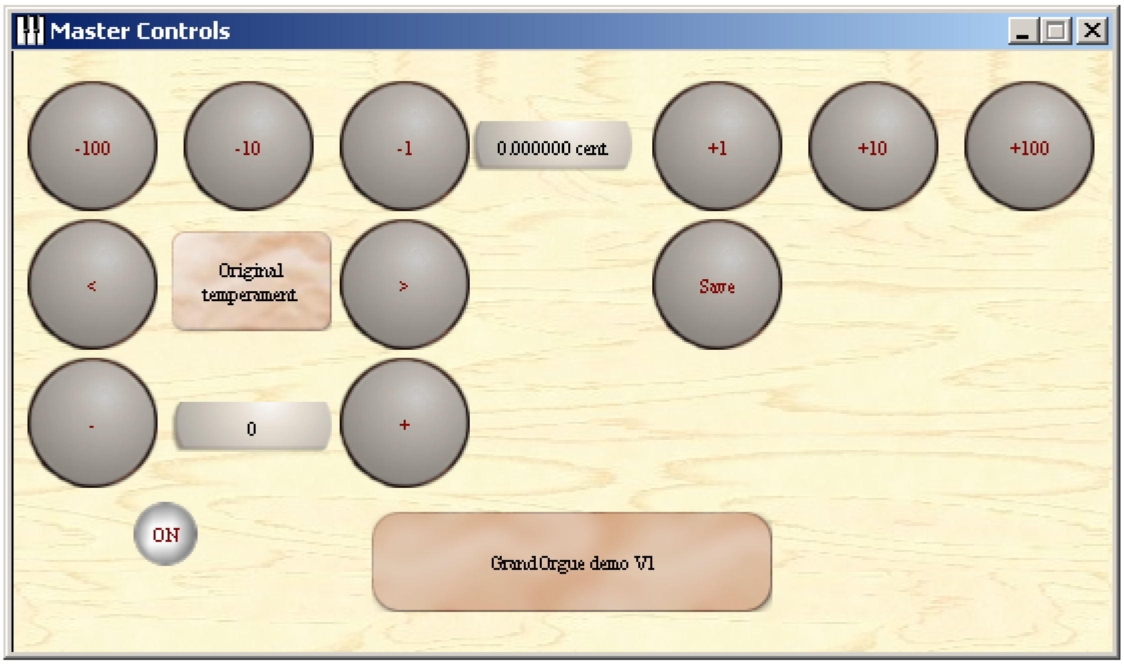
8.8.3 Beat per minute settings

This section manages the number of metronome beats per minute. The initial value when the panel is first opened is 80.

-10	Decreases the number of beats per minute by 10.
-1	Decreases the number of beats per minute by 1.
+1	Increases the number of beats per minute by 1.
+10	Increases the number of beats per minute by 10.
Label in between	This label displays the current beats per minute number. It is an *active* label, meaning that it can be configured to send MIDI messages.

Note
The metronome settings can be saved in the current settings file and remembered the next time the sampleset is loaded. Use the File > Save menu item, as there is no warning to save when the sampleset is closed with changed metronome settings.

8.9 Master Controls



This panel groups at the same place some controls that are used while playing. Each displayed element can be associated to a MIDI event and thus linked to a physical control on the console. This enables the user to ignore the computer and concentrate on his playing.

Master controls are specific to the console hardware. As such, their MIDI configuration can also be defined in the *Midi & Audio settings*, Initial Midi Configuration tab.

8.9.1 Tuning

The first row controls the ability to tune the whole organ. All tuning done from the master controls panel is mirrored in the Organ Settings, *Settings* frame, *Tuning (Cent)* spinner and will be saved in the current preset file.

-100, -10, -1 buttons	Tune down by amount of 100, 10, 1 cent(s) respectively
+100, +10, +1 buttons	Tune up by amount of 100, 10, 1 cent(s) respectively
Label in between	This label displays the current tuning amount. It is an "active" label, and as such supports a Midi send configuration

Reminder
The tuning unit is 1/100th of a semitone. The allowed range is {-1200, +1200 }, i.e. {-1, +1} octave. A semitone is 100 tuning units.

8.9.2 Temperaments

The second row controls the ability to navigate the temperaments list. The temperament selection in the master controls panels is mirrored in the Temperaments menu item of the *Audio/Midi* menu.

< button	Go to the previous temperament in the list
> button	Go to the next temperament in the list
Label in between	This label displays the current temperament name. It is an "active" label, and as such supports a Midi send configuration

8.9.3 Save

The *Save* button in the middle row saves the current sampleset customization in the current preset file. It has the same usage as the Save menu item in the *File* menu.

8.9.4 Transposer

The third row controls the transposer. The transposer state is mirrored in the transposer spinner in the toolbar.

< button	Shift all keyboards down one semitone
> button	Shift all keyboards up one semitone
Label in between	This label displays the current transposer state. It is an "active" label, and as such supports a Midi send configuration

8.9.5 Readiness indicators

The fourth row displays readiness indicators which change state when the sample set finishes loading

ON "LED"	This "LED" is rendered as a lighted button and as such supports a Midi send configuration
Label	This label displays the organ name. It is an "active" label, and as such supports a Midi send configuration

Chapter 9

Frequently Asked Questions

This section will be populated once questions become frequently asked.

Chapter 10

The Grand Orgue file format

10.1 General information

It is a text file in ISO-8859-1 encoding. The standard extension is *.organ*. As an alternative, the file might be encoded in UTF-8, if it starts with an approriate byte order marker.

Comment lines are started with *;* in the first column. Empty lines are ignored.

Various settings are grouped in blocks. Each block is started with a line starting with *[*, followed by the section name and ended with *]*. Each block consists of list of name-values pairs separated by =. Each setting may only occur once. The content is case sensitive. No other elements are allowed in the file.

File paths in this format are relative to the location of the organ file. The directory separator in these paths must be \. The paths should not contain */*. The paths should be considered case sensitive, even if this is not enforced on all platforms.

Boolean values are represented as *Y* for "true" and *N* for "false".

A color can be one of the following colors (not case sensitive): BLACK, BLUE, DARK BLUE, GREEN, DARK GREEN, CYAN, DARK CYAN, RED, DARK RED, MAGENTA, DARK MAGENTA, YELLOW, DARK YELLOW, LIGHT GREY, DARK GREY, WHITE, BROWN. Additionally, the HTML syntax #RRGGBB is supported.

A font size can be one of the following values: SMALL, NORMAL, LARGE or an integer number between 1 and 50.

A panel size can be one of the following values: SMALL, MEDIUM, MEDIUM LARGE, LARGE or an integer number between 100 and 4000.

The following image formats are supported: bmp, gif, jpg, ico, png

A bitmap number is a value between 1 and 64. It refers to a predefined (wood) background. An odd number and the following even number represent the same type of wood. The odd number has vertical grain, while the odd has horizontal grain.

Floating point numbers use the following format: *-?[0-9]+(.[0-9]*)?* That means: a optional minus sign, followed by at least one digit. The decimal separator is a period.

The unit "samples" counts the number of samples from the start of the WAV file. On sample includes the values of all channels, eg: for a stereo WAV file at 44.1 kHz, 1 second is equivalent to 44100 samples.

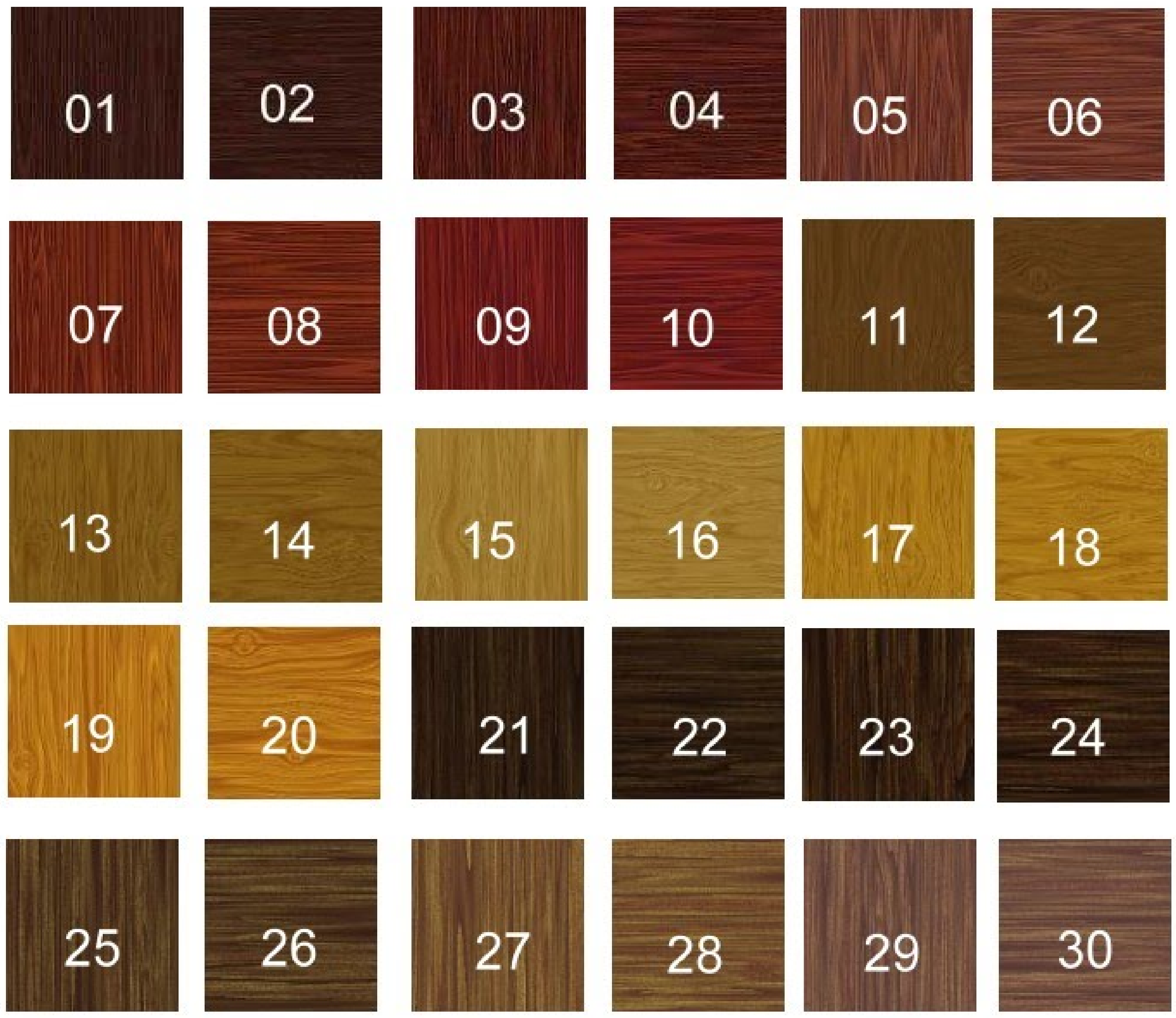


Figure 10.1: background bitmaps 1 to 30

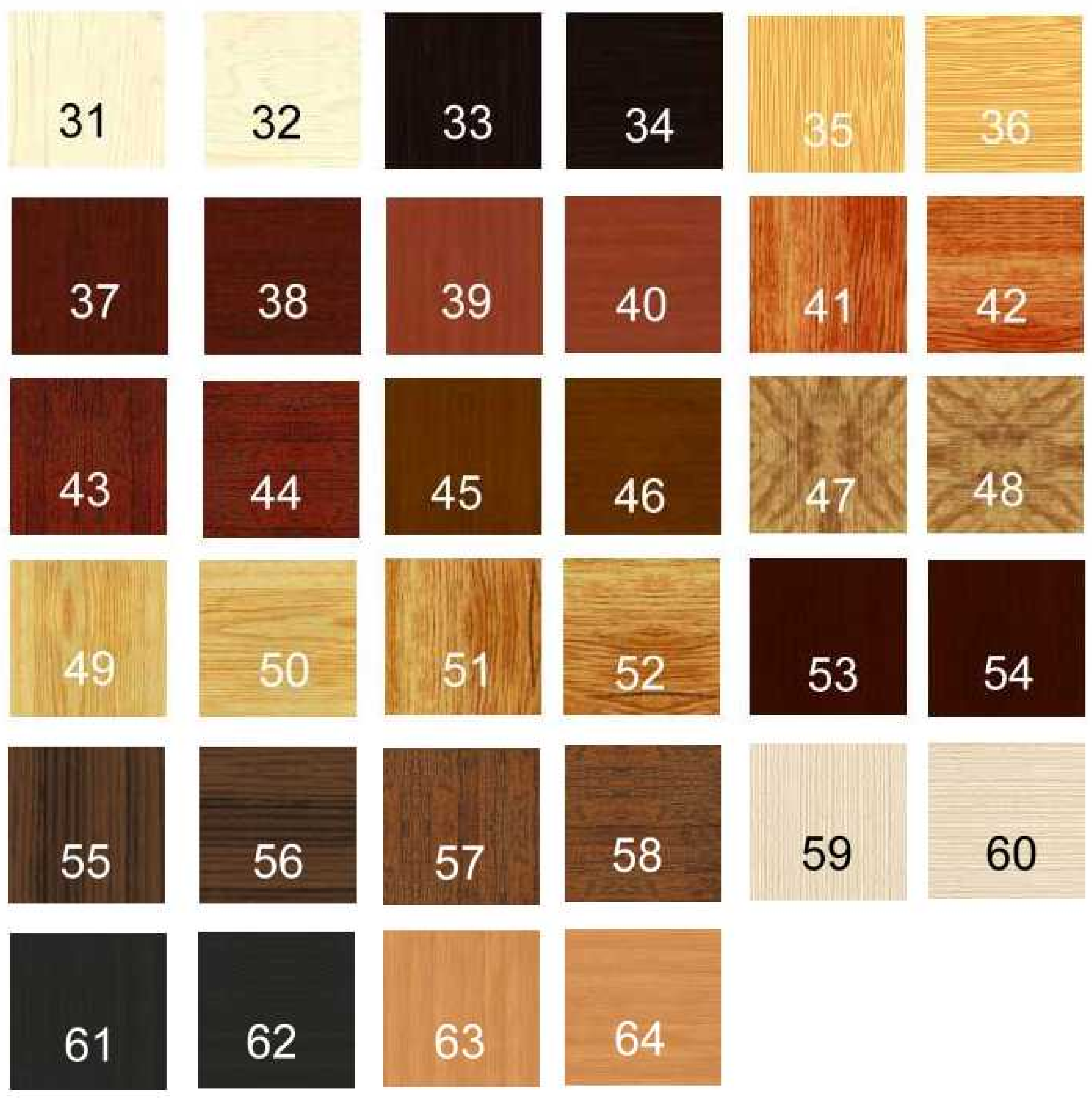


Figure 10.2: background bitmaps 31 to 64

Objects in GrandOrgue are e.g. Manuals, Stops, Generals, ... Each object can consist of a backend part representing the object, and its configuration (eg. list of sample file names) and multiple GUI representations.

The main section is called *Organ*. It defines the main panel. Objects like stops only displayed on a non-main panel need an invisible definition for the backend part on the main panel too.

10.2 Sample format

The samples are stored as WAV files according to the WAV file specification. The supported formats are: 8 bit, 16 bit and 24 bit PCM or 32 bit IEEE float, either mono or stereo. The preferred sample rates are 44100 or 48000 Hz - GO supports any sample rate between 22000 and 96000 Hz. GO only supports a single data chunk. To play looped samples, they must include cue points (cue chunk) and loops (smpl chunk). If there are multiple loops, each loop should overlap another loop. Attack samples include the attack phase and the loops - they may contain a release too. If release samples include a cue point, the release is loaded starting with this position, else the whole file is loaded. dwMIDIUnityNote and dwMIDIPitchFraction of the smpl chunk are used for retuning an organ to other temperaments.

If multiple sample files are specified for one pipe, they must match each other in regards to pitch, amplitude and other things. GrandOrgue only allows adjustments at pipe level.

The samples in the format above may also be packed with wavpack http://www.wavpack.com/. GO only supports the wavpack V4 format without hybrid compression and it must include all metadata. A good starting point for the wavpack compression options is *-x6*

10.3 Shortcut keys

The following key codes for shortcut keys are supported by GO:

8 Backspace key

48 0 key

49 1 key

50 2 key

51 3 key

52 4 key

53 5 key

54 5 key

55 6 key

56 8 key

57 9 key

65 A key

66 B key

67 C key

68 D key

69 E key

70 F key

71 G key

72 H key

73 I key

74 J key

75	K key
76	L key
77	M key
78	N key
79	O key
80	P key
81	Q key
82	R key
83	S key
84	T key
85	U key
86	V key
87	W key
88	X key
89	Y key
90	Z key
112	F1 key
113	F2 key
114	F3 key
115	F4 key
116	F5 key
117	F6 key
118	F7 key
119	F8 key
120	F9 key
121	F10 key
122	F11 key
123	F12 key

10.4 Organ section

This section describes the whole organ. For the old panel format, it includes the main panel, therefore the section includes all display metrics attributes. The new panel format separates the display metrics of the main panel into a section named *Panel000*. Additionally it includes the following attributes:

ChurchName (string, required) Name of the organ/church. This string should be unique, as setting files for organs with the same *ChurchName* are considered compatible. Grand Orgue will not load a settings file if the ChurchName does not match.

ChurchAddress (string, required) informational text displayed in the property dialog

OrganBuilder (string, required) informational text displayed in the property dialog

OrganBuildDate (string, required) informational text displayed in the property dialog

OrganComments (string, required) informational text displayed in the property dialog

RecordingDetails (string, required) informational text displayed in the property dialog

InfoFilename (string, not required) relative path to an html file with more information about the organ. This setting is currently not supported for organ packages.

NumberOfManuals (integer 1-16, required) number of manuals. It does not include the pedal keyboard. The manual information for each manual is available in sections called *Manual999*. 999 is a number defining each manual, starting with 001.

HasPedals (boolean, required) Determines if the pedal, which is defined as section *Manual000*, is present.

NumberOfEnclosures (integer 0-50, required) number of enclosures. The details of each enclosure are contained in a section called *Enclosure999*

NumberOfTremulants (integer 0-10, required) number of tremulants. The details of each tremulant are contained in a section called *Tremulant999*

NumberOfWindchestGroups (integer 1-50, required) number of windchests. The details of each windchest are in a section called *WindchestGroup999*.

NumberOfReversiblePistons (integer 0-32, required) number of reversible pistons. The details of each reversible piston are in a section called *ReversiblePiston999*.

NumberOfGenerals (integer 0-99, required) number of generals. The details are in a section called *General999*.

NumberOfDivisionalCouplers (integer 0-8, required) number of divisional couplers. The details are in a section called *DivisionalCoupler999*.

NumberOfPanels (integer 0-100, required) number of additional panels. The details are in a section called *Panel999*.

NumberOfSwitches (integer 0-999, default: 0) number of switches. The details are in a section called *Switch999*.

NumberOfRanks (integer 0-400, default: 0) number of additional ranks. The details are in a section called *Rank999*.

DivisionalsStoreIntermanualCouplers (boolean, required) determines if divisionals store/change the state of associated intermanual couplers.

DivisionalsStoreIntramanualCouplers (boolean, required) determines if divisionals store/change the state of associated intramanual couplers.

DivisionalsStoreTremulants (boolean, required) determines if divisionals store/change the state of associated tremulants.

GeneralsStoreDivisionalCouplers (boolean, required) determines if divisionals store/change the state of divisional couplers.

CombinationsStoreNonDisplayedDrawstops (boolean, default: true) determines, if the state of invisible objects (on the main panel) is stored in divisionals, generals and the setter.

NumberOfImages (integer 0-999, default: 0) Number of images on the panel. The section of the label GUI definitions are called *Image999*. This setting is not supported for the new panel format.

NumberOfSetterElements (integer 0-999, default: 0) Number of setter elements on the panel. The section of the GUI definitions are called *SetterElement999*. This setting is not supported for the new panel format.

NumberOfLabels (integer 0-999, required) Number of labels on the panel. The section for each label GUI definition is called *Label999*. This setting is not supported for the new panel format.

AmplitudeLevel (float 0-1000, default: 100) Linear amplitude scale factor applied to the whole organ. 100 means no change.

Gain (float -120 - 40, default: 0) Amplitude scale factor in dB applied to the whole organ. 0 means no change.

PitchTuning (float -1200-1200, default: 0) Retune the whole organ the specified number of cents.

TrackerDelay (integer 0 - 10000, default: 0) Delay introduced by the tracker applied to the whole organ.

10.5 Setter elements

It is possible to display various setter elements on the panels. The *Type* attribute describes the function - the other attributes depend on the function:

Swell Represents the crescendo swell. The section must contain the GUI attributes of an enclosure, eg the images for each position of the swell pedal, etc.

CrescendoLabel Represents a label with crescendo state (ie the current step that the crescendo pedal is at, from 1 to 32.) The section must contain the GUI attributes of a label.

Label Represents the current number of the setter. The section must contain the GUI attributes of a label.

Prev, Next, Set Prev/next/set button of the setter. The section must contain the GUI attributes of a button (displayed as button per default).

M100, M10, M1, P1, P10, P100 +/- 1/10/100 button of the setter. The section must contain the GUI attributes of a button (displayed as button per default).

Home Move to 000 button of the setter. The section must contain the GUI attributes of a button (displayed as button per default).

Current Recall current number button of the setter. The section must contain the GUI attributes of a button (displayed as button per default).

GC General cancel button of the setter. The section must contain the GUI attributes of a button (displayed as button per default).

L0, L1, L2, L3, L4, L5, L6, L7, L8, L9 Recall combination with the specified digit as last digit button of the setter. The section must contain the GUI attributes of a button (displayed as button per default).

Regular, Scope, Scoped Button to switch between the various setter modes. The section must contain the GUI attributes of a button (displayed as button per default).

Full Button to enable storing all elements in the setter (restrictions from the ODF are ignored). The section must contain the GUI attributes of a button (displayed as button per default).

Insert, Delete Buttons to insert/remove a combination. The section must contain the GUI attributes of a button (displayed as button per default).

General01 - General50 Buttons for the setter generals. The section must contain the GUI attributes of a button (displayed as button per default).

GeneralPrev, GeneralNext Buttons for switching banks of the setter generals. The section must contain the GUI attributes of a button (displayed as button per default).

GeneralLabel Represents a label with the general bank number. The section must contain the GUI attributes of a label.

CescendoA, CrescendoB, CrescendoC, CrescendoD Buttons to switch between the various crescendo modes. The section must contain the GUI attributes of a button (displayed as button per default).

CrescendoPrev, CrescendoNext, CrescendoCurrent Buttons for controling the crescendo combinations. The section must contain the GUI attributes of a button (displayed as button per default).

PitchP1, PitchP10, PitchP100, PitchM1, PitchM10, PitchM100 Buttons for controlling the organ pitch (+1, +10, +100, -1, -10, -100 cent).

PitchLabel Label displaying the current pitch shift of the organ.

TemperamentPrev, TemperamentNext Buttons for switching temperaments.

TemperamentLabel Label displaying the current temperament of the organ.

TransposeDown, TransposeUp Buttons for transposing.

TransposeLabel Label displaying the current transpose setting.

Save Save button.

10.6 Display metrics

The layout of any panel is described by a layout model. The available space is split vertically into three columns: The left and the right column contain drawstops laid out via a mesh. The middle row is vertically divided:

At the bottom, the pedal and its buttons are placed (if present). Above, an extra row of buttons may follow. The next row contains the enclosures. Then all manuals with their associated buttons follow. The two rows are a block of buttons and pistons. The exact order can be specified via an attribute.

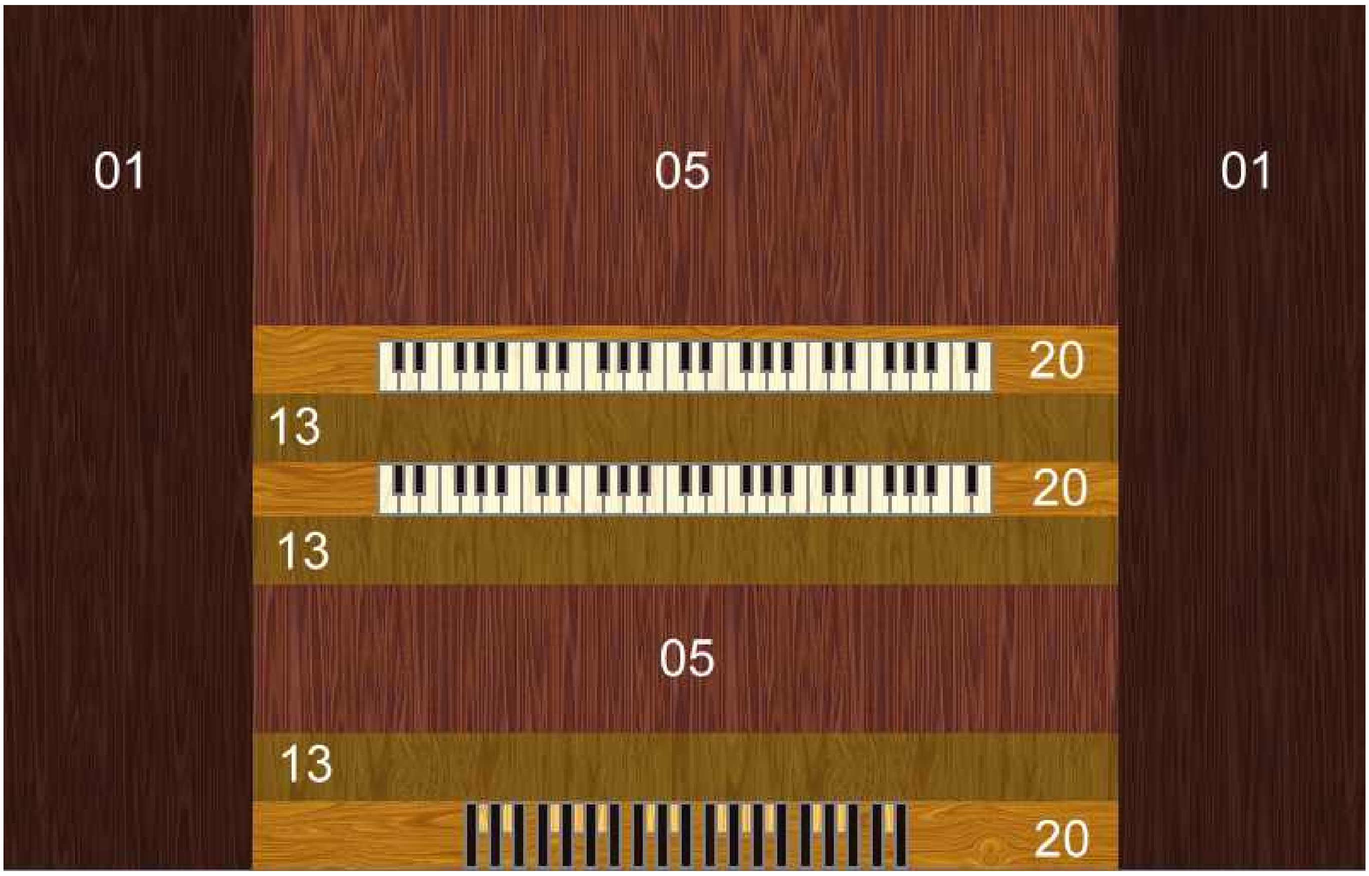


Figure 10.3: background regions

Has the following attributes:

DispScreenSizeHoriz (panel size, required) Height of the panel.

DispScreenSizeVert (panel size, required) Width of the panel.

DispDrawstopBackgroundImageNum (bitmap number, required) shown as 01 in the above image.

DispConsoleBackgroundImageNum (bitmap number, required) shown as 05 in the above image.

DispKeyHorizBackgroundImageNum (bitmap number, required) shown as 13 in the above image

DispKeyVertBackgroundImageNum (bitmap number, required) shown as 20 in the above image

DispDrawstopInsetBackgroundImageNum (bitmap number, required)

DispControlLabelFont (string, required) Name of the font for button labels.

DispShortcutKeyLabelFont (string, required) Name of the font for keyboard labels

DispShortcutKeyLabelColour (color, required) Color for shortcut labels

DispGroupLabelFont (string, required) font name for labels

DispDrawstopCols (integer 2-12, required) number of drawstop columns. Must be even. NOTE: If you want more than 12 drawstop columns, you must use absolute positioning.

DispDrawstopRows (integer 1-20, required) Number of drawstop rows. NOTE: If you want more than 20 drawstop rows you must use absolute positioning.

DispDrawstopColsOffset (boolean, required) If true, each second row of drawstops on the left/right is displayed vertically shifted.

DispDrawstopOuterColOffsetUp (boolean, required if DispDrawstopColsOffset is true) Determines if second row is shifted up or down.

DispPairDrawstopCols (boolean, required) group two drawstop rows together. Number of drawstop rows must be divisible by 4.

DispExtraDrawstopRows (integer 0-99, required) number of drawstop rows in the center block. The row numbers start with 100.

DispExtraDrawstopCols (integer 0 - 40, required) number of drawstop cols in the center block

DispButtonCols (integer 1-32, required) Number of columns for displaying pistons in the center block

DispExtraButtonRows (integer 0-99, required) number of rows for displaying extra pistons in the center block. The row numbers start with 100.

DispExtraPedalButtonRow (boolean, required) display an extra piston row with row number 9.

DispExtraPedalButtonRowOffset (boolean, required if DispExtraPedalButtonRow is true) move extra pistons row slightly to the left.

DispExtraPedalButtonRowOffsetRight (string, required if DispExtraPedalButtonRow is true) move extra pistons row slightly to the right.

DispButtonsAboveManuals (boolean, required) Display the pistons associated with the manual above (true) or below (false) the manual.

DispTrimAboveManuals (boolean, required)

DispTrimBelowManuals (boolean, required)

DispTrimAboveExtraRows (boolean, required)

DispExtraDrawstopRowsAboveExtraButtonRows (boolean, required) Display extra drawstop block above or below the extra piston block.

DispDrawstopWidth (integer 1-150, default: 78) Drawstop width used for layout calculation

DispDrawstopHeight (integer 1-150, default: 69) Drawstop height used for layout calculation

DispPistonWidth (integer 1-150, default: 44) Piston width used for layout calculation

DispPistonHeight (integer 1-150, default: 40) Piston height used for layout calculation

DispEnclosureWidth (integer 1-150, default: 52) Enclosure width used for layout calculation

DispEnclosureHeight (integer 1-150, default: 63) Enclosure width used for layout calculation

DispPedalHeight (integer 1-500, default: 40) Pedal height used for layout calculation

DispPedalKeyWidth (integer 1-500, default: 7) Width of one pedal key used for layout calculation

DispManualHeight (integer 1-500, default: 32) Manual height used for layout calculation

DispManualKeyWidth (integer 1-500, default: 12) Width of one manual key used for layout calculation

10.7 Enclosure objects

An enclosure represents a swell pedal. It consists of non-gui attributes describing its function. If it is displayed, it contains additional gui attributes. Best practise is to specify enclosures in natural layout order (leftmost first) and give them incremental values of MIDIInputNumber to make initial configurations easy for the user.

Name (string, required) Name of the control

AmpMinimumLevel (integer 0-100, required) Minimum volume, if the enclosure is closed.

MIDIInputNumber (integer 0 - 200, default: 0) This number is used while building the initial MIDI configuration to map the enclosure object to one MIDI device the user can specify for the respective enclosure. A value of 0 means no association, 1 means enclosure 1, 2 is enclosure 2 etc. Please note, that the GUI only allows the association of the first few enclosures.

Displayed (boolean, default: false for the new panel format, otherwise true) If true, the enclosure is visible on the main panel.

If the enclosure is displayed, it contains the following gui attributes:

DispLabelColour (color, default: Dark Red) Color for the label text.

DispLabelFontSize (font size, default: 7) Size of the label font

DisplLabelFontName (string, default: empty) Font for the text. Empty means use the default font.

DispLabelText (string, default: Name of the button) Content of the text label. You should edit it if you need to display a shorter string.

EnclosureStyle (integer 1 - 4, default: implementation dependent) Select a built-in enclosure style.

BitmapCount (integer 1 - 127, default: implementation dependent) Number of bitmaps/steps.

Bitmap999 (string, default: use internal bitmap) Specify the file name of an image to use as on bitmap for position 999. If the bitmap contains a mask for transparency, it will be used. All bitmaps must have the same size.

Mask999 (string, default: empty) File name for a external mask for bitmap 999. If empty, no mask is added.

PositionX (integer 0 - panel width, default: according to layout model) Allow to override X position for enclosure.

PositionY (integer 0 - panel height, default: according to layout model) Allow to override Y position for enclosure.

Width (integer 0 - panel width, default: bitmap width) Width of the enclosure. If larger than the bitmap, the bitmap is tiled.

Height (integer 0 - panel height, default: bitmap height) Height of the enclosure. If larger than the bitmap, the bitmap is tiled.

TileOffsetX (integer 0 - bitmap width, default: 0) X position on the bitmap of the left pixel of the enclosure

TileOffsetY (integer 0 - bitmap width, default: 0) Y position on the bitmap of the top pixel of the enclosure

MouseRectLeft (integer 0 - Width, default: 0) relative X of left border of the mouse rectangle

MouseRectTop (integer 0 - Height, default: implementation dependent) relative Y of top border of the mouse rectangle

MouseRectWidth (integer 0 - Width, default: Width) width of the mouse rectangle

MouseRectHeight (integer 0 - Height, default: implementation dependent) height of the mouse rectangle

MouseAxisStart (integer 0 - MouseRectHeight, default: implementation dependent) top Y coordinate of the axis

MouseAxisEnd (integer MouseAxisStart - MouseRectHeight, default: implementation dependent) bottom Y coordinate of the axis

TextRectLeft (integer 0 - Height, default: 0) relative X of left border of the text rectangle

TextRectTop (integer 0 - Height, default: implementation dependent) relative Y of top border of the text rectangle

TextRectWidth (integer 0 - Width, default: Width) width of the text rectangle

TextRectHeight (integer 0 - Height, default: implementation dependent) height of the text rectangle

TextBreakWidth (integer 0 - text rectangle width, default: TextWidth) If 0, no text is displayed. Otherwise the value specifies the maximum line length used for text breaking.

10.8 Combination data store

This object is used to store one combination It has the following attributes:

Protected (boolean, default: false) If true, the stored combination cannot be changed.

NumberOfStops (integer 0 - 999, required) Number of stop states stored in this combination. The entries are called *StopNumber999* and *StopManual999*.

NumberOfCouplers (integer 0 - 999, required) Number of coupler states stored in this combination. The entries are called *CouplerNumber999* and *CouplerManual999*.

NumberOfTremulants (integer 0 - 999, required) Number of tremulant states stored in this combination. The entries are called *TremulantNumber999*.

NumberOfSwitches (integer 0 - 999, default: 0) Number of switch states stored in this combination. The entries are called *SwitchNumber999*.

NumberOfDivisionalCouplers (integer 0 - divisional coupler count, required if storing of divisional coupler in the generals is enabled) Number of divisional coupler state stored in this combination. The entries are called *DivisionalCouplerNumber999*.

StopNumber999 (integer -manual stop count - manual stop count, required) Number of the stop on the manual. If the value is negative, it is turned off, else turned on.

StopManual999 (integer manual number, required) Number of the manual, which contains the stop.

CouplerNumber999 (integer -999 - 999, required) Number of the coupler on the manual. If the value is negative, it is turned off, else it is turned on.

CouplerManual999 (integer manual number, required) Number of the manual, which contains the coupler.

TremulantNumber999 (integer -tremulant count - tremulant count, required) Number of the tremulant. If the value is negative, it is turned off, else turned on.

SwitchNumber999 (integer -switch count - switch count, required) Number of the switch. If the value is negative, it is turned off, else it is turned on.

DivisionalCouplerNumber999 (integer -divisional coupler count -divisional coupler count, required) Number of the divisional coupler. If the value is negative, it is turned off, else it is turned on.

10.9 General object

A general object consists of a push button and a combination data store.

10.10 Divisional coupler objects

A divisional coupler is a drawstop object. If enabled, activating a divisional on one controlled manual will activate the corresponding divisional on all other manuals. Additionally, it has the following attributes:

BiDirectionalCoupling (boolean, required) If false, the coupler only couples upward in the manual list of the coupler, else upward and downward.

NumberOfManuals (integer 1 - manual count, required) Number of manuals affected by this coupler. The list entries are stored in *Manual999* setting.

Manual999 (integer manual number, required) Manual affected by the coupler

10.11 Divisional objects

A divisional is a pushbutton object. Additionally, it has the following attributes:

Protected (boolean, default: false) If true, the stored combination cannot be changed.

NumberOfStops (integer 0 - stop count of the manual, required) Number of stop states stored in this combination. The entries are called *Stop999*.

NumberOfCouplers (integer 0 - coupler count of the manual, required) Number of coupler states stored in this combination. The entries are called *Coupler999*.

NumberOfTremulants (integer 0 - tremulant count of the manual, required) Number of tremulant states stored in this combination. The entries are called *Tremulant999*.

NumberOfSwitches (integer 0 - switch count of the manual, default) Number of switch states stored in this combination. The entries are called *Switch999*.

Stop999 (integer -manual stop count - manual stop count, required) Number of the stop. If the value is negative, it is turned off, else turned on.

Coupler999 (integer -999 - 999, required) Number of the coupler. If the value is negative, it is turned off, else turned on.

Tremulant999 (integer -manual tremulant count - manual tremulant count, required) Number of the tremulant. If the value is negative, it is turned off, else it is turned on.

Switch999 (integer -manual switch count - manual switch count, required) Number of the switch. If the value is negative, it is turned off, else it isturned on.

10.12 Manual objects

A manual is associated with a number of stops, tremulants, divisionals and couplers. The accessible range can be played via MIDI, the rest of the logical keys can only be triggered by (octave) couplers. Best practise is to specify the visible manuals in the order of appearance, lowest first. Invisible manuals and those used for special effects should be specified after the visible ones. Manual objects contain the following non-gui attributes:

Name (string, required) Name of the manual

NumberOfLogicalKeys (integer 1-192, required) Number of keys on this manual (including non-playable ones).

FirstAccessibleKeyLogicalKeyNumber (integer 1 - NumberOfLogicalKeys, required) number of the first usable key

FirstAccessibleKeyMIDINoteNumber (integer 0 - 127, required) MIDI note number of the first MIDI acessible key.

NumberOfAccessibleKeys (integer 0 - 85, required) number of MIDI accessible keys.

MIDIInputNumber (integer 0 - 200, default: 0) This number is used while building the initial MIDI configuration to map the manual object to what MIDI device the the user has set for the respective pedal/manual. 0 means no association. 1 maps to pedal, 2 to first manual, 3 to second manual etc. NOTE: the GUI only allows the association of the first few manuals. Second touch manuals can be set to the same number as the main manual as the user then only has to configure the low velocity to make it work.

Displayed (boolean, default: false) If true, the manual is visible on the main panel.

NumberOfStops (integer 0-999, required) Number of stops associated with this manual. Starting with 1, for each stop, there is a *Stop999* setting.

Stop999 (integer, required) Number of the *Stop999* section containing the stop details.

NumberOfCouplers (integer 0-999, default: 0) Number of couplers associated with this manual. Starting with 1, for each coupler, there is a *Coupler999* setting.

Coupler999 (integer, required) Number of the *Coupler999* section containing the coupler details.

NumberOfDivisionals (integer 0-999, default: 0) Number of divisionals associated with this manual. Starting with 1, for each divisional, there is a *Divisional999* setting.

Divisional999 (integer, required) Number of the *Divisional999* section containg the coupler details.

NumberOfTremulants (integer 0 - number of tremulants, default: 0) Number of tremulants associated with this manual. Starting with 1, for each tremulant, there is a *Tremulant999* setting.

Tremulant999 (integer, required) Number of the *Tremulant999* section containg the tremulant details.

NumberOfSwitches (integer 0 - number of switches, default: 0) Number of switches associated with this manual. Starting with 1, for each switch, there is a *Switch999* setting.

Switch999 (integer, required) Number of the *Switch999* section containg the switch details.

MIDIKey000 - MIDIKey127 (integer 0-127, default: same MIDI key number) Allows to map the MIDI note in *MIDIKey999* to a different number. This mapping is used by the default manual MIDI matching type - others may or may not use this mapping table.

The various GUI manual attributes are specified for the different key types. They are named: C, Cis, D, Dis, E, F, Fis, G, Gis, A, Ais, B. If it is the first key on the manual, it is prefixed with First. If it is the last key on the manual, it is prefixed with Last. So valid values are eg. Gis, FirstDis, LastAis. In the following, these values will be marked as *KEYTYPE*.

If the manual is displayed, it contains the following gui attributes:

PositionX (integer 0 - panel width, default: according to layout model) Allow to override X position for manual.

PositionY (integer 0 - panel height, default: according to layout model) Allow to override Y position for manual.

ImageOn_*KEYTYPE* (string, default: implementation dependent bitmap) Bitmap for the specified key type, if the key is pressed. The bitmap may contain a mask.

ImageOff_*KEYTYPE* (string, default: implementation dependent bitmap) Bitmap for the specified key type, if the key is not pressed. The bitmap may contain a mask.

MaskOn_*KEYTYPE* (string, default: empty string) Mask for the corresponding On bitmap. If empty, no external mask is loaded.

MaskOff_*KEYTYPE* (string, default: corresponding on mask) Mask for the corresponding Off bitmap. If empty, no external mask is loaded.

Width_*KEYTYPE* (integer 0 - 500, default: implementation dependent) This value is added to the x position of the current key to determine the position of the next key.

Offset_*KEYTYPE* (integer -500 - 500, default: implementation dependent) This value can be used to adjust the display of the current key, eg. to place a sharp key overlapped with the previous key.

YOffset_*KEYTYPE* (integer 0 - 500, default: 0) This value is can be used to adjust the Y coordinate of the current key.

Key999ImageOn (string, default: corresponding ImageOn_*KEYTYPE*) Allows to set the on bitmap for the 999 key.

Key999ImageOff (string, default: corresponding ImageOff_*KEYTYPE*) Allows to set the off bitmap for the 999 key.

Key999MaskOn (string, default: corresponding MaskOn_*KEYTYPE*) Allows to set the on mask for the 999 key.

Key999MaskOff (string, default: corresponding MaskOff_*KEYTYPE*) Allows to set the off mask for the 999 key.

Key999Width (integer 0 - 500, default: corresponding Width_*KEYTYPE*) Allows to set the width of the 999 key.

Key999Offset (integer -500 - 500, default: corresponding Offset_*KEYTYPE*) This value is can be used to adjust the display of the 999 key, eg. to place a sharp key overlapped with the previous 999 key.

Key999YOffset (integer 0 - 500, default: corresponding YOffset_*KEYTYPE*) This value is can be used to adjust the Y coordinate of the 999 key.

Key999MouseRectLeft (integer 0 - key bitmap width - 1 , default: 0) relative X of left border of the mouse rectangle

Key999MouseRectTop (integer 0 - key bitmap height - 1, default: 0) relative Y of top border of the mouse rectangle

Key999MouseRectWidth (integer 0 - key bitmap width, default: key bitmap width) width of the mouse rectangle

Key999MouseRectHeight (integer 0 - key bitmap height, default: key bitmap height) height of the mouse rectangle

DispKeyColourInverted (boolean, required) True means, the *black* keys are drawn in a light color while the *white* keys are drawn in a dark color.

DispKeyColourWooden (boolean, default: false) True means, that a wood background is used for the keys.

DisplayFirstNote (integer 0 - 127, default: FirstAccessibleKeyMIDINoteNumber) Display the first key as the following note.

DisplayKeys (integer 1 - NumberOfAccessibleKeys, default: NumberOfAccessibleKeys) number of keys to display.

DisplayKey999 (integer 0 - 127, default: FirstAccessibleKeyMIDINoteNumber + *999*) The number in the key (*999*) is between 1 and *DisplayKeys*. It contains the midi number of the backend key, that is connected to this GUI key.

10.13 Label objects

Label object allows to display a text label. The background is an image. It is tiled, if the image is smaller than the label area. A label has the following attributes:

FreeXPlacement (boolean, default: true) True means that the X position is determined by DispXpos, else by DispDrawstopCol and DispSpanDrawstopColToRight.

FreeYPlacement (boolean, default: true) True means that the Y position is determined by DispYpos, else by DispAtTopOf-DrawstopCol.

DispXpos (integer 0-panel width, default: 0) absolute X position

DispYpos (integer 0-panel height, default: 0) absolute Y position

DispAtTopOfDrawstopCol (boolean, required if FreeYPlacement is true) If true, the label is displayed above the drawstop, else below.

DispDrawstopCol (integer 1- number of drawstop columns, required if FreeXPlacement is false) Position label at the specified drawstop column.

DispSpanDrawstopColToRight (boolean, required if FreeXPlacement is false) If true, move label half of the drawstop to the right.

DispLabelColour (color, default: black) Color for the label text.

DispLabelFontSize (font size, default: normal) Size of the label font

DisplLabelFontName (string, default: empty) Font for the text. Empty means use the group label font of the panel.

Name (string, default: empty) The text to display on this object

DispImageNum (integer 1-12, default: 1) Builtin bitmap set to use.

Image (string, default: use internal bitmap according to DispImageNum) Specify the file name of an image to use as bitmap. If the bitmap contains a mask for transparency, it will be used.

Mask (string, default: empty) File name for a external mask for the bitmap. If empty, no mask is added.

PositionX (integer 0 - panel width, default: according to the definitions above) Allow to override X position for button

PositionY (integer 0 - panel height, default: according to defintions above) Allow to override Y position for button

Width (integer 0 - panel width, default: bitmap width) Width of the button. If larger than the bitmap, the bitmap is tiled.

Height (integer 0 - panel height, default: bitmap height) Height of the button. If larger than the bitmap, the bitmap is tiled.

TileOffsetX (integer 0 - bitmap width, default: 0) X position on the bitmap of the left pixel of the button

TileOffsetY (integer 0 - bitmap width, default: 0) Y position on the bitmap of the top pixel of the button

TextRectLeft (integer 0 - Height, default: 0) relative X of left border of the text rectangle

TextRectTop (integer 0 - Height, default: 0) relative Y of top border of the text rectangle

TextRectWidth (integer 0 - Width, default: Width) width of the text rectangle

TextRectHeight (integer 0 - Height, default: Height) height of the text rectangle

TextBreakWidth (integer 0 - text rectangle width, default: Width) If 0, no text is displayed. Otherwise the value specifies the maximum line length used for text breaking.

10.14 Image objects

Image objects allow to display an image on a panel. They tile the images, if it is bigger then image size. It has the following attributes:

Image (string, required) Specifies the file name of an image to use as a bitmap. If the bitmap contains a mask for transparency, it will be used.

Mask (string, default: empty) File name for a external mask for the bitmap. If empty, no mask is added.

PositionX (integer 0 - panel width, default: 0) X coordinate of the left side. to override X position for button

PositionY (integer 0 - panel height, default: 0) Y coordinate of the left side.

Width (integer 0 - panel width, default: bitmap width) Width of the button. If larger than the bitmap, the bitmap is tiled.

Height (integer 0 - panel height, default: bitmap height) Height of the button. If larger than the bitmap, the bitmap is tiled.

TileOffsetX (integer 0 - bitmap width, default: 0) X position on the bitmap of the left pixel of the button

TileOffsetY (integer 0 - bitmap width, default: 0) Y position on the bitmap of the top pixel of the button

10.15 Button objects

Button objects are defined on the main panel. Their section name is based on their function. They share the following common properties:

Name (string, required) Name of the object. The name may be presented to the user in lists too, therefore it should be descriptive. If a GUI representation requires a shorter name, please override this value locally.

Displayed (boolean, default: false) If true, the section also includes the GUI properties for the main panel. Otherwise it is not displayed on the main panel.

DisplayInInvertedState (boolean, default: false) If true, off is displayed as on and on as off.

StopControlMIDIKeyNumber (integer 0-127, default: no MIDI event specified) Only used for building the initial configuration during the first load - provided just for HW1 compatibility. DEPRECATED.

MIDIProgramChangeNumber (int 1-128, default: no MIDI event specified) Only used for building the initial configuration during the first load - provided just for HW1 compatibility. DEPRECATED.

ShortcutKey (integer 0-255, default: 0) 0 means no shortcut, else it specifies the key code of the shortcut key.

The GUI properties of button objects are:

DisplayAsPiston (boolean, default: true for divisionals, generals and pistons, else false) True means to display as button, false as drawstop

DispLabelColour (color, default: Dark Red) Color for the label text.

DispLabelFontSize (font size, default: normal) Size of the label font

DisplLabelFontName (string, default: empty) Font for the text. Empty means use the control label font of the panel.

DispLabelText (string, default: Name of the button) Content of the text label. You should edit it, if you need to display a shorter string on the label.

DispKeyLabelOnLeft (boolean, default: true) If displayed as a piston and this attribute is false, move it a little bit left. Otherwise ignored.

DispImageNum (integer 1- type dependent, default: see below) Builtin bitmap set to use. GrandOrgue has 6 for drawstops and 5 for pistons. The default is 3 (piston) or 4 (drawstops) for read-only buttons, otherwise the default is 1.

DispButtonRow (button row, default: 1) If displayed as piston, it contains the button row according to the layout model. Otherwise ignored.

DispButtonCol (button column, default: 1) If displayed as piston, it contains the button column according to the layout model. Otherwise ignored.

DispDrawstopRow (drawstop row, default: 1) If displayed as drawstop, it contains the drawstop row according to the layout model. Otherwise ignored.

DispDrawstopCol (drawstop column, default: 1) If displayed as drawstop, it contains the drawstop column according to the layout model. Otherwise ignored.

ImageOn (string, default: use internal bitmap according to DispImageNum) Specifiy the file name of an image to use as on bitmap. If the bitmap contains a mask for transparency, it will be used.

ImageOff (string, default: use internal bitmap according to DispImageNum) Specifiy the file name of an image to use as off bitmap. If the bitmap contains a mask for transparency, it will be used. The size must match the on bitmap.

MaskOn (string, default: empty) File name for a external mask for the on bitmap. If empty, no mask is added.

MaskOff (string, default: value of MaskOn) File name for a external mask for the off bitmap. If empty, no mask is added.

PositionX (integer 0 - panel width, default: according to layout model) Allow to override X position for button

PositionY (integer 0 - panel height, default: according to layout model) Allow to override Y position for button

Width (integer 0 - panel width, default: bitmap width) Width of the button. If larger than the bitmap, the bitmap is tiled.

Height (integer 0 - panel height, default: bitmap height) Height of the button. If larger than the bitmap, the bitmap is tiled.

TileOffsetX (integer 0 - bitmap width, default: 0) X position on the bitmap of the left pixel of the button

TileOffsetY (integer 0 - bitmap width, default: 0) Y position on the bitmap of the top pixel of the button

MouseRectLeft (integer 0 - Width, default: 0) relative X of left border of the mouse rectangle

MouseRectTop (integer 0 - Height, default: 0) relative Y of top border of the mouse rectangle

MouseRectWidth (integer 0 - Width, default: Width) width of the mouse rectangle

MouseRectHeight (integer 0 - Height, default: Height) height of the mouse rectangle

MouseRadius (integer 0 - max(MouseRectHeight, MouseRectWidth), default: max(MouseRectHeight, MouseRectWidth) / 2) If 0, the mouse events are captured inside the mouse rectangle. Otherwise they must be inside a circle of the specified size too.

TextRectLeft (integer 0 - Height, default: 0) relative X of left border of the text rectangle

TextRectTop (integer 0 - Height, default: 0) relative Y of top border of the text rectangle

TextRectWidth (integer 0 - Width, default: Width) width of the text rectangle

TextRectHeight (integer 0 - Height, default: Height) height of the text rectangle

TextBreakWidth (integer 0 - text rectangle width, default: slightly smaller than Width) If 0, no text is displayed. Otherwise the value specifies the maximum line length used for text breaking.

Inside the button, the on/off bitmap (depending on the button state) is tiled. If a text width is set, a text label is displayed on it. Mouse events are only captured inside the mouse rectangle.

10.16 Drawstop objects

Drawstop objects are buttons with toogle functions. They contain these additional non-GUI properties:

Function (enumeration, default: Input) Logical function of the drawstop. If the value is *Input*, it is a normal user controllable drawstop and has no input switches. *Not* has one only input and negates the state of the input switch. *And*, *Xor*, *Nand*, *Nor* as well as *Or* has a variable number of inputs.

SwitchCount (integer 1 - switch count, required) Contains the number of input ports, if the logical function allows a variable number number of input. *Switch999* contains the referenced switches.

Switch999 (integer 1 - switch count, required) Lists the input switches of the logical function of the drawstop. If the drawstop is a switch, it can only reference switches with a lower number. The number of this settings depends on the function.

DefaultToEngaged (boolean, required) State of the button after loading.

GCState (integer -1 - 1, default: implementation defined) State of the button after pressing GC. -1 means no change, 0 off and 1 on.

StoreInDivisional (boolean, default: dependent on various settings) Determines, if the button should be stored in divisionals without *FULL*.

StoreInGeneral (boolean, default: dependent on various settings) Determines, if the button should be stored in generals without *FULL*.

10.17 Coupler objects

Couplers are drawstop objects. They forward key presses from one manual to other manuals/keys. They have the following additional attributes:

UnisonOff (boolean, required) If true, this coupler decouples the manual from the stops (turn it into a floating manual).

DestinationManual (integer manual number, required if not a unison off coupler) manual to forward key presses to.

DestinationKeyshift (integer -24 - 24, required if not a unsion off coupler) specifies the keyboard shift between source and destination manual in terms of absolute MIDI note numbers

CoupleToSubsequentUnisonIntermanualCouplers (boolean, required if not a unison off/melody/bass coupler) Triggers further inter-manual coupler with a destination key shift of zero.

CoupleToSubsequentUpwardIntermanualCouplers (boolean, required if not a unison off/melody/bass coupler) Triggers further inter-manual coupler with a destination key shift greater than zero.

CoupleToSubsequentDownwardIntermanualCouplers (boolean, required if not a unison off/melody/bass coupler) Triggers further inter-manual coupler with a destination key shift less than zero.

CoupleToSubsequentUpwardIntramanualCouplers (boolean, required if not a unison off/melody/bass coupler) Triggers further intra-manual coupler with a destination key shift greater than zero.

CoupleToSubsequentDownwardIntramanualCouplers (boolean, required if not a unison off/melody/bass coupler) Triggers further intra-manual coupler with a destination key shift less than zero.

CouplerType (enumeration, default: Normal) Type of the coupler: Normal, Bass or Melody.

FirstMIDINoteNumber (integer 0-127, default: 0) first MIDI note number to forward

NumberOfKeys (integer 0-127, default: 0) number of keys to forward starting with FirstMIDINoteNumber.

10.18 Switch objects

A switch object is a drawstop objects without any additional attributes. It can be used, for example, to trigger stop action noises and key action noises. See the Demo organ and the Kalvtrask organ for examples of its use.

10.19 Rank objects

A rank represents a row of pipes. It can be either be part of a stop section or appear in its own section.

Name (string, required) Name of the rank. The name may be presented to the user in lists too, therefore it should be descriptive.

FirstMidiNoteNumber (integer 0-256, if the rank is part of a stop section, the default value is derived from the associated manuals. Otherwise required) Midi note number of the first pipe

NumberOfLogicalPipes (integer 1-192, required) Number of pipes in this rank

AmplitudeLevel (float 0-1000, default: 100) Linear amplitude scale factor applied to the whole rank. 100 means no change.

Gain (float -120 - 40, default: 0) Amplitude scale factor in dB applied to the whole rank. 0 means no change.

PitchTuning (float -1200-1200, default: 0) Retune the rank by the specified number of cents.

TrackerDelay (integer 0 - 10000, default: 0) Delay introduced by the tracker for that rank.

HarmonicNumber (float 1-1024, default: 8) Harmonic number (= 64 / rank size), eg. 2 2/3 => 64 / (2 2/3) = 24. The harmonic number is used determining alternative tunings.

PitchCorrection (float -1200-1200, default: 0) Correction factor in cent for the pitch specified in the sample. This setting is used for retuning to other temperaments.

WindchestGroup (integer 1 - number of windchests, required) specify the windchest on which the pipes of the are placed.

Percussive (boolean, required) If false, the samples are played as is (without any loop/release handling)

MinVelocityVolume (float 0-1000, default: 100) Linear amplitude scale factor at low velocity applied to the whole rank. 100 means no change.

MaxVelocityVolume (float 0-1000, default: 100) Linear amplitude scale factor at high velocity applied to the whole rank. 100 means no change.

AcceptsRetuning (boolean, default: true) Determines if the rank will be retuned according to the current temperament. Retuning should be only disabled for sound effects.

A rank section contains the attributes of each of its pipes too. The attributes of each pipe are prefixed with *Pipe999* (number starting with 1). The supported attributes are:

Pipe999Percussive (boolean, default: rank percussive setting) If false, the samples are played as is (without any loop/release handling)

Pipe999AmplitudeLevel (float 0-1000, default: 100) Linear amplitude scale factor applied to the pipe (in addition to the organ/rank factor). 100 means no change.

Pipe999Gain (float -120 - 40, default: 0) Amplitude scale factor in dB applied to the pipe (in addition to the organ/rank factor). 0 means no change.

Pipe999PitchTuning (float -1200-1200, default: 0) Retune this pipe the specified number of cents (in addition to the organ/rank factor).

Pipe999TrackerDelay (integer 0 - 10000, default: 0) Delay introduced by the tracker for that pipe.

Pipe999 (string, required) Relative path to the sample WAV file of the first attack. It may be listed as REF:aa:bb:cc too. In that case, it means that this pipe is borrowed from manual aa, first rank of stop bb, the pipe cc. It may contain DUMMY, which defines a non-sounding placeholder.

Pipe999LoadRelease (boolean, default: reverse of percussive setting) If true, the release part is loaded from the first attack sample.

Pipe999AttackVelocity (int 0 - 127, default: 0) minimum velocity to use this attack sample.

Pipe999IsTremulant (int -1 - 1, default: -1) 1 means, that it is played, if the associated wave-based tremulant is on. 0 means, that it is played, if the associated wave-based tremulant is off. -1 means, that it is not affected by a wave-based tremulant.

Pipe999MaxKeyPressTime (int -1 - 100000, default: -1) Up to this time value in ms, the release sample is chosen. -1 means infinite.

Pipe999AttackStart (int 0 - 158760000, default: 0) Allows to override the start of the sample. This option is specified in samples.

Pipe999CuePoint (int -1 - 158760000, default: -1) Allows to override the cue point for the release. -1 means use from the wave file. This option is specified in samples.

Pipe999ReleaseEnd (int -1 - 158760000, default: -1) Allows to override the end of the release. -1 means play till the end of the wav. This option is specified in samples.

Pipe999LoopCount (int 0 - 100, default: 0) Allows to override the loops in the WAV file. 0 means use loops from the wave file.

Pipe999Loop999Start (int 0 - 158760000, default: 0) Start sample of the loop. The value must be within the WAV file. This option is specified in samples.

Pipe999Loop999End (int Pipe999Loop999Start + 1 - 158760000, required if Pipe999LoopCount is not zero) End sample of the loop. The value must be within the WAV file. This option is specified in samples.

Pipe999HarmonicNumber (float 1-1024, default: rank harmonic number) Harmonic number (= 64 / rank size), eg. 2 2/3 => 64 / (2 2/3) = 24.

Pipe999MIDIKeyNumber (integer -1 - 127, default: -1) If -1, use pitch information from the Pipe999 sample, else override the information in the sample with this midi note number (Pipe999PitchCorrection is used for specifing the fraction). Specifing the midi note number also reset the pitch fraction in the sample to 0.

Pipe999PitchCorrection (float -1200-1200, default: rank pitch correction) Correction factor in cent for the pitch specified in the sample. This setting is used for retuning to other temperaments.

Pipe999AcceptsRetuning (boolean, default: rank setting) Determines if the pipe will be retuned according to the current temperament. Retuning should be only disabled for sound effects.

Pipe999WindchestGroup (interger 1 - number of windchests, default: rank windchest) specify the windchest, on which this pipe is placed.

Pipe999MinVelocityVolume (float 0-1000, default: corresponding rank setting) Linear amplitude scale factor at low velocity applied to the pipe. 100 means no change.

Pipe999MaxVelocityVolume (float 0-1000, default: corresponding rank setting) Linear amplitude scale factor at high velocity applied to the pipe. 100 means no change.

Pipe999AttackCount (int 0 - 100, default: 0) Number of additional attack samples

Pipe999Attack999 (string, required) Relative path to the sample WAV file.

Pipe999Attack999LoadRelease (boolean, default: reverse of percussive setting) If true, the release part is loaded.

Pipe999Attack999AttackVelocity (int 0 - 127, default: 0) minimum velocity to use this attack sample.

Pipe999Attack999IsTremulant (int -1 - 1, default: -1) 1 means, that it is played, if the associated wave-based tremulant is on. 0 means, that it is played, if the associated wave-based tremulant is off. -1 means, that it not affected by a wave-based tremulant.

Pipe999Attack999MaxKeyPressTime (int -1 - 100000, default: -1) Up to this time value in ms, the release sample is choosen. -1 means infinite.

Pipe999Attack999AttackStart (int 0 - 158760000, default: 0) Allows to override the start of the sample. This option is specified in samples.

Pipe999Attack999CuePoint (int -1 - 158760000, default: -1) Allows to override the cue point for the release. -1 means use from the wave file. This option is specified in samples.

Pipe999Attack999ReleaseEnd (int -1 - 158760000, default: -1) Allows to override the end of the release. -1 means play till the end of the wav. This option is specified in samples.

Pipe999Attack999LoopCount (int 0 - 100, default: 0) Allows to override the loops in the WAV file. 0 means use loops from the wave file.

Pipe999Attack999Loop999Start (int 0 - 158760000, default: 0) Start sample of the loop. The value must be within the WAV file. This option is specified in samples.

Pipe999Attack999Loop999End (int Pipe999Loop999Start + 1 - 158760000, required if Pipe999LoopCount is not zero) End sample of the loop. The value must be within the WAV file. This option is specified in samples.

Pipe999ReleaseCount (int 0 - 100, default: 0) Number of additional release samples

Pipe999Release999 (string, required) Relative path to the sample WAV file.

Pipe999Release999IsTremulant (int -1 - 1, default: -1) 1 means, that it is played, if the associated wave-based tremulant is on. 0 means, that it is played, if the associated wave-based tremulant is off. -1 means, that it not affected by a wave-based tremulant.

Pipe999Release999MaxKeyPressTime (int -1 - 100000, default: -1) Up to this time value in ms, the release sample is choosen. -1 means infinite.

Pipe999Release999CuePoint (int -1 - 158760000, default: -1) Allows to override the cue point for the release. -1 means use from the wave file.

Pipe999Release999ReleaseEnd (int -1 - 158760000, default: -1) Allows to override the end of the release. -1 means play till the end of the wav. This option is specified in samples.

Pipe999LoopCrossfadeLength (int 0 - 120, default: 0) Crossfade length between loop start and loop end. A cross fade requires enough samples before the start of the loop.

10.20 Stop objects

Stop objects are drawstop objects. A stop consists of a number of ranks. If the number of ranks is set to zero, the stop contains one rank, which is defined in the stop section - else it references a list of ranks. Only the accessible pipes can be triggered from the manual. A stop has the following additions attributes:

NumberOfRanks (integer 0 - rank count specified in the organ section, default: 0) Number of referenced ranks. If zero, one rank definition is included in the stop section. The lists of references is specified via the *Rank999...* settings.

FirstAccessiblePipeLogicalKeyNumber (integer 1-128, required) The key number on the manual of the first accessible pipe.

NumberOfAccessiblePipes (integer 1 - 192, required) Number of pipes, that are playable from the manual starting from the first acessible pipe.

FirstAccessiblePipeLogicalPipeNumber (integer 1 - 192, required if NumberOfRanks=0) The number of the first pipe accessible from the manual. If NumberOfRanks is not 0, this setting is not necessary.

Rank999 (integer 0 - rank count specified in the organ section, required) Reference to a rank from the organ section.

Rank999FirstPipeNumber (integer 1 - number of pipes in the rank, default: 1) Number of first mapped pipe from the rank

Rank999PipeCount (integer 0 - remaining number of pipes in the rank, default: remaining number of pipes in the rank) Number of pipes mapped from the rank.

Rank999FirstAccessibleKeyNumber (integer 1 - NumberOfAccessiblePipes, default: 1) Key number offset (starting with FirstAccessiblePipeLogicalKeyNumber) for the pipe referenced by Rank999FirstPipeNumber.

10.21 Pushbutton objects

Pushbuttons are buttons without any state. It is displayed as a piston.

10.22 Piston objects

A piston is a pushbutton which triggers other elements. It contains the following additional attributes:

ObjecType (string, required) Type of the element to trigger. Value can be STOP, COUPLER, SWITCH or TREMULANT.

ManualNumber (integer first manual index - last manual index, required for stops and coupler) The manual, to which the referenced object belongs.

ObjectNumber (integer, required) Determines the number of the object. Depending on the object it must be a valid stop/coupler/switch number on the referenced manual or a valid global tremulant number.

10.23 Tremulant objects

Tremulants are drawstop objects with the following additional attributes:

TremulantType (enumeration, default: Synth) Type of the tremulant. Valid values are: Synth (synthesised tremulant) and Wave (tremulant based on different wave samples).

Synthesised tremulants have the following attributes:

Period (integer 32-44100, required) Period of the tremulant in ms

StartRate (integer 1-100, required) Determines the startup time of the tremulant.

StopRate (integer 1-100, required) Determines the stop time of the tremulant.

AmpModDepth (integer 1-100, required) Determines, how much the volume will be changed.

10.24 Windchest objects

Windchest objects represent a windchest on which pipes of ranks are placed. It has the following attributes:

Name (string, default: implementation dependent) Display name of this windchest

NumberOfEnclosures (integer 0 - enclosure count, required) Number of enclosures, which influence this windchest. The list is specified by the *Enclosure999* entries in the windchest section.

NumberOfTremulants (integer 0 - tremulant count, required) Number of tremulants, which influence this windchest. The list is specifed by the *Tremulant999* entries in the windchest section.

Enclosure999 (integer 1 - enclosure count, required) Number of an enclosure, which influences this windchest.

Tremulant999 (integer 1 - tremulant count, required) Number of a tremulant, which influences this windchest.

10.25 Additional panels (old format)

It is possible to define additional panels. The section is called *Panel999* with a number starting from 001. All panels use the same layout engine as the main panel, therefore the section includes all display metrics attributes. Additionally it includes the following attributes:

Name (string, required) Name of the panel

Group (string, default: empty) If not empty, place it in the submenu with the specified name, else directly in the panel menu.

NumberOfManuals (integer 1 - number of defined manuals) number of manuals to display on this panel

HasPedals (boolean, required) Includes a manual displayed as pedal

NumberOfSetterElements (integer 0-999, default: 0) Number of setter elements on the panel. The section of the GUI definitions are called *Panel999SetterElement999*.

NumberOfEnclosures (integer 0 - number of defined enclosures, required) Number of enclosures on the panel. The section of the enclosures GUI definitions are called *Panel999Enclosure999*.

Enclosure999 (valid enclosure number, required) Reference to the enclosure on the main panel.

NumberOfTremulants (integer 0 - number of defined tremulants, required) Number of tremulant on the panel. The section of the tremulants GUI definitions are called *Panel999Tremulant999*.

Tremulant999 (valid tremulant number, required) Reference to the tremulant on the main panel.

NumberOfReversiblePistons (integer 0 - number of defined reversible pistons, required) Number of reversible pistons on the panel. The section of the reversible pistons GUI definitions are called *Panel999ReversiblePiston999*.

ReversiblePiston999 (valid reversible piston number, required) Reference to the reversible piston on the main panel.

NumberOfSwitches (integer 0 - number of defined switches, default: 0) Number of switches on the panel. The section of the switches GUI definitions are called *Panel999Switchn999*.

Switch999 (valid switch number, required) Reference to the switch on the main panel.

NumberOfGenerals (integer 0 - number of defined generals, required) Number of generals on the panel. The section of the generals GUI definitions are called *Panel999General999*.

General999 (valid general number, required) Reference to the general on the main panel.

NumberOfDivisionalCouplers (integer 0 - number of defined divisional couplers, required) Number of divisional couplers on the panel. The section of the divisional coupler GUI definitions are called *Panel999DivisionalCoupler999*.

DivisionalCoupler999 (valid divisional coupler number, required) Reference to the divisional coupler on the main panel.

Manual999 (valid manual number, required) Number of the manual to use a specified manual on the panel.

NumberOfStops (integer 0-999, required) Number of stops on the panel. The section of the stops GUI definitions are called *Panel999Stop999*.

Stop999Manual (valid manual number, required) Reference to the manual of the stop on the main panel.

Stop999 (valid stop number, required) Reference to the stop on the main panel.

NumberOfCouplers (integer 0-999, required) Number of couplers on the panel. The section of the couplers GUI definitions are called *Panel999Coupler999*.

Coupler999Manual (valid manual number, required) Reference to the manual of the coupler on the main panel.

Divisional999 (valid coupler number, required) Reference to the divisional on the main panel.

NumberOfDivisionals (integer 0-999, required) Number of divisionals on the panel. The section of the divisionals GUI definitions are called *Panel999Divisional999*.

Divisional999Manual (valid manual number, required) Reference to the manual of the divisional on the main panel.

Divisional999 (valid divisional number, required) Reference to the divisional on the main panel.

NumberOfLabels (integer 0-999, required) Number of labels on the panel. The section of the label GUI definitions are called *Panel999Label999*.

NumberOfImages (integer 0-999, required) Number of images on the panel. The section of the image GUI definitions are called *Panel999Image999*.

10.26 Panels (new format)

The section is called *Panel999*. *Panel000* is the main panel. Additional panels start with number 001. The section includes the display metrics. Additionally it includes the following attributes:

Name (string, required for non-main panels) Name of the panel

Group (string, default: empty, only for non-main panels) If not empty, place it in the submenu with the specified name, else directly in the panel menu.

HasPedals (boolean, required) Includes a manual displayed as pedal

NumberOfImages (integer 0-999, required) Number of images on the panel. The section of the label GUI definitions are called *Panel999Image999*.

NumberOfGUIElements (integer 0-999, required for the main panel, default 0 for any other panel) Number of elements on the panel. The section of the GUI elements are called *Panel999Element999*.

10.27 GUI elements

The attribute *Type* can be one of the following values:

Divisional The section contains the GUI attributes of a divisional. Additionally it includes the following attributes:

- **Manual** (valid manual number, required) Number of the manual.
- **Divisional** (valid divisional number, required) Number of the divisional on the manual.

Coupler The section contains the GUI attributes of a coupler. Additionally it includes the following attributes:

- **Manual** (valid manual number, required) Number of the manual.
- **Coupler** (valid coupler number, required) Number of the coupler on the manual.

Stop The section contains the GUI attributes of a stop. Additionally it includes the following attributes:

- **Manual** (valid manual number, required) Number of the manual.
- **Stop** (valid stop number, required) Number of the stop on the manual.

Enclosure The section contains the GUI attributes of an enclosure. Additionally it includes the following attributes:

- **Enclosure** (valid enclosure number, required) Number of the enclosure.

Tremulant The section contains the GUI attributes of a tremulant. Additionally it includes the following attributes:

- **Tremulant** (valid tremulant number, required) Number of the tremulant.

DivisionalCoupler The section contains the GUI attributes of a divisional coupler. Additionally it includes the following attributes:

- **DivisionalCoupler** (valid divisional coupler number, required) Number of the divisional coupler.

General The section contains the GUI attributes of a general. Additionally it includes the following attributes:

> **General** (valid general number, required) Number of the general.

ReversiblePiston The section contains the GUI attributes of a reversible piston. Additionally it includes the following attributes:

> **ReversiblePiston** (valid reversible piston number, required) Number of the reversible piston.

Switch The section contains the GUI attributes of a switch. Additionally it includes the following attributes:

> **Switch** (valid switch number, required) Number of the switch.

Label The section contains the GUI attributes of a label.

Manual The section contains the GUI attributes of a manual. Additionally it includes the following attributes:

> **Manual** (valid manual number, required) Number of the manual.

Any setter element type See the setter element section for further description.

Chapter 11

The GrandOrgue organ package format

11.1 General information

An organ package is a ZIP file using only the *STORED* compression [= compression level 0] with the extension *orgue*. It may use the ZIP64 extensions - no other extensions are allowed. It is possible to store file names containing directories.

The ID of an organ package is its SHA1 sum with all letter converted to upper case.

Each organ package must contain an index file named *organindex.ini*. This file lists all contained organs as well as any other organ package they might depend on.

Referenced files are only loaded from the organ package - not from the file system. If a file is missing in the main organ package, it will search *Dependency001*, then *Dependency002* and so on.

Use wavpack to compress samples. Use PNG for lossless compression of images.

11.2 Index file specification

It has the same syntax rules as an organ definition file. The *General* section has the following elements:

Title (string, required) Name of the organ package

OrganCount (integer 0-100, required) Number of organs installed by this package. The organs are defined in a section named *Organ999*.

DepenencyCount (integer 0-100, default: 0) Number of organs packages required by this organ package. The dependencies are defined in a section named *Dependency999*.

11.3 Dependency section

The section has the following elements:

PackageID (string, required) ID of the package file, that should be searched for missing files.

Title (string, required) Name of the package file.

11.4 Organ section

This section has the following elements:

Filename (string, required) Filename of organ definition file in the organ package.

ChurchName (string, required) Name of the organ/church. This setting must match the organ definition file.

OrganBuilder (string, required) informational text displayed in the property dialog. This setting must match the organ definition file.

RecordingDetails (string, required) informational text displayed in the property dialog. This setting must match the organ definition file.

Chapter 12

Technical Support

GrandOrgue is an open source project distributed under the GNU General Public License. For information, comments, questions, or problems, please visit the project pages on sourceforge.

Chapter 13

History

GrandOrgue is a project based on a previous GNU GPL project with a very similar name that that has been discontinued.

Appendix A

Short octave

GrandOrgue can use three different methods to make a short octave keyboard:

- *Tie samples to keys*

  This is the build scheme:

  - The keyboard defines e.g. 45 keys (full compass: 49)
  - The first accessible key on the physical keyboard is low E
  - The first sample (Pipe001) refers to low C (read: press low E, hear low C)
  - The second sample (Pipe002) refers to low F (read: press low F, hear low F)
  - The third sample (Pipe003) refers to low D (read: press low F#, hear low D)
  - The fourth sample (Pipe004) refers to low G (read: press low G, hear low G)
  - The fifth sample (Pipe005) refers to low E (read: press low G#, hear low E)
  - From the sixth sample and upwards, the keyboard layout is "normal"

  The pipe layout to the keys being hardwired, many useful features which expect 12-note octaves are broken in the low octave:

  - Octave and suboctave couplers (Even if this feature is unlikely when reproducing old organs)
  - Bass and melody couplers
  - Tuning to other temperaments

  This method should be considered as legacy and used only for backward compatibility with Hauptwerk™ version 1 samplesets. The preferred method for designing a short octave keyboard is through a key mapping (see MIDIKey999) and below.

- *MIDI setting: 9x Note short octave at low key*

  Use this setting to simulate a short octave keyboard when the Organ Definition File only provides a full compass keyboard. GrandOrgue renders sound as if the keyboard were fitted with a short octave, but the visual feedback is that of a full compass keyboard (e.g. press low E and the keyboard image in the GUI displays a depressed low C).

- *MIDI mapping on full compass*

  This is the build scheme:

  - The keyboard defines e.g. 49 keys, full compass
  - The first accessible key on the physical keyboard is low C
  - *Pipe999* entries refer to all pipes starting a low C, as usual
  - Configure the GUI:

DisplayKeys=45	Display only 45 keys
DisplayFirstNote=40	The fist displayed key is low E

– Add *MIDIKey999* entries.

MIDIKey036=0	Disable low C
MIDIKey037=0	Disable low C#
MIDIKey038=0	Disable low D
MIDIKey039=0	Disable low D#
MIDIKey040=36	Map physical low E to sound low C
MIDIKey041=41	Map physical low F to sound low F
MIDIKey042=38	Map physical low F# to sound low D
MIDIKey043=43	Map physical low G to sound low G
MIDIKey044=40	Map physical low G# to sound low E
MIDIKey045=45	Map physical low A to sound low A. From this key up to the top end of the keyboard, the mapping is usual and can be omitted.

– Add DisplayKey999 entries to tie GUI keys to Midi note numbers:

DisplayKey001=36	1st Key is depressed in the GUI when Midi note 36 (low C) is received
DisplayKey002=41	2nd Key is depressed in the GUI when Midi note 41 (low F) is received
DisplayKey003=38	3rd Key is depressed in the GUI when Midi note 38 (low D) is received
DisplayKey004=43	4th Key is depressed in the GUI when Midi note 43 (low G) is received
DisplayKey005=40	5th Key is depressed in the GUI when Midi note 40 (low E) is received
DisplayKey006=45	6th Key is depressed in the GUI when Midi note 45 (low A) is received. From this key up to the top end of the keyboard, the mapping is usual and can be omitted.

Note
When sampling a real-world organ, missing samples in the low octave need to be generated to get a complete low octave. If this is not done, holes will appear in the low octave if the keyboard is converted to full compass using the 9x Note without map MIDI setting.

Chapter 14

Index