ZSP

ZPP

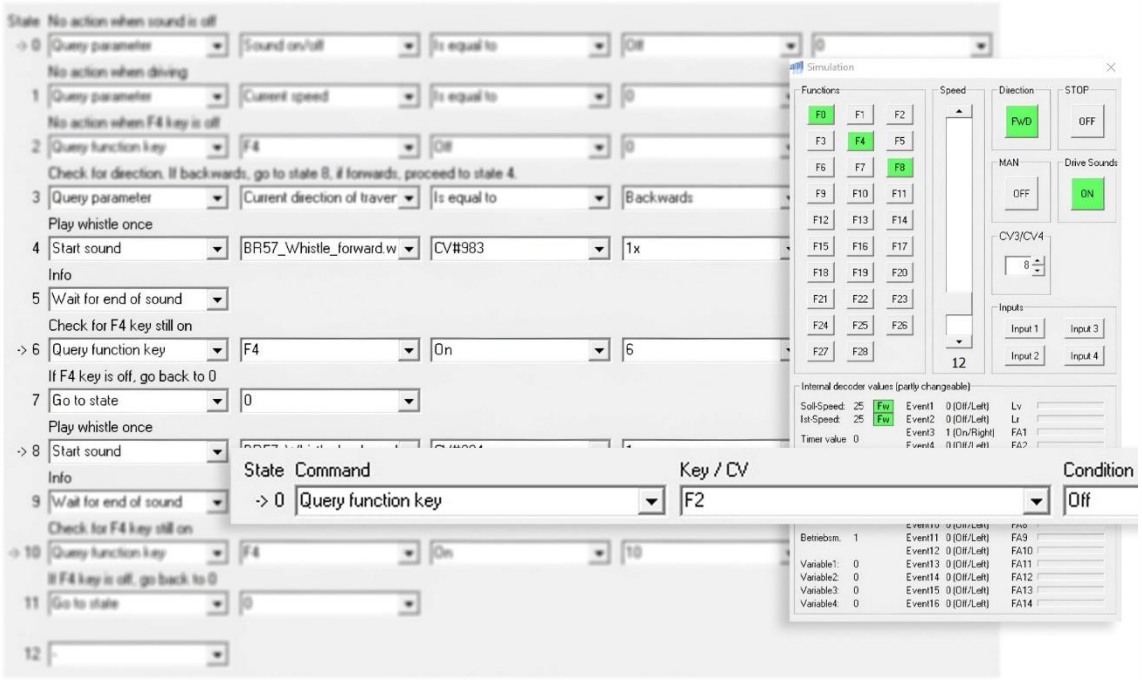
INSTRUCTION MANUAL

Scripts

for software version
ZSP V1.24.12

Introduction to programming scripts in ZSP

ISSUES:
2020 nov
2024 mar.
2025 jun

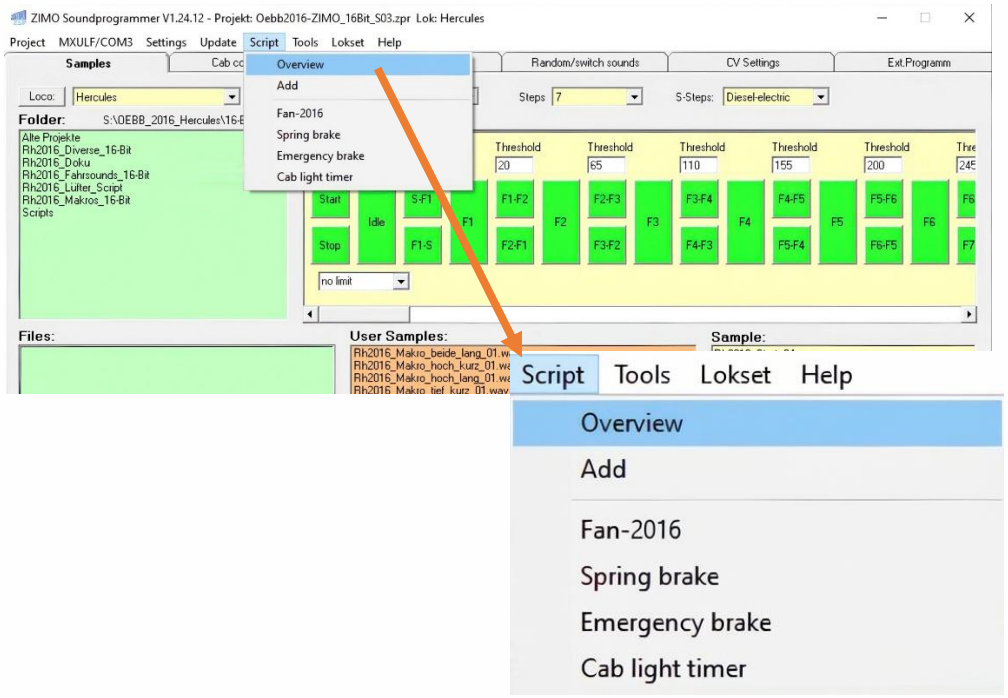


General Info:

To program sequences and dependencies, ZSP offers the option of working with 'scripts'. Scripts can be thought of as programmed sequences. However, you do not need any programming skills to create these, as ZSP has the option of creating the desired sequences in an input mask. For each sound project, it is possible to create (or import) up to 16 scripts with MS decoders from MS-FW 4.207 onwards and up to 8 scripts with MX decoders from MX-FW 39.0 onwards. However, MX decoders are subject to some restrictions compared to MS decoders, e.g. only 6 independently playable sound channels are available, and two samples can only be played simultaneously in a script in certain cases. The following applies to both decoder families: up to 256 states (commands or lines) can be programmed per script. The individual scripts can be deactivated via the corresponding bits of CVs #837 and #843 (CV #837 Bit0-7 = scripts 1-8, CV #843 Bit0-7 = scripts 9-16).

Many values and parameters can be exported as script CVs, which can be changed later in the decoder without the necessity to reload the entire sound project. These CVs are #980 to #1019. Common applications include sample volumes, timer lengths or speed steps. Of course, the permitted values of the individual CVs they represent must be followed – see the decoder instructions for MX and MS decoders.

Starting with MS-FW 4.250, scripts in CV sets can also be used in MN decoders. The scripts can be created in ZSP or ZPP Konfig and loaded into the decoder via the CV Set Creator. However, it should be noted that scripts with sounds are not possible, or sounds (including thyristor and motor samples, diesel stages, etc.) will be ignored. It makes more sense to use scripts to alternate function outputs or driving behaviour of MN decoders. However, if necessary, timers with the length of sound samples can be used as a workaround.



Create and edit scripts:

From version V1.21.22 onwards, scripts can be created and edited in both ZSP and ZPP Konfig. However, existing scripts in coded projects cannot be viewed, edited or deleted in ZPP Konfig, as these have been specifically adapted by the sound provider to the sequence of the project. It is only possible to deactivate them via the corresponding bits in CVs #837 and #843, but this can significantly limit the functionality of the sound project. However, you can also create or import your own scripts in coded projects

The scripts can be found in ZSP under the Script button. There, you can either select Overview or Add, or select one of the existing scripts directly. Clicking on Overview opens the separate main script window.

In ZPP Konfig, you can find the scripts under the Scripts tab. The following editing methods are the same in both programmes.

To create a new script, click on the **Add button** and a field labelled 'Script1' will appear. Right-click on 'Script 1' to give the script a distinctive name so that it is easier to recognise later.

Double-click on a script to open the script editor.

Export to file: A script can be exported to any location, for example to use it in another sound project or to share it with someone else. It is saved in .zsc file format with any file name.

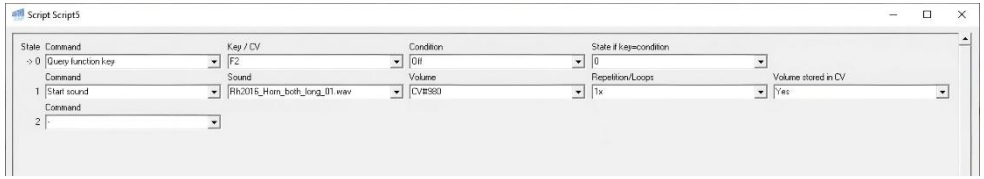
Import from file: Once a script has been saved and exported, it can be imported into another sound project. ATTENTION: Only the index number of sound samples within the current sound project is saved, and this may vary from one sound project to another. This means that sound samples must be checked manually and adjusted if necessary after importing a script into another sound project!

Window size: The script editor window can be resized and maximised. This is useful for long and complex scripts, as it allows you to see more states on your screen.

Delete: A script can be deleted by simply clicking on it and using the Delete key.

The overview of both programmes also shows all elements linked to the respective script. These include function keys, function outputs, script CVs and assigned sound samples.

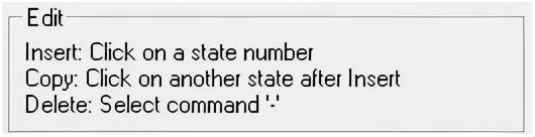
Script Editor



In the **ZSP script editor**, there is a checkbox at the bottom labelled 'Show comments. If this is activated, you can give each individual state a name. This is helpful for keeping track of more complex scripts so that you can still understand exactly what you wanted to achieve with each command later on. You can enter the name or text of your choice by clicking on 'Info'. You can then start typing straight away. ZSP saves the name and you can switch between the original names of the commands and your own descriptions at any time (by selecting or deselecting the check box). Next to it is the Single line box, which can be used to hide the information for each parameter. This makes the script more compact, but sometimes also more confusing. Here, the individual parameters can now be pushed together using the slider to display the explanatory text for the states to the left.

This function is currently not available in ZPP Konfig.

At the bottom left of the script editor, there is an info box titled 'Edit' for useful actions while editing scripts.



These are:

- Insert: Click on the state number (inserts an empty state above). "Go to's" to already existing states are changed accordingly.
- Copy: Click on another state after insert (pastes the selected state into the last inserted one, while it is still empty. This only works as long as the inserted state is still highlighted in red).
- Delete: Select the "-" command at the very top of the "Command" parameter. For security reasons, you will be asked again whether you really want to delete a state.

The 'Simulation' field is located in the lower center area of the script editor. This will be explained in more detail later, as it is used to test a completed script.

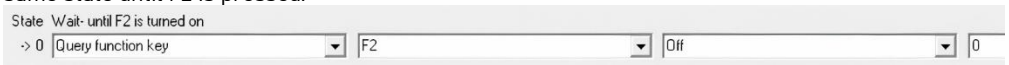
Read and write scripts:

Scripts are best 'read' as follows: start at the top left and read line after line, like a text, until you reach the bottom right. This is exactly what ZSP does, and later the decoder, when playing the scripts. Depending on the type, several parameters can be queried for each command. In principle, you can generate many different sequences with the scripts (within the predefined actions and parameters). However, there are a few basic rules for configuration that should always be followed.

Query the contrary:

Scripts are similar to the if/else principle in programming. For example: if pressing a button should to trigger something, the off state of the button should be checked.

The commands work by querying a state, and if this is true, the program jumps to the state set in the last parameter and continues the sequence at this point. However, if the query is negative, the sequence continues in the next state in line. In the following example, the sequence remains in the same state until F2 is pressed.



This can be read as follows: The command 'Query function key' checks whether the function key F2 is deactivated. If this is the case, the script jumps to state 0, i.e. remains in the current line.

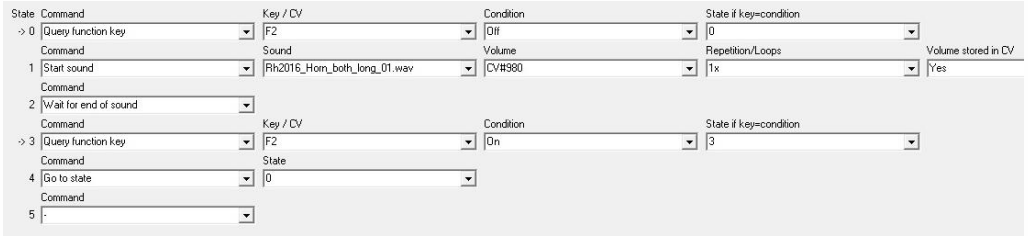
If you press F2 now – either on the digital command or via the MXULF controller from ZSP or ZPP Konfig – the query gets a negative result and the script automatically continues to the next line, i.e. to the next state.

In this state, an action should be defined and executed when the function key is activated – for example, the start of a sound.

After the action has been performed, the status of the key must be queried again. In the example shown, the status must now be checked for On.

If something is started in a script it must also be terminated there!

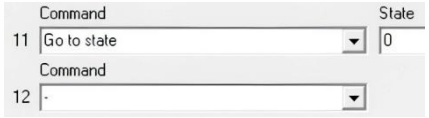
No matter if it is sound samples, function outputs or alternation of the diesel sequence: if an action is triggered in a script, you have to ensure that it can also be terminated again.



The end of a script must not remain open!

A final command (e.g. "Go to State X") is essential to ensure a continuous sequence. If this is missing, the script may stop and cannot be continued.

The final state of a script should therefore always contain a concluding command. An example could look like this:



It is not necessary to jump from the last state to state 0; the important thing is that the script forms a completed loop and does not remain open or stop at any point.

Description of commands:

Start sound: Starts a sound sample selected in the 'Sound' parameter, which lists all samples in the sound project. The next parameter determines the volume at which this sound is played. You can determine whether the sound should be looped or how often it should be played (from 1x to 8x). Loop and Short are only used if the selected sound sample is looped and the loop is to be ended prematurely. If a sound sample continues beyond the end of a loop marker, this end part will also be played. The loops of the sound samples must be edited in ZSP as usual. The volume of a sample can also be set via CVs #980-#1019. To do this, select 'Yes' in the last drop-down 'Volume stored in CV' and select the desired CV in the 'Volume' drop-down. The advantage of this is that the script does not have to be adjusted and the entire sound project reloaded into the decoder and the volume can be easily changed later via simple CV programming.
Stop Sound: If the sound was started with Loop or Loop and Short, this command must be set to end the sound. If the sound was started with a certain number of repetitions, this command is not necessary (since the sound stops automatically when the repetitions are finished), unless the sound needs to be stopped early by other conditions in the sequence of the script.

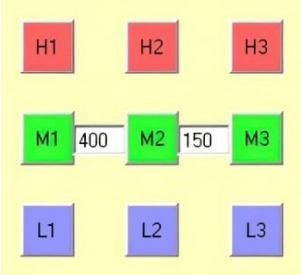
Wait for end of sound: Before a new sound sample is started, you can wait with this command until the current sound sample (including all loops and repetitions) has been played entirely. Furthermore this command will ensure an audio transition to the next sound sample – sound gaps will be prevented. However, this command has no relevance for the following sequence of the script itself.

Query function key: Checks whether a function key (set in parameter 1) is switched on or off (according to parameter 2) and determines where to go to if the check is positive. If, for example, a sound is to be triggered with the F5 key, check whether F5 is switched off– and go to (remain at) the same state. As soon as the key is active, the script then automatically continues to the next state. One of the script CVs #980-#1011 can also be queried again in parameter 1; values =0 to =28 for keys F0 to F28.

Query parameter: Compares whether the queried parameter (selection field *If parameter*; see following description of the parameters) is equal, not equal, greater, or less (selection field *Test*) than the defined value (selection field *comparative value*). The test can also be queried using a corresponding value in script CVs #980-#1019. Just like *Query function key*, the last parameter defines where the script should go to if this check is positive. If this check is negative, the sequence will continue with the following state.

The following parameters can be set in the ***If parameter*** drop down:

- ***Current speed***: Current internal speed step (0-255). If the 'Test' is performed through a CV, the selection for the next parameter changes from 'Comparative value' to 'CV'. This applies to all subsequent queries.
- ***Target speed***: Speed step set on the controller (0-255) to which acceleration or deceleration is to take place.
- ***Sound on/off:*** Checks whether driving sound is *off* or *on*. This query should preferably be used to check if the sound of the project on or off instead of checking for the function key, e.g. F8, for sounds.
- ***Target set***: Checks which diesel sound set will be played next (e.g. by pressing the sound switch key). The term 'diesel' set or level is used in the sense of drive levels set up in ZSP. These can be diesel sounds or, for example, the fan of an electric locomotive. The set index is in the script: Diesel set 1 = index 0, diesel set 2 = index 1, etc. If transition sets are used (for dual vehicles with electric and diesel drive, CV #346 Bit2), the first diesel set has the index =0 and the 4th complete diesel set has the index =1. The transition sets cannot be queried separately and the index change takes place at the end of the transition set.
- ***Current set***: Checks which diesel set is currently being played. In steam sound projects the set change happens instantly. Current sets and target sets are basically the same and so there's only one of these queries needed.
- ***Diesel step***: Checks the level of the diesel sound.
- ***Timer value***: Checks the countdown from the value set in 'Load timer' to 0. This parameter allows actions to be triggered during the countdown.
- ***Acceleration value***: Number of speed steps by which acceleration is increased. If the vehicle decelerates during the test, the script returns the value 0.
- ***Deceleration value:*** Number of speed steps by which the delay is applied.
- ***Current direction of travel***: Checks the current direction of travel (forward, backwards).
- ***Target direction of travel***: Checks which direction the locomotive should travel in according to the control centre (e.g. locomotive still braking in the 'old' direction)
- ***Consist active***: Checks whether the locomotive is currently in consist mode.
- ***Consist inverted***: Checks whether bit 7 in CV #19 is set = 1 (inverts the direction of travel in consist mode) and whether consist mode is active.
- ***HLU limit***: Current limit of the HLU brake section. The values in the script CVs are:
- =0: HLU emergency stop (braking without delay), =2: Halt, =3: UH, 4= U, =5: LU, =6: L, =7: FL, =8: F.
- ***ABC limit***: Current limit of the ABC brake section. The values in the script CVs are:
- =2: Stop, =5: Slow, =8: Driving.
- ***DC limit***: Current limit of the DC brake section. The values in the script CVs are: =2: Stop, =8: Driving.
- ***Current limit***: Lowest limit of the three previous parameters. To avoid having to query all three in the script, this parameter tests for the lowest value found.
- ***Chuff interval*** (only MS): queries the time in ms between the individual chuffs (for more details, see the ZSP manual)



- ***Chuff/revolution*** (only MS): queries the current chuff for comparison. The number is specified exclusively in ZSP (with No. of chuffs, 2-16). Please note that the index of the chuff applies, i.e. chuff 1 = value 0, chuff 2 = 1, etc.
- ***Chuff speed index*** (only MS): Compares the level that is currently being played in the sequence (H1, H2, etc.).
- ***Chuff load index*** (only MS): Compares the current load level H, M or L.
- ***Drain active*** (only MS): compares the drainage function.
- ***Current operating mode*** (only MS): queries the current mode, digital or analogue. 'None' = value 0, 'Digital mode' = 1, analogue mode = 2.

Control loco: Sets parameters for controlling the locomotive and defines the type of influence using 'Control':

- ***No influence***: is used to undo the following commands.
- ***Set drive step***: Regardless of the currently set drive step, a specific drive step is forced. This can also be drive step 0.
- ***Limit drive step***: This sets a maximum drive step that cannot be exceeded.
- ***Scale down drive step***: Reduces the internal drive step (255) in proportion to the set drive step, e.g.: value 128 reduces to approximately half, value 64 to approximately a quarter.
- ***Set direction***: forces a direction of travel (forward, backward, east, west, etc.).
- ***Emergency stop***: This initiates an emergency stop while driving. The braking delay for the emergency stop can be set in CV #111.
- **Inhibit acceleration**: When this command is active, you can only decelerate or drive at the same speed, but not accelerate.

The **internal drive** step 0-255 is defined in 'Drive step'. Parameter 3 defines the direction in which the influence applies (in driving direction = current driving direction, against driving direction = against the current driving direction, forward = forward driving direction of the locomotive, backward = backward driving direction of the locomotive, east, west). If 'Drive step via CV – Yes' is selected, one of the script CVs #980-#1019 must be used.

Thyristor Sample: Select a sound sample for the thyristor effect. The dynamics can be controlled either through the corresponding CVs #289-295, #357-358, or via the script command 'Thyristor Pitch'. The script does not distinguish which sound project is currently being edited, which is why this function is also visible in steam and diesel projects. Thyristor samples are usually looped. If manual transitions are needed, for example between individual thyristor stages, the loop can also be deactivated. Each thyristor sample started in the script replaces the previous one, so that it is not necessary to manually stop a playing sample.

Wait for end of thyristor sample: This command can be used to control the sequence so that the script waits for a thyristor sample to play completely before starting the next sample. Without this function, the current sample would be replaced immediately by the newly started one, which could lead to premature termination or, in the worst case, to an audible gap in the sound.

Thyristor pitch: This command defines the pitch of the thyristor, linked to certain drive levels. Parameter 1 defines the pitch of the thyristor at medium speed step (parameter 3) and parameter 2 defines the maximum pitch of the thyristor at maximum speed step (parameter 4). The settings here correspond to CVs #291-#293 and #838. Once used in a script, the effect of these CVs cannot be returned to the decoder sequence and all changes must be made in the script.

I²C command: ZIMO internal test function for MX645 for outputting signals to I²C extensions; currently not implemented.

Load timer: A timer is set here that works in the background. Depending on the selected time unit, the timer counts down from the set value to zero. The units are 1 ms, 10 ms, 100 ms, 1 s, 10 s and 100 s for MS decoders. For MX decoders, only 100 ms units (0.1 seconds) can be selected. If an action is to be triggered after a certain time, the 'Timer value' parameter is checked in 'Query Parameter'! The maximum value that can be set for a timer on MS decoders is 255 with a unit of 100s (i.e. 25500 seconds, approximately 7 hours). Several timers with different units can be set up subsequently. However, they do not work simultaneously and the last timer started is used. The time value can also be written as a script CV.

Set diesel step: Activates a fixed diesel stage (value: 0= idle... 20= F20). This allows the script to take control of the diesel sequence, i.e. to fix a certain stage in the sequence regardless of the actual sequence. The decoder plays all the way through the stages up to the selected one, and it is recommended to query this 'target' diesel stage in a further query if it is relevant for the course of the script. If 'Fix diesel stage' is deactivated in one of the following states, it is irrelevant which stage is specified there – the sequence finds its own way back to the current diesel stage.

Suppress brake squeal: Defines whether break squeal should be suppressed.

Switch output on: Switches a function output on and also defines the dimming (the PWM value in %) for this output in parameter 2. The output can also be switched off again in the same parameter. In addition, it is possible to set the output, the PWM value or both via script CVs. However, outputs switched on via normal mapping or in samples cannot be switched off via a script; these would have to be managed completely via the script. The only option is to assign such an output to a PWM group in the mapping (for more details, see the MX and MS decoder instructions in the 'Swiss mapping' chapter). This group can be set to 0, for example, using Set PWM Group. The settings for the effects are described below.

Set event: First, make sure that the event to which you want to assign a servo in the script, for example, is defined as the control source in the 'Sound sequence' tab. Then you need to set an event marker in a .wav file. To do this, select the sound file in the User Samples window in ZSP, open it and set an event marker in it (for more details, see the instructions for ZSP and ZPP Konfig). Save the sound file with the event marker. The servo will then be moved to the left, right or centre position at the marker location within the script using the 'Set event' action, or a function will be switched on or off. The servo positions can be used to program any complex coupling waltz, for example. The end positions of the servos are programmed in the corresponding CVs #161-#173 and #770-#773 (see MX and MS decoder instructions); they cannot be changed in the scripts.

Moreover, events can also be used as a communication tool between several scripts. Of course, the event should not be used for something else. If a certain stage is reached in one script, an event 1-16 can be set to any value 0-255 and this exact moment can be queried in another script.

EMotor sample: Select a sound sample for the electric motor and determine whether it should be looped or not. The dynamic settings for the electric motor are controlled via the corresponding CVs. Just like with the thyristor samples, each newly started electric motor sample replaces the previous one.

Wait for end of EMotor sample: Wait until the electric motor sample has finished playing till end.

Take over diesel sequence: As soon as the diesel sequence is taken over (with 'Yes'), it ends in ZSP or the decoder. The desired diesel sequence sounds must be realised using sound samples in a script. When accepting, parameter 2 'Set current diesel stage' is initially irrelevant and only becomes important when returning (with "No") to the sequence in ZSP or the decoder. To ensure that the correct diesel stage is transferred during driving, the command 'Fix diesel stage' can be used in advance to define a specific stage in the sequence..

Set effect: This allows you to apply effects to the function output defined in parameter 2. You can select not only from lighting effects (Mars, Ditch, Gyra, etc.), but also smoke, SoftStart, dimming and many more. The function output defined here does not necessarily have to be switched on via a script; this command overwrites the effect that was set in ZSP or ZPP Konfig.

Query reed input: The status of a reed input (1-4) is queried. The status can be either 'On' or 'Off', or 'No event' or "Event". If the status of a reed input changes, an 'Event' is triggered, which can then be processed in the script. If the status remains unchanged (On or Off), no event occurs. The input can be selected via the corresponding CV, with values 0-3 assigned to inputs 1-4.

Query event: Comparatively queries the status of an event (1-16). These can be set in sound samples or triggered in a script. The following values apply in scripts in the 'comparison value': 0 = off/left, 1 = on/right, 2 = centre. Any other values up to 255 can also be queried as well.

Query variable: Queries the value of a set variable (1-4). These variables are used to count certain actions. For example, an additional sound can be triggered after every third time a function key is pressed (if its function is also controlled by the script). For example, the variable is assigned the value 3, and after each time the button is pressed, the value is decreased by 1 using the 'Change variable' command. As soon as the value reaches 0, the sound is played.

Set variable: This allows any variable 1-4 to be set and assigned a value between 0 and 255.

Change variable/event: Adds or subtracts a value of 1 from a variable 1-4, or an event 1-16.

Generate random value: Generates a random value between the specified 'Min' and "Max" values and applies it either to a variable (1-4) or to a timer, which is started immediately. The time unit can be selected as described above. The timer value can be queried directly afterwards in the 'Query parameter'.

Set Parameter: The following parameters are available (as of MS-SW 5.15 and MX-SW 40.22):

- *SET offset*: sets a diesel set. Value 0 = diesel set 1 (MS and MX)
- *Chuff rate CV828*: Optionally uses the steam cycle of CV #828 (not yet implemented).
- *Cylinder valves on*: Enables or disables 'Drain' (not yet implemented)
- *Activate mute*: Activates or deactivates the mute function including fade time (MS and MX)
- *Activate pulse output*: Applies the pulse of the smoke fan to a function output (not yet implemented)
- *deactivate momentum*: Enables or disables acceleration deactivation according to CV #124 (currently only for MX).
- *Invert motor direction*: Reverses the direction of the drive motor (only MS).
- *Driving sound on/off*: Regardless of the sound's running status, this command forces the running sound to OFF or ON. This command can be compared to 'Fix diesel level' above. Any diesel levels in between are played in sequence. (only MS).
- *Invert lights*: Inverts the direction of all direction-dependent lights (only MS).
- *Invert loco direction*: Inverts the feedback of the protocol direction (only MS).

Set PWM group: This allows you to set or change the settings of the PWM groups in the ZIMO mapping. The PWM (dimming) value can be set in 3% steps and, just like in the ZIMO mapping, you can also set up a flashing pattern or switch off an effect.

Set volume: (only MS): sets the overall volume of the entire sound project to a selectable percentage from 0% = off to 200% = double volume, 100% corresponds to the value =64 in CV #266, 200% corresponds to the value =128. The fade time (from the current volume to the volume in the script) can be set in 0.1s steps.

Set CV (only MS): Sets any CV to a specified value (the permissible CV values are specified in the operating instructions for MS decoders and must be observed!). This value can either be set as a constant or applied to a variable or an event.

Set CV Bit (only MS): here, bits 0-7 can be set to 0 or 1 in any CVs.

Read CV (only MS): here, the value of any CV can be read out and then applied to a variable 1-4.

Read CV Bit (only MS): reads bits 0-7 of any CV and applies them to variable 1-4.

Example:

The easiest way to explain how to use the scripts is with an example (here is a sample script, which can be downloaded from the ZIMO Sound database):

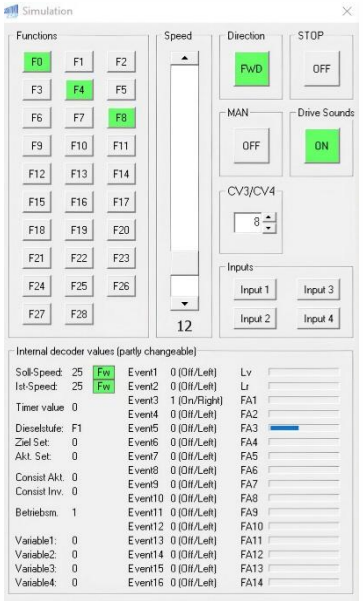
How this script is 'read':

0. Is driving sound off?
   a. Yes? → stay in State
   b. No? → proceed to State 1
1. Is the driving speed 0?
   a. Yes? → go back to 0
   b. No? → proceed to the next state, i.e. state 2
2. If function key F4 is off?
   a. Yes? → go back to 0
   b. No? → continue to State 3
3. Which direction is currently set?
   a. Backwards? → Go to State 8
   b. Forward? → proceed to the next state, i.e. state 4
4. Starts whistle sound forwards.
5. Wait until the whistle has finished.
6. Is function key F4 still on?
   a. Yes? → stay on State 6
   b. No? → proceed to the next state, and then to 0.

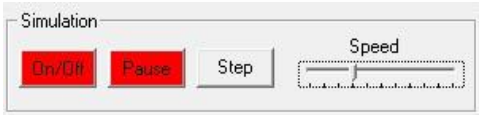
The query sequence for states 4-7 is repeated in states 8-11 for the whistle in reverse driving direction. The query for the driving sound could also be carried out directly before a 'Start sound' command.

Simulate scripts:

Once the script has been finished, it can be tested for errors in both the ZSP and ZPP Konfig. It is also possible to test the script and check how it works by executing it in a simulation. This is done by clicking the 'Simulation On/Off' button in the middle at the lower part of the script editor to start the simulation.



The simulation currently (ZSP V1.24.12) does not work with all sequences and parameters, as ZSP and ZPP Config are unable to replicate certain functions in some cases (such as servo movements, queries for diesel set, etc.). However, every script can be run to check whether it gets stuck at a certain point or forms a loop. If the script simulation is executed too quickly and the individual steps cannot be traced or an error occurs which cannot be determined exactly, it is possible to test the script step by step. To do so, click on 'Pause' to stop the simulation. Then, with "Step", each step in the script can be checked individually. The speed of the simulation can also be adjusted using the 'Speed' option.



The acceleration and deceleration behaviour can be set in the CV3/CV4 field. If the driving sound is checked in a script, 'Driving sound ON' or the corresponding function key can be activated in the simulation. It is possible to simulate inputs 1-4, and variables, events and function outputs that are triggered in this script are displayed. For target set and act. set, you can click in the field next to it and enter the desired values.

Only the currently open script can be tested – possible interactions with other scripts cannot be tested at this time.

Next steps once the script is working successfully:

Click ‘OK’ at the bottom right to save the script and return to the overview window of the script.

If the script is finished but does NOT work as desired:

Working with scripts quickly reveals that they can become very extensive and complex. The simulator is a useful tool for analysing errors. If the error cannot be identified this way, save the script, create a new script and recreate the faulty section with as few states as possible. It can also be helpful to write down the desired behaviour that the script should trigger.

ZIMO Elektronik GmbH