Stanze 51663

Stanzmaschine mit Transportband No. 51 663

Freier Nachbau der Stanzmaschine aus dem Katalog 2001/2002 aus der Rubrik Trainingsmodelle (dazu mehr) versehen mit Betriebsprogrammen in acht Programmiersprachen : LLWin 3.0, Visual Basic 6, Delphi4, C++Builder4, C#, VB.NET, JavaSwing und VBScript.

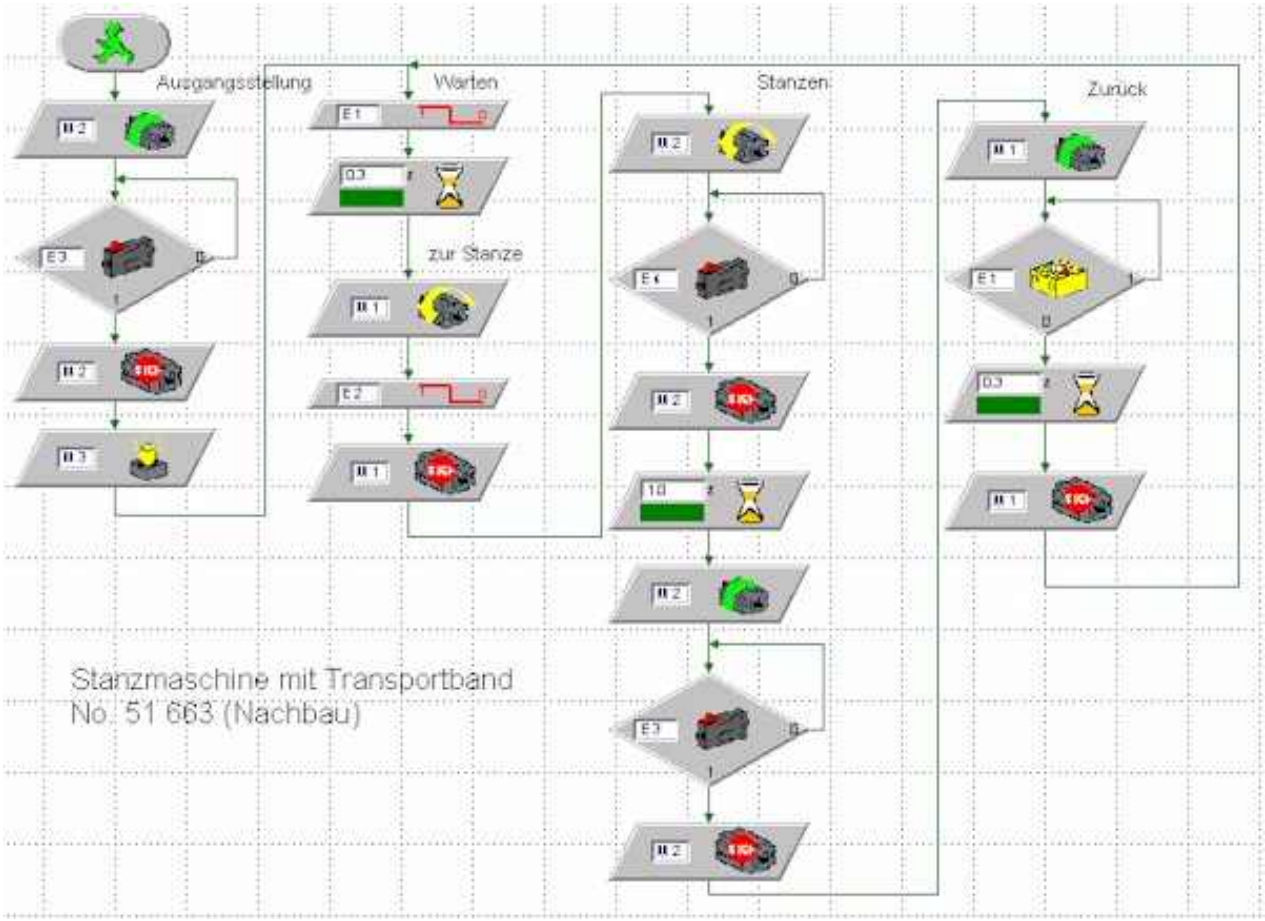
Modelldaten :

```
Private Const mBand = 1              ' Bandmotor an M1
Private Const cbVor = ftiLinks       ' zur Stanze linksdrehend
Private Const cbRuck = ftiRechts     ' und zurück

Private Const mStanze = 2            ' Stanzenmotor an M2
Private Const eOben = 3              ' Taster oben an E3
Private Const eUnten = 4             ' Taster unter an E4
Private Const csAb = ftiRechts       ' abwärts : linksrum
Private Const csAuf = ftiLinks       ' auf : rechtsrum

Private Const ePhotoV = 1            ' Photowiderstand vorn an E1
Private Const ePhotoS = 2            '     bei Stanze       an E2
Private Const mLampen = 3            ' Lampen für Photow.   an M3
```

Neu am Modell ist die Kiste, in die die gestanzten Teile abgelegt werden. Man kann dafür natürlich auch einen Industry Robot einsetzen.... Für das Transportband wurden 20er Zahnräder verwendet (das Band schleift etwas), Original 15er. Als Werkstücke wurden 2 aufeinandergesteckte Radscheiben (23 mm) genommen.

LLWin 3.0

Der **Ablauf** wird anhand des **LLWin-Programms** beschrieben :

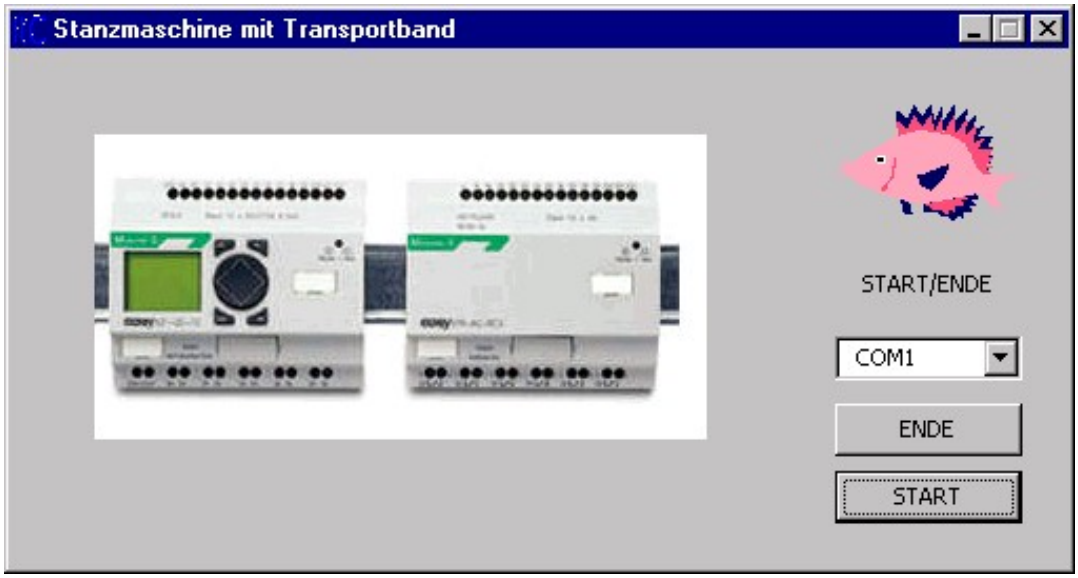


- Bei Start des Programms wird das Modell in Ausgangslage gebracht : Stanze fährt nach oben und die Lampen für die Lichtschranken werden eingeschaltet.
- Warten auf einen true/false-Durchgang an der vorderen Lichtschranke (true heißt kein Teil in der Lichtschranke, wenn da noch ein altes lag muß man es wegnehmen).
- Damit das neue Teil nicht gleich wegreißt 0,3 Sek. warten.
- Dann gehts zur Stanze bis wieder ein true/false-Durchgang (Lichtschranke offen/geschlossen) das Band stoppt.
- Die Stanze fährt nach unten bis Endtaster.
- GedenkSekunde
- Es geht wieder aufwärts.
- und zurück bis die vordere Lichtschranke schließt.
- und darüber hinaus in die Ablage P. Der Bandmotor hat 0,3 Sekunden Nachlauf

Nach Oben

Visual Basic 6

Das **Visual Basic Programm** (erstellt mit dem Template ftComputing50, Einsatz von FishFa50.OCX/umfish20.DLL) sieht dann so aus :



Außer dem Bild (das kommt später) sieht man wenig und das Programm (VB6-Version) dazu funktioniert denn auch wie oben beschrieben :

```
Private Sub Action()

' --- Routine für den Modellbetrieb, kann weitere Subs aufrufen ---
-

ft.ClearMotors
ft.SetMotor mStanze, csAuf
ft.WaitForInput eOben
ft.SetMotor mStanze, ftiAus
ft.SetMotor mLampen, ftiEin

Do

lblStatus = "Wartet"
ft.WaitForLow ePhotoV
ft.WaitForTime 500

lblStatus = "Zur Stanze"
ft.SetMotor mBand, cbVor
ft.WaitForLow ePhotoS
ft.SetMotor mBand, ftiAus

lblStatus = "Stanzen"
ft.SetMotor mStanze, csAb
ft.WaitForInput eUnten
ft.SetMotor mStanze, ftiAus
ft.WaitForTime 1000
ft.SetMotor mStanze, csAuf
ft.WaitForInput eOben
ft.SetMotor mStanze, ftiAus

lblStatus = "Zurück"
ft.SetMotor mBand, cbRuck
```

```
ft.WaitForInput ePhotoV, False
ft.WaitForTime 300
ft.SetMotor mBand, ftiAus

Loop Until ft.Finish(0)

End Sub
```

Neu ist eigentlich nur die zusätzliche Anzeige des Betriebszustandes.

Nach Oben

Jetzt noch eine Übersicht weiterer Sprachversionen

Weitere Sprach-Versionen von Stanze

Die eigentlichen Betriebsprogramme für das Stanze-Modell sind alle sehr ähnlich, da sie alle die FishFace Methoden nutzen. Es werden deswegen auch nur beispielhaft zwei Statements gezeigt um den Variantenreichtum heutiger Programmiersprachen zu characterisieren. Zusätzlich ein paar allgemeine Anmerkungen zur Implementierung. Die Variablenschreibweise entspricht Empfehlungen zur jeweiligen Sprache (soweit ich sie kapiert und akzeptiert habe).

Die Programme haben eine Sperre gegen unkontrolliertes Beenden. Sie ist durch Enabled = False der entsprechenden Buttons, Änderung der Beschriftung und Sperre des (x) rechts oben realisiert.

Alle Programme nutzen umFish30.DLL bzw. umFish20.DLL. Beide DLLs setzen den MultiMediaTimer ein, der in einem Extra-Thread eine CallBack-Routine zur Steuerung des Interfaces aufruft. so ist eine solide Zeitbasis zum Pollen der E-Eingänge auf dem nicht Realtime-fähigen Windowssystem erreichbar.

Delphi4

```
lblStatus.Caption := 'Wartet';

ft.SetMotor(mStanze, csAuf);
```

Genutzt wird die FishFa50.DCU auf Basis von umFish20.DLL

Nach Oben

C++Builder4

```
lblStatus->Caption = "Wartet";

ft.SetMotor(mStanze, csAuf);
```

Genutzt wird die FishFa30.H/CPP Unit auf der Basis von umFish30.DLL.
Es wird #include <vcl.h> eingesetzt. Die Oberfläche ist also nahe an Delphi.

Nach Oben

C#

```
lblStatus.Text = "Wartet";

ft.SetMotor(mStanze, csAuf);
```

Genutzt wird die FishFa30.CS Assembly als Source auf Basis von umFish30.DLL (cs-Variante).

Nach Oben

VB.NET

```
lblStatus.Text = "Wartet"

ft.SetMotor(mStanze, csAuf)
```

Genutzt wird die in C# geschriebene FishFa30.DLL in kompilierter Form (Basis umFish30.DLL, cs-Variante).

Nach Oben

JavaSwing

```
lblStatus.setText("Wartet");

ft.setMotor(mStanze, csAuf);
```

Genutzt werden die Klassen JavaFish und FishThread, die den Zugriff auf die Wrapper.DLL javaFish.DLL (VC++ mit speziellen Java-Konstrukten (JNI-Interface) kapselt. javaFish.DLL nutzt dann umFish20.DLL. Da Java ein Konstrukt wie Application.ProcessMessage (Abgabe der Prozesszeit an den Windowsloop) nicht kennt, wurde hier das eigentliche Betriebsprogramm (class Stempel extends FishThread) in einen eigenen Thread gelegt um die Reaktionsfähigkeit der Anwenderoberfläche sicherzustellen.

Nach Oben

VBScript

```
---

ft.SetMotor mStanze, csAuf
```

Genutzt wird FishFa50D.DLL auf Basis von umFish20.DLL

Nach Oben

Download

Im Download-Päckchen Stanze sind die Source zusammengefaßt. Zusätzlich ist noch eine DLL und weitere Zugriffssoftware erforderlich, die Angaben dazu sind im ReadMe des Stanze-Päckchens aufgelistet.

Nach Oben

EasyTrainer von Moeller

Die Firma Moeller (www.moeller.net) hat für ihre speicherprogrammierbaren Relais ein Mini-Trainer Set zusammengestellt, das auch eine Anschlußplatine für fischertechnik enthält (24V), Anschüsse wie beim Interface. Preis liegt liegt das Set im privaten Rahmen (ca. 250 Euro). Die 24V-Teile (Motoren Lampen) kann man problemlos bei www.knobloch-gmbh.de bestellen. Sie passen gut zu den normalen fischertechnik-Teilen.

Der Mini-Trainer bietet einen einfachen Einstieg in die Speicherprogrammierbare Steuerrung (SPS) auf Basis von Stromlaufplänen. Programmiert werden kann direkt am Relais oder mit einem komfortablen Editor am PC.

Ist etwas für Berufsschulen. (Punkt) Sowie Elektriker und E-Ing.s, die es auch zu Hause nicht lassen können.

Man sollte sich mal im (Heizungs)Keller umsehen, vielleicht gibt es da schon so ein Relais zum Thermo-Management oder mal hinter (Schrank)türen sehen da könnte es beim Security-Management (Tag/Nachtlichtschaltung, Bewegungssensor oder Rasensprenger). Man weiß es dann richtig zu schätzen.

Nach Oben