Riesenrad

Programmgesteuertes Riesenrad

Schrittweise Entwicklung eines Betriebsprogrammes für das Riesenrad aus Fun Park 57 484
(Anleitung allein 62 959)

Das mit Motorantrieb ausgerüstete Riesenrad wurde dazu nur wenig umgerüstet :

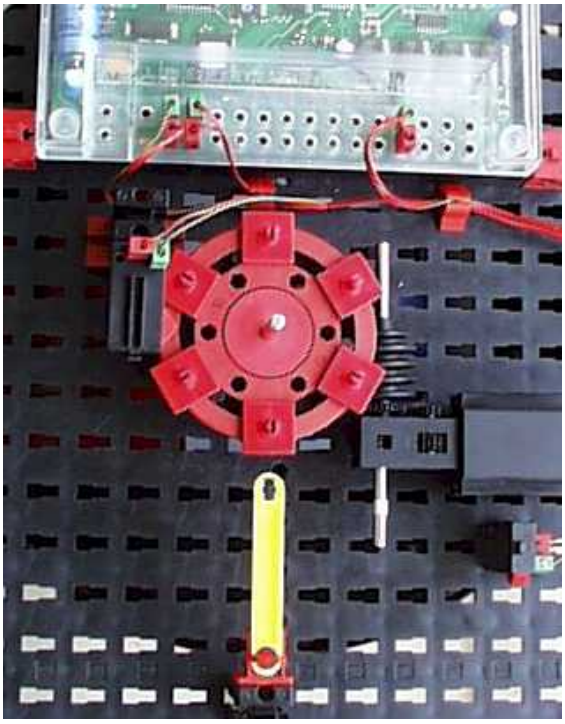
- Antriebsmotor an M1
- Im Fußbereich wurden das Intelligent Interface und der Taster E1 untergebracht.
- Am hinteren Scheibenrad wurden Schaltnocken (siehe Bild) und der Taster E2 montiert.

Das ist alles.

Für diejenigen, die rein zufällig gerade kein Riesenrad haben oder denen ein Riesenrad auf dem Schreibtisch zu sperrig ist, aber trotzdem die gezeigten Programme ausprobieren wollen :

Ein Simulationsmodell.

Es kann die gleichen Steuerfunktionen ausführen wie das Original. Das Zeitverhalten entspricht nicht ganz dem großen Modell. Besonders beim Anfahren



und Anhalten macht sich das bemerkbar. Die verwendeten Zeitkonstanten sind anzupassen (zu verkleinern).

Der gelbe Zeiger markiert die Position der unteren Gondel.

Und nun geht's los :

- mit LLWin 3.0
- mit **VBA** und der Entwicklungsumgebung vbaFish30 gleich hier :

Schritt 1 : Aus- und Einsteigen der Fahrgäste

```
Sub Main
  Dim i%
  For i = 1 To 6
    SetMotor ftiM1, ftiLinks
    WaitForHigh ftiE2
    Pause 1500
    SetMotor ftiM1, ftiAus
    Pause 4000
  Next i
End Sub
```

Alle Gondeln werden nacheinander zu Einsteigeposition gefahren. Zum Aus- und Einsteigen wird gehalten.

Kernpunkt der Positionserkennung ist der Befehl WaitForHigh (Warten auf einen False/True-Durchgang). Ein schlichtes Warten auf E2 = True reicht nicht, da noch auf True stehen kann.
Da mit E2 = True die exakte Einsteigeposition nicht gewährleistet ist (Lage des Taster, Masse des Riesenrades), wird nach E2 = True noch ein paar (1,5) Sekunden weitergefahren und dann erst abgeschaltet. Danach folgt die Pause für das Aus- und Einsteigen. Das ganze immer schön in einer 6er-Schleife.

ACHTUNG : Die Länge der Pause (hier `Pause 1500`) hängt beim Original Riesenrad von der Beladung und der Befestigung der Hauptachse ab. Sie lag bei mir zwischen 1500 (12 Männeken) und 3200 (leer).

Schritt 2 : Die Fun-Runden :

```
Sub Main
  Dim i%
  Do
    PrintStatus "--- Ein- und
Aussteigen ---"
' --- weiter wie oben ...

    PrintStatus "--- Tour linksrum -
--"
    SetMotor ftiM1, ftiLinks
    Pause 15000 * EA
    SetMotor ftiM1, ftiAus
    Pause 1000
    PrintStatus "--- Tour rechtsrum
---"
    SetMotor ftiM1, ftiRechts
    Pause 15000 * EA
    SetMotor ftiM1, ftiAus
    Pause 1000
  Loop Until Finish(ftiE1)
End Sub
```

Das ganze Programm wird in eine Do ... Loop-Schleife gepackt, so erhält man ein schönes Demo-Programm, das durch E1 = True, die ESC-Taste oder den HALT-Button der Entwicklungsumgebung abgebrochen werden kann. Zusätzlich wird mit PrintStatus in der Statuszeile der IDE die aktuelle Funktion angezeigt.

Nach dem Ein-/Aussteigen (wie bisher) wird 15 Sek. links und dann 15 Sek rechts gedreht. Da es bei sofortiger Richtungsumschaltung richtig knirschen kann, wird dazwischen 1 Sek. Pause eingelegt.

Wenn man die Runden gerne länger hätte, kann man im EA-Feld der IDE einen Faktor eingeben, der die Rundenzeit entsprechend multipliziert.

Schritt 3a : Echt-Betrieb

```
Sub Main
  Dim i%
  For i = 1 To 6
    SetMotor ftiM1, ftiLinks
    WaitForHigh ftiE2
    Pause 1500
    SetMotor ftiM1, ftiAus
    WaitForLow ftiE1
  Next i
  PrintStatus "Starten des
Fahrbetriebs : E1"
  Pause 3000
  If Not GetInput(ftiE1) Then Exit
Sub
' --- weiter wie gehabt ...
End Sub
```

Bei einem Echt-Betrieb sind die Zeiten für Aus- und Einsteigen nicht vorhersehbar, die Pause 4000 wird deswegen durch ein WaitForLow ftiE1 ersetzt. Das Programm wartet bis E1 gedrückt und wieder freigegeben wird.

Nach dem Aus-/Einsteigen muß der Betrieb durch erneutes Drücken der E1-Taste innerhalb von 3 Sek. freigegeben werden, sonst wird das Programm beendet : Feierabend.

Man kann sich zum Betrieb natürlich noch mehr einfallen lassen.

Schritt 3b : Symbolische Namen

```
Sub Main
Const mRadMotor = ftiM1, eRadPos =
ftiE2, _
                        eQuittung
= ftiE1
....
    WaitForHigh eRadPos
....
End Sub
```

Anstelle der allgemeinen Bezeichnungen für die Ein- und Ausgänge des Interface sollte man, wenn's was größeres wird, besser spezielle symbolische Namen setzen. Hier mRadMotor, eRadPos, eQuittung.
Sie können dann überall verwendet werden, wo jetzt die fti-Namen stehen.

Schritt 4a : Zählen statt Warten

```
Sub Main
....
SetMotor mRadMotor, ftiLinks
PrintStatus "--- Tour linksrum "
WaitForChange eRadPos,12 * EA
SetMotor mRadMotor, ftiAus
....
End Sub
```

Die Pause 15000 * EA wurde durch ein WaitForChange ersetzt. Da WaitForChange den Wechsel von True auf False und umgekehrt separat zählt, muß für eine Runde 12 angegeben werden. Die Anzahl Runden kommt aus dem Feld EA.

Schritt 4b : Runden-Anzeige

```
Sub Main
....
SetMotor mRadMotor, ftiLinks
For i = 1 To EA
  PrintStatus "--- Runde : " & i &
" linksrum"
  WaitForChange eRadPos, 12
Next i
SetMotor mRadMotor, ftiAus
....
End Sub
```

Wenn man im Statusfeld die Nummer der aktuellen Runde anzeigen will, kann man das über eine For .. Next-Schleife über jeweils eine Runde tun. Die Rundenzahl kommt wieder von EA

Schritt 5 : Überkreuz Beladen

```
Sub Main
Dim i%
Do
  PrintStatus "--- Ein- und
Aussteigen ---"
  For i=1 To 3
    Beladen 1, i*2-1
    Beladen 3, i*2
  Next i
  PrintStatus "Starten des
Fahrbetriebs : E2"
......
End Sub

Sub Beladen(Position%, Runde%)
Dim n%
  SetMotor mRadMotor,ftiLinks
  For n = 1 To Position
    WaitForHigh eRadPos
  Next n
  Pause 1500 ' --- 3200 bei
Original
              ' 1500 bei Simulation
  SetMotor mRadMotor,ftiAus
  PrintStatus "Gondel : " & Runde
  WaitForLow eQuittung
End Sub
```

In Praxis werden bei einem Riesenrad die Gondeln selten in ihrer direkten Reihenfolge "beladen". Das Beladen übernimmt hier das gleichnamige Unterprogramm. Das Unterprogramm selber entspricht weitgehend dem bisherigen Belade-Code, lediglich die Ansteuerung der Position findet jetzt in einer Schleife statt.

Anstelle des Belade-Codes in Sub Main findet man dort eine For .. Next Schleife, die Beladen aufruft. Der erste Parameter gibt die relative Nummer der nächsten Beladeposition an (Anzahl Gondeln, die zu überspringen sind, + 1). Der zweite Parameter gibt die wievielte Gondel zu beladen ist, sie wird aus der äußeren For .. Next Schleife abgeleitet.

Beladen wird in der Reihenfolge 1 - 4 - 5 - 2 - 3 - 6. Also die gegenüberliegende Gondel oder die nächste.
Denkbar sind natürlich auch noch andere Beladepläne.

Schritt 6 ff : Weiterer Aus- und Umbau

- Anbringen eines zusätzlichen Tasters, der die Gondel Nr. 1 identifiziert.
- Anbringen eines Impulsrades (siehe Industry Robots) anstelle des Taster eRadPos und der 6 Nocken.

Lösung unter Einsatz von vbaFish30. Die Source der vorgestellten Lösung ist in RiesenTut.ZIP enthalten. Zusätzlich wird vbaFish30 benötigt.

Stand : 17.05.2004

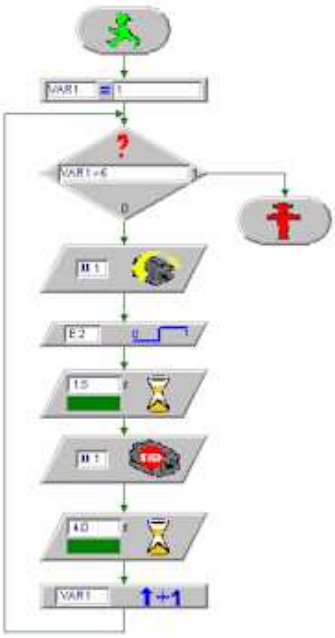
Riesenrad LLWin

ftComputing : Programme für die fischertechnik-Interfaces und -konstruktionskästen

Programmgesteuertes Riesenrad : LLWin Version

Schrittweise Entwicklung eines Betriebsprogrammes für das Riesenrad aus Fun Park unter Nutzung von LLWin 3.0. Basiert auf der Modellbeschreibung von Riesenrad.

Schritt 1 : Aus- und Einsteigen der Fahrgäste



Alle Gondeln werden nacheinander zu Einsteigeposition gefahren. Zum Aus- und Einsteigen wird gehalten.

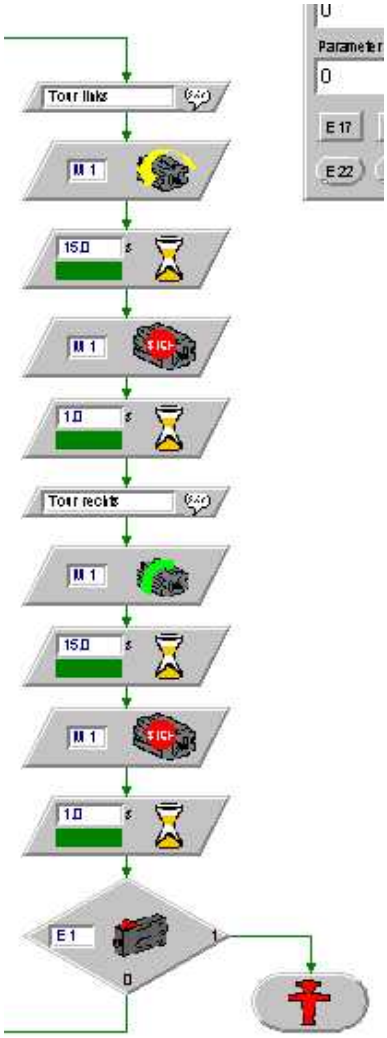
Kernpunkt der Positionserkennung ist der Baustein Flanke (Warten auf einen 0/1-Durchgang). Ein schlichtes Warten auf E2 = 1 reicht nicht, da es noch auf 1 stehen kann.
Da mit E2 = 1 die exakte Einsteigeposition nicht gewährleistet ist (Lage des Taster, Masse des Riesenrades), wird nach E2 = 1 noch ein paar (1,5) Sekunden weitergefahren und dann erst abgeschaltet. Danach folgt die Pause für das Aus- und Einsteigen. Das ganze immer schön in einer 6er-Schleife.

ACHTUNG : Die Länge der Pause (hier `Warte 1.5`) hängt beim Original Riesenrad von der Beladung und der Befestigung der Hauptachse ab. Sie lag bei mir zwischen 1,5 (12 Männeken) und 3,2 (leer).

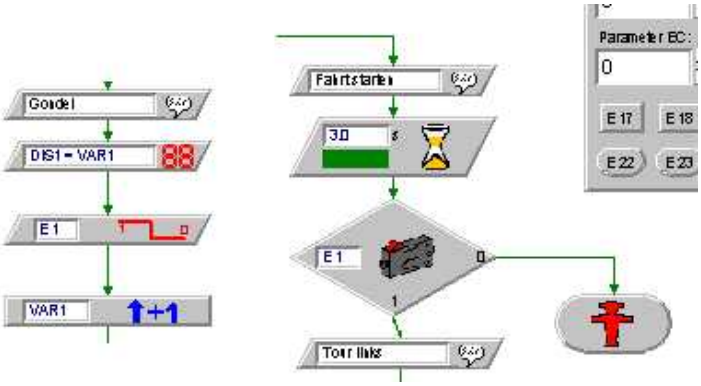
Schritt 2 : Die Fun-Runden :

Das ganze Programm wird in eine Endlosschleife gepackt, so erhält man ein schönes Demo-Programm, das durch E1 = 0 und über die Entwicklungsumgebung abgebrochen werden kann. Zusätzlich wird mit dem Baustein Meldung im Baustein Terminal die aktuelle Funktion angezeigt.

Nach dem Ein-/Aussteigen (wie bisher) wird 15 Sek. links und dann 15 Sek rechts gedreht. Da es bei sofortiger Richtungsumschaltung richtig knirschen kann, wird dazwischen 1 Sek. Pause eingelegt.



Schritt 3 : Echt-Betrieb

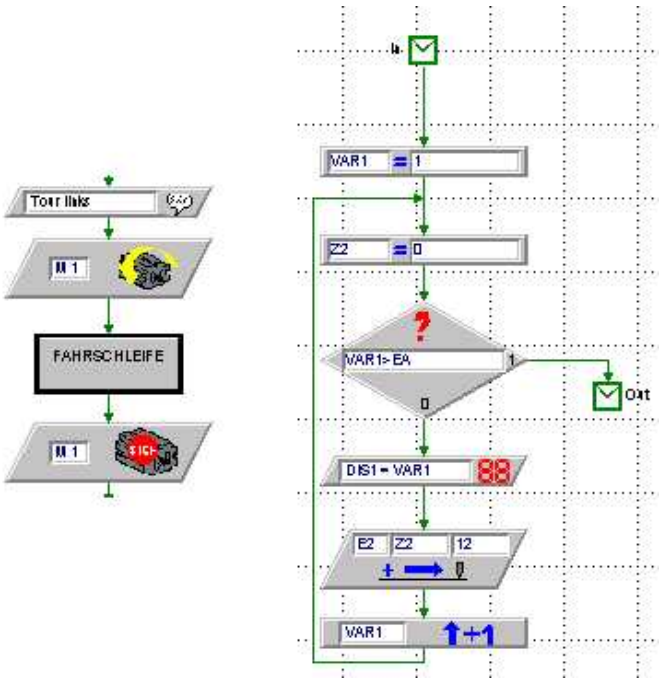


Bei einem Echt-Betrieb sind die Zeiten für Aus- und Einsteigen nicht vorhersehbar, der Baustein Warte 4 wird deswegen durch eine Flanke E1 ersetzt. Das Programm wartet bis E1 gedrückt und wieder freigegeben wird.

Nach dem Aus-/Einsteigen muß der Betrieb durch erneutes Drücken der E1-Taste innerhalb von 3 Sek. freigegeben werden, sonst wird das Programm beendet : Feierabend.

Man kann sich zum Betrieb natürlich noch mehr einfallen lassen.

Schritt 4 : Zählen statt Warten, Rundenanzeige

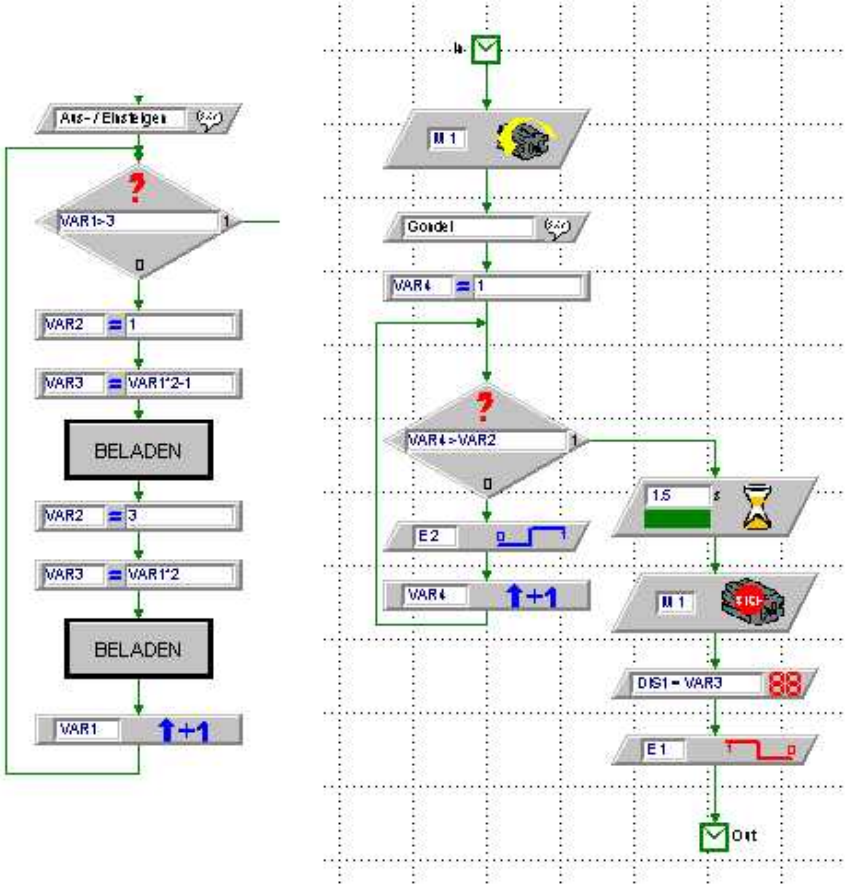


Anstelle des Bausteins Warte 15 wurde in $MAIN das Unterprogramm Fahrschleife verwendet.

Hier wird anstelle des Warte 15 ein Baustein Position 12 in einer Schleife verwendet. Die 12 steht für eine Runde (zur Erinnerung Position zählt 0/1- und 1/0-Durchgänge getrennt). Der Baustein Position liegt in einer Schleife,Beginn mit VAR1 = 1, Ende mit VAR1 > EA des Terminal Bausteins. Der von Position verwendete Zähler Z2 muß dazu vor jeder Nutzung auf Z2 = 0 gesetzt werden.

Grund für die Schleifenkonstruktion ist der Wunsch, die aktuell gefahrene Runde anzeigen zu können. Dabei zählt VAR1 die vollen Runden, die mit Baustein Display in Display1 des Terminal-Bausteins angezeigt werden.

Schritt 5 : Überkreuz Beladen



In Praxis werden bei einem Riesenrad die Gondeln selten in ihrer direkten Reihenfolge "beladen". Das Beladen übernimmt hier das gleichnamige Unterprogramm. Das Unterprogramm selber entspricht weitgehend dem bisherigen Belade-Code, lediglich die Ansteuerung der Position findet jetzt in einer Schleife statt.

Anstelle des Belade-Codes in $MAIN findet man dort eine Schleife, die Beladen aufruft. Beladen entnimmt VAR2 die Anzahl der zu überspringenden

Gondel (VAR2-1). VAR3 enthält die laufende Gondelnummer, die in Display1 des Terminal-Bausteins angezeigt wird.

Beladen wird in der Reihenfolge 1 - 4 - 5 - 2 - 3 - 6. Also die gegenüberliegende Gondel oder die nächste. Denkbar sind natürlich auch noch andere Beladepläne.

In einem separaten Ablaufplan kann man dann noch über den Baustein Display in Display2 des Terminals den Zähler Z2 anzeigen, um einen genaueren Überblick zu bekommen.

Schritt 6 ff : Weiterer Aus- und Umbau

- Anbringen eines zusätzlichen Tasters, der die Gondel Nr. 1 identifiziert.
- Anbringen eines Impulsrades (siehe Industry Robots) anstelle des Taster eRadPos und der 6 Nocken.

Lösung unter Einsatz von LLWin 3.0. Die Source der vorgestellten Lösung ist in RiesenTut.ZIP enthalten. Zusätzlich wird LLWin 3.0 benötigt.

Stand : 17.05.2004