Übungen zu imc FAMOS II – Digital Kurs

- Block 5 - Doc. Rev.: 1.2- 28.08.2025

Gezielter Wissenstransfer - höhere Produktivität

Übung A

Übungsziel:

In dieser Übung lernen Sie den Umgang mit Untersequenzen und deren optionaler Übergabe-/bzw. Rückgabeparametern. Als Beispiel dient die Sequenz aus Übung A Block 1, in der Berechnungen auf verschiedene Datensätze zur Anwendung kommen, welche in dieser Übung in Untersequenzen ausgelagert werden. Durch die Auslagerung wird eine Vereinfachung sowie eine größere Flexibilität der Auswertesequenz erreicht.

Aufgabenstellung:

A) Vereinfachen Sie die Sequenz aus Übung A Block 1 *(auch enthalten in den Beispieldateien)*, indem Sie den Abschnitt zum Setzen der Marker in eine Untersequenz ausgliedern. Übergabeparameter sollen sein:
   - Der Kanal, an dem der Marker erstellt werden soll.
   - X-, und Y-Koordinaten des Markers.
   - Ein beliebiger Zusatztext, der zusätzlich vor dem bestehenden Markertext erscheinen soll.

B) Gliedern Sie dann in derselben Sequenz die Erstellung der 600-s-Teilstücke sowie die Berechnung von deren Maximalwerte in eine 2. Untersequenz aus:
   - Übergabe-Parameter: Kanal, X-Wert für linken Schnittpunkt, Y-Wert für rechten Schnittpunkt.
   - Rückgabe-Parameter: Ausgeschnittenes Teilstück, Maximalwert des Teilstücks.

Ergebnis:

Sie erhalten eine Auswertesequenz mit 2 Untersequenzen, die an den entsprechenden Positionen in der Hauptsequenz aufgerufen werden.

Übungsschritte:

A) Untersequenz zum Setzen der Marker:

- Ein Untersequenzaufruf soll gemäß der Übungsbeschreibung in folgender allgemeiner Form realisiert werden (Parameternamen nur zur Veranschaulichung):

```
Seq SetMarker Channel, X-Pos, Y-Pos, TextBeforeMax
```

- Laden Sie zunächst die gespeicherte Sequenz aus Block 1 Übung A und erstellen Sie eine Untersequenz mit dem Namen **SetMarker** (Menü-Icon ).
  Hinweis: Es kann stattdessen auch mit einer separaten Sequenz-Datei gleichen Namens gearbeitet werden.

- Kopieren Sie einen der Blöcke aus der Hauptsequenz, der die Marker im Kurvenfenster setzt, bspw. den **Speed**-Block, und fügen ihn in die Untersequenz ein:

```
CwSelectByChannel("line", Speed)
CwNewElement("marker")
CwMarkerSet("line.selected", 1)
CwMarkerSet("x.type", 1)
CwMarkerSet("y.type", 1)
CwMarkerSet("x", Pos(CutSpeed, MaxSpeed))
CwMarkerSet("y", MaxSpeed)
CwMarkerSet("text", TForm(MaxSpeed, "f1.3"))
```

- Ersetzen Sie die Variable **Speed** mit **PA1**, dem 1. Parameter aus unserem Sequenzaufruf, der den Kanal enthält:

```
CwSelectByChannel("line", PA1)
```

- Erstellen Sie für die restlichen Übergabeparameter zur besseren Lesbarkeit aussagekräftige lokale Variablen, die Sie zu Beginn der Untersequenz ganz oben zuweisen:

```
Local xpos = PA2
Local ypos = PA3
Local AddText = PA4
```

- Passen Sie die letzten 3 Sequenzzeilen entsprechend der neuen Variablen an und ergänzen Sie außerdem den freien Text des Markers:

```
CwMarkerSet("x", xpos)
CwMarkerSet("y", ypos)
CwMarkerSet("text", AddText + TForm(ypos, "f1.3")); Text vor dem
Max-Wert
```

  Hinweis: Die Zuweisung zu einer lokalen Variable kann in diesem Fall nicht für PA1 verwendet werden, da ansonsten der Befehl **CwSelectByChannel()** im Kurvenfenster erfolglos nach dieser lokalen Variable suchen würde.
- Die Untersequenz ist damit fertiggestellt. Ersetzen Sie in der Hauptsequenz die 3 Blöcke, die die Marker setzen, durch entsprechende Aufrufe der Untersequenz:

```
Seq SetMarker Speed, Pos(CutSpeed, MaxSpeed), Maxspeed, "Max = "
Seq SetMarker Torque, Pos(CutTorque, MaxTorque), MaxTorque, "Max = "
Seq SetMarker Power_Engine, Pos(CutPower, MaxPower),
          MaxPower, "Max = "
```

B) Untersequenz zur Erstellung der Teilstücke sowie zur Berechnung deren Maximalwerte:

- Gemäß der Übungsbeschreibung soll ein Untersequenzaufruf für die Erstellung der 600-s-Teilstücke sowie die Berechnung von deren Maximalwerte in folgender allgemeiner Form realisiert werden:

```
Seq  CutMax  Channel,  X_Left,  X_Right,  Part,  MaxPart
```

  Die Parameter 1 - 3 sollen an die Untersequenz übergeben werden, die Parameter 4 und 5 beinhalten die Rückgabewerte.

- Erstellen Sie eine neue Untersequenz mit dem Namen **SetMarker** und definieren Sie in dieser den **Cut()** sowie den **Max()** Befehl mit Hilfe der entsprechenden Parameter:

```
PA4 = Cut(PA1, PA2, PA3)
PA5 = Max(PA4)
```

- Ersetzen Sie in der Hauptsequenz den gesamten Block aller **Cut()** und **Max()** Befehle durch entsprechende Untersequenzaufrufe:

```
Seq CutMax Speed, Range0, Range1, CutSpeed, MaxSpeed
Seq CutMax Torque, Range0, Range1, CutTorque, MaxTorque
Seq CutMax Power_Engine, Range0, Range1, CutPower, MaxPower
```


Übung B

Übungsziel:

In dieser Übung lernen Sie wie Sie die FAMOS Funktionsbibliothek durch eigene selbsterstellte Funktionen erweitern können. Die erstellten Funktionen verhalten sich dabei wie mitgelieferte Funktionen. Sie können unter anderem über die Suchfunktion in der Funktionsbibliothek gefunden und mit Hilfe des Funktionsassistenten konfiguriert werden. Auch Hilfetexte zur Bedienung der Funktion können in der Definition enthalten sein.

Neben der Erstellung solcher Sequenzfunktionen werden in dieser Übung Techniken zum Abfragen des Datentyps sowie der Umgang mit der Fehlerbehandlung behandelt.

Übungsbeschreibung:

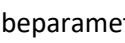
Schreiben Sie eine Sequenzfunktion, die aus einem normalen oder aus einem XY-Datensatz ohne Segmente und ohne Events seine Zeitspur extrahiert. Fangen Sie Fehler bei der Übergabe eines falschen Datentyps auf. Diese Funktion ist mit FAMOS 2025 als ***CmpX***(Data) -> ComponentX als fertige Funktion implementiert. Zur Verdeutlichung der Grundprinzipien für Sequenzfunktionen wird hier ein vereinfachtes Beispiel gebastelt.

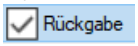
Ergebnis:

Wird der erstellten Funktion eine ND- oder XY-Variable ohne Events und ohne Segmente übergeben, wird eine Zeitspur erzeugt, ansonst erfolgt eine Fehlermeldung.

Übungsschritte:

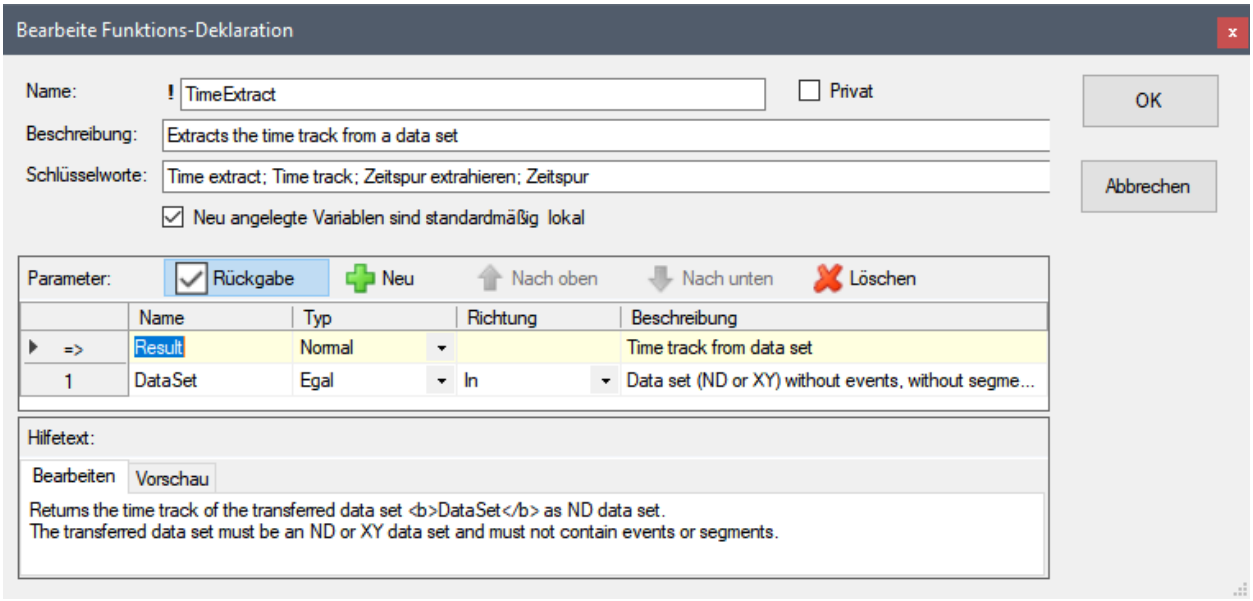
Schritt 1: Deklaration erstellen

- Öffnen Sie das Eingabefenster und erstellen Sie eine neue Sequenzfunktion durch Klick auf das entsprechende Menü-Icon  . Im Deklarationsdialog definieren Sie anschließend folgendes:
    - Vergeben Sie einen Namen für die Sequenz (z.B. **TimeExtract**).
    - Geben Sie eine Beschreibung der Sequenz ein (z.B. **Extracts the time track from a data set**).
    - Verwenden Sie einige Schlüsselworte, durch die Ihre Sequenz später in der Suchfunktion der Funktionsbibliothek gefunden werden kann (z.B. **Time extract; Time track; Zeitspur**).
    - Neu angelegte Variablen sollen standardmäßig lokal sein.
    - Die Funktion soll nicht als private Funktion deklariert werden.
    - Geben Sie einen Hilfetext ein, der im Hilfefenster angezeigt wird (z.B. **Returns time track of an ND or XY dataset, no events or segments allowed**).
    - Fügen Sie einen Übergabeparameter durch Klick auf 

hinzu. Dieser soll dem übergebenen Datensatz (z.B. **DataSet**) entsprechen und zunächst keinem speziellen Datentyp angehören.
    - Erstellen Sie einen Rückgabeparameter als normalen Datensatz, der die extrahierte Zeitspur zurückgibt (z.B. **Result**). Aktivieren Sie dazu das Feld 

.
    - Der vollständige Deklarationsdialog sollte etwa wie in der Abbildung auf der nächsten Seite aussehen.

- Bestätigen Sie den Deklarationsdialok mit OK und speichern Sie die Sequenz im FAMOS Standardordner für Sequenzen. Daraufhin öffnet sich das Editorfenster für die Sequenzfunktion.



Schritt 2: Quellcode im Editor erstellen

- Da die Extraktion der Zeitspur für die erlaubten Datensätze unterschiedlich ist, soll zu Beginn der Sequenzfunktion mit Hilfe der Funktion **VerifyVar()** der Datentyp des übergebenen Datensatzes mit Hilfe von mehreren **If** Bedingungen eingeordnet werden:

```
If VerifyVar(DataSet, "ND(->)")
    ; Sequenz für ND Datensätze ohne Events und Segmente
ElseIf VerifyVar(DataSet, "XY(->)")
    ; Sequenz für XY Datensätze ohne Events und Segmente
Else
    ; Sequenz im Fehlerfall
End
```

  In den auskommentierten Codeblöcken erfolgt später die Generierung bzw. Extraktion der Zeitspur und im Fehlerfall die Bildung einer Fehlermeldung.
- Im Falle eines normalen Datensatzes (ND) erzeugen Sie eine Zeitspurrampe mit dem Startzeitwert des übergebenen Datensatzes, seiner Abtastrate und der Anzahl der Samples des übergebenen Datensatzes und übergeben das Ergebnis **Result**:

```
Result = Ramp(xOff?(DataSet), xDel?(DataSet), Leng?(Dataset))
```

- Die Zeitspur aus einem XY-Datensatz erhalten Sie direkt aus der X-Spur des übergebenen Datensatzes:

```
Result = DataSet.X
```

- Für den Fehlerfall (kein reiner ND- bzw. XY-Datensatz) formulieren Sie eine Fehlermeldung und geben diese mit der Funktion **ThrowError()** aus:

```
ThrowError("Command '!TimeExtract': Parameter must contain a ND or
            XY dataset without events or segments.")
```


Hinweis: Mit **ThrowError()** wird ein Fehler generiert. FAMOS unterbricht standardmäßig bei Laufzeitfehlern, aber auch bei generierten Fehlern, die Sequenzabarbeitung und öffnet den Sequenzfunktionseditor. Dieses Standardverhalten kann mit der Funktion **OnError()** verändert werden.

- Passen Sie zu Beginn der Sequenzfunktion die Option für das Verhalten im Fehlerfall so an, dass die Sequenz zum Ende springt und der Editor nicht geöffnet bleibt:

```
OnError("ReturnFail")
```

- Die vollständige Sequenz sollte etwa wie folgt aussehen:

```
OnError("ReturnFail")
If VerifyVar(DataSet, "ND(->)")
   Result = Ramp(xOff?(DataSet), xDel?(DataSet), Leng?(Dataset))
ElseIf VerifyVar(DataSet, "XY(->)")
   Result = DataSet.X
Else
   ThrowError("Command '!TimeExtract': Parameter must contain a
           normal Dataset or XY dataset without events or segments")
End
```

- Speichern Sie die Sequenzfunktion ab.

Schritt 3: Neue Funktion testen und debuggen

- Laden Sie den Datensatz **slope.dat** aus den Beispieldaten in die Variablenliste.
- Öffnen Sie das Eingabefenster und suchen Sie in der Funktionsbibliothek nach der Sequenzfunktion mit Hilfe des Namens oder eines der definierten Stichworte.
- Markieren Sie die Funktion und rufen Sie den Funktionsassistenten auf (Tastenkombination **Umschalt + F1**). Wählen Sie den Datensatz **slope** als Parameter und geben Sie einen Ergebnisnamen (z.B. **slope_timetrack**) an. Kopieren Sie die Befehlszeile in das Eingabefenster und schließen sie den Assistenten anschließend wieder. Die Befehlszeile sollte lauten:

```
slope_timetrack = !TimeExtract(slope)
```

- Markieren Sie die Zeile, sodass der Compiler-Pfeil vor der Zeile steht und wählen Sie im Menü **Ausführen** den Punkt Einzelschritt in Untersequenz (Tastenkombination **Umschalt + F9**). Daraufhin öffnet sich der Editor der Sequenzfunktion und Sie können gewohnt per Einzelschritt ( ) oder gesamter Sequenzausführung ( ) die Sequenzabarbeitung testen. Bei erfolgreicher Ausführung erhalten Sie anschließend den Datensatz **slope_timetrack** in der Variablenliste.
- Öffnen Sie die berechnete Zeitspur sowie den Originaldatensatz im Dateneditor und vergleichen Sie die Zeitinformationen.
- Laden Sie die Datei **rpm_V.dat** aus den Beispieldaten und erstellen Sie eine Zeitspur für den importierten Datensatz mit der erstellten Sequenzfunktion. Führen Sie die Zeile diesmal direkt aus, ohne in die Untersequenz zu springen.
- Zum Testen des Fehlerfalls erstellen Sie eine Textvariable und versuchen von dieser eine Zeitspur zu berechnen.

