Das Umkehrmodul eignet sich zum "Kopfmachen" in Stumpf- und Durchgangsgleisen. In der Basis -konfiguration sind lediglich zwei elementare Parameter (*) erforderlich.

used_TurnModul_SpDr = { {Datensatz1}, {Datensatz..n} }

Die Tabelle wird zur Moduldefinition verwendet. Mehrere Datensätze in der Tabelle werden durch ein Komma voneinander getrennt.

Ein Datensatz in der Tabelle ***kann*** die folgenden Parameter enthalten (Beispiel):

{ MasterSigId=8, MemoId="#21", FlagSigId=23, FlagSigActivPos=3, Delay=5, OpenTrack=true }

**MasterSigId=8*

Signal-Id des übergeordneten Signals, in der Regel das Ausfahrtsignal des Gleises, für das das Modul eingerichtet wird. Der Datentyp ist eine Ganzzahl.

**MemoId = "#41"*

Immobilien-Identnummer. Wird als Speichermedium für die Echtzeitdaten des Moduls benötigt. Der Datentyp ist eine String.

[optionale Parameter]

FlagSigId = 23

Signal-Id der Status-LED, falls installiert. Signalisiert einen vorhandenen Dateninhalt im Speichermedium. Der Datentyp ist eine *Ganzzahl*.

FlagSigActivPos = 3

Aktive Signalposition der Status-LED, falls eine installiert ist. Ändert den Standardwert 2 (Dauerlicht) der Status-LED in die Position 3 (Blinklicht). Der Datentyp ist eine Ganzzahl.

Delay = 10

Startet einen *modulinternen Funktions-Timer.* Die Signalfreigabe (nicht der Begriffswechsel) wird um die angegebenen Sekunden verzögert. Der Datentyp ist eine Ganzzahl (Sekunden).

OpenTrack=true

Parameter zur Angabe des Gleistyps, in dem sich das Modul befindet. Es gibt zwei Typen von Gleisen:

- *Stumpfgleis: OpenTrack = false*
- *Durchgansgleis: OpenTrack = true*

Der Datentyp ist eine boolescher Ausdruck. *Der Defaultwert ist "false".*

*Wird OpenTrack = true gesetzt, müssen die als "Pendelzüge" verkehrenden Zugverbände mit einem sog. "Tag" im Zugnamen markiert werden. Dabei reicht es, dem Zugnamen die Erweiterung " **:TM** " hinzuzufügen. Wichtig dabei ist, die genaue Schreibweise des "Tags" zu beachten !*

Um das Modul auszulösen, ist ein Einfahrt-Gleiskontakt erforderlich. In das Lua-Funktionsfeld des Gleiskontakt wird der Funktionsaufruf eingegeben. Den "CopyCode" dazu finden wir in der Datei "SpDrS60_TmpFunctionFile.txt" unter der entsprechenden Modulbezeichnung. Hier ein Beispiel:

```
------ Kontaktfunktionen fuer TurnModul 8 ------
Gleiskontakt:  SpDrTurnModCtrl_ID8_getTrainData
Signalkontakt:  SpDrTurnModCtrl_ID8_clearMemo
```

Um das Modul auszulösen, verwendet man den Funktionsaufruf zum "Gleiskontakt" und kopiert ihn in die Lua-Zeile des Kontaktdialogfeldes.

Um den Inhalt des Modulspeicher zu löschen, kann die Funktion für den "Signalkontakt" in die Lua-Zeile des Mastersignal-Rückfallkontakts kopiert werden.

Die Methoden des Moduls (Info für Selbstprogrammierer)

Hinweis: Die Methoden stehen nur bei Definition des Delay-Parameter im Moduldatensatz zur Verfügung!

SpDrTurnModCtrl_ID8.setEventTbl(_argEventTable)

übergibt dem Controller eine Eventtable. Die Tabelle muss in der folgenden Form vorliegen:

```
MyEvents01 = {
        {4,"EEPSetSignal(3,3)"},
        {6,"EEPSetSignal(17,2)"},
};
```

Dabei definiert der erste Parameter eines Datensatzes die Ausführungszeit in Sekunden nach Start des Timers und der zweite Parameter das auszuführende Event als *String*.

Die Zuweisung: `SpDrTurnModCtrl_ID8.setEventTbl(MyEvents01)`

SpDrTurnModCtrl_ID8.setEvent(_argNum,"_argString")

übergibt dem Controller ein einzelnes Event.

Die Zuweisung: `SpDrTurnModCtrl_ID8.setEvent(6,"EEPSetSignal(17,2)")`

SpDrTurnModCtrl_ID8.clearEvents()

entleert den Eventpuffer des Moduls.