E-Tec Module

ftComputing : Programme für die fischertechnik-Interfaces und -konstruktionskästen

ftComputing.de

Home
Back
Alarmanlage
Garagentor

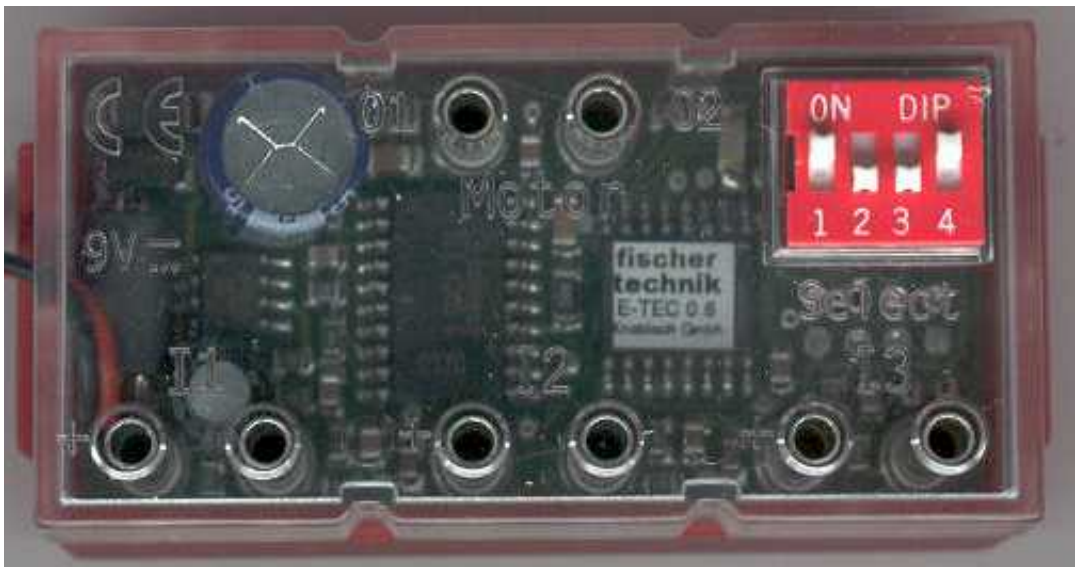
Sitemap
Index
Links
Impressum

Mail

Der Kasten Profi E-Tec 91 083

Der Kasten Profi E-Tec (aus 2003) ist Nachfolger des Kastens Profi Sensoric (aus 1991). Als zentrale Komponente enthält er jetzt einen Microprozessor mit festen Programmen (E-Tec Module) der den FLIP-FLOP des Profi Sensoric ersetzt. Das Modell angebot ist irgendwo zwischen Profi Sensoric und Computing Starter Kit angesiedelt. Vorerst ist der E-Tec Module nur mit dem kompletten Kasten lieferbar.

E-Tec Module



Gehäuse : von 9V Blockbatterie (32 263) mit Klarsicht-Deckel

3 Digitale Eingänge (I1, I2, I3)

1 Motor Ausgang = 2 Lampen Ausgänge (O1, O2)

1 grüne Leuchtdiode zur Anzeige des Betriebszustandes

8 fest gespeicherte Programme über DIP einstellbar

Label : fischertechnik E-Tec 0.6 Knobloch GmbH

Programme

1. Grundprogramm

    I1 Motor links
    I2 Motor rechts
    I3 Motor aus
    über DIP Schalter können die einzelnen Eingänge invertiert (NOT) werden.

2. Spezialprogramme

    2.1 Händetrockner
    2.2 Alarmanlage
    2.3 Garagentor / Parkhausschranke
    2.4 Wechselblinker

3. Digitalfunktionen

    Eingebaut sind eine Reihe von Digitalfunktionen (AND, OR, FlipFlop, MonoFlop), die im Handbuch des Kasten nicht beschrieben sind. Eine Beschreibung ist unter www.fischertechnik.de > service vorgesehen.

Programmbeispiele

Auf den folgenden Seiten werden Beispiele für die Umsetzung der im E-Tec Module gespeicherten Programme in verschiedene Programmiersprachen gezeigt. Für den Nachbau der Modelle ist Phantasie gefragt, wenn man keinen Profi E-Tec Kasten besitzt. Die benötigten Teile werden aber meist vorhanden sein :

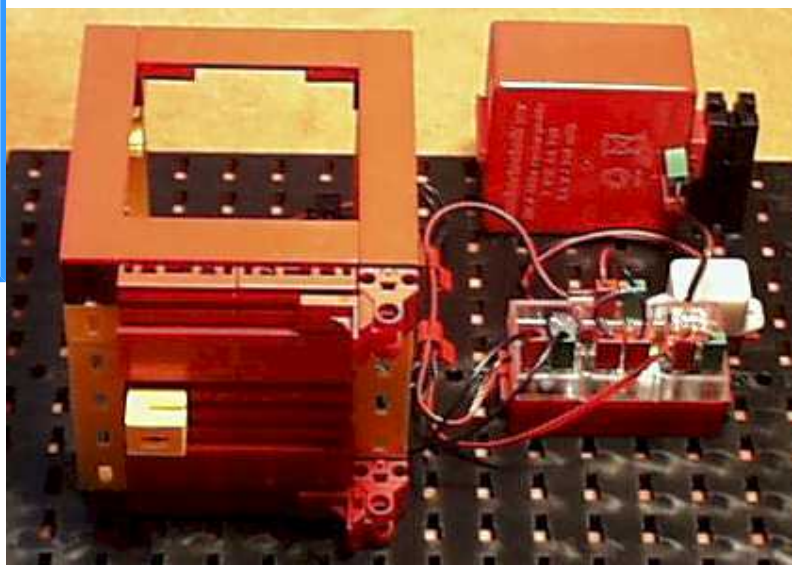
- Alarmanlage (VBA (vbaFish), Python und Siemens LOGO!)
- Garagentor (LLWin, Python, Delphi, VBA (vbaFish))

Alle Sources in eTec.ZIP

Stand : 30.10.2003

Alarmanlage

Programmbeispiel : Alarmanlage



Modell Alarmanlage aus Kasten Profi E-Tec

Funktion :
Sobald I1 unterbrochen wird, ertönt der Summer mit Unterbrechungen. Wird I3 geschlossen, schaltet der Summer ab, aber nur, wenn vorher I1 wieder geschlossen wurde. Durch Überbrücken von I2 kann die Dauer des einzelnen Summtons verändert werden (Text Handbuch). An I1 ist im Original ein Reed-Kontakt angeschlossen (Tür zu : ein, Schaltung durch einen Magneten).

```
Sub Main
Dim PZ%
  Do
    If Not GetInput(ftiE1) Then
      PZ = IIf(GetInput(ftiE2),256,512)
      Do
        SetMotor ftiM1, ftiEin
        Pause PZ
        SetMotor ftiM1,ftiAus
        Pause PZ
      Loop Until ( GetInput(ftiE1) And GetInput(ftiE3) Or
Finish())
    End If
  Loop Until Finish()
End Sub
```

Programmiersprache **VBA**, erstellt mit vbaFish. Gegenüber E-Tec etwas vereinfacht : Die Alarmanlage kann nur am Ende einer Hup-Phase zurückgesetzt werden (Also max. 1 Sekunde drücken).

```
ft = FishFace()

ft.OpenInterface("LPT")

while not ft.Finish():
  if not ft.GetInput(eReed):
    if ft.GetInput(eTime): PZ = 256
    else: PZ = 512
    while not(ft.GetInput(eReed) and ft.GetInput(eStop) or
ft.Finish()):
      ft.SetMotor(mHupe, ftiEin)
      ft.Pause(PZ)
      ft.SetMotor(mHupe, ftiAus)
      ft.Pause(PZ)
```

Programmiersprache **Python** (Programmausschnitt). Die Lösung entspricht der obigen VBA-Version. Da Python keinen "Until" Loop kennt, wurde hier die Abbruchabfrage für die Hup-Schleife "umgedreht" (in Klammern gesetzt und not davor).

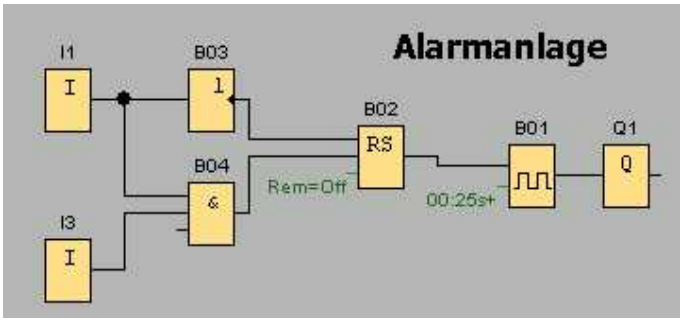
```
class FishFaceP(FishFace):
  def Pause(self, mSek, TermInput):
    sZeit = windll.kernel32.GetTickCount()
    while (windll.kernel32.GetTickCount() - sZeit) < mSek:
      if (windll.user32.GetAsyncKeyState(self.ESCAPE) != 0
         or self.GetInput(TermInput)): return
      windll.kernel32.Sleep(self.PollInterval)

ft = FishFaceP()

ft.OpenInterface("LPT") # --- ACHTUNG : PortName

while not ft.Finish():
  if not ft.GetInput(eReed):
    if ft.GetInput(eTime): PZ = 256
    else: PZ = 512
    while not(ft.GetInput(eReed) and ft.GetInput(eStop) or
ft.Finish()):
      ft.SetMotor(mHupe, ftiEin)
      ft.Pause(PZ, eStop)
      ft.SetMotor(mHupe, ftiAus)
      ft.Pause(PZ, eStop)
```

Programmiersprache **Python** (Programmausschnitt) : Die Alarmanlage kann in jedem Augenblick zurückgesetzt werden. Dazu wurde die Methode Pause der Klasse FishFace in einer hiervon abgeleiteten Klasse FishFaceP durch eine neue Methode Pause überschrieben. Sie entspricht voll der bisherigen Methode Pause, hinzugekommen ist lediglich ein Abbruchparameter TermInput, der in der bereits vorhandenen Abbruchabfrage zusätzlich abgefragt wird.



Und dann noch eine Lösung mit dem Intelligenten Steuerrelais **Siemens LOGO!** Hier wurde auf die Änderung der Hupfrequenz durch Überbrücken von I2 verzichtet, da der Baustein B01 bequem von außen parametrisiert werden kann. B02 : Selbsthalterelais, B01 : Symmetrischer Taktgeber.

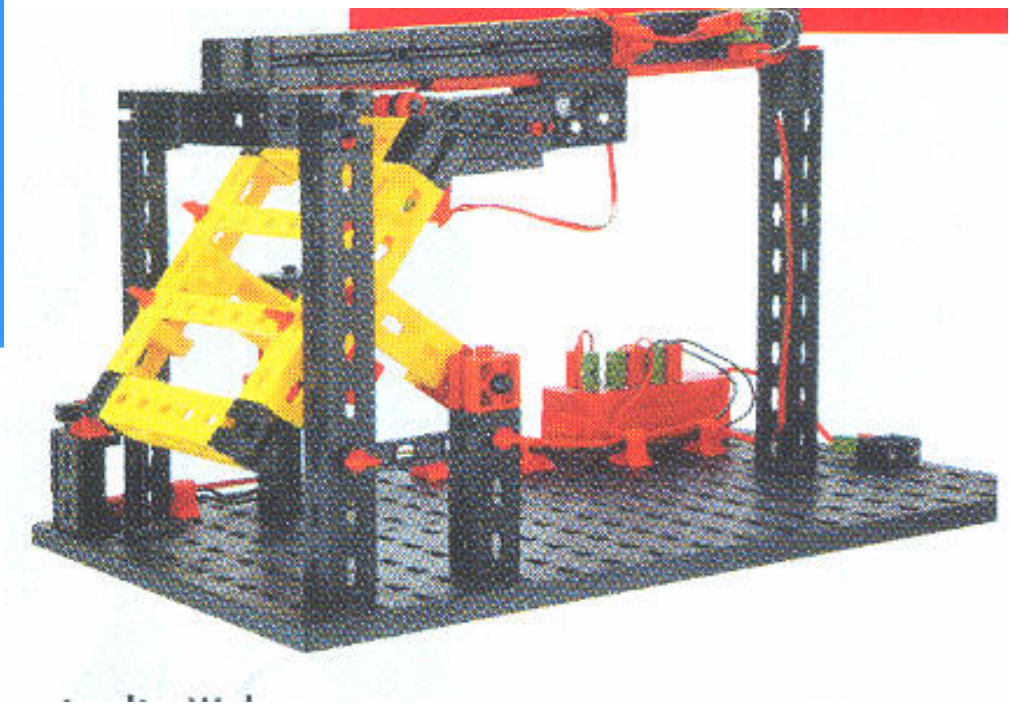
Alle Sources in eTec.ZIP

Stand : 30.10.2003

Garagentor

ftComputing : Programme für die fischertechnik-Interfaces und -konstruktionskästen

Programmbeispiel : Garagentor



Modell Garagentor aus Kasten Profi E-Tec

Funktion :
Tor wird zuerst geschlossen (Motor rechts). Tor wird geöffnet (Motor links) durch Schließen von I1. Tor wird geschlossen (Motor rechts) durch Schließen von I2. Schranke kann nur geschlosssen werden, wenn sie vorher geöffnet wurde und umgekehrt. (nach Text Handbuch). Begrenzung des Torweges durch parallel geschaltete Taster an I3. An I1 ist ein Reed-Kontakt angeschlossen, Betätigung durch einen Magneten (ein schlichter Taster tuts auch). Programmeinstellung am DIP : 1 on, 2 on, 3 off, 4 on. Anmerkung : der Serien E-Tec Module hat ein transparentes Oberteil.

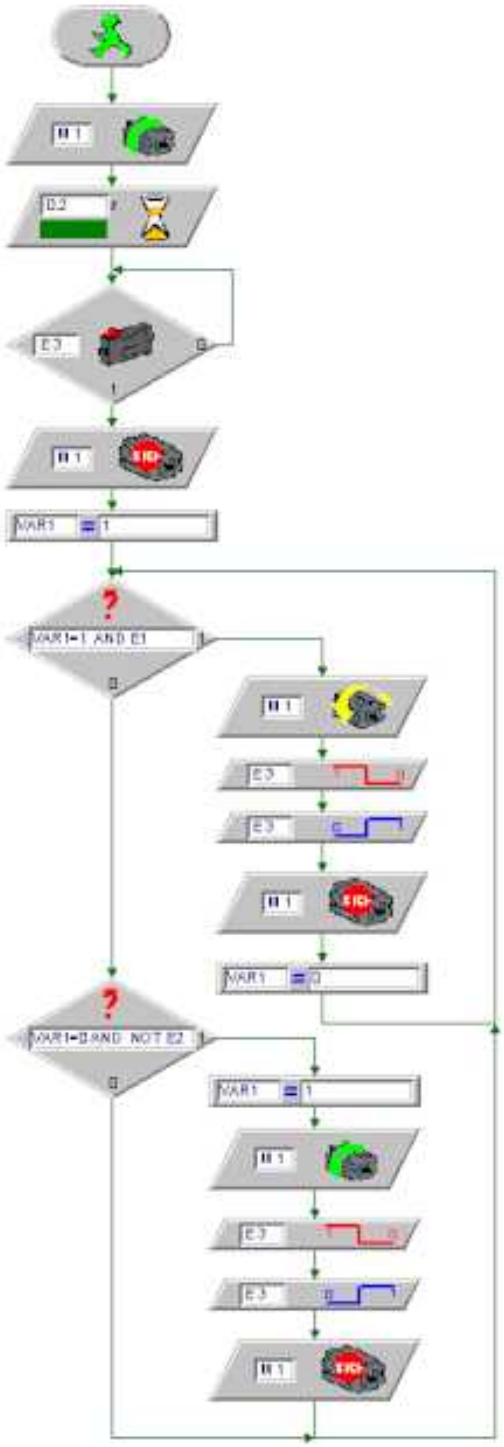
Version 1

Version 2a

```
Sub Main
Dim Geschlossen As
Boolean

  SetMotor ftiM1,
ftiRechts
  WaitForTime 300
  WaitForInput
ftiE3
  SetMotor ftiM1,
ftiAus
  Geschlossen =
True

  Do
    If Geschlossen
And GetInput(ftiE1)
Then
      SetMotor
ftiM1, ftiLinks
      WaitForLow
ftiE3
      WaitForHigh
ftiE3
      SetMotor
ftiM1, ftiAus
      Geschlossen =
False
    ElseIf (Not
Geschlossen) And _
           (Not
GetInput(ftiE2))
Then
      SetMotor
ftiM1, ftiRechts
      WaitForLow
ftiE3
      WaitForHigh
ftiE3
      SetMotor
ftiM1, ftiAus
      Geschlossen =
True
    End If
  Loop Until
Finish()
End Sub
```



LLWin 3.0

Programmiersprache : **VBA**, erstellt mit vbaFish.

Version 2b

```
Sub Main
  If Not
GetInput(ftiE4)
Then
    SetMotor ftiM1,
ftiRechts

WaitForInput(ftiE4)
    SetMotor ftiM1,
ftiAus
  End If
  Do
    If
GetInput(ftiE4) And
GetInput(ftiE1)
Then
      SetMotor
ftiM1, ftiLinks
      WaitForInput
ftiE3
      SetMotor
ftiM1, ftiAus
    ElseIf
GetInput(ftiE3) And
GetInput(ftiE2)
Then
      SetMotor
ftiM1, ftiRechts

WaitForInput(ftiE4)
      SetMotor
ftiM1, ftiAus
    End If
  Loop Until
Finish()
End Sub
```

Programmiersprache : **VBA**, erstellt mit vbaFish.

Anmerkungen :

- Die Versionen 1 (LLWin) und 2a entsprechen der E-Tec Lösung. Version 2a nutzt einen zusätzlichen E-Eingang des Interfaces.
- Im ersten Block wird bei Anlauf des Programms das Tor geschlossen.
  - Bei der E-Tec Lösung mit parallel geschalteten Endtastern kann nicht erkannt werden, ob das Tor offen, geschlossen oder irgendwo dazwischen steht. Deswegen wird verdachtsweise erstmal 0,3 Sek. nach rechts gefahren, um vom möglichen Endtaster E3 Tor offen wegzukommen, dann erst wird der Endtaster E3 Tor zu angefahren. Das ergibt bei geschlossenem Tor ein wenig Ruckeln.
  - Bei einem zusätzlichen E-Eingang für Tor zu (E4) kann direkt der Endtaster E4 Tor zu angefahren werden. Die If-Klammer verhindert ein Ruckeln bei Tor zu.
- In der Do ... Loop Schleife wird endlos auf Anforderungen zum Öffnen bzw. Schließen gewartet. Das wird nur ausgeführt, wenn das Tor in der zulässigen Stellung steht.
- Da bei der ersten Variante nicht erkannt werden kann, ob das Tor offen oder zu ist, wird eine entsprechende Variable eingeführt. Hier auch wieder ein "Freifahren" vom jeweiligen Endtaster und dann ein Anfahren des Ziel-Endtasters. Da hier sicher ist, daß immer ein Endtaster gedrückt ist, kann auf das immer ein wenig problematische Warten verzichtet werden. An seine Stelle tritt ein Freifahren über WaitForLow und ein Zielanfahren mit WaitForHigh.
- Bei Version 2a tritt E4 anstelle der Variablen. Da immer ein anderer E-Eingang angefahren wird, kann auf das Freifahren verzichtet werden.
- Der Abbruch des gesamten Programms erfolgt über die ESC-Taste (Entf) bzw. den HALT-Button auf der Form von vbaFish.

Alle Sources in eTec.ZIP

Stand : 30.10.2003