ZSP

BETRIEBSANLEITUNG

Scripts

Einführung in die Programmierung
von Skripten in ZSP

Gültig ab Software Version
ZSP V1.21.22

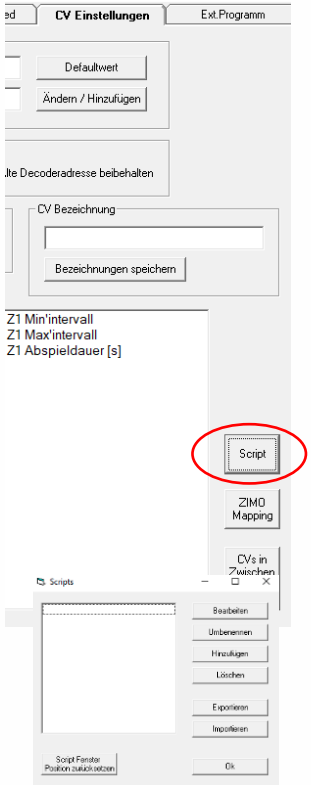
AUSGABEN:

2022 Okt
2024 Mar

Allgemeines:

Um Abläufe und Abhängigkeiten zu programmieren, gibt es in ZSP die Möglichkeit, mit „Scripts“ zu arbeiten. Scripts kann man sich als programmierte Abläufe vorstellen. Um sie zu erstellen braucht man allerdings keine Programmierkenntnisse, denn ZSP bietet die Möglichkeit, die gewünschten Abläufe in einer Eingabemaske zu programmieren. Pro Soundprojekt ist es auf MS-Decodern ab
MS-FW 4.207 möglich, bis zu 16 Scripts zu erstellen (oder zu importieren) und auf MX-Decodern ab MX-FW 39.0 bis zu 8 Scripts. Es gibt bei MX-Decodern auch einige Einschränkungen im Vergleich zu MS-Decodern, z.B. dass nur 6 unabhängig abspielbare Soundkanäle zur Verfügung stehen und in einem Script nur in bestimmten Fällen zwei Samples gleichzeitig abgespielt werden können. Für beide Decoder-Familien gilt: pro Script kann man bis zu 255 States (Befehle / Zeilen) programmieren. Deaktivieren kann man die einzelnen Scripts in den zugehörigen Bits von CVs #837 und #843.

Scripts erstellen und bearbeiten:

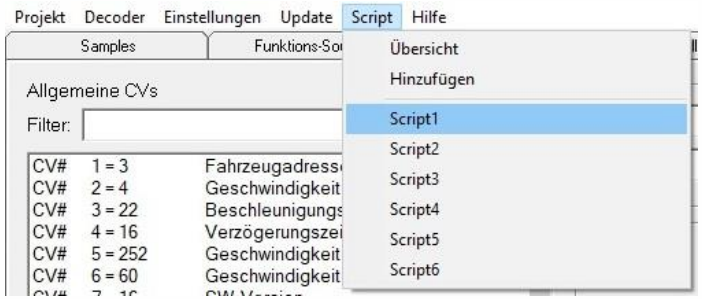


ZSP wie gewohnt starten. Die Scripts findet man im Reiter **CV Einstellungen**, dort klickt man rechts unten auf die Schaltfläche **Script**.

Daraufhin öffnet sich das Script-Hauptfenster mit einem leeren weißen Feld. Um ein Script zu erstellen, klickt man auf den Button **Hinzufügen**, und es erscheint ein blau markiertes Feld „Script1“. Mit **Umbenennen** kann dem Script ein charakteristischer Name zugeordnet werden, damit man es in weiterer Folge leichter erkennt.

Über **Bearbeiten** oder mit Doppelklick auf ein Script öffnet sich der Script-Editor. Falls sich dieser nicht öffnen sollte (oder am Bildschirm nicht auffindbar ist), kann man ihn mit dem Button **Script Fenster Position zurücksetzen** in die linke obere Ecke des Bildschirms setzen.

Ein Shortcut direkt zum Script-Editor findet sich zudem in der obersten Menüleiste des allgemeinen ZSP-Fensters mit der Bezeichnung **Script**. Beim Anklicken klappt ein Menü auf und man kann direkt ein neues Script **Hinzu-fügen**, oder bereits vorhandene Scripts öffnen. Ebenso kann man über **Übersicht** das Script-Hauptfenster aufrufen, in dem man wieder Scripts erstellen, bearbeiten, importieren, ex-portieren und löschen kann.



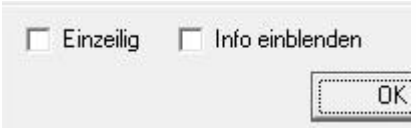
Ab ZSP Version V1.21.22 ist es möglich, über ZPP Konfig Scripts und zugehörige Sounds einer .zpp-Datei hinzuzufügen, zu importieren und dort zu bearbeiten. Allerdings ist das Bearbeiten von bereits vorhandenen Scripts in einer .zpp-Datei nicht möglich, da diese vom Soundprovider speziell an den Ablauf des Projekts angepasst sind und eine Änderung der oft komplexen Scripts zu unerwünschtem Verhalten führen kann.

Exportieren: ein Script kann man an einen beliebigen Speicherort exportieren, um es z.B. anschließend in einem anderen Soundprojekt zu verwenden, oder jemandem weiterzugeben. Es wird als .zsc[1]-Datei gespeichert.

Importieren: wenn ein Script gespeichert und exportiert wurde, kann man es in einem anderen Soundprojekt wieder importieren. ACHTUNG: zu Sound-Samples wird nur ihre Index-Nummer innerhalb des aktuellen Soundprojektes abgespeichert und können daher von Soundprojekt zu Soundprojekt unterschiedlich sein; dadurch müssen die Sound-Samples nach dem Import in ein anderes Soundprojekt manuell überprüft und angepasst werden!

Fenstergröße: Das Fenster des Script-Editors kann man in der Größe verstellen und auch maximieren. Dies bietet bei langen und komplexen Scripts den Vorteil, dass mehr States auf einen Blick sichtbar sind.

Script Editor



Rechts unten im Script-Editor (ein geöffnetes Script) befindet sich ein Kästchen mit der Beschreibung **Info einblenden**. Wenn dieses aktiviert ist, kann man jedem State einen Namen geben. Dies kann bei komplexen Scripts helfen, den Überblick zu bewahren, um auch später noch genau nachzuvollziehen, was man mit welchem Befehl bezwecken wollte. Den frei wählbaren Namen oder Text gibt man ein, indem man mit der Maus auf Info klickt und einfach zu schreiben beginnt. ZSP speichert die Bezeichnung ab und man kann jederzeit (durch Setzen oder Entfernen des Häkchens) zwischen den ursprünglichen Bezeichnungen der Befehle und der eigenen Beschreibung hin- und herschalten. Daneben ist das Kästchen **Einzeilig**, mit dem die Info zu jedem Parameter ausgeblendet werden kann. Damit wird das Script optisch kompakter, mitunter aber auch unübersichtlich. Hier können nun mit dem Schieber die einzelnen Parameter zusammengeschoben werden, um links daneben den erklärenden Text zu den States anzuzeigen.

[1] für ZIMO Script

Links unten im Script-Editor befindet sich ein Infokästchen mit dem Titel „Bearbeiten“ für nützliche Aktionen während der Bearbeitung der Scripts.
Diese sind:

Bearbeiten
Einfügen: Klick auf State-Nummer
Kopieren: Klick auf anderen State nach Einfügen
Löschen: Befehl '-' auswählen

- Einfügen: Klick auf State-Nummer (fügt einen leeren State oberhalb ein). Verknüpfungen zu bereits bestehenden States ändern sich entsprechend automatisch.
- Kopieren: Klick auf einen anderen State nach Einfügen (fügt den kopierten State in den zuletzt eingefügten – noch leeren – State ein. Dies funktioniert nur, solange der eingefügte State noch rot hinterlegt ist).
- Löschen: Befehl ´-´ auswählen, im Parameter „Befehl“ ganz oben. Zur Sicherheit wird noch einmal abgefragt, ob ein State wirklich gelöscht werden soll.

In der Mitte unten im Script-Editor ist noch das Feld „Simulation“, welches aber erst später erklärt wird, da es das fertig geschriebene Script testweise abspielt.

Scripts lesen und schreiben:

Scripts „liest“ man am besten folgendermaßen: man beginnt links oben und liest es Zeile für Zeile, wie einen Text, bis rechts unten. Genauso macht es auch ZSP, und später der Decoder, beim Abspielen der Scripts. Bei jedem Befehl können je nach Art mehrere Parameter abgefragt werden. Prinzipiell kann man mit den Scripts (innerhalb der vorgefertigten Aktionen und Parameter) sehr viele verschiedene Abläufe generieren. Bei der Konfiguration gibt es allerding ein paar Grundregeln, die man in jedem Fall beachten muss. Diese sind folgende folgende:

⚠ Wenn das Einschalten einer Taste abgefragt wird, muss auch das Ausschalten der Taste abgefragt werden

Dies erfolgt über die Aktion „Abfrage Funktion“. Die Abfragen funktionieren so, dass ein Zustand (einer Taste) abgefragt wird, und wenn dieser zutrifft, springt das Programm in den State, der in der letzten Spalte eingestellt ist, und führt den Ablauf an dieser Stelle fort. Wenn die Abfrage allerdings negativ ist, wird der Ablauf im nächsten State fortgeführt. Im folgenden Beispiel bleibt der Ablauf im gleichen State bis F2 eingeschaltet wird.

State	Befehl	Taste	Zustand	State wenn Taste=Zustand
0	Abfrage Funktion	F2	Aus	0

Zu lesen ist dies so: Der Befehl *Abfrage Funktion* prüft, ob *F2 Aus*geschaltet ist. Wenn dies der Fall ist, springt das Programm zu *State 0* (also bleibt in derselben Zeile). Wenn F2 nun (am Fahrpult oder über ZSP) eingeschaltet wird, ist die Abfrage negativ, und das Script springt automatisch in die nächste Zeile, also zum nächsten State. In diesem nächsten State sollte eine beliebige Aktion eingestellt werden, die mit dieser Funktionstaste ausgeführt werden soll.

Nach erfolgter Aktion muss man die Taste wieder ausschalten, was im Script aussieht, wie das gezeigte Beispiel, mit dem Unterschied, dass der „Zustand“ auf *Ein* geprüft werden muss, und der State auf sich selbst verweisen muss. Siehe folgendes Beispiel:

State	Befehl	Taste	Zustand	State wenn Taste=Zustand
0	Abfrage Funktion	F2	Aus	0
	Befehl	Sound	Lautstärke	Wiederholungen
1	Sound starten	Sieden_01_32_ID(A100	-0 dB	Loop
	Befehl			
2	Auf Sound Ende warten			
	Befehl	Taste	Zustand	State wenn Taste=Zustand
3	Abfrage Funktion	F2	Ein	3

⚠ Am Ende eines jeden Scripts muss wieder zum Anfang gesprungen werden

Dies ist äußerst wichtig, da das Script sonst anhält und nicht mehr funktioniert. Die letzte Zeile in jedem Script könnte also folgendermaßen aussehen:

	Befehl	State
11	Gehe zu State	0
	Befehl	
12	-	

Es muss nicht zwingend vom letzten State aus auf State 0 gesprungen werden, es ist nur wichtig, dass das Script in sich einen Kreislauf bilden kann und nicht an irgendeiner Stelle offen ist. Grundsätzlich müssen auch alle Aktionen (Sounds, Funktionsausgänge, Events, etc.), die über ein Script gestartet werden, dort wieder beendet werden.

Beschreibung der Befehle:

Sound starten: Startet einen Sound, der im Parameter „Sound“, in dem alle Sound-Samples des Soundprojekts aufgelistet sind, ausgewählt wird. Der nächste Parameter bestimmt die Lautstärke, mit der der Sound abgespielt wird. Anschließend kann man einstellen, ob der Sound geloopt werden, oder wie oft der Sound abgespielt werden soll (1x bis 8x). Loop und Short ist nur relevant, wenn das ausgewählte Sound-Sample geloopt ist und der Loop vorzeitig beendet werden soll. Wenn ein Sound-Sample hinter einem Loopmarker-Ende noch weitergeht, wird dieses Ende ebenfalls noch abgespielt. Die Loops der Sound-Samples müssen wie gewohnt in ZSP bearbeitet sein. Die Lautstärke eines Samples kann auch über CVs #980-#1019 eingestellt werden. Hierzu im letzten Fenster „Lautstärke über CV“ – „Ja“ auswählen und im Fenster „Lautstärke“ die gewünschte CV auswählen. Der Vorteil daran ist, dass die Lautstärke später einfach per CV-Programmierung geändert werden kann und nicht das Script angepasst und das ganze Soundprojekt neu auf den Decoder geladen werden muss.

Sound beenden: Wenn der Sound mit Loop oder Loop und Short gestartet wurde, muss dieser Befehl gesetzt werden, um den Sound wieder zu beenden. Wenn nur einmalig oder mit Wiederholungen gestartet wurde, ist dieser Befehl nicht notwendig (da bei fix definierter Anzahl der Wiederholungen der Sound von selbst beendet wird), es sei denn, die Wiederholungen sollen durch andere Bedingungen im Ablauf frühzeitig beendet werden.

Auf Sound Ende warten: Der weitere Ablauf des Scripts wartet so lange, bis das Sound-Sample (inklusive aller eingestellter Loops und Wiederholungen) zu Ende gespielt wurde und springt erst dann zum nächsten State.

Abfrage Funktion: Prüft, ob eine Funktionstaste (welche man in Parameter 1 einstellt) ein- oder ausgeschaltet (Parameter 2) ist und definiert, wohin gesprungen werden soll, wenn diese Prüfung positiv ausfällt. Wenn ein Sound z.B. mit Taste F5 ausgelöst werden soll, prüft man, ob F5 ausgeschaltet

ist – und springt zum (bleibt beim) eigenen State. Sobald die Taste aktiv ist, springt das Script dann automatisch zum nächsten State. Es kann in Parameter 1 auch wieder eine der Script-CVs #980-#1011 abgefragt werden; Werte =0 bis =28 für Tasten F0 bis F28.

Abfrage Parameter: Vergleicht ob der abgefragte Parameter (Auswahlfeld *Wenn Parameter*; siehe anschließende Beschreibung der Parameter) gleich, ungleich, größer, kleiner (Auswahlfeld *Test*), als der definierte Wert (Auswahlfeld *Vergleichswert*) ist. Zudem kann man den Test auch über einen entsprechenden Wert in den Script-CVs #980-#1019 abfragen. Genau wie bei *Abfrage Funktion* definiert man im letzten einstellbaren Parameter, wohin das Script springen soll, wenn diese Prüfung positiv ausfällt. Wenn diese Prüfung negativ ausfällt, wird der Ablauf im nächsten State fortgeführt.

Die Folgenden Parameter können im Auswahlfeld *Wenn Parameter* eingestellt werden:

- Fahrgeschwindigkeit: Aktuelle interne Fahrstufe (0-255[2]). Wenn der „Test" über eine CV erfolgt, ändert sich die Auswahl beim nächsten Parameter von „Vergleichswert" auf „CV". Dies ist für alle folgenden Abfragen gültig.
- Zielgeschwindigkeit: Am Regler eingestellte Fahrstufe (0-255), auf die beschleunigt oder verzögert werden soll.
- Fahrsound: Prüft, ob Fahrsound *Aus*- oder *Ein*geschaltet ist. Diese Abfrage ist bei Sounds zu bevorzugen, statt der Abfrage nach der Sound-Taste, z.B. F8.
- Ziel-Set: Prüft, auf welches Diesel-Set als nächstes gesprungen wird (z.B. Set-Wechsel per Taste). Folgend wird der Begriff „Diesel"-Set oder -Stufe im Sinne von in ZSP eingerichteten Fahrstufen verwendet. Diese können vom Sound her Diesel-Geräusche sein, oder etwa der Lüfter einer Elektrolok. Der Set-Index ist im Script: Diesel-Set 1 = Index 0, Diesel-Set 2 = Index 1, usw. Wenn in einem Projekt mit Übergangs-Sets gearbeitet wird (bei Dual-Fahrzeugen mit Elektro- und Dieselantrieb, CV #346 Bit2), hat das erste Diesel-Set den Index =0 und das 4. vollständige Diesel-Set den Index =1. Die Übergangs-Sets können nicht explizit abgefragt werden und der Index-Wechsel findet jeweils mit dem Ende des Übergangs-Sets statt.
- Aktuelles Set: Prüft, welches Diesel-Set momentan abgespielt wird.
- Diesel-Stufe: Prüft, auf welcher Stufe sich der Diesel-Sound befindet.
- Timer-Wert: Zählt von in „Timer laden" gesetztem Wert hinunter auf 0. Mit diesem Parameter ist es möglich, innerhalb des Countdowns Aktionen auszulösen.
- Beschleunigung: Anzahl an Fahrstufen, um die beschleunigt wird. Wenn das Fahrzeug bei der Prüfung verzögert, liefert das Script den Wert 0.
- Verzögerung: Anzahl an Fahrstufen, um die verzögert wird.
- Ist-Fahrrichtung: Prüft auf aktuelle Fahrrichtung (vorwärts, rückwärts).
- Soll-Fahrrichtung: Prüft, welche Fahrrichtung die Lok laut Zentrale haben soll (z.B: Lok noch im Bremsvorgang in „alter" Richtung).
- Consist aktiv: Prüft, ob sich Lok momentan im Consist-Betrieb befindet.
- Consist invers: Prüft, ob in CV #19, Bit 7 = 1 gesetzt (invertiert die Fahrrichtung im Consist-Betrieb) und ob Consist-Betrieb aktiv ist.
- HLU-Limit: Aktuelles Limit der HLU-Bremsstrecke. Die Werte über die Script-CVs sind: =0: HLU-Notstop, =2: Halt, =3: UH, 4= U, =5: LU, =6: L, =7: FL, =8: F.
- ABC-Limit: Aktuelles Limit der ABC-Bremsstrecke. Die Werte über die Script-CVs sind: =2: Halt, =5: Langsam, =8: Fahrt.
- DC-Limit: Aktuelles Limit der DC-Bremsstrecke. Die Werte über die Script-CVs sind: =2: Halt, =8: Fahrt.
- Akt. Limit: Niedrigstes Limit der drei vorigen Parameter. Damit diese nicht alle drei im Script abgefragt werden müssen, testet dieser Parameter auf den niedrigsten gefundenen Wert.

Lok steuern: Setzt Parameter für die Steuerung der Lok. Unter Parameter „Kontrolle" wird die Art der Beeinflussung definiert:

- Keine Beeinflussung: wird verwendet, um die folgenden Befehle wieder aufzuheben.
- Fahrstufe vorgeben: ungeachtet der aktuell eingestellten Fahrstufe wird hiermit eine bestimmte Fahrstufe vorgegeben. Dies kann auch Fahrstufe 0 sein.
- Fahrstufe begrenzen: hiermit wird eine maximale Fahrstufe vorgegeben, die nicht überschritten werden kann.
- Fahrstufe reduzieren: reduziert die interne Fahrstufe (255) verhältnismäßig zur eingestellten Fahrstufe, z.B.: Wert 128 reduziert etwa auf die Hälfte, Wert 64 etwa auf ein Viertel.
- Nur Richtung setzen: erzwingt eine Fahrtrichtung (vorwärts, rückwärts, Ost, West, etc.).
- Notstop: hiermit wird ein Notstop während der Fahrt eingeleitet. Die Bremsverzögerung für den Notstop kann in CV #111 eingestellt werden.
- Beschleunigung verhindern: wenn dieser Befehl aktiv ist, kann man nur verzögern oder mit gleicher Geschwindigkeit fahren, aber eben nicht beschleunigen.

In „Fahrstufe" wird die interne Fahrstufe 0-255 definiert. Parameter 3 definiert die Richtung, in der die Beeinflussung gilt (In Fahrrichtung = aktuelle Fahrrichtung, gegen Fahrrichtung = entgegen der aktuellen Fahrrichtung, Vorwärts = Fahrrichtung vorwärts der Lok, Rückwärts = Fahrrichtung rückwärts der Lok, Ost, West). Wenn „Fahrstufe über CV – Ja" ausgewählt wird, ist eine der Script-CVs #980-#1019 auszuwählen.

Thyristor Sample: Wählt ein Sound-Sample für den Thyristoreffekt aus. Das Script unterscheidet nicht, welches Soundprojekt momentan bearbeitet wird, weshalb es diese Funktion auch bei Dampf- und Dieselprojekten gibt. In der Regel werden Thyristor-Samples geloopt. Wenn man manuelle Übergänge einbauen will, kann man den Loop auch abschalten.

Auf Thyristor Ende warten: Wenn ein Thyristor-Sample beendet wird, kann hiermit auf das Ende gewartet werden bevor das nächste Sample geladen wird. Die Samples würden sonst übereinander abgespielt werden, oder es kommt möglicherweise zu einer Soundlücke.

Thyristor Tonhöhe: Bei diesem Befehl definiert man den Anstieg der Tonhöhe des Thyristors, gebunden an bestimmte Fahrstufen. So definiert man in Parameter 1 die Tonhöhe des Thyristors bei mittlerer Fahrstufe (Parameter 3) und in Parameter 2 die maximale Tonhöhe des Thyristors bei maximaler Fahrstufe (Parameter 4). Die Einstellungen hier entsprechen den CVs #291-#293 und #838. Einmal in einem Script verwendet, kann die Wirkung dieser CVs nicht mehr an den Ablauf des Decoders zurückgegeben werden und alle Änderungen müssen erneut im Script verwirklicht werden.

[2] Je nach eingestelltem Fahrstufenmodus bitte nachrechnen. Bei 128 Fahrstufen muss der Wert dementsprechend ungefähr mal 2 gerechnet werden

I²C Command: ZIMO-interne Testfunktion zur Ausgabe von Signalen an I²C-Erweiterungen; aktuell aber nicht implementiert.

Timer laden: Hier wird ein Timer eingestellt, der im Hintergrund abläuft. Der Timer zählt in Zehntel-Sekunden vom eingestellten Wert auf Null zurück. Wenn nach einer bestimmten Zeit eine Aktion ausgelöst werden soll, wird der Parameter „Timer-Wert" in *Abfrage Parameter* geprüft! Maximal kann ein Timer auf den Wert 255 (25,5 Sekunden) gestellt werden. Wenn ein längerer Zeitraum gewünscht ist, können mehrere Timer hintereinander ablaufen. Gleichzeitig funktioniert das aber nicht und es wird jeweils der letzte gestartete Timer abgefragt.

Diesel-Stufe fixieren: Aktiviert eine fixe Dieselstufe (Wert: 0= Stand... 20= F20). Damit kann das Script die Kontrolle über den Dieselablauf[3] übernehmen, also eine bestimmte Stufe im Ablauf fixieren ungeachtet des Ablaufs. Der Decoder läuft alle bis zur ausgewählten Stufe durch und es empfiehlt sich in einer weiteren Abfrage diese „Ziel"-Diesel-Stufe abzufragen, falls diese für den Verlauf des Scripts relevant ist. Wenn man „Dieselstufe fixieren" in einem der nachfolgenden States wieder deaktiviert, ist es nicht relevant, welche Stufe man dort angibt – der Ablauf findet selbst den Weg zurück zur aktuellen Dieselstufe.

Bremsenquietschen unterdrücken: Definiert, ob Bremsenquietschen unterdrückt werden soll.

Ausgang einschalten: Schaltet einen Funktionsausgang ein, und definiert in Parameter 2 auch die Dimmung (den PWM-Wert in %) für diesen Ausgang. Im selben Parameter kann der Ausgang auch wieder ausgeschaltet werden. Zudem ist es hier möglich den Ausgang, oder den PWM-Wert oder beides über Script-CVs einzustellen. Über das normale Mapping oder in Samples eingeschaltete Ausgänge können allerdings nicht über ein Script ausgeschaltet werden; diese müssten komplett über das Script verwaltet werden. Die Einstellung der Effekte wird weiter unten beschrieben.

Event setzen: Zuerst muss man sichergehen, dass im Tab „Ablauf Sound" das Event, auf das man beispielsweise einen Servo im Script legen möchte, als Ansteuerungsquelle definiert ist. Danach muss man einen Eventmarker in einem .wav-file setzen. Dafür muss man im allgemeinen ZSP-Fenster das Soundfile auswählen, öffnen und darin einen Event-Marker setzen. Speichern Sie das Soundfile mit dem Eventmarker, daraufhin wird innerhalb des Scripts mit der Aktion "Event setzen" an der Stelle des Markers der Servo auf die Position Links, Rechts oder Mitte bewegt, oder eine Funktion Aus- oder Eingeschaltet. Mit den Servopositionen kann so ein beliebig komplizierter Kupplungswalzer programmiert werden. Die Endpositionen der Servos werden in den entsprechenden CVs programmiert, sie können in den Scripts nicht verstellt werden.

EMotor Sample: Wählt ein Sound-Sample für den E-Motor aus und definiert ob es geloopt wird oder nicht. Es gelten die Einstellungen für E-Motor in den CVs.

Auf E-Motor-Ende warten: Wartet, bis das E-Motor-Sample komplett zu Ende gespielt wurde.

Diesel-Ablauf übernehmen: Sobald der Diesel-Ablauf (mit „Ja") übernommen wird, endet er in ZSP und die gewünschten Diesel-Ablauf-Geräusche müssen über Sound-Samples in einem Script nachgebildet werden. Bei der Übernahme ist Parameter 2 zunächst irrelevant und wird erst bei der Rückgabe (mit „Nein") an den Ablauf in ZSP benötigt. Um sicher zu gehen, dass hier z.B. während der Fahrt in die passende aktuelle Diesel-Stufe übergeben wird, kann vorher über den Befehl „Diesel-Stufe fixieren" eine Stufe im Ablauf gesetzt werden.

FA Effekt setzen: Damit kann man Effekte auf den in Parameter 2 definierten Funktionsausgang legen. Auswahl stehen nicht nur Lichteffekte (Mars, Ditch, Gyra, etc.), sondern auch Rauch, SoftStart, Dimmen und viele mehr. Der hier definierte Funktionsausgang muss nicht zwingend über ein Script eingeschaltet werden; dieser Befehl überschreibt den Effekt, der über ZSP eingestellt wurde.

Abfrage Reed-Input: Hier wird ein Reed-Input (1-4) nach seinem Zustand abgefragt. Der Zustand kann sein: Ein oder Aus, oder „kein Event" oder „Event". Wenn sich der Zustand eines Reed-Input ändert, kommt es zu einem „Event", das dann abgearbeitet werden kann. Bleibt der Zustand gleich (Ein oder Aus) gibt es kein Event. Jede Änderung erzeugt somit ein Event.

Abfrage Event: Fragt vergleichend den Zustand eines Events (1-8) ab. Diese können in Soundsamples gesetzt werden oder in einem Script ausgelöst werden. Im „Vergleichswert" gelten in Scripts folgende Werte: 0 = aus/links, 1 = ein/rechts, 2 = Mitte.

Abfrage Variable: Fragt vergleichend den Wert einer gesetzten Variablen 1-4 ab. Diese Variablen dienen zum Abzählen von bestimmten Aktionen. So kann man z.B. nach jeder 3. Betätigung einer Funktionstaste (wenn ihre Funktion auch über ein Script gesteuert wird) einen zusätzlichen Sound auslösen. Der Variablen gibt man den Wert 3 und zieht nach jeder Betätigung der Taste über „Variable ändern" einen Wert ab. Sobald der Wert auf 0 ist, soll der Sound gespielt werden.

Variable setzen: Hiermit kann eine beliebige Variable 1-4 gesetzt werden und ihr ein Wert gegeben werden.

Variable ändern: Addiert oder subtrahiert einen Wert von 1 von einer Variablen 1-4.

Zufallszahl generieren: Generiert einen zufälligen Wert zwischen „Min" und „Max" und wendet ihn wahlweise auf eine Variablen 1-4 oder einen Timer an, der gleichzeitig geladen wird. Es kann direkt im Anschluss der Timer-Wert in „Abfrage Parameter" abgefragt werden.

Set Parameter: Folgende Parameter stehen zur Auswahl (Stand MS-FW 4.225 nicht implementiert):

- SET-Offset: setzt ein Diesel-Set. Wert 0 = Diesel-Set 1.
- Dampftakt CV828: verwendet wahlweise den Dampfschlagtakt von CV #828.
- Zylinderventil öffnen: aktiviert oder deaktiviert „Entwässern".
- Mute aktivieren: aktiviert oder deaktiviert die Mute-Funktion.
- Takt aktivieren: legt den Impuls des Rauchventilators auf einen Funktionsausgang.

[3] Dieselablauf: Beim Beschleunigen oder Bremsen durchläuft der Dieselmotor, oder bei E-Loks der Lüfter, verschiedene Stufen, also Stand, F1, F2, etc. und Übergangssamples (F1-F2, F2-F3, bzw. F4-F3, F2-F1, usw.). Siehe dazu die ZSP-Anleitung.

Set PWM-Gruppe: Hiermit können die Einstellungen der PWM-Gruppen im ZIMO Mapping gesetzt oder geändert werden. Der PWM-(Dimm-)Wert kann in 3%-Schritten eingestellt werden und genau wie im ZIMO Mapping auch kann man ein Blinken einrichten, oder einen Effekt ausschalten.

Beispiel:

Am einfachsten lässt sich die Anwendung der Scripts durch ein Beispiel erklären (hier eines unserer Muster-Scripts, welche hier auf der ZIMO Sound Datenbank heruntergeladen werden können):

State					
	Wenn Sound off, tu nix				
-> 0	Abfrage Parameter	Fahrsound	Ist Gleich	Aus	0
	Wenn Stillstand, tu nix				
1	Abfrage Parameter	Fahrgeschwindigkeit	Ist Gleich	0	0
	Teste Status Funktionstaste. Wenn aus, tu nix. Wenn ein, gehe weiter				
2	Abfrage Funktion	F4	Aus	0	
	Teste aktuelle Fahrtrichtung. Wenn vorwärts, gehe weiter. Wenn rückwärts, gehe zu 8				
3	Abfrage Parameter	Ist-Fahrrichtung	Ist Gleich	Rückwärts	8
	Starte Pfiff vor, einmal abspielen				
4	Sound starten	Pfiff_vor.wav	-0 dB	1x	Nein
	Info				
5	Auf Sound Ende warten				
	Warten, bis Funktionstaste aus				
-> 6	Abfrage Funktion	F4	Ein	6	
	Info				
7	Gehe zu State	0			
	Wenn Fahrtrichtung rückwärts, starte Pfiff rück, einmal abspielen				
-> 8	Sound starten	Pfiff_rück.wav	-0 dB	1x	Nein
	Info				
9	Auf Sound Ende warten				
	Warten, bis Funktionstaste aus				
-> 10	Abfrage Funktion	F4	Ein	10	
	Info				
11	Gehe zu State	0			

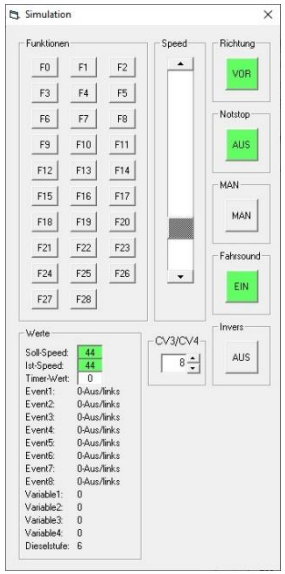
Wie dieses Script „gelesen" wird:

0. Ist Fahrsound aus?
   a. Ja? → bleibe in State
   b. Nein? → aktiviere das Script und gehe weiter zu State 1
1. Ist Fahrgeschwindigkeit 0?
   a. Ja? → gehe zurück zu 0
   b. Nein? → gehe weiter zum nächsten State, also State 2
2. Ist Funktionstaste F4 ein?
   a. Ja? → weiter zu State 3
   b. Nein? → gehe zurück zu 0
3. Welche Fahrtrichtung ist aktuell eingestellt?
   a. Rückwärts? → gehe zu State 8
   b. Vorwärts? → gehe weiter zum nächsten State, also State 4
4. Starte Sound von Pfiff vorwärts.
5. Warte, bis der Pfiff zu Ende gespielt hat.
6. Ist Funktionstaste F4 noch ein?
   a. Ja? → bleibe auf State 6
   b. Nein? → gehe weiter zum nächsten State, und anschließend zu 0.

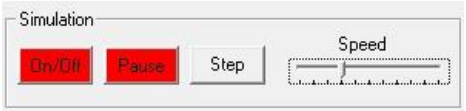
Der Abfragen-Ablauf von States 4-7 wiederholt sich in den States 8-11 für den Pfiff für Fahrtrichtung rückwärts. Die Abfrage nach dem Fahrsound könnte man auch jeweils direkt vor einem „Sound starten" machen.

Scripts simulieren

Wenn das Script fertig programmiert ist, kann man es auf Fehler testen. Auch wenn man zwischendurch testen will, ob und wie das Script läuft, hat man die Möglichkeit, das Script in einer Simulation durchlaufen zu lassen. Dies geschieht wie folgt: In der Mitte unten im Script-Editor gibt es den Button Simulation **On/Off** um sie zu starten.



Die Simulation funktioniert aktuell (ZSP V1.21.25) nur mit bestimmten Abläufen und Parametern, da ZSP bei einigen Dingen nicht in der Lage ist, sie nachzubilden (Servo-bewegungen, Abfragen nach Diesel-Set, etc.). Grundsätzlich kann man aber jedes Scripts durchlaufen lassen, um zu sehen, ob es an einer Stelle hängen bleibt oder in einem Loop durchläuft (weil man vergessen hat, etwas wieder rückgängig zu machen). Wenn das Script zu schnell läuft und die einzelnen Schritte nicht nachverfolgt werden können, oder falls ein Fehler auftritt und man das Problem nicht exakt nachvollziehen kann, hat man die Möglichkeit, das Script Schritt für Schritt durchlaufen zu lassen. Dafür klickt man auf „Pause", und dann geht man mit „Step" jeden Schritt im Script einzeln durch. Ebenso kann die Geschwindigkeit des Durchlaufs über „Speed" angepasst werden. Wenn in einem Script der Fahrsound geprüft wird, muss in der Simulation auch „Fahrsound EIN" aktiv sein, die Taste genügt hier nicht.



Nächste Schritte, wenn das Script fertig ist und funktioniert wie gewünscht:

Mit einem Klick rechts unten auf „OK" wird das Script wird gespeichert und man kommt zum Script-Hauptfenster zurück.

Wenn das Script fertig ist, aber es funktioniert NICHT wie gewünscht:

Wer sich ausführlicher mit den Scripts befasst, wird schnell feststellen, dass diese sehr umfangreich und komplex werden können. Ein gutes Hilfsmittel für die Fehleranalyse bietet der Simulator. Falls sich der Fehler auf diese Weise nicht zeigt, speichern Sie das Script ab, legen Sie ein neues Script an und versuchen Sie, den Teil in dem der Fehler auftritt mit so wenigen States wie möglich nachzustellen. Es kann auch hilfreich sein, sich das gewünschte Verhalten, das mit dem Script ausgelöst werden soll, aufzuschreiben.

ZIMO Elektronik GmbH

RailCom ist ein Markenzeichen der Lenz GmbH, mfx ist ein Markenzeichen der Märklin & Cie GmbH