Software LLWin 3.0

fischertechnik®

(D) INHALT:

(GB+USA) CONTENTS:

INHALTSVERZEICHNIS

1. Einleitung – fischertechnik Modelle steuern mit LLWin 3.0

Sicherlich hast du dich auch schon manchmal gefragt, wie das funktioniert, wenn Roboter, wie von Geisterhand bewegt, bestimmte Aufgaben ausführen. Aber nicht nur bei echten Robotern, in vielen anderen Bereichen begegnen wir der Steuerungs- und Automatisierungstechnik; auch bei fischertechnik. Bereits im übernächsten Kapitel werden wir gemeinsam ein kleines Steuerungsprogramm für ein automatisches Garagentor entwerfen und dabei lernen, wie man mit Hilfe der Software Lucky Logic für Windows, kurz LLWin genannt, solche Steuerungsaufgaben lösen und testen kann. Die Bedienung von LLWin ist dabei sehr einfach. Auf der grafischen Bedienoberfläche lassen sich die Steuerungsprogramme, genauer gesagt die Ablaufpläne, wie wir noch lernen werden, fast ausschließlich mit Hilfe der Maus erstellen.

Damit du deine fischertechnik Modelle über den PC ansteuern kannst, benötigst du neben der Steuerungssoftware LLWin außerdem noch ein Interface als Bindeglied zwischen Rechner und Modell. Es wandelt die Befehle der Software so um, dass beispielsweise Motoren angesteuert und Signale von Sensoren verarbeitet werden können. Von fischertechnik gibt es das Intelligent Interface Art.-Nr. 30402 und das ältere parallele Interface Art.-Nr. 30520. Beide Interfaces kannst du zusammen mit LLWin 3.0 verwenden.

Noch ein paar Worte zum Aufbau dieses Handbuches. Es unterteilt sich in zwei Teile. Der erste Teil von Kapitel 1 bis Kapitel 6 beschreibt die grundsätzliche Vorgehensweise beim Programmieren mit LLWin 3.0. Dabei bekommst du viele Informationen und Hintergründe zum Programmieren allgemein und zur Bedienungsweise der Software LLWin speziell.
Der zweite Teil umfasst die Kapitel 7 bis 9. Sie sind eher ein Nachschlagewerk. Wenn du also nach dem ersten Teil mit der Bedienung von LLWin vertraut bist und ganz spezielle Informationen suchst, findest du dort die ausführliche Erklärung zu den einzelnen Bausteinen und Menübefehlen.
In den letzten Kapiteln findest du dann noch Hinweise, Tipps und Tricks zur Programmierung sowie Antworten auf spezielle Fragen (z. B. was geschieht mit alten LLWin Projekten?).
Nun aber los! Sicherlich bist du schon sehr gespannt, welche Möglichkeiten du mit der Software LLWin zum Programmieren deiner fischertechnik-Modelle hast. Viel Spaß!

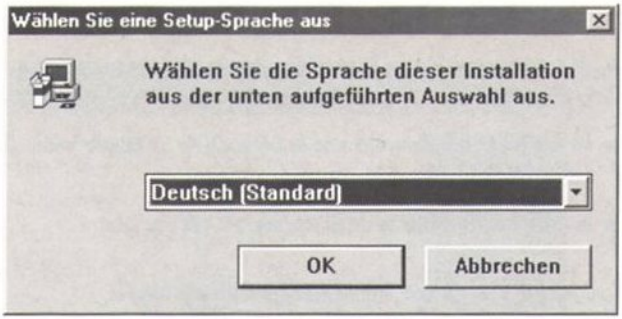
1.1 Installation von LLWin

Voraussetzungen zur Installation von LLWin 3.0 sind:

- ein IBM-kompatibler PC mit Pentium-Prozessor mit mindestens 100 MHz Taktfrequenz, 32 MB RAM und ca. 20 MB freier Speicherkapazität auf der Festplatte
- Microsoft, Windows, Version Windows 95, 98, 2000 oder NT
- Eine freie RS232-Schnittstelle COM1 bis COM4 zum Anschluss des Intelligent Interface - Art.-Nr. 30402 (Druckerschnittstelle LPT1 oder LPT2 für das ältere parallele Interface – Art.-Nr. 30520)

Zunächst musst du natürlich den Computer starten und warten, bis das Betriebssystem (Windows) vollständig geladen ist. Lege anschließend die Installations-CD in das CD-ROM Laufwerk ein. Das Installationsprogramm auf der CD wird dann automatisch gestartet. In verschiedenen Dialogfenstern musst du noch Folgendes angeben:

- Zunächst kannst du die gewünschte Sprache auswählen:

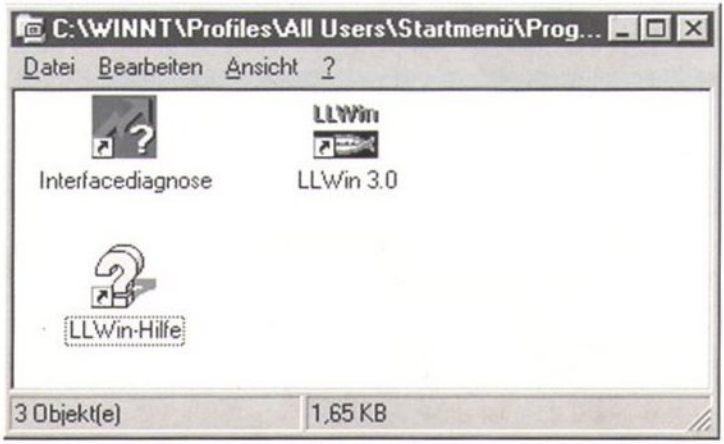


- Das nächste Hinweisfenster liest du aufmerksam durch. Wenn du noch andere Programme gestartet hast, solltest du diese schließen. Weiter geht's mit OK.
- In der dritten Dialogbox musst du den gewünschten Ordner bzw. Verzeichnispfad auswählen, in welchem das Programm LLWin installiert werden soll. Normalerweise ist dies der Pfad `C:\Programme\LLWin`. Du kannst aber auch ein anders Verzeichnis eingeben!
- Im letzten Dialogfenster kannst du noch angeben, wo die Programmsymbole zum Starten der LLWin Programme im Startmenü eingefügt werden sollen. Am Besten du bestätigst hier einfach mit OK. Dann wird im Startmenü eine eigene Programmgruppe für LLWin erzeugt.

Nun werden die benötigten Programmdateien kopiert. Nach Abschluss dieses Vorgangs meldet ein Informationsfenster die erfolgreiche Installation. Mit einem Klick auf OK wird die Installation abgeschlossen.

Wichtiger Hinweis für die Installation unter Windows NT:
Für das parallele Interface Art.-Nr. 30520 wird unter Windows NT ein spezieller Treiber zur Ansteuerung der parallelen Schnittstelle installiert (sog. Parallel-Port-Treiber). Dieser Treiber kann nur von einem Anwender installiert werden, der an dem PC Administratorrechte besitzt. Sollte das Installationsprogramm melden, dass du den Parallel-Port-Treiber nicht installieren darfst, musst du entweder deinen Systemadministrator bitten, den Treiber zu installieren oder LLWin ohne diesen Treiber installieren. Dann kannst du LLWin allerdings nur mit dem Intelligent Interface Art.-Nr. 30402 betreiben.

Das folgende Fenster zeigt die Programmgruppe von LLWin mit den verschiedenen Programmsymbolen:

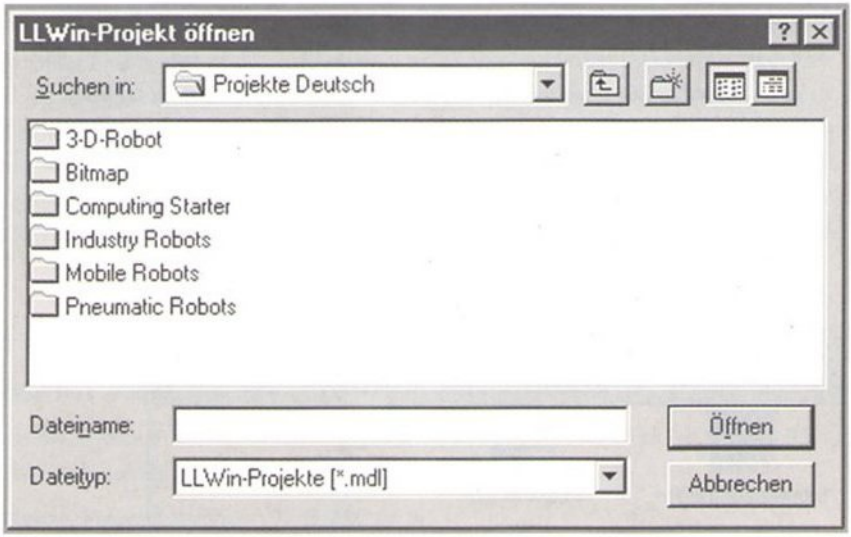


- Mit dem Programm "Interface-Diagnose" kann das angeschlossene Interface getestet werden. Dies werden wir im Kapitel 2 auch gleich tun.
- Mit dem zweiten Programmsymbol kann die Software LLWin 3.0 gestartet werden. Damit werden die Steuerungsprogramme für die fischertechnik-Modelle erstellt.
- Das letzte Programmsymbol spricht für sich, denn hier geht's zur Hilfe!

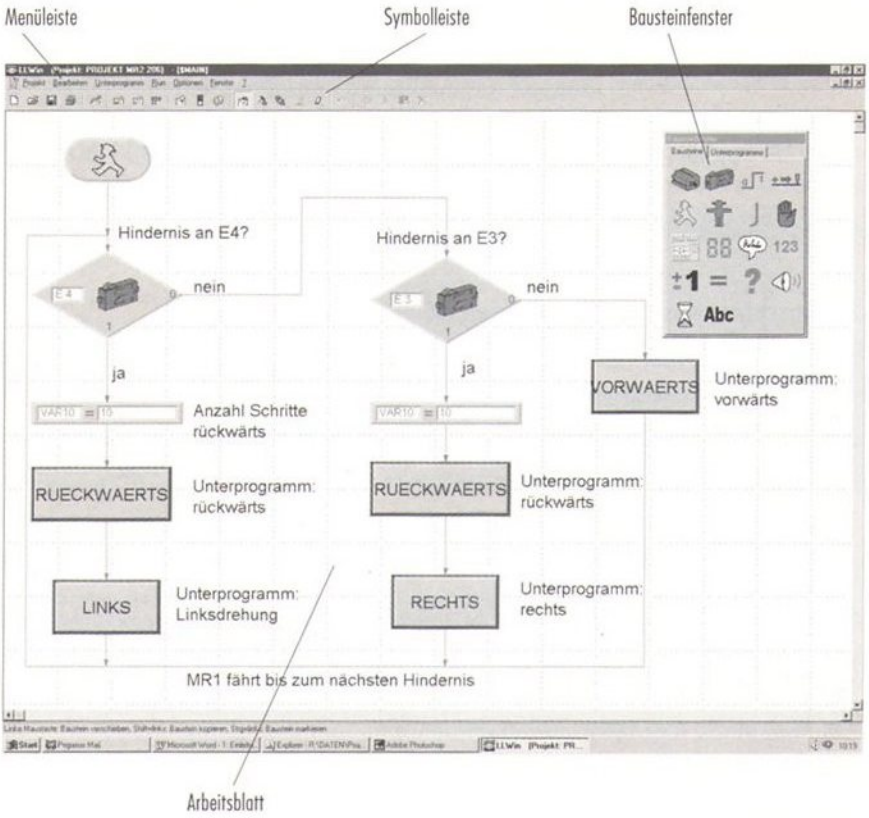
Durch einen Klick auf das X der Titelzeile kannst du das Fenster schließen.

1.2 Die Bedienoberfläche von LLWin

Neugierig? Dann starte doch einfach mal das Programm LLWin. Hierfür klickst du auf den Startknopf der Taskleiste und wählst anschließend PROGRAMME-LLWIN 3.0-LLWIN 3.0 aus. Daraufhin wird das Programm gestartet. Nun hast du die Möglichkeit, entweder ein neues Projekt zu erstellen, oder aber ein bereits existierendes Projekt zu öffnen. Ein neues Projekt wollen wir erst im Kapitel 3 Wir erstellen unser erstes Steuerungsprogramm anlegen. Um die Bedienoberfläche kennenzulernen, öffnen wir deshalb einmal eines der bereits existierenden Beispielprojekte. Dazu klickst du im Menü



PROJEKT auf den Eintrag ÖFFNEN. Daraufhin erscheint die Dialogbox "LLWin-Projekt öffnen".
Hier wählst du im Verzeichnis "Mobile Robots" das Projekt "MR2" aus. Mit einem Klick auf "Öffnen" wird dieses Projekt geöffnet und dir präsentiert sich die Bedienoberfläche der Software LLWin.:



In der Menüleiste findest du alle Befehle, die zum Bedienen der Software benötigt werden. Neben den Befehlen zum Verwalten, Erstellen und Testen der Projekte sind hier auch die verschiedenen Befehle zu den Programmoptionen und die Online-Hilfe zu finden. Für die wichtigsten Menübefehle sind in der Symbolleiste Icons vorhanden. Und wie werden nun Steuerungsprogramme erstellt? Mit den Bausteinen des Bausteinfensters werden beim Programmieren auf dem Arbeitsblatt die Ablaufpläne der Steuerungsprogramme erstellt. Die fertigen Ablaufpläne können dann überprüft und mit einem angeschlossenen fischertechnik Interface getestet werden. Aber immer mit der Ruhe, wir werden in den nächsten Kapiteln schrittweise das Programmieren kennenlernen! Nachdem du so einen ersten Eindruck von der Bedienoberfläche bekommen hast, schließt du das Projekt über den Befehl BEENDEN im Menü PROJEKT wieder. Die Abfrage, ob du das Projekt speichern möchtest, kannst du mit "Nein" beantworten.

2. Vor der Programmierung ein kurzer Test der Hardware

Damit wir die Steuerungsprogramme, die wir später erstellen werden, auch testen können, muss das Interface an den PC angeschlossen werden, das ist klar. Aber je nach verwendetem Interface (Intelligent Interface Art.-Nr. 30402 oder älteres paralleles Interface Art.-Nr. 30520) muss auch die Software entsprechend eingestellt und die Verbindung getestet werden. Dies wollen wir im folgenden Kapitel tun.

2.1 Anschluss des Interface an den PC

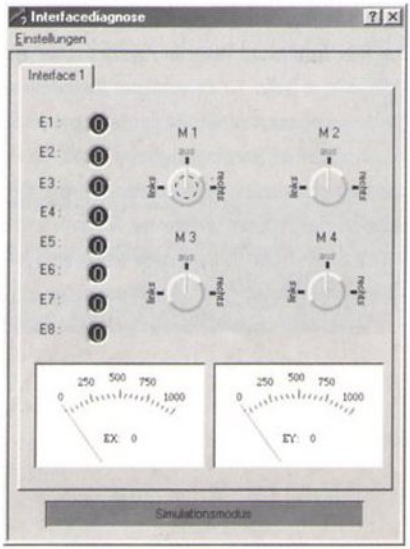
Dies dürfte kein größeres Problem sein. Das mit dem Interface mitgelieferte Verbindungskabel wird am Interface und an einer Schnittstelle des PCs angeschlossen:

- Beim Intelligent Interface (Art.-Nr. 30402) muss eine serielle Schnittstelle COM1 bis COM4 verwendet werden.
- Beim älteren parallelen Interface (Art.-Nr. 30520) muss hierfür die parallele Schnittstelle LPT1 oder LPT2 verwendet werden. An dieser Schnittstelle ist meistens auch der Drucker angeschlossen, so dass dieser evtl. abgetrennt werden muss.

Die Anschlüsse dieser Schnittstellen befinden sich auf der Rückseite deines Computers. Die genaue Lage der verschiedenen Anschlüsse ist in der Bedienungsanleitung deines PCs genau beschrieben, bitte dort nachlesen.
Vergiss nicht, das Interface mit Strom zu versorgen (Netzgerät oder Batterie). Die einzelnen Anschlüsse des Interfaces sind in der Bedienungsanleitung des jeweiligen Interfaces genau beschrieben.

2.2 Damit die Verbindung stimmt – die Interfaceeinstellungen

Damit die Verbindung von PC und Interface korrekt funktioniert, muss das jeweils verwendete Interface in LLWin eingestellt werden. Hierzu kannst du das Programm Interface-Diagnose verwenden, das sich über `START-PROGRAMME-LLWIN-INTERFACEDIAGNOSE` starten lässt.
Es erscheint folgendes Fenster: ►



Es zeigt die am Interface vorhandenen Eingänge und Ausgänge. Der grüne Balken ganz unten zeigt den Verbindungsstatus vom PC zum Interface an:

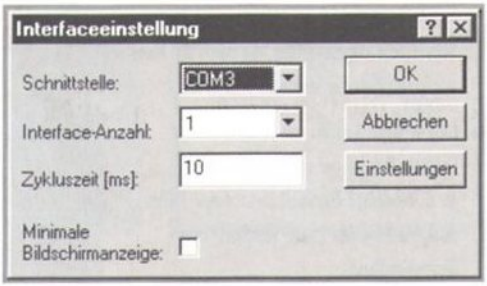
- "Simulationsmodus" bedeutet, dass bei den Interfaceeinstellungen keine Schnittstelle eingestellt wurde.
- "Keine Verbindung zum Interface" deutet darauf hin,

dass die Verbindung nicht korrekt eingestellt wurde und der PC keine Verbindung zum Interface aufbauen konnte. Der Balken erscheint dann in roter Farbe.

- "Verbindung zum Interface o.k." bestätigt eine korrekte Verbindung zum Interface.

Um die Interface- und damit die Verbindungseinstellungen verändern zu können, musst du im Menü EINSTELLUNGEN auf den Befehl INTERFACEEINSTELLUNGEN klicken, worauf folgendes Fenster erscheint:

Hier kannst du die "Schnittstelle" auswählen, an der du dein Interface angeschlossen hast. Um die weiteren Einstellungen wie "Interface-Anzahl", "Zykluszeit" und "Minimale Bildschirmanzeige" brauchst du dich im Moment noch nicht zu kümmern. Du kannst sie bei Bedarf (wenn z. B. zwei Interfaces angeschlossen werden) im Kapitel 8.5 nachlesen.



Hinweis: Solltest du kein Interface angeschlossen haben, kannst du bei "Schnittstelle" den Eintrag "keine" auswählen und damit die Software LLWin nur zum Simulieren deiner Steuerungsprogramme verwenden.

Wenn du so die Verbindung von PC und Interface einstellen konntest, kannst du das nächste Kapitel getrost überspringen. Wenn nicht, können dir vielleicht die folgenden Tipps weiterhelfen.

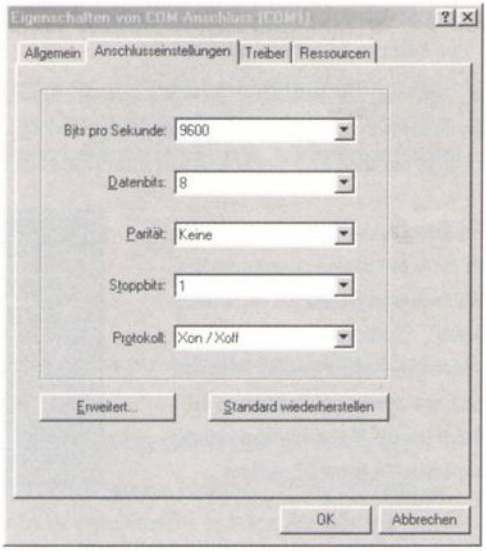
2.3 Falls die Verbindung nicht stimmt – keine Verbindung zum Interface!?

Falls beim Intelligent Interface trotz korrekt eingestellter serieller COM-Schnittstelle (s. oben) die Meldung "Keine Verbindung zum Interface" erscheint, solltest du nachfolgende Punkte überprüfen. Eventuell musst du hierfür auch einen "Computerkenner" zu Rate ziehen:

- Stromversorgung:
  Wird das Interface richtig mit Strom versorgt? Verwendest du als Stromversorgung Batterien oder Akkus, dann besteht die Möglichkeit, dass leere Akkus nicht mehr genug Spannung liefern. Sinkt die Spannung der Batterie unter 6 V, funktioniert der Prozessor des Interfaces nicht mehr. Die Leuchtdiode leuchtet trotzdem noch, wenn auch schwach. Dann solltest du die Akkus neu laden bzw. neue Batterien verwenden, oder aber das Interface falls möglich mit einem Netzgerät testen.

- Stimmen die Einstellungen der Schnittstelle?
  Hierzu musst du START-EINSTELLUNGEN-SYSTEMSTEUERUNG starten und anschließend bei SYSTEM die Registerkarte "Gerätemanager" aufrufen. Unter ANSCHLÜSSE kannst du die Eigenschaften der betreffenden Schnittstelle einstellen. Es gelten folgende Anschlusseinstellungen:

- Funktioniert die Schnittstelle überhaupt? Dies kannst du herausfinden, indem du eine serielle Maus daran anschließt und den PC neu startest. Funktioniert die Maus, schaltest du den PC aus, schließt die Maus an eine andere serielle Schnittstelle an und startest den PC neu. Dann erst wird das Interface an die freie Schnittstelle angeschlossen, die durch die Maus erfolgreich getestet wurde.



- Gibt es einen Konflikt mit einem anderen Gerätetreiber an der selben Schnittstelle (z. B. Modem)? Eventuell muss dieser Treiber deaktiviert werden (siehe Windows- oder Gerätehandbuch).
- Nur für Windows NT Benutzer:
  Ist das Interface beim Hochfahren des PCs bereits mit dem Rechner und der Stromversorgung verbunden, wird es von Windows NT unglücklicherweise in den Download-Modus umgeschaltet. Um die Verbindung zum PC wieder herzustellen musst du einfach die Stromversorgung am Interface kurz unterbrechen.
- Falls du immer noch keine Verbindung zum Interface herstellen kannst, ist wahrscheinlich das Interface oder das Verbindungskabel defekt. In diesem Fall wendest du dich an den fischertechnik Service (Adresse: siehe Menü: `? – Info über`).

Falls beim älteren parallelen Interface keine Verbindung zum Interface zu Stande kommt, gibt es folgende Fehlermöglichkeiten:

- Falsche parallele Schnittstelle ausgewählt (LPT1 oder LPT2).
- Keine Stromversorgung angeschlossen.

Hinweis: Die rote LED an diesem Interface leuchtet nicht, wenn die Stromversorgung angeschlossen wird, sondern erst, wenn das Interface vom PC angesprochen wird.

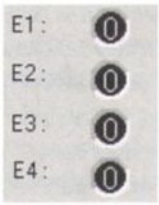
- Nur für Windows- NT-Benutzer:
  Der Treiber für die parallele Schnittstelle (Parallel-Port-Treiber) konnte nicht installiert werden. Dann müsste beim Installieren eine entsprechende Meldung erschienen sein. Dieser Treiber kann auf dem PC nur von demjenigen installiert werden, der "Administrator-Rechte" hat. Wende dich in diesem Fall an deinen Systemadministrator.
- Interface defekt. Dann wendest du dich an den fischertechnik Service (Adresse: siehe Menü: `? – Info über`).

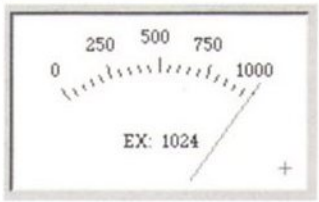
2.4 Funktioniert alles – die Interfacediagnose

Nachdem die Verbindung korrekt eingestellt ist, können wir mit Hilfe der Interfacediagnose das Interface selbst und daran angeschlossene Modelle testen.
Wie bereits erwähnt, zeigt das Diagnosefenster die verschiedenen Eingänge und Ausgänge des Interfaces:

• Eingänge

E1 – E8 sind die digitalen, EX und EY die analogen Eingänge des Interfaces. An die Eingänge des Interfaces werden sog. Sensoren angeschlossen. Dies können z. B. Taster, Schalter, Fototransistoren, etc. sein.





Du kannst die Funktion dieser Eingänge überprüfen, indem du am Interface z. B. an E1 einen Mini-Taster (Art.-Nr. 37783) anschließt  B2.6
(verwende am Taster die Kontakte 1 und 3). Sobald du den Taster drückst, wechselt die Anzeige von E1 von "0" nach "1". Hast du den Taster anders herum angeschlossen (Kontakte 1und 2), erscheint sofort die "1" und wechselt bei Tastendruck auf "0". Das Arbeiten mit den Analogeingängen wird im Kapitel 5.3 näher erläutert.

• Ausgänge

M1 – M4 sind die Ausgänge des Interfaces. Hier werden die sog. Aktoren angeschlossen. Dies können z. B. Motoren, Elektromagneten oder Lampen sein.

Wenn du einen Ausgang testen möchtest, schließt du an diesen Ausgang, z. B. M1, einen Motor an. Wenn du nun mit der linken Maustaste auf den Knopf des Ausgangs klickst, dreht sich der Motor links, bei der rechten Maustaste rechts herum. Drückst du zuerst die STRG-Taste und klickst dann den Ausgang an, bleibt er eingeschaltet. Ausschalten lässt er sich mit einem erneuten Mausklick.

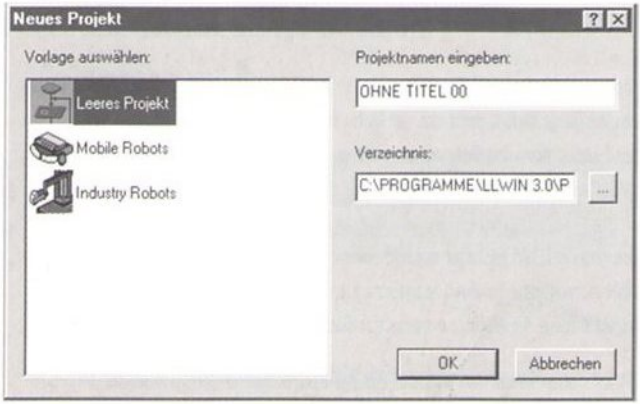
Natürlich kann die Interfacediagnose auch in LLWin gestartet werden. Solltest du das Diagnosefenster noch geöffnet haben, schließe es, indem du auf den Befehl BEENDEN im Menü EINSTELLUNGEN, oder auf das X in der Titelzeile klickst. Starte dann LLWin (kein Projekt öffnen). Im Menü OPTIONEN findest du die bereits bekannte INTERFACE-DIAGNOSE, die du mit einem Klick starten kannst (ein Klick auf das das Icon  in der Symbolleiste bewirkt übrigens das Gleiche). Für die Einstellung des Interfaces findest du im gleichen Menü den Befehl INTERFACE-EINSTELLUNGEN. Schließe die Interfacediagnose, lasse aber LLWin geöffnet, denn jetzt geht's ans Programmieren.

3. Wir erstellen unser erstes Steuerungsprogramm

Nachdem wir im letzten Kapitel die Hardware, also das Interface und daran angeschlossene Schalter und Motoren, getestet haben, wollen wir uns nun mit dem Programmieren befassen. Was aber bedeutet "Programmieren" eigentlich? Nun stell dir einmal vor, dass an unserem Interface z. B. ein Roboter angeschlossen ist. Dieser Roboter ist aber so dumm, dass er von alleine nicht funktioniert. Zum Glück sind wir da ein bisschen schlauer. Wir können dem Roboter ganz genau sagen, was er tun soll. Wie? Nun, was passierte, als wir im letzten Kapitel mit der linken Maustaste auf den Button Motor 1 geklickt haben? Richtig, wir haben den Motor eingeschaltet. Würde dieser Motor z. B. die Greifzange unseres Roboters bewegen, hätten wir nichts anderes getan, als dem Roboter gesagt: "Greife den Gegenstand!" Nun wollen wir aber nicht jeden Schritt von Hand auslösen, sondern der Roboter soll dies "automatisch" tun. Dazu müssen wir die einzeln auszuführenden Schritte so abspeichern, dass der Roboter sie nacheinander abarbeiten kann, d. h. wir müssen ein Programm erstellen, welches an unserer Stelle den Roboter steuert. In der Fachsprache nennt man das dann logischerweise ein **Steuerungsprogramm**.

3.1 Ein neues Projekt ist notwendig

Mit der Software LLWin haben wir nun ein tolles Werkzeug zur Hand, um solche Steuerungsprogramme zu entwerfen und mit Hilfe eines angeschlossenen Interfaces zu testen. Keine Angst, wir wollen nicht gleich einen Roboter programmieren. Wir begnügen uns zunächst mit einfachen Steuerungsaufgaben. Hierzu müssen wir ein **neues Projekt** erstellen. In der Menüleiste findest du den Eintrag PROJEKT. Wenn du mit der linken Maustaste darauf klickst, öffnet sich ein Untermenü mit verschiedenen Befehlen, unter anderem dem Befehl NEU. Wenn du diesen anklickst öffnet sich ein Dialogfenster "Neues Projekt", in dem bereits einige Projektvorlagen zu verschiedenen fischertechnik-Modelltypen (z. B. für Mobile Robots oder Industry Robots) vorhanden sind.

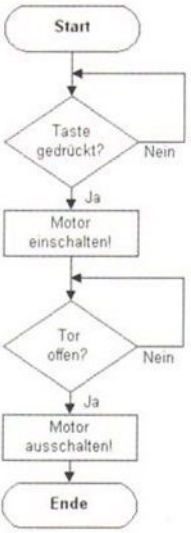


Da wir kein bestimmtes Modell angeschlossen haben, wählen wir für unsere Zwecke den Eintrag "Leeres Projekt" (einfach mit der linken Maustaste anklicken). Auch vergeben wir noch keinen Projektnamen, sondern schließen den Dialog mit OK. Damit wird unser neues Projekt angelegt und geöffnet.

3.2 Bausteine – Die Elemente des Steuerungsprogramms

Nun können wir uns an die Aufgabe machen, unser erstes Steuerungsprogramm zu erstellen. Dieses wollen wir an einer konkreten Steuerung tun:

Funktionsbeschreibung:
Stell Dir ein Garagentor vor, das sich automatisch öffnen lässt. Vielleicht besitzt ihr sogar ein solches zu Hause! Kommt man mit dem Auto zur Garage, genügt ein Tastendruck beim Sender und das Garagentor wird, von einem Motor angetrieben, geöffnet. Der Motor muss so lange laufen, bis das Garagentor vollständig geöffnet ist.



Nun ist es recht umständlich und auch nicht sehr anschaulich, eine Steuerung in Worten zu beschreiben. Deshalb verwendet man für die Darstellung der nacheinander auszuführenden Aktionen und die Bedingungen, die für diese Aktionen erfüllt sein müssen, sogenannte **Ablaufpläne**. Die Bedingung für die Aktion "Einschalten des Motors" ist im Fall unserer Steuerung, dass der Taster gedrückt wird.
Das Lesen eines solchen Ablaufplanes ist ganz einfach: Immer schrittweise den Pfeilen nach! Diese ergeben dann die genaue Funktionsweise der Steuerung – die einzelnen Schritte können nur in der durch die Pfeile vorgegebenen Reihenfolge ausgeführt werden, niemals anders. Sonst bräuchten wir uns die ganze Arbeit nicht zu machen – oder?

Mit Hilfe unserer Software LLWin können wir nun genau diesen Ablaufplan zeichnen und damit das **Steuerungsprogramm** für die angeschlossene Hardware (Interface, Motoren, Schalter, etc.) erstellen. Den Rest übernimmt die Software, was im Übrigen bei großen, industriellen Anwendungen auch nicht anders ist! Damit können wir uns ganz auf die Erstellung des Ablaufplans konzentrieren.

Vielleicht hast du bereits festgestellt, dass, nachdem wir unser neues Projekt erstellt haben, automatisch das erste Element des Ablaufplans auf der Arbeitsoberfläche eingefügt wurde, nämlich der Baustein START.

Wieder ein neuer Begriff? Halb so wild! In LLWin werden die einzelnen Funktionselemente, mit denen der Ablaufplan erstellt wird, **Bausteine** genannt. Sie heißen deshalb so, weil wir ja auch unsere fischertechnik-Modelle mit genau diesen Bausteinen aufbauen. Die Aktion "Motor einschalten" heißt doch nichts anderes, als dass das Interface tatsächlich diesen Motor, der am Interface angeschlossen ist, einschalten soll! Alle verfügbaren Bausteine findest du im ebenfalls geöffneten "Bausteinfenster". Sollte dieses nicht geöffnet sein, einfach in der Menüleiste `BEARBEITEN` und anschließend `BAUSTEIN EINFÜGEN` mit der linken Maustaste anklicken. Sollte es dann immer noch nicht erscheinen, aktiviere es in Menü Optionen `Bausteinfenster`. Hier kannst du es auch deaktivieren, falls es dich einmal stören sollte.

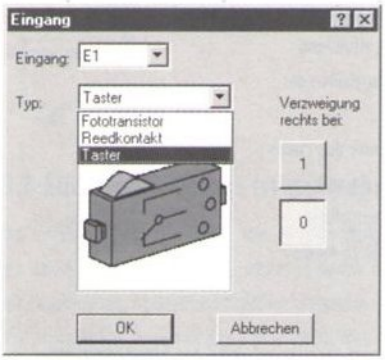
3.3 Bausteine einfügen, verschieben und ändern

Jetzt geht es darum, mit den im Bausteinfenster enthaltenen Bausteinen den Ablaufplan für unsere Garagentorsteuerung zu erstellen. Alle verfügbaren Bausteine können, wenn das Bausteinfenster sichtbar ist, aus diesem geholt und auf der Arbeitsoberfläche eingefügt werden.

Dazu bewegst du die Maus auf das Symbol des gewünschten Bausteins. Der Name des Bausteins wird dann zusätzlich eingeblendet. Jetzt drückst du die linke Maustaste und ziehst den Baustein bei gedrückter Maustaste an die gewünschte Stelle auf der Arbeitsoberfläche. Lässt du jetzt die Maustaste los, erscheint das Dialogfeld des jeweiligen Bausteins. Da unser zuerst benötigtes Bauteil ein Taster

ist, der an einem Eingang des Interfaces angeschlossen ist, probierst du es am besten gleich mal mit diesem Baustein aus.

Folgendes Dialogfeld erscheint: ▼



Hier kannst du nun einstellen, an welchem Eingang des Interfaces der Taster angeschlossen ist. Der Eingang E1 ist bereits voreingestellt, den wir auch verwenden wollen. Der "Typ" des Eingangs sollte auf "Taster" eingestellt und die "Verzweigung rechts bei" "0" hell unterlegt sein.

- "0" bedeutet, dass der Kontakt durch den Taster nicht geschlossen ist.
- "1" bedeutet, dass der Kontakt durch den Taster geschlossen ist.

Achtung: Hierzu musst du den Taster auch in dieser Weise am Eingang E1 des Interfaces anschließen (Kontakte 1 und 3)!

Nach Eingabe aller benötigten Werte wird der Dialog mit einem Klick auf OK geschlossen und der Baustein ist auf der Arbeitsoberfläche platziert.

Ein Baustein lässt sich auch nach dem Einfügen bei gedrückter linker Maustaste an die gewünschte Stelle **verschieben**. Ein **Ändern** der Bausteinwerte ist ebenfalls sehr leicht möglich. Ein Klick mit der rechten Maustaste auf den Baustein öffnet das Dialogfeld, in dem die betreffenden Werte geändert werden können.

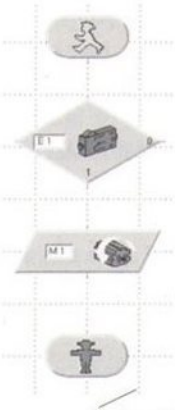
Die Größe der Bausteine auf dem Arbeitsblatt kannst du mit der Zoomfunktion verändern. Im Menü `Optionen Arbeitsblatt` kannst du mit dem Schieberegler den Zoomfaktor einstellen. Noch schneller geht es mit den Plus- und Minustasten am Nummernblock deiner Tastatur.

Füge als Nächstes nach der oben beschriebenen Vorgehensweise den Baustein AUSGANG ein. 

Mit diesem Baustein kannst du sowohl einen Motor als auch eine Lampe oder einen Elektromagneten ein- oder ausschalten. Da wir in diesem Fall einen Motor ansteuern wollen, wählst du im Dialog bei "Typ" den Motor aus.

Den Ausgang, den wir ansteuern wollen, ist "M1", die Drehrichtung "links".

Ergänze abschließend noch den Baustein ENDE, so dass dein Ablaufplan in etwa so aussieht: ▶



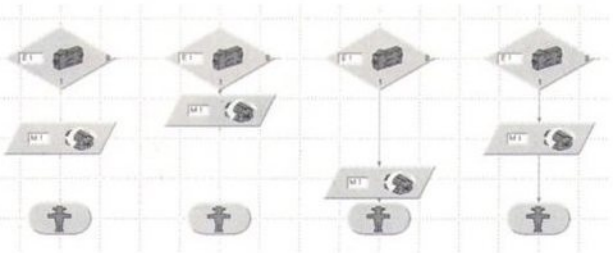
Rasterpunkt

3.4 Verbinden der Bausteine

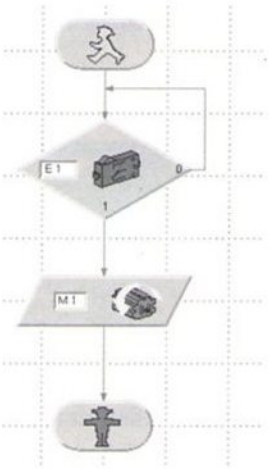
Was uns nun noch fehlt, sind die Verbindungslinien zwischen den einzelnen Bausteinen. Vielleicht hast du die Bausteine so platziert, dass die Verbindungslinien automatisch gezogen wurden. Dieses automatische Verbinden zweier Bausteine wird "Auto Connect" genannt. Es funktioniert immer dann, wenn du einen Baustein neu einfügst und der Abstand zu dem vorhergehenden Baustein nicht mehr als 9 Rasterpunkte beträgt.

Zum manuellen Zeichnen von Linien muss der Befehl LINIEN ZEICHNEN im Menü BEARBEITEN ausgeführt werden. Ein Klick auf das Bleistiftsymbol in der Symbolleiste bewirkt dasselbe. Aus dem Cursor wird ein Stift. Nun kannst du zwischen zwei Bausteinen eine Linie ziehen, indem du zuerst mit der linken Maustaste auf den Ausgang des ersten Bausteins klickst. Lässt du die Maustaste wieder los, wird aus dem Cursor ein Fadenkreuz. Daran erkennst du, dass der Ausgang "getroffen" wurde. Danach klickst du auf den Eingang des zweiten Bausteins, die Linie wird gezogen. Den Verlauf von Linienteilstücken oder des gesamten Linienzuges kannst du auch selbst bestimmen. Durch Klick auf einen Eingang, Ausgang oder eine Linie wird das Zeichnen begonnen. Bewegst du dann das Fadenkreuz genau senkrecht oder waagerecht zum ersten Punkt und betätigst erneut die linke Maustaste, wird die erste Teillinie gezeichnet. In gleicher Weise können weitere Teilstücke gezeichnet werden, bis der Zielpunkt erreicht ist. Referenzpunkt für die nächste Zeichenoperation ist immer der Endpunkt der zuletzt gezeichneten Teillinie. Möchtest du den restlichen Linienzug automatisch zeichnen lassen, klickst du einfach auf den Zielpunkt.

Verbindungslinien lassen sich auch durch Verschieben der Bausteine erzeugen. Wenn du einen Baustein so verschiebst, dass sein Eingang auf dem Ausgang eines anderen liegt, wird zwischen beiden Bausteinen eine Verbindungslinie erzeugt. Das gilt auch bei einem Ausgang, der auf einen Eingang geschoben wird. Danach kannst du den Baustein in seine Endposition schieben oder weitere Verbindungen für die verbleibenden Ein- und Ausgänge zeichnen.



Auch das **Löschen** von Verbindungslinien ist sehr einfach. Gleich dem Radieren auf dem Papier existiert auch ein Radiergummi auf der Symbolleiste (rechts neben dem Bleistift). Ein Klick und der Cursor wird zum Radiergummi. Klickst du jetzt mit der linken Maustaste auf eine Verbindungslinie, so wird nur diese gelöscht. Fehlende Teilstücke an anderen angrenzenden Linien werden vervollständigt. Betätigst du die rechte Maustaste wird die getroffene Linie inklusive aller angrenzenden Linien gelöscht. Alles klar? Nein! Keine Sorge, mit der Zeit wird dies ein Kinderspiel für dich.



Der fertige Ablaufplan sollte nach dem Verbinden der Bausteine so aussehen: ▶

Die Verbindung vom Ausgang "0" des Tasters zurück zu dessen Eingang wird gezeichnet, indem zunächst auf den Ausgang und anschließend auf die Verbindungslinie START-TASTER geklickt wird. Das Programm läuft dann so lange in dieser Schleife, bis der Taster betätigt und damit der Motor für das Garagentor eingeschaltet wird.

3.5 Testen des ersten Steuerungsprogramms

Um unser erstes Steuerungsprogramm testen zu können, solltest du ein kleines Modell aufbauen. Dazu genügt es, an das Interface an E1 einen Taster und an M1 einen Motor anzuschließen. Falls du den Baukasten "Computing Starter" besitzt, kannst du auch daraus das Modell "Motorsteuerung" aufbauen. Es eignet sich hervorragend zum Testen dieses Programms.

> Hinweis: Der Anschluss des Interfaces an den PC und die Einstellung des Interfaces wurde bereits im vorigen Kapitel durchgeführt und kann dort nachgelesen werden.

Bevor du das Steuerungsprogramm testest, solltest du das Projekt auf der Festplatte abspeichern. Klicke mit der Maus auf den Befehl SPEICHERN UNTER im Menü PROJEKT. Darauf erscheint folgende Dialogbox:

Wähle bei "Speichern in" das Verzeichnis, in dem du das Projekt abspeichern willst. Gebe bei "Dateinamen" einen noch nicht vergebenen Namen ein, z. B. GARAGENTOR und bestätige mit einem linken Mausklick auf "Speichern".

Zum Testen eines neu erstellten Projekts solltest du zunächst den Befehl INIT im Menü RUN aufrufen (oder klicke auf das entsprechende Icon  der Symbolleiste). Bei Eintritt in den Init-Modus verschwindet das Gitternetz auf der Programmieroberfläche. Hier wird getestet, ob bei allen Bausteinen die Anschlüsse verbunden sind. Nicht verbundene Bausteine leuchten violett auf. Dann musst du zurück in Bearbeiten Hauptprogramm wechseln, am Schnellsten mit der Taste <F6>. In diesem sog. "Editiermodus" kannst du die fehlenden Linien ergänzen. Falls alle Bausteine verbunden sind und kein Eingabefeld mehr violett leuchtet, wähle im Menü RUN den Befehl START oder drücke die Taste <F5> (auch ein Klick auf das Icon  ist möglich). Zunächst wird das Programm übersetzt, wobei eine Balkenanzeige über den Bearbeitungsstand informiert. Nach der Übersetzung wird das Projekt im "Online-Betrieb" gestartet.

> Hinweis: Eine ausführliche Erklärung zu dieser Betriebsart und zur zweiten Betriebsart "Download-Betrieb" findest du im Kapitel 3.7.

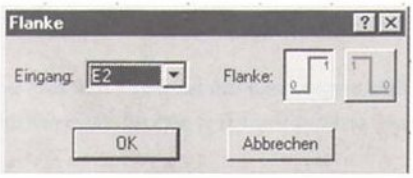
Der Baustein EINGANG leuchtet rot auf. Es wird angezeigt, dass der Ablauf in diesem Baustein auf ein Ereignis wartet, nämlich dass der Taster gedrückt wird. Drücke den am Eingang 1 angeschlossenen Taster. Damit ist die Bedingung zum Weiterschalten im Baustein EINGANG erfüllt und der Baustein MOTOR wird aufgerufen. Der Motor wird eingeschaltet und der Ablauf ruft im nächsten Schritt den unteren Baustein ENDE auf. Du kannst das Programm stoppen, indem du auf die rote Ampel in der Symbolleiste klickst. Mit der grünen Ampel kannst du das Programm nochmals starten (wie oben) oder mit <F6> in den Editiermodus zurückschalten.

Hat alles geklappt? Herzlichen Glückwunsch! Du hast damit Dein erstes Steuerungsprogramm erstellt und getestet. Falls es nicht funktioniert – nicht verzagen, einfach noch einmal alles genau nachprüfen, bestimmt hat sich irgendwo ein kleiner Fehler versteckt. Jeder Programmierer macht mal Fehler und aus Fehlern lernt man am meisten. Deshalb, nur Mut!!!

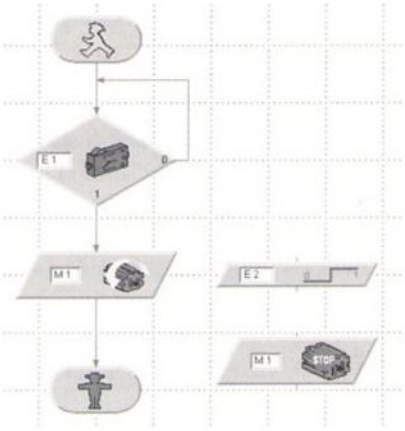
3.6 Mehrere Bausteine gleichzeitig bearbeiten

Nun können wir uns an die Aufgabe machen, unser Steuerungsprogramm zu erweitern. Denk mal zurück an die Funktionsbeschreibung der Garagentorsteuerung - fehlt da nicht noch etwas? Richtig, zwar haben wir den Motor per Tastendruck eingeschaltet, nachdem das Tor geöffnet ist, muss er aber wieder automatisch abgeschaltet werden! In der Praxis geschieht dies mit einem sog. Endschalter. Das ist ein Taster, der so am Garagentor angebracht ist, dass er in dem Moment betätigt wird, in dem der Motor das Tor ganz geöffnet hat. Und wie beim Einschalten des Motors kann dieses Signal verwendet werden, um den Motor wieder auszuschalten.
Für die Abfrage des Endschalters könnten wir wieder den Baustein EINGANG verwenden. Wir wollen aber gleich einen neuen Baustein kennenlernen, die FLANKE.

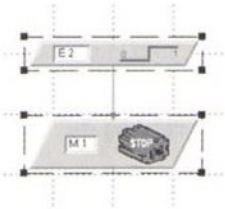
Der Baustein Flanke wartet darauf, dass ein digitaler Eingang entweder von "0" auf "1" oder von "1"auf "0" wechselt. Diesen Übergang nennt man "Flanke". Im Bausteindialog musst du angeben, an welchem Eingang auf welche Flanke gewartet werden soll:



Zum Abschalten des Motors verwenden wir wieder den Baustein AUSGANG.
Hol dir diese beiden Bausteine aus dem Bausteinfenster, verbinde sie miteinander und platziere sie zunächst einmal neben dem Ablaufplan.



Für den Endschalter musst du einen zweiten Taster an einem anderen Eingang (z. B. E2) anschließen. Mit dem Ausgangsbaustein wollen wir natürlich den am Ausgang M1 angeschlossenen Motor abschalten. Zum Einfügen dieser Bausteine in das Ablaufdiagramm muss noch die Verbindungslinie zum Baustein ENDE gelöscht und dieser etwas weiter nach unten geschoben werden.

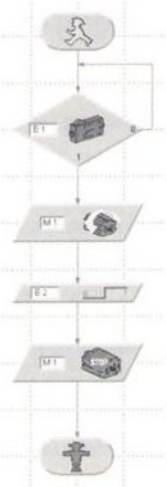


Um die beiden neuen Bausteine gemeinsam verschieben zu können, musst du sie zunächst **markieren**. Klicke hierzu bei gedrückter STRG-Taste die beiden Bausteine nacheinander mit der linken Maustaste an. Jeder so markierte Baustein erhält einen schwarzen Rahmen.

Anschließend lässt du die STRG-Taste los und kannst dann die markierten Bausteine an die gewünschte Position befördern. Einfach einen der markierten Bausteine mit der linken Maustaste anklicken und alle markierten Bausteine bei gedrückter Maustaste verschieben.
Anstelle der STRG-Taste kannst du auch die SHIFT-Taste zum Markieren verwenden. Dann werden alle Bausteine zwischen den beiden mit der Maus angeklickten markiert. Probiere es einfach mal aus, indem du bei gedrückter SHIFT-Taste den Start- und Ende-Baustein anklickst. Alle Bausteine (in dem Fall der gesamte Ablaufplan) werden markiert und können zusammen verschoben werden. Auf diese Weise werden übrigens auch Dateien im Windows-Explorer markiert.

Hinweis: Diese Möglichkeit, einzelne Bausteine oder ganze Gruppen von Bausteinen zu markieren, ist nicht nur zum Verschieben von Bausteinen bei Änderungen wichtig. Wie wir später noch sehen werden, können damit auch ganze Bausteingruppen über ie Befehle des Menüs BEARBEITEN ausgeschnitten, gelöscht, kopiert und in Unterprogramme oder andere Projekte eingefügt werden.

Nachdem du die noch notwendigen Verbindungen gezeichnet hast, sollte der fertige Ablaufplan so aussehen: ▶



Unser Garagentormotor läuft nun solange, bis er durch den Taster am Eingang E2 abgeschaltet wird. Am Besten testest du es gleich mal. Vergiss nicht dein Projekt zu speichern!!

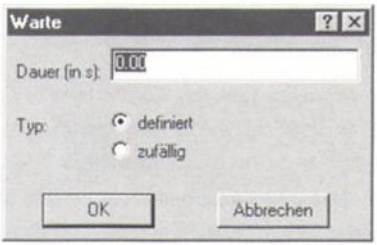
Das Tor ist nun offen. Wie aber wird es wieder geschlossen? Natürlich könnten wir den Motor wieder durch Drücken eines Tasters starten! Wir wollen aber eine andere Lösung testen und dabei zwei neue Bausteine kennenlernen. Hierfür erstellen wir ein neues Projekt (den jetzigen Ablaufplan benötigen wir später noch einmal). Um aber nicht alles noch einmal zeichnen zu müssen, kopieren wir den alten Ablaufplan in das neue Projekt. Dazu gehst du folgendermaßen vor:

1. Du lässt das bisherige Projekt geöffnet und startest LLWin ein zweites Mal (Windows-Start-Button – Programme – LLWin – LLWin 3.0). Hier erzeugst du ein neues Projekt. Du kannst nun über die Taskleiste sehr schnell zwischen den beiden Projekten hin und her schalten.
2. Im bisherigen Projekt markierst du den gesamten Ablaufplan und führst den Befehl KOPIEREN im Menü BEARBEITEN aus (Icon  der Symbolleiste).
3. Wechsle in das neue Projekt und führe den Befehl EINFÜGEN im Menü BEARBEITEN aus (Icon  der Symbolleiste).
4. Nun kannst du den Ablaufplan auf dem Arbeitsblatt des neuen Projekts platzieren. Das alte Projekt kannst du schließen (PROJEKT BEENDEN).

Bevor wir den Ablaufplan erweitern können, musst du die Verbindung von "Motor abschalten" und "Ende" löschen und den Baustein ENDE nach unten verschieben. Die neuen Bausteine kannst du nun nach dem Baustein, mit dem wir den Motor abgeschaltet haben, einfügen.

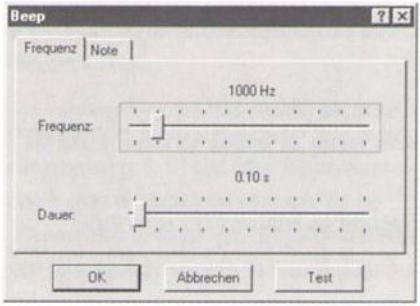
Das Garagentor soll nach einer Zeit von 10 Sekunden automatisch geschlossen werden. Hierfür kannst du den Baustein WARTE:

verwenden. Im Bausteindialog musst du die "Dauer" von 10 Sekunden und als "Typ" definiert einstellen (was man mit einer "zufällig" langen Zeit anfangen kann, kannst du in Kapitel 7.18 nachlesen).

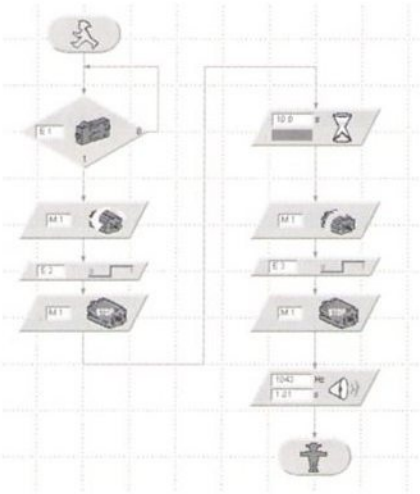


Zum Schließen des Garagentores muss der Motor natürlich in die andere Richtung laufen. Abgeschaltet wird der Motor durch einen weiteren Endschalter an E3.
Nachdem das Tor geschlossen ist, soll ein Signalton ausgegeben werden. Hierfür dient der Baustein BEEP:

Der Baustein BEEP gibt diesen Signalton über den Lautsprecher des PCs aus, wobei du Tonhöhe und Tondauer im Bausteindialog einstellen kannst:



Dein fertiges Ablaufdiagramm sollte etwa so aussehen ▶ (die neuen Bausteine sind zur besseren Darstellung nach rechts verschoben):



Hinweis: Im Init-Modus kannst du die Nummern und Zahlenwerte in den Bausteinen ändern, indem du mit der linken Maustaste auf den betreffenden Baustein klickst. Der Bausteindialog erscheint und du kannst die gewünschten nderungen durchführen. Du brauchst also nicht in den Editiermodus zurückschalten.

Sind im Ablaufplan keine Fehler mehr enthalten, kannst du die erweiterte Garagentorsteuerung wie gewohnt testen. Bei Betätigung des Tasters an E1 wird der Motor eingeschaltet, bei Betätigung von E2 wieder abgeschaltet. Damit ist das Garagentor geöffnet. Nun leuchtet der Baustein WARTE für 10 Sekunden, unserer eingestellten Wartezeit, rot auf. Dann wird der Motor mit anderer Drehrichtung eingeschaltet bis der Taster an E3 betätigt wird. Zur Quittierung des geschlossenen Garagentores ertönt der Signalton. Verändere im Init-Modus die Wartezeit und den Signalton. Du kannst auch eine kleine Melodie spielen, indem du mehrere Bausteine BEEP hintereinander schaltest.

3.7 Online- oder Download-Betrieb – Wo ist denn da der Unterschied?

Bisher haben wir unsere Steuerungsprogramme im sog. "Online-Betrieb" getestet. Das heißt, der PC, auf dem wir unseren Ablaufplan erstellt haben, übernahm die Steuerung der an das Interface angeschlossenen Bausteine (Motor und Schalter). Das Interface diente hierbei als Schnittstelle zwischen PC und Bausteinen.

Vielleicht ist dir bereits aufgefallen, dass es im Menü RUN auch einen Befehl DOWNLOAD gibt. Nachdem du mit INIT dein Steuerungsprogramm überprüft hast und keine Fehler mehr enthalten sind, kannst du mit dem Befehl DOWNLOAD dieses Steuerungsprogramm auf das angeschlossene Interface übertragen. Anschließend kann das Verbindungskabel zwischen PC und Interface entfernt werden. Nun kann das Interface das Steuerungsprogramm unabhängig vom PC ausführen. Wichtig ist dieser "Download-Betrieb" z. B. bei der Programmierung von mobilen Robotern aus dem Baukasten "Mobile Robots", bei denen ein Verbindungskabel zwischen PC und Roboter sehr hinderlich wäre. Trotzdem sollten Steuerungsprogramme zuerst im "Online-Betrieb" getestet werden, da sich hier mögliche Fehler besser finden lassen. Das "fehlerfreie" Programm kann dann per Download auf das Interface übertragen werden.

Achtung: Der Download-Betrieb ist nur mit dem Intelligent Interface (Art.-Nr. 30402) möglich! Deshalb ist der Befehl DOWNLOAD auch deaktiviert, wenn das ältere parallele Interface (Art.-Nr. 30520) verwendet wird und hierzu bei den Interfaceeinstellungen LPT1 oder LPT2 als Schnittstelle ausgewählt ist.

Solltest du also das neuere Intelligent Interface besitzen, kannst du unsere Garagentorsteuerung mit dem Befehl DOWNLOAD auf das Interface übertragen. Eine Balkenanzeige informiert dich dabei wieder über den Bearbeitungsstatus. Nach erfolgreicher Übertragung kannst du das Verbindungskabel zum PC vom Interface entfernen und dein Steuerungsprogramm läuft wie bisher. Da das Interface keinen eigenen Lautsprecher besitzt, funktioniert die Signalausgabe mit BEEP in dieser Betriebsart natürlich nicht.

Wichtig: Um nach einem Download die Verbindung zwischen dem Intelligent Interface und dem PC wieder herzustellen musst du die Stromversorgung des Interfaces für ca. 3-4 Sekunden unterbrechen. Das auf dem Interface laufende Programm wird dadurch gelöscht, das Interface kehrt in den Online-Modus zurück und kann wieder Verbindung zum PC aufnehmen. Das geht natürlich nur, wenn du das Verbindungskabel wieder am Interface anschließt!!

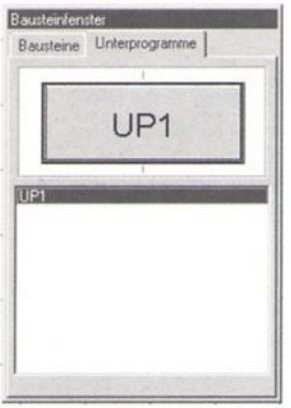
Hinweis: Die übertragenen Steuerungsprogramme werden nur so lange im Interface gespeichert, wie dieses mit Strom versorgt wird. Wird also der Netzstecker gezogen oder ist der Akku leer, muss das Steuerungsprogramm erneut übertragen werden.

4. Hilfe, ich verlier die Übersicht! – Arbeiten mit Unterprogrammen

Zwar sind unsere bisher entworfenen Ablaufpläne noch nicht so umfangreich, dass wir bereits die Übersicht verlieren, aber sicherlich kannst du dir vorstellen, dass dies bei größeren Projekten mit umfangreicheren Ablaufplänen sehr leicht der Fall sein kann. Plötzlich ist das Arbeitsblatt voller Bausteine, überall sind Verbindungslinien und auf dem Bildschirm muss man mit den Scroll-Balken ständig hin- und herfahren. "Wo war jetzt dieser oder jener Ausgang?" Kurz – es droht ein kleines Chaos! Was tun? Gibt es nicht eine Möglichkeit, hier wieder etwas mehr Ordnung zu schaffen? Doch – und sie nennt sich **Unterprogramm**!

4.1 Grundlagen zur Unterprogrammtechnik

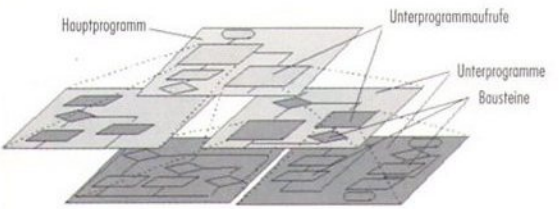
In LLWin stellt ein Unterprogramm zunächst einmal nur einen Baustein dar. Wenn du im Bausteinfenster auf dem Register Unterprogramm und dann auf den Eintrag "UP1" klickst, siehst du im oberen Feld einen Baustein mit dem Namen "UP1". UP steht übrigens für Unterprogramm – logisch, oder?



Diesen Baustein kannst du wie einen anderen Baustein auch auf die Arbeitsoberfläche ziehen und dort ablegen. Auch besitzt er bereits zwei Anschlüsse für Verbindungslinien: einen Eingang oben und einen Ausgang unten, also nichts Besonderes, wirst du sagen! Aber, das Besondere steckt im wahrsten Sinne des Wortes im Detail! Die Funktion dieses Bausteins kannst du selber vorgeben bzw. programmieren. Wie? So wie wir bisher die Ablaufpläne für unser Steuerungsprogramm erstellt haben, können wir für die Funktion dieses Bausteins ebenfalls einen Ablaufplan, oder eben ein Unterprogramm, erstellen. Wofür aber die Ein- und Ausgänge bei Unterprogrammen? Wir wollen unser Unterprogramm ja an einer bestimmten Stelle in unser Steuerungsprogramm einbauen, d. h. mit Verbindungslinien versehen. Die Verbindungslinie zum Eingang des Unterprogrammbausteins heißt nichts anderes, als dass hier das Unterprogramm aufgerufen und abgearbeitet wird. Beim Ausgang ist dies nicht anders: Nachdem das Unterprogramm abgearbeitet wurde, folgt der nächste Baustein, der mit dem Ausgang des Unterprogramms verbunden ist.
Damit wird eine zweite sehr wichtige Aufgabe von Unterprogrammen deutlich: Wird die Funktion, die dieses Unterprogramm erfüllt, häufiger benötigt, muss sie nicht jedesmal neu erstellt werden. Es genügt, das entsprechende Unterprogramm einmal zu erstellen. Anschließend läßt es sich beliebig oft in einem Steuerungsprogramm verwenden. Das spart eine Menge Aufwand und dient der Übersichtlichkeit. Darüber hinaus kann das Unterprogramm auch in andere Projekte kopiert und dort verwendet werden. Toll, was?

4.2 Projektaufbau bei Verwendung von Unterprogrammen

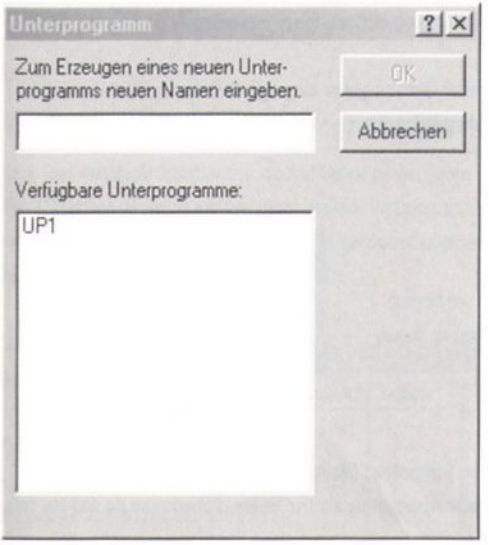
Die folgende Abbildung zeigt den Aufbau eines Projekts, in dessen Ablauf Unterprogramme enthalten sind:



Der Unterprogrammbaustein wird im Ablaufplan des Steuerungsprogramms eingefügt und stellt dort den Unterprogrammaufruf dar. Den Ablaufplan des Unterprogramms selber kannst du dir eine Ebene tiefer vorstellen, denn in der (höheren) Ebene des Unterprogrammaufrufs ist er ja nicht sichtbar. Natürlich kann ein Unterprogramm wiederum Unterprogrammbausteine enthalten, deren Ablaufpläne noch eine Ebene tiefer liegen. Die oberste Ebene wird sinnvollerweise Hauptprogramm genannt, denn darüber liegen keine anderen Ebenen!

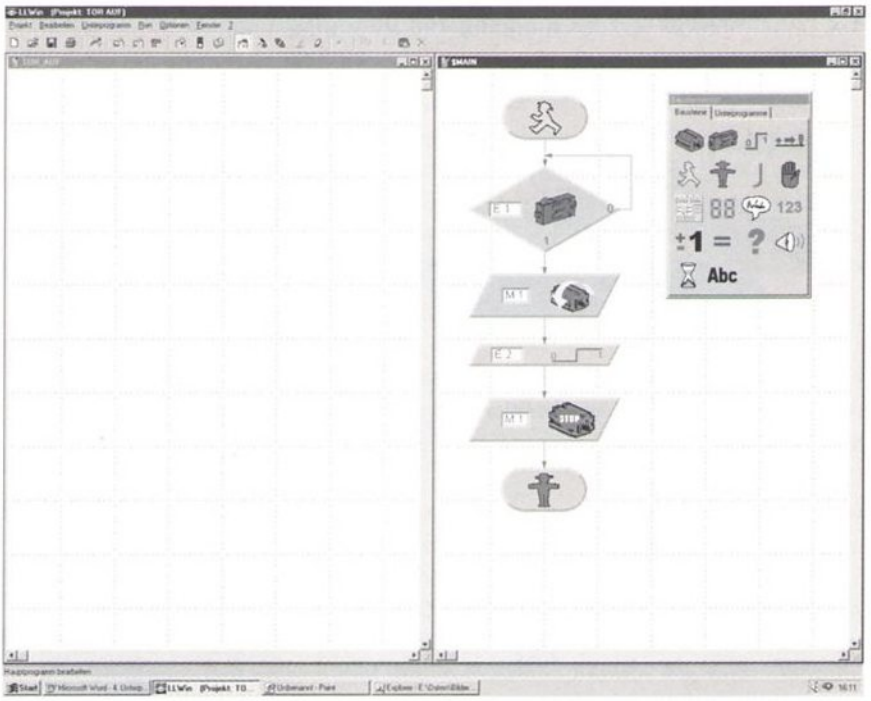
Den praktischen Einsatz von Unterprogrammen wollen wir an unserer Garagentorsteuerung kennenlernen. Dazu wollen wir in unserem ersten Projekt das Öffnen des Garagentores von einem Unterprogramm erledigen lassen. Öffne bitte dieses Projekt.

Zunächst müssen wir uns ein neues Unterprogramm erstellen. Hierzu klickst du im Menü BEARBEITEN auf den Befehl UNTERPROGRAMM oder auf das Icon  in der Symbolleiste. Darauf erscheint folgendes Dialogfenster:

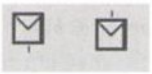


Im oberen Eingabefeld gibst du als neuen Unterprogrammnamen "TOR_AUF" ein und klickst anschließend auf OK. Eine leere Arbeitsoberfläche erscheint. Diese ist das Arbeitsblatt, auf dem nun der Ablaufplan für das Unterprogramm erstellt werden kann. Wenn du nochmals auf die Abbildung zu Beginn schaust, befinden wir uns jetzt also eine Ebene unter dem Hauptprogramm.

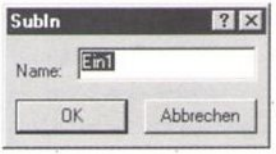
Die Software LLWin hat hierfür ein neues Bearbeitungsfenster erzeugt. Wenn du im Menü FENSTER auf den Befehl NEBENEINANDER klickst, werden auf der Arbeitsoberfläche zwei Fenster angezeigt: Ein Fenster mit dem Namen $MAIN, welches unser bereits erstelltes Steuerungsprogramm (das Hauptprogramm) enthält, und ein Fenster TOR_AUF für unser Unterprogramm.



Zunächst musst du den bereits beschriebenen Eingang und Ausgang einfügen. Hierzu verwendest du die Bausteine SUBIN und SUBOUT aus dem Bausteinfenster.
SUBIN steht für "Unterprogramm Eingang". SUBOUT heißt "Unterprogramm Ausgang". Diese beiden Bausteine sind im Bausteinfenster nur verfügbar, wenn man ein Unterprogramm bearbeitet.



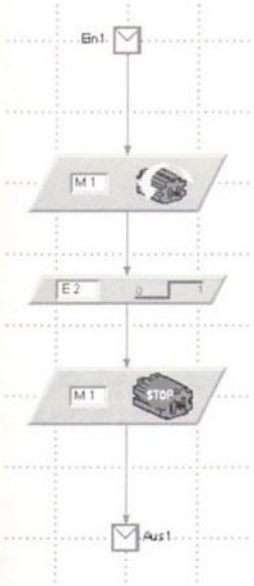
Beim Platzieren wirst du jeweils nach einem Namen für den Eingang bzw. Ausgang gefragt. Den vorgeschlagenen Namen kannst du verwenden und bestätigst ihn mit einem Klick auf OK.



Als Nächstes musst du die drei Bausteine zum Öffnen des Garagentores (Motor einschalten, Endschalter und Motor ausschalten) einfügen. Da wir diese ja im bisherigen Steuerungsprogramm ersetzen wollen, schneiden wir sie dort aus und fügen sie in unserem Unterprogramm ein:

- Markiere zunächst die betroffenen Bausteine im Fenster $MAIN
- Führe den Befehl AUSSCHNEIDEN im Menü BEARBEITEN aus (oder klicke auf das Icon  in der Symbolleiste!).

- Vor dem Einfügen musst du in den hierfür gewünschten Bereich wechseln. Einfach mit der Maus auf die Arbeitsfläche im Fenster TOR_AUF klicken.
- Ein weiterer Klick auf den Befehl EINFÜGEN im dem Menü BEARBEITEN genügt, und die zuvor ausgeschnittenen Bausteine können mit der Maus im Fenster TOR_AUF zwischen dem Ein- und Ausgang platziert werden.



Wenn du nun noch die fehlenden Verbindungslinien zu Ein- bzw. Ausgang zeichnest, ist das Unterprogramm fast fertig. Es sollte so aussehen:

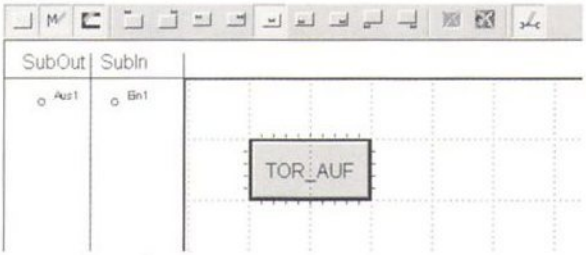
◀

Fast fertig? Ja, denn wenn du im Bausteinfenster auf das Register "Unterprogramme" klickst, siehst du, dass am Unterprogrammbaustein TOR_AUF noch keine Anschlüsse vorhanden sind (klicke zum Vergleich einfach mal auf das Unterprogramm UP1, dann siehst du den Unterschied).

Diese Anschlüsse müssen wir noch hinzufügen, doch dazu mehr im nächsten Kapitel.

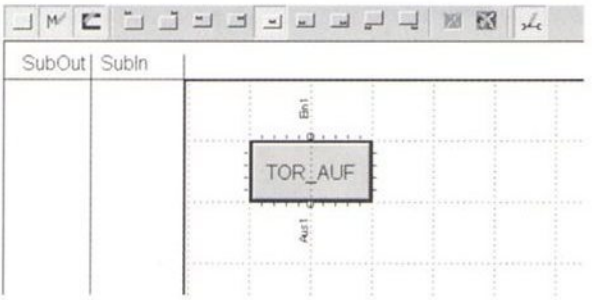
4.3 Bearbeiten von Unterprogrammbausteinen (Unterprogramm-Design)

Unter dem Menüpunkt UNTERPROGAMM findest du den Eintrag DESIGN. Mit diesem Befehl öffnet sich zunächst eine Auswahlbox, in der du angeben musst, welchen Unterprogrammbaustein du bearbeiten möchtest. Nach der Bestätigung mit OK öffnet sich ein neues Fenster, in dem du die Eigenschaften des ausgewählten Unterprogrammbausteins verändern kannst.



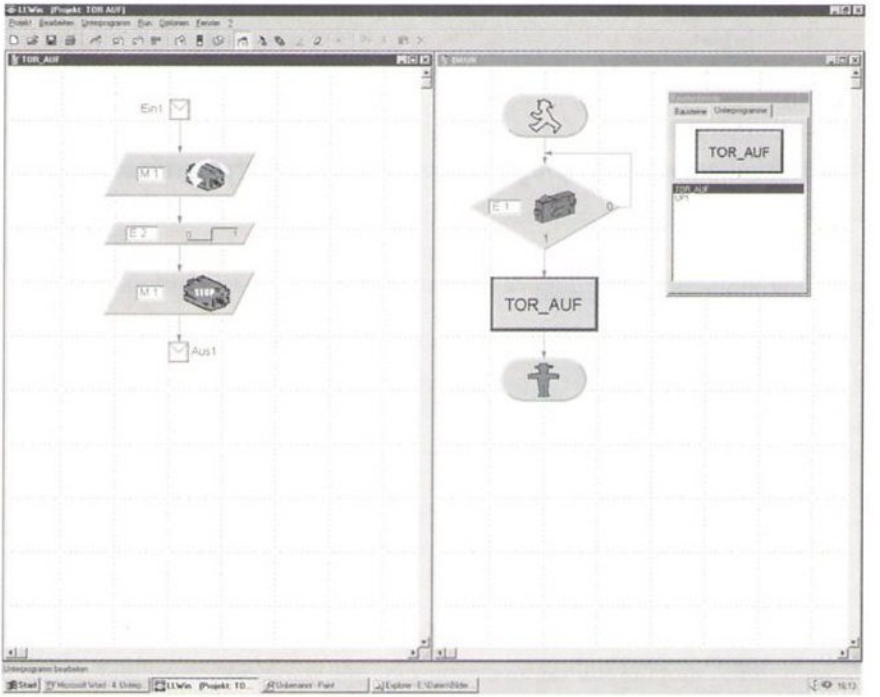
Die im Ablaufdiagramm eingefügten Bausteine SubIn und SubOut sind in der links angeordneten Tabelle als Kreis abgebildet. Diese musst du nun mit der Maus greifen (linke Maustaste gedrückt halten) und auf dem Bausteinrahmen an der gewünschten Position absetzen. Die Kreise stellen die Anschlüsse dar, die den Unterprogrammbaustein mit den anderen Bausteinen im Hauptprogramm verbinden.

Der fertige Baustein sieht dann so aus: ▶



Hinweis: Weitere Möglichkeiten, die Eigenschaften des Unterprogrammbausteins zu ändern, sind im Kapitel 8.3 ausführlich beschrieben. Am Besten dort nachlesen.

Nachdem du die Anschlüsse eingefügt hast, kannst du dieses Bearbeitungsfenster schließen, indem du auf das X in der Menüleiste (nicht in der Titelleiste) klickst. Es erscheint wieder das Bearbeitungsfenster des Hauptprogramms $MAIN. Hier kannst du nun aus dem Bausteinfenster das Unterprogramm "TOR_AUF" in den Ablaufplan zur Garagentorsteuerung einfügen. Wenn du die Fenster "$MAIN" und "TOR_AUF" nebeneinander darstellst, müsste das fertige Projekt so aussehen:



Du siehst, LLWin erzeugt für das Hauptprogramm, jedes Unterprogramm und Unterprogrammdesign ein separates Fenster. Zum Wechseln und Anordnen können die Befehle des Menüs FENSTER benutzt werden. Mit etwas Übung behältst du so beim Erstellen deiner Steuerungsprogramme sehr leicht die Übersicht zwischen dem Hauptprogramm und den verschiedenen Unterprogrammen.

Nun wollen wir unser Programm testen. Schalte dazu in den Init-Modus um. Wenn du jetzt im Menü BEARBEITEN nachsiehst, wirst du feststellen, dass nur noch das Fenster "$MAIN" geöffnet ist. Im Init-Modus kannst du aber sehr leicht in das Unterprogramm wechseln, indem du mit der linken Maustaste auf den Unterprogrammbaustein klickst. Du landest eine Ebene tiefer im Ablaufplan des Unterprogramms. Zurück geht's mit einem Klick auf die rechte Maustaste. Praktisch, nicht?

Außerdem kannst du auch im Init-Modus mehrere Fenster öffnen. Jedes Mal, wenn du, z. B. in der Symbolleiste, auf "Init" drückst, wird ein neues Fenster geöffnet. Die verschiedenen Fenster kannst du wieder mit den Befehlen aus dem Menü Fenster gleichzeitig am Bildschirm anordnen. Klickst du nun in einem Fenster auf das Unterprogramm "TOR_AUF", kannst du nachher, wenn das Programm läuft, gleichzeitig das Haupt- und das Unterprogramm betrachten.

Starte das Programm mit dem Icon  . Wieder läuft der Motor, sobald du E1 drückst und hält an, sobald du E2 betätigst. Probiere ruhig auch noch einmal den Downloadbetrieb aus, das Programm funktioniert dort genauso, nur wird am Bildschirm nichts angezeigt. Und nicht vergessen: zur Rückkehr in den Online-Betrieb die Stromversorgung am Interface kurz unterbrechen.

Hinweis: Unterprogramme lassen sich im Editiermodus auch sehr leicht umbenennen und löschen. Die Vorgehensweise hierzu ist in Kapitel 8.3 ausführlich beschrieben. Dort ist auch beschrieben, wie Unterprogramme im gleichen Projekt kopiert bzw. unter neuem Namen vervielfältigt werden können.

4.4 Unterprogramme in andere Projekte kopieren

Wie wir bereits zu Beginn festgestellt haben, bieten Unterprogramme die Möglichkeit, dass du sie auch in anderen Projekten verwenden kannst. Du kannst z. B. eine Art Bibliothek erstellen, indem du in einem Projekt nur Unterprogramme mit bestimmten, immer wieder benötigten, Funktionen abspeicherst. Auf diese Funktionen kannst du dann wie beim Bausteinfenster zugreifen, indem du das betreffende Unterprogramm in dein aktuelles Projekt kopierst. Wie? Dazu musst du folgende Schritte ausführen:

- Im Projekt 1, worin sich das Unterprogramm befindet, in das zu kopierende Unterprogramm wechseln (BEARBEITEN-UNTERPROGRAMM).
- Alle Bausteine markieren in die Zwischenablage kopieren.
- LLWin ein zweites Mal starten, damit du bequem zwischen Projekt1 und Projekt2, deinem aktuellen Projekt, wechseln kannst. Dann Projekt2 öffnen oder neu erstellen.
- In Projekt 2 über BEARBEITEN-UNTERPROGRAMM ein neues Unterprogramm mit dem gleichen Namen wie in Projekt1 erstellen.

- Mit "Einfügen" den kopierten Ablaufplan aus der Zwischenablage in das Unterprogramm einfügen.
- Mit UNTERPROGRAMM-DESIGN das Aussehen des Unterprogrammbausteins festlegen. Dieser muss nicht identisch sein mit dem Aussehen des Bausteins in Projekt1. Nur der Name des Unterprogramms muss gleich sein.

Wenn das Unterprogramm, das du kopieren willst, weitere Unterprogramme enthält musst du folgendermaßen vorgehen:

- Zuerst musst du alle Unterprogramme, die in dem Ablauf, der kopiert werden soll, enthalten sind, von Projekt 1 nach Projekt 2 kopieren (s.o.)
- In Projekt 1 alle Bausteine (auch die Unterprogrammbausteine), die kopiert werden sollen, markieren und in die Zwischenablage kopieren.
- In Projekt 2 wechseln und den Ablauf aus der Zwischenablage einfügen.

Achtung: Enthält der Ablauf, der kopiert wird, Unterprogramme, die es in Projekt 2 nicht gibt, werden diese Unterprogramme nicht mit kopiert. Gibt es in Projekt 2 ein Unterprogramm mit gleichem Namen aber mit anderem Inhalt als in Projekt 1, wird der Unterprogrammbaustein mit dem Inhalt des Unterprogramms von Projekt 2 eingefügt.

Was sich hier zunächst etwas kompliziert anhört, ist spätestens dann, wenn du es ein paar Mal gemacht hast, recht einfach. Deshalb – Übung macht den Meister!

5. Die Verwendung von Variablen im Steuerungsprogramm

Sicherlich wirst du dich jetzt fragen, wofür du Variablen in einem Steuerungsprogramm benötigst. Was sind Variablen überhaupt? Nun, Variablen sind Zwischenspeicher für bestimmte Informationen. Welche Informationen? Das kann z. B. die Information darüber sein, ob ein Taster gedrückt ist oder nicht. In diesem Fall besitzt die Variable zwei verschiedene Werte. Die Information kann aber auch sein, wie oft ein bestimmtes Ereignis aufgetreten ist. In diesem Fall wird die Variable dazu verwendet, um die Häufigkeit eines Ereignisses zu zählen und zwischenzuspeichern. Beides wollen wir in den nächsten Kapiteln praktisch einsetzen. Zuvor jedoch halten wir etwas Grundlegendes für den Umgang mit Variablen fest:

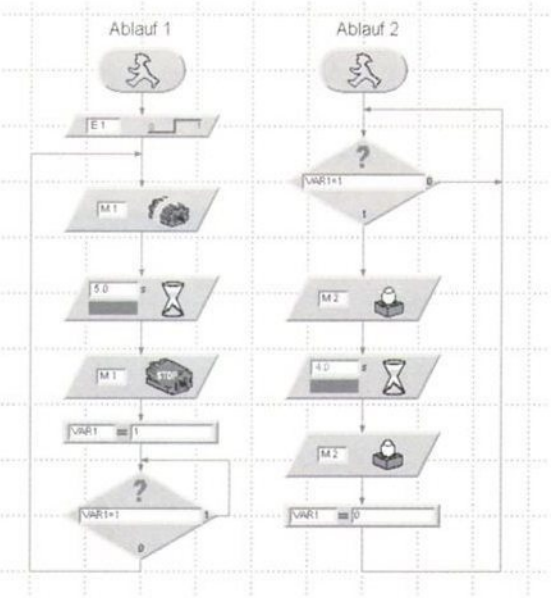
Variablen besitzen zum Einen einen festen Namen, über den man auf die Variable zugreifen kann. Zum Anderen haben sie einen Inhalt oder Wert, der veränderbar oder variabel ist. Daher auch der Name "Variable"!

In einem LLWin-Projekt kannst du maximal 99 verschiedene Variablen verwenden, die mit VAR1-VAR99 bezeichnet werden. Der Wert einer Variablen, eine ganze Zahl, kann zwischen -32567 bis +32567 liegen. Während ein Programm im Online-Modus läuft, können die Werte der Variablen am Bildschirm angezeigt werden.

5.1 Variablen zum gegenseitigen Steuern zweier Abläufe

Bisher haben wir bei unseren Steuerungsprogrammen immer nur einen Ablaufplan erstellt. Nun kannst du innerhalb deines Steuerungsprogramms auch mehrere Ablaufpläne erstellen, die dann gleichzeitig oder parallel, wie man das in der Fachsprache nennt, durchlaufen werden.

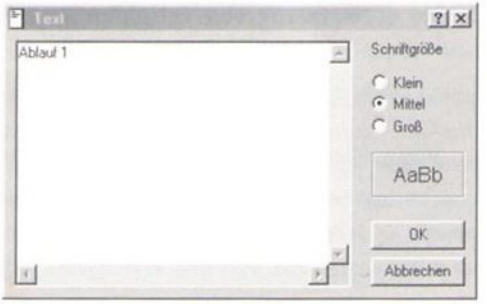
Meistens wird es aber so sein, dass die beiden Abläufe nicht völlig unabhängig voneinander sind. Zum Beispiel kann bei einem Ablauf an einer Stelle erst dann fortgefahren werden, wenn beim zweiten Ablauf eine bestimmte Bedingung oder ein Zustand erreicht ist. Dieses gegenseitige Steuern von Abläufen lässt sich sehr einfach mit Variablen realisieren. Erstelle dir hierzu in einem neuen Projekt den folgenden Ablaufplan: ▶



Wie du siehst sind es genau genommen 2 Ablaufpläne, die jeweils mit dem Baustein START beginnen. Die Ablaufpläne enthalten keinen Baustein ENDE, da jeder Ablauf nach Durchlaufen der einzelnen Bausteine wieder von vorne beginnen soll. In den Ablaufplänen benötigst du drei neue Bausteine:

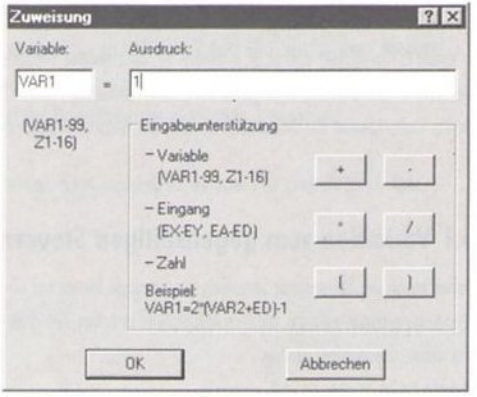
● **Text**

Der Textbaustein wird benötigt um den Ablaufplan beschriften zu können. Den im Dialog eingegebenen Text kannst du an einer beliebigen Stelle des Arbeitsblattes platzieren. Die Schriftgröße kannst du ebenfalls im Dialog einstellen.



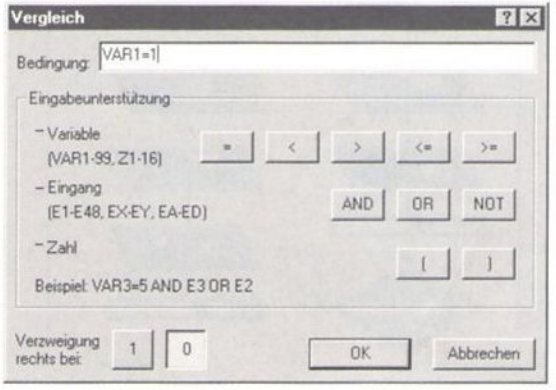
● **Zuweisung**

Mit diesem Baustein kann einer Variablen ein bestimmter Wert zugewiesen werden. Im Bausteindialog werden folgende Eingaben vorgenommen: Im linken Feld der Variablenname Var1, im rechten Feld der zugewiesene Wert (1 bzw. 0).
Die Buttons in der Eingabeunterstützung benötigen wir an dieser Stelle noch nicht.



● **Vergleich**

In den Baustein VERGLEICH wird eine Bedingung eingetragen. Abhängig davon, ob die Bedingung erfüllt ist oder nicht, verzweigt der Ablauf nach rechts oder wird am unteren Ausgang des Bausteins fortgesetzt. Die Zahlen 0 und 1 an den Ausgängen stehen für "Bedingung erfüllt" (1) bzw. "Bedingung nicht erfüllt" (0).



Die Bedingung ist in unserem Fall, ob der Wert der Variablen "0" oder "1" ist. Erst wenn im 2. Ablauf der Wert der Variablen "Var 1" auf "1" gesetzt wird, soll im 1. Ablauf fortgefahren werden. Deshalb lautet die Weiterschaltbedingung "Var1=1".
Sie wird im Bausteindialog als Formel eingegeben:
Das "=" Zeichen kannst du entweder über die Tastatur oder per Mausklick über die Eingabeunterstützung eingeben.

Hinweis: Das Arbeiten mit Formeln wird in Kapitel 6 noch ausführlicher beschrieben.

Im zweiten Ablauf wählst du in den beiden Bausteinen AUSGANG als Typ die Lampe aus.
Im Baustein WARTE des zweiten Ablaufs stellst du den Typ der Wartezeit auf "zufällig", für eine zufällig lange Wartezeit zwischen 1 und 4 Sekunden.

Wenn im Init-Modus keine Fehler mehr festgestellt werden, kannst du das Programm wie gewohnt im "Online-Betrieb" testen.

Der Motor an M1 startet, sobald an E1 eine Flanke von "0" nach "1"auftritt, und läuft anschließend 5 Sekunden lang. Danach wird "Var1" auf "1" gesetzt und die Lampe an M2 wird für eine zufällig lange Zeit zwischen 0 und 4 Sekunden eingeschaltet. Danach wird "VAR1" zurück auf "0" gesetzt und der Motor geht wieder an. Während die Ablaufpläne abgearbeitet werden, kannst du am Bildschirm über die rot markierten Bausteine sehr gut erkennen, wie über die Variable die beiden Abläufe gesteuert werden.

Hinweis: Für dieses gegenseitige Steuern kann auch eine andere Funktion von LLWin genutzt werden. Sie nennt sich "Abfragen der Motorzustände". Näheres hierzu findest du in Kapitel 10!

5.2 Variablen zum Zählen von Impulsen

Sehr häufig werden Variablen auch zum Zählen verwendet. Dies wollen wir uns an einem praktischen Beispiel näher betrachten. Wenn ein Roboter seinen Greifer immer genau zwischen zwei Ablagepunkten bewegen soll, dann ist es erforderlich, den Weg elektronisch zu messen. In den fischertechnik-Modellen werden die Motoren mit speziellen Zahnrädern (sog. Impulsrädern) verbunden, die einen Taster bei jeder Umdrehung 4 mal öffnen und schließen. Solche Signale nennt man in der Elektrotechnik auch Impulse.

Das Zählen der Impulse übernimmt in LLWin der Baustein POSITION. 

Dieser enthält einen Zähler, der mit Hilfe einer Zählvariablen bis zu einem einstellbaren Endwert zählt. Damit du die Funktionsweise dieses Bausteins besser verstehst, wollen wir ihn durch andere Bausteine selber aufbauen (wir schreiben sozusagen ein eigenes Programm POSITION). Dabei wollen wir auch gleich ein paar neue Bausteine für unsere Steuerungsprogramme kennenlernen:

- Für dieses Beispiel benötigen wir kein Interface. Als Eingangssignal verwenden wir einen Schalter vom Baustein TERMINAL, 

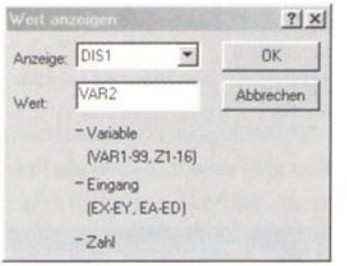
denn ein Impulsrad an einem Motor würde den Taster so schnell öffnen und schließen, dass es nicht möglich wäre, im Ablaufplan die Funktion zu verfolgen. Mit dem Terminal hingegen können wir, während das Programm läuft, bestimmte Werte eingeben (Terminalparameter), verschiedene Werte anzeigen (Display), Meldetexte ausgeben und mit Schaltern

digitale Eingangssignale erzeugen. Der Baustein wird ohne Verbindungslinien zu anderen Bausteinen auf dem Arbeitsblatt platziert. Eine genau Beschreibung des Bausteins findest du in Kapitel 7.9.

- Mit Hilfe des Bausteins MELDUNG  wollen wir die Meldung "Endwert erreicht" am Terminal ausgeben.

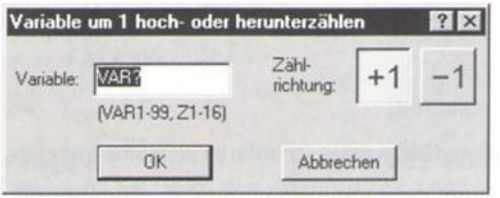
- Mit dem Baustein DISPLAY 

lässt sich auf einem der beiden Displays des Terminals ein Wert oder eine Variable ausgeben. Im Bausteindialog wird eingestellt, welches Display verwendet und welcher Wert (Var1) dort angezeigt wird.

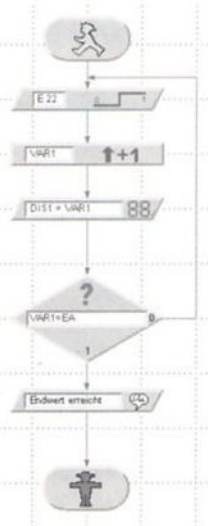


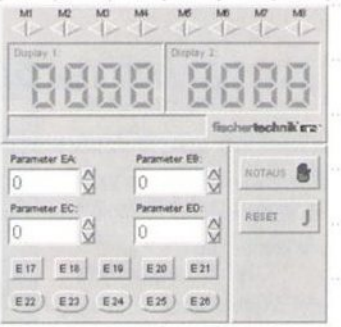
- Der Baustein VARIABLE +/-1 

dient dazu, den Wert einer Variablen um eins zu erhöhen oder zu verringern. Die Variable und die Zählrichtung muss im Bausteindialog angegeben werden.

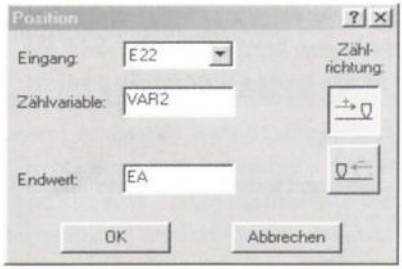


Damit bist du nun in der Lage, den folgenden Ablaufplan zu erstellen. Im Baustein EINGANG musst du den Eingang E22 angeben. Dies ist der Taster E22 auf dem Terminal. Im unteren Baustein VERGLEICH wollen wir zur Eingabe des Endwerts anstatt einer Zahl den Terminalparameter EA verwenden. Diesem Parameter kann man dann im Baustein TERMINAL einen Wert, z. B. "4", zuweisen und diesen Wert dann während das Programm läuft verändern.





Nun kannst du mit <F8> den Befehl INIT ausführen. Klicke mit der Maus auf das Eingabefeld des Terminalparameters EA im Baustein TERMINAL, gebe den Wert 4 ein und bestätige mit <ENTER>. Sind alle Bausteine richtig verbunden, dann starte das Programm mit einem Klick auf das Start-Icon. Beobachte die Displayanzeige im Baustein TERMINAL. Bei jedem Klick auf den Schalter E22 im TERMINAL wird der Wert um eins erhöht. Ist der Endwert erreicht, so erscheint im TERMINAL die entsprechende Meldung! Das bedeutet, mit der Variablen "Var1" werden die Impulse (Mausklicks) gezählt. Beim Roboter wären dies die Impulse der Impulsräder. Über Zählvariable und Endwert kann somit immer eine genaue Position angefahren werden (Motor nach Erreichen des Endwerts abschalten).

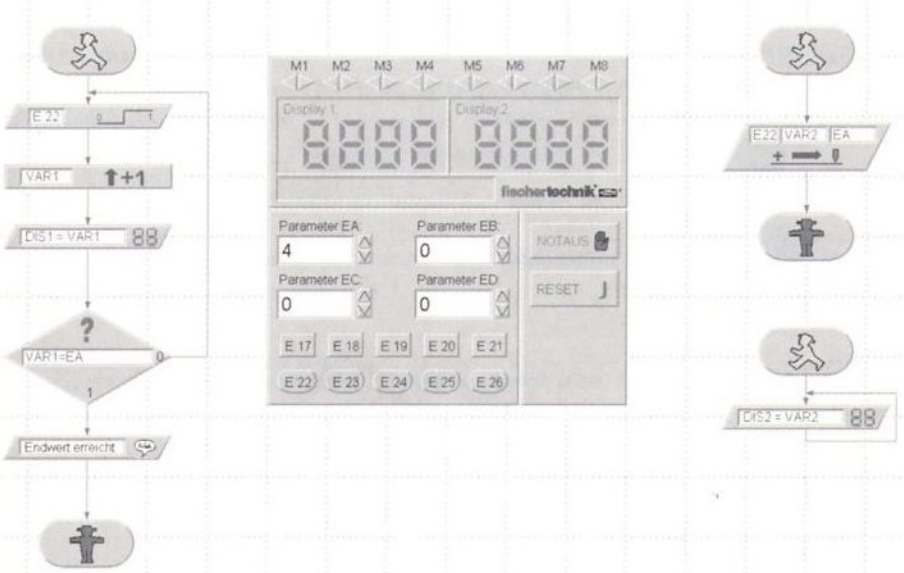


Nun wollen wir unseren eigenen Impulszähler mit dem Baustein POSITION vergleichen. Füge deshalb deinem Projekt einen neuen Ablaufplan hinzu, in dem du nur den Baustein POSITION zum Zählen der "gleichen" Impulse (Mausklicks) verwendest.

Im Bausteindialog musst du hierfür als Eingang E22 einstellen. Als Zählvariable verwendest du "VAR2" und der Endwert ist natürlich ebenfalls EA.

Hinweis: Im Baustein POSITION kann jedem Zähleingang E1 – E16 eine Standardzählvariablen Z1 – Z16 zugeordnet werden. Hierzu musst du im Bausteindialog den Button "Standardzählvariable" aktivieren. Dann wird für E1 automatisch Z1, für E2 Z2 usw. verwendet. Du brauchst du dir also keine Gedanken mehr darum zu machen, welche Variable du nun für den Zählvorgang verwendest und ob diese Variable nicht eventuell schon anderweitig vergeben ist (siehe auch Kapitel 7.4).

Um die Funktionen besser vergleichen zu können, fügst du deinem Projekt noch einen dritten Ablaufplan hinzu, in dem die Variable VAR2 auf dem Display 2 des Terminals anzeigst. Der fertige Ablaufplan sieht dann so aus:

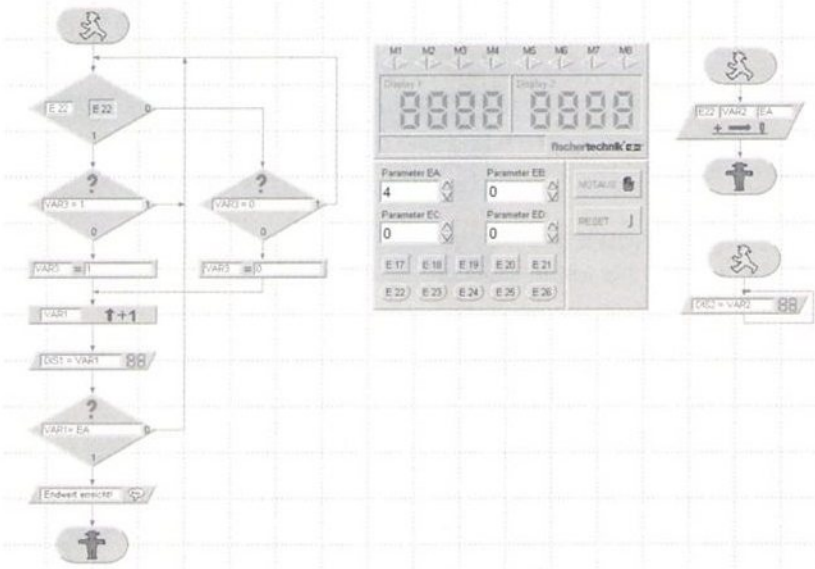


Du kannst übrigens den Terminalparameter EA auch im Bearbeiten-Modus (Bearbeiten Hauptprogramm) eingeben, indem du den Cursor auf das Feld des Parameters bewegst und dann die rechte Maustaste drückst.

Starte das Programm neu und vergleiche, ob die beiden Abläufe nach der gleichen Anzahl von Mausklicks (Impulsen) in den Baustein ENDE schalten.

Wie du sicherlich festgestellt hast, ist dies nicht der Fall. Warum? Nun, der Baustein POSITION registriert sowohl den Übergang von 0 nach 1 als auch von 1 nach 0 als jeweils ein Impuls. Ein Mausklick wird bei unserem selbst erstellten Impulszähler als ein Impuls registriert. Der Baustein POSITION hingegen zählt bei einem Mausklick zwei Impulse, nämlich einen beim Drücken und einen beim Loslassen der Taste. Deshalb schaltet der rechte Ablaufplan bereits nach zwei Mausklicks in den Baustein ENDE!

Das heißt, wir müssen unseren selbst entworfenen Impulszähler so erweitern, dass er sowohl die "0-1-Flanke" als auch die "1-0-Flanke" als einen Impuls registriert. Der Ablaufplan sieht dann so aus:



Schon ziemlich kompliziert, oder? Als Eingang verwenden wir wieder E22. In Var3 muss der Zustand des Eingangs E22 zwischengespeichert werden. Var1 dient als Zählvariable im linken Ablauf, Var2 als Zählvariable im Baustein POSITION. Beobachte die grünen Dreiecke in den Bausteinen EINGANG und VERGLEICH und versuche, den Ablauf Schritt für Schritt nachzuvollziehen. Du siehst, wie umfangreich ein derartiges Zählen von Impulsen sein kann. Da verwenden wir doch lieber den Baustein POSITION!!

5.3 Verarbeiten der Analogeingänge EX und EY mit Variablen

Neben den digitalen Eingängen bietet das Interface auch zwei analoge Eingänge EX und EY. Worin besteht dabei der Unterschied? Nun, mit digitalen Eingängen können nur zwei Zustände ausgewertet werden, nämlich "0" und "1" bzw. Kontakt geschlossen oder Kontakt geöffnet. Möchtest du aber z. B. eine Temperatur messen, benötigst du einen Eingang, mit dem man mehrere Zustände (je nach Temperatur) auswerten kann. Hierfür dienen die beiden Analogeingänge des Interfaces. Der am Interface an EX und/oder EY angeschlossene Widerstand im Bereich von 0...5 KΩ wird beim Intelligent Interface in einen Zahlenwert zwischen 0 und 1024, beim parallelen Interface zwischen 0 und 999, umgewandelt und kann dann im Steuerungsprogramm verwendet werden.

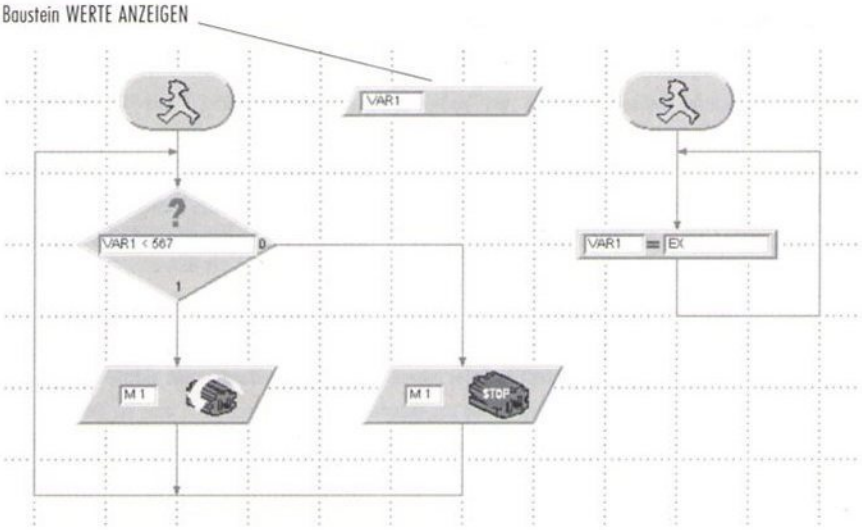
Dies wollen wir gleich praktisch ausprobieren und eine Ventilatorsteuerung für "heiße" Tage entwerfen. Überlegen wir uns zunächst, wie eine solche Steuerung funktioniert! Als Erstes müssen wir mit einem Temperatur- oder Wärmesensor die Temperatur messen. Ist die Temperatur höher als die von uns eingestellte "kritische Temperatur", muss der Ventilator eingeschaltet werden. Sinkt die Temperatur wieder unter den kritischen Punkt, muss der Ventilator abgeschaltet werden. Der Vorgang Messen – Vergleichen – Ventilator aus-/einschalten beginnt dabei immer wieder von vorne, der Ablaufplan bildet also einen Kreis – in der Technik nennt man diese Steuerung deshalb auch einen **Regelkreis**!

Um die Temperatur messen zu können, benötigen wir einen Wärmesensor. Das ist ein temperaturabhängiger Widerstand. Je höher die Temperatur, desto geringer der Widerstand. Wir verwenden dazu den NTC-Widerstand 1,5kΩ Art.-Nr. 36437 von fischertechnik. Solltest du einen solchen Wärmesensor besitzen, schließe ihn zum Test an den Eingang EX des Interfaces an. Über die Interface-Diagnose kannst du feststellen, welchen Wert EX dann bei Zimmertemperatur besitzt. Wenn du den Wärmesensor erwärmst, wirst du feststellen, dass der Wert von EX kleiner wird.

Bei der Erstellung des Ablaufplans ergibt sich allerdings ein kleines Problem! Wie lässt sich der Analogwert des Wärmesensors verwerten? Bei unserem bisher verwendeten Baustein EINGANG können wir nur E1 bis E48, nicht aber EX oder EY einstellen. Probiere es einmal aus! Wenn du jetzt sagst, das könne ja gar nicht sein, hast du vollkommen recht. Denn mit dem Baustein EINGANG lässt sich nur eine "digitale" Abfrage erstellen!! Was tun? Hier hilft uns nur die Verwendung einer Variablen weiter. Wenn wir der Variablen den Wert des Analogeingangs zuweisen, können wir mit dem Baustein VERGLEICH unseren Steuerungsablaufplan erstellen. Zur Anzeige des Werts der Variablen könnten wir das bereits bekannte TERMINAL verwenden. Wir wollen aber noch einen neuen Baustein kennenlernen: Er heißt "Werte anzeigen"

Im Bausteindialog musst du angeben, welche Variable oder welcher Eingang angezeigt werden soll. Der Baustein wird ohne Verbindung zu anderen Bausteinen eingefügt.

Die fertige Ablaufsteuerung sieht dann so aus:



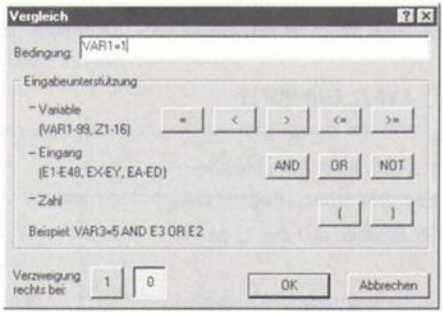
Die Ablaufpläne enthalten drei Besonderheiten:

- Der Ventilator muss bei allen Temperaturen, die höher als unsere kritische Temperatur sind, eingeschaltet werden/bleiben. Die Bedingung im Baustein VERGLEICH kann also nicht lauten "Var1 = 567", sonst würde der Ventilator ja nur bei dieser Temperatur eingeschaltet werden. Wir können für die Bedingung aber auch "Var1<567" eintragen, dann wird der Ventilator tatsächlich bei jeder höheren Temperatur eingeschaltet!
- Wie in den Ablaufplänen gezeichnet, sollte die Zuweisung von Analogwert "EX" und Variablen "VAR1" in einem separaten Ablauf geschehen. Damit wird gewährleistet, dass "VAR1" zu jedem Zeitpunkt den aktuellen Wert von "EX" besitzt. Genaueres hierzu findest du im Kapitel 8.5!
- Zuletzt besitzen beide Ablaufpläne keinen Baustein ENDE, da ja im Sinne des Regelkreises das Programm, einmal gestartet, immer wieder von vorne beginnen soll.

Erstelle den Ablaufplan in einem neuen Projekt und teste das Steuerungsprogramm. Für den Wert der kritischen Temperatur trägst du einfach einen Wert ein, der etwas kleiner ist als der Wert der momentanen Zimmertemperatur. Wenn du nun den Wärmesensor erwärmst, schaltet sich der Ventilator ein, beim Abkühlen wieder aus. Solltest du keinen Wärmesensor besitzen, verwendest du anstelle des Wertes EX den Terminalparameter EA. Dann kannst du die Ventilatorsteuerung simulieren, indem du über EA bestimmte Werte um die kritische Temperatur herum eingibst.

6. Ein klein wenig Mathematik leistet Erstaunliches - Formeln verwenden

Nicht erschrecken, wir machen jetzt keine höhere Mathematik oder ähnliches. Auch wollen wir den Mathematikunterricht nicht durch das LLWin-Handbuch ersetzen! Es geht darum, die Mathematik sehr praktisch in unseren Steuerungsprogrammen anzuwenden. Wie? In den Bausteinen VERGLEICH, NOTAUS und RESET kann im Bausteindialog als Bedingung eine mathematische Formel eingegeben werden.



Eine solche Formel kann aus folgenden Bestandteilen aufgebaut werden:

- Digitale Eingänge (E1 - E48)

Die einfachste Bedingung heißt dann "E1". Dies ist nichts anderes als unser bereits bekannter Baustein EINGANG, bei dem der Ablaufplan, in Abhängigkeit vom Zustand des Eingangs "0" oder "1", verzweigt.

- Variablen

Dazu gehören die Variablen Var1 - Var99 und die Standardzählvariablen aus dem Baustein POSITION Z1-Z16.

- Analoge Eingänge EX und EY.
- Terminalparameter EA, EB, EC und ED.
- Ganze Zahlen von -32567 bis +32567

Im Gegensatz zu den Zahlenwerten der Variablen nennt man diese Zahlen "Konstanten".

- Vergleichsoperatoren

=	gleich
<	kleiner als
>	größer als
<=	kleiner oder gleich als
>=	größer oder gleich als

Den Vergleichsoperator "<" haben wir bereits im letzten Kapitel verwendet. Dort lautete die Bedingung "Var1 < 567"!

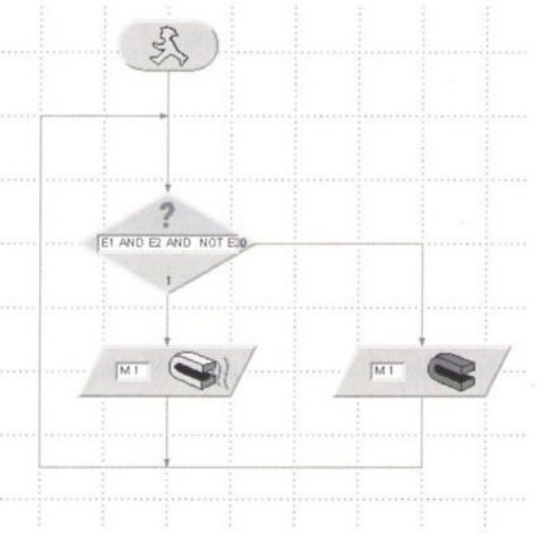
- Verschiedene Ausdrücke können mit AND, OR, NOT (und, oder, nicht) logisch verknüpft werden. Hierzu folgt gleich noch ein Beispiel.
- Schließlich ist auch das Setzen von Klammern ( ) möglich.

An einer kleinen Steuerungsaufgabe wollen wir die Verwendung der AND- und NOT-Verknüpfung praktisch anwenden. In einem Sicherheitsbereich wird die Zugangstür von einem Elektromagneten offengehalten. Zwei Lichtschranken, angeschlossen an E1 und E2, überwachen den Bereich. Sobald eine dieser Lichtschranken unterbrochen wird (Signal "0"), soll die Türe automatisch geschlossen werden (Elektromagnet aus). Zusätzlich soll sich die Tür durch Betätigung eines Notschalters

(Signal "1"), angeschlossen an E3, schließen lassen. Nun könnten wir drei Bausteine vom Typ EINGANG hintereinander schalten und damit die Bedingungen zum Schließen der Tür an den Eingängen abfragen. Wir können es aber auch etwas einfacher machen! Die Bedingung für den eingeschalteten Elektromagneten lautet doch "E1 und E2 und nicht E3". Diese Bedingung können wir beim Baustein VERGLEICH im Bausteindialog folgendermaßen eingeben:

E1 ***AND*** E2 ***AND NOT*** E3

und erhalten damit die Einschaltbedingung für unseren Elektromagneten. Der Ablaufplan sieht dann so aus: ▶



Erstelle diesen Ablaufplan in einem neuen Projekt und teste die Funktion! Du siehst: Ein klein wenig Mathematik kann doch schon Erstaunliches leisten! Du kannst auch Variablen in die Vergleiche mit einbeziehen. So sind z. B. die folgenden Vergleiche durchaus erlaubt:

- Var1 = 5 OR VAR2 = 3
- Var1<2 OR E1

Achtung: Folgende Ausdrücke sind jedoch nicht erlaubt:

Var2>E1
E1=1
E2=0

Der Grund besteht darin, dass zwischen den Zuständen 0 und 1 der digitalen Eingänge und den Zahlenwerten 0, 1, 2, 3,..., die eine Variable annehmen kann, ein gravierender Unterschied beseht. Die Werte 0 und 1 der digitalen Eingänge werden als sog. "Boolsche Werte" verarbeitet, d. h. sie nehmen nur die Zustände "0" (z. B. "Taster nicht gedrückt") oder "1" (z. B. "Taster gedrückt") an. Die Werte 0, 1, 2, 3,..., die Variablen und Analogeingänge annehmen können, sind dagegen ganze Zahlen (in der Fachsprache "Integer" genannt). Mit diesen Werten kannst du rechnen, kannst sie z. B. addieren, subtrahieren, multiplizieren und dividieren, was mit den Boolschen Werten so nicht geht. Deshalb musst du aufpassen, dass du diese verschiedenen Arten von Zahlen (auch "Datentypen" genannt) nicht durcheinander wirfst. Die oben genannten Ausdrücke enthalten daher folgende Fehler:

- **Var2>E1** Die Variable Var2 hat den Wert einer ganzen Zahl, während der Eingang E1 ein Boolscher Wert ist. Diese beiden Werte können nicht miteinander verglichen werden. Bei der Eingabe dieser Formel würde eine Fehlermeldung erscheinen.
- **E1=1, E2=0** Die Zahlen 0 und 1 haben in Verbindung mit einem =Zeichen den Wert der ganzen Zahl 0 oder 1, während der Eingang E1 bzw. E2 ein Boolscher Wert ist. Der Vergleich ist deshalb nicht zulässig. Auch hier erscheint

eine Fehlermeldung. Die richtige Eingabe würde lauten:
anstatt "E1=1" "E1" (z. B. Taster E1 betätigt)
anstatt "E2=0" "NOT E2" (Taster E2 nicht betätigt)

Alles klar? Nicht? Gut, dann wollen wir es auf einen einfachen Nenner bringen:

Digitale Eingänge dürfen nicht mit Variablen, Zahlen oder anderen Eingängen verglichen werden, sonst erscheint eine Fehlermeldung.
Logische Verknüpfungen (AND, OR, NOT) zwischen verschiedenen Eingängen sind hingegen erlaubt (wie in unserem Beispiel: E1 AND E2 AND NOT E3).

Wem das zu kompliziert ist, der soll sich mit einfachen Bedingungen wie z. B. Var5>2 begnügen. Die Möglichkeit, komplexe Formeln einzugeben, wurde vor allem für die Profis unter den LLWin-Programmierern aufgenommen.

Hinweise: Formeln lassen sich aber nicht nur zum Aufstellen von Bedingungen verwenden. Im Baustein ZUWEISUNG kann der Variablen auch ein Wert zugewiesen werden, der Ergebnis einer Berechnung ist, z. B. Var1 = (Var2 + 3)/5.

7. Die Bausteine in LLWin 3.0

Alle Bausteine, die in LLWin 3.0 zur Verfügung stehen, sind in der Reihenfolge beschrieben, in der sie im Bausteinfenster abgebildet sind.

7.1 Ausgang

Mit dem Baustein "Ausgang" schaltet man einen der Ausgänge M1-M4 des Interface (bei zwei Interfaces M1-M8). An einem Ausgang am Interface kann entweder ein Motor, eine Lampe oder ein Elektromagnet angeschlossen sein. Deshalb kann man auch beim Einfügen des Bausteins im Dialogfeld "Typ" zwischen diesen drei Typen auswählen. Das Symbol des gewählten Typs wird im Baustein abgebildet.

Außerdem stellt man im Dialog den gewünschten Zustand des Ausgangs ein:
Bei einem Motor: links, rechts oder aus.
Bei einer Lampe oder einem Elektromagneten: ein oder aus.

Der eingestellte Zustand wird ebenfalls im Bausteinsymbol abgebildet.

Lampe (aus)

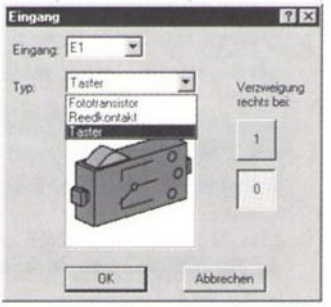
Elektromagnet (ein)

Motor (links)

7.2 Eingang

Der Baustein "Eingang" fragt den Zustand eines digitalen Eingangs E1-E8 am Interface ab (bei zwei Interfaces E1-E16). Ein digitaler Eingang kann nur zwei Zustände annehmen, nämlich 0 und 1. An den digitalen Interfaceeingängen schließt man folgende digitale fischertechnik Sensoren an:
Taster: gedrückt – nicht gedrückt
Fototransistor: hell – dunkel
Reedkontakt (Magnetsensor): geschaltet – nicht geschaltet

Einen dieser Sensortypen kann man beim Einfügen des Bausteins im Dialog auswählen. Das Symbol des Sensors wird dann im Baustein abgebildet.



Taster

Fototransistor

Reedkontakt

Je nach Zustand (0 oder 1) des Eingangs kann der Ablauf nach rechts verzweigen oder am unteren Anschluss des Bausteins fortgesetzt werden. Ob die Verzweigung nach rechts bei 0 oder 1 erfolgen soll, kann im Dialog ausgewählt werden.

Außer den Interfaceeingängen E1-E16 stehen in LLWin noch weitere digitale Eingänge zur Verfügung:

E17-E26:
Schalter und Taster am Terminal, die mit der Maus gedrückt werden können. Als Bild erscheint ein gedrückter Terminaltaster im Baustein.

Terminaleingang

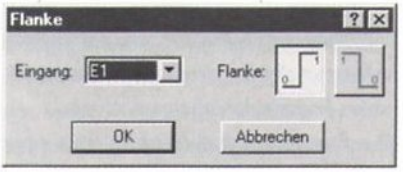
E31-E38, E41-E48:
In diesen Eingängen speichert LLWin intern die Drehrichtung der Motoren, die am Interface betrieben werden. E31-E38 stehen für die linke Drehrichtung der Motoren M1-M8, E41-E48 für die rechte Drehrichtung von M1-M8.

Motordrehrichtung

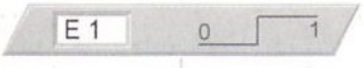
Dreht sich z. B. der Motor M1 nach links, dann ist der Eingang E31 gesetzt (1) E41 ist 0. Dreht sich M2 nach rechts, dann ist E42 gesetzt (1) usw. Es besteht damit die Möglichkeit, in Abhängigkeit der Motordrehrichtung den Programmablauf zu verzweigen. Als Bild erscheint im Baustein ein Motor, der sich nach rechts oder links dreht (siehe auch Kapitel 10.3 Rücklesen der Motorzustände).

7.3 Flanke

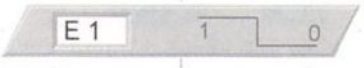
Der Baustein Flanke wartet darauf, dass ein digitaler Eingang entweder von 0 auf 1 oder von 1 auf 0 wechselt. Diesen Übergang nennt man "Flanke"
Im Dialog des Bausteins kann eingestellt werden, auf welche Flanke (1-0 oder 0-1) an einem digitalen Eingang gewartet werden soll.



Die Art der ausgewählten Flanke wird im Bausteinsymbol dargestellt.

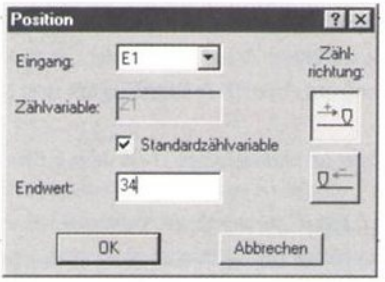


0-1 Flanke



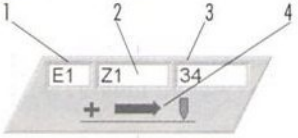
1-0 Flanke

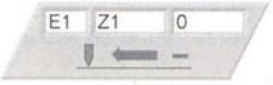
7.4 Position



Der Baustein Position zählt Impulse an einem digitalen Eingang so lange, bis der gewünschte Endwert erreicht ist. Gezählt wird jede Flanke am Eingangs, d. h. wird ein Taster gedrückt und wieder losgelassen, werden zwei Impulse gezählt.
Der aktuelle Zählwert wird in der Zählvariable festgehalten. Aktiviert man im Dialog den Button "Standardzählvariable" wird dem ausgewählten digitalen Zähleingang E1-E16 automatisch eine Zählvariable Z1-Z16 zugeordnet.
Dann muss man nicht lange überlegen, welche Variable man für diesen Zählvorgang verwenden soll, und ob diese schon anderweitig vergeben ist.
Die 16 Zählvariablen können auch von allen anderen Bausteinen, in denen Variablen vorkommen, verarbeitet werden (z. B. Vergleich). Will man trotzdem lieber eigene Zählvariablen vergeben, deaktiviert man die Funktion "Standardzählvariable" und trägt im Feld "Zählvariable" eine der Variablen Var1 bis Var99 ein, die in LLWin zur Verfügung stehen.

Schließlich stellt man im Dialog noch die Zählrichtung ein. Bei "+" wird der Wert der Zählvariable mit jedem Impuls um 1 erhöht, bei "_" um 1 verringert.





Position, Zählrichtung +

Position, Zählrichtung -

Im Bausteinsymbol werden der Zähleingang (1), die Zählvariable (2), der Endwert (3) und die Zählrichtung (4) dargestellt.

Am Häufigsten wird der Baustein Position dazu verwendet, einen Motor in eine bestimmte Position zu fahren. Man schaltet den Motor ein, zählt über einen Taster, der von einem Impulsrad gedrückt wird, die Impulse und schaltet den Motor ab, nachdem der gewünschte Endwert erreicht wurde.
Will man eine Zählvariable wieder auf Null setzen, verwendet man dazu den Baustein "Zuweisung" und trägt dort als Formel z. B. Z1=0 ein.

7.5 Start

Ein Ablaufplan beginnt immer mit einem Startbaustein. Fehlt dieser Baustein am Anfang, wird der Ablauf nicht abgearbeitet. Wenn ein Projekt mehrere Abläufe enthält, muss jeder dieser Abläufe einen Startbaustein enthalten. Die verschiedenen Abläufe werden dann gleichzeitig gestartet.

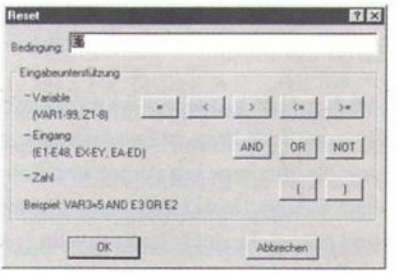
7.6 Ende

Soll ein Ablauf beendet werden, verbindet man den Ausgang des letzten Bausteins mit dem Ende-Baustein. Ein Ablauf kann auch an verschiedenen Stellen mit einem Ende-Baustein beendet werden. Es besteht auch die Möglichkeit mehrere Ausgänge mit einem einzigen Ende-Baustein zu verbinden. Es kann aber auch durchaus vorkommen, dass ein Ablauf als Endlosschleife ausgeführt wird und kein Ende-Symbol enthält.

7.7 Reset

Der Baustein Reset setzt alle Abläufe eines Projekts zurück auf Start, sobald die im Dialog eingegebene Bedingung erfüllt ist.

Der Reset-Baustein wird ohne Verbindungslinien zu anderen Bausteinen auf der Programmieroberfläche platziert.



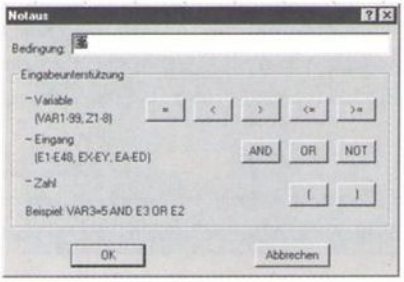
In einem Projekt darf nur ein Reset-Baustein verwendet werden.
Beim Einfügen des Bausteins gibt man im Dialogfenster die Bedingung ein, bei der Reset ausgelöst wird. Die einfachste Reset-Bedingung kann z. B. "E1" heißen. Dann wird Reset ausgelöst, sobald diese Bedingung erfüllt ist, E1 also 1 wird (z. B. indem ein Taster, der am Interface angeschlossen ist, gedrückt wird).
Das Programm startet wieder von vorne, sobald die Bedingung nicht mehr zutrifft, E1 also wieder 0 wird (z. B. indem man den Taster loslässt).
So kann z. B. ein Projekt, das auf dem Intelligent Interface im Download-Betrieb ohne Verbindung zum PC läuft, über einen digitalen Eingang angehalten und neu gestartet werden, ohne dass man es löschen muss.

Als Bedingung kann im Dialog auch eine mathematische Formel eingegeben werden, z. B. Var5 = 6. In der Eingabeunterstützung des Dialogs können die Vergleichsoperatoren, die logischen Verknüpfungen und die Klammern direkt angeklickt werden. Die ganze Formel kann aber auch komplett über die Tastatur eingetippt werden. Die maximale Länge der Formel beträgt 40 Zeichen. Längere Formeln können nicht bearbeitet werden. Bereits vor Erreichen der maximalen Länge kann die Formel nicht mehr komplett im Anzeigefenster des Bausteins abgebildet werden. Sie wird einfach abgeschnitten, aber dennoch richtig berechnet. Im Dialog, der bei bereits eingefügten Bausteinen über die rechte Maustaste aktiviert wird, ist die komplette Formel zu sehen.
Gibt man in eine Formel einen Ausdruck ein, der mathematisch nicht korrekt bzw. nicht erlaubt ist, erscheint eine Fehlermeldung sobald man den Dialog mit OK bestätigen will. Um detailierte Hilfe zu erhalten, kann man dann durch Drücken der Funktionstaste F1 auf der Tastatur zu dem entsprechenden Thema in der Online-Hilfe gelangen.

Hinweis: Die Verwendung von Formeln wird im Kapitel 6 ausführlicher besprochen. Auf jeden Fall lesen! Das hilft Fehler zu vermeiden.

7.8 Notaus

Der Baustein Notaus schaltet, wenn er ausgelöst wird, alle Ausgänge am Interface ab.
Er wird ohne Verbindungslinien zu anderen Bausteinen auf der Programmieroberfläche platziert.
In einem Projekt darf nur ein Notaus-Baustein eingefügt werden.
Beim Einfügen des Bausteins gibt man im Dialogfenster die Bedingung ein, bei der Notaus ausgelöst wird.
Eine einfache Notaus-Bedingung kann z. B. "E1" heißen.



Dann werden alle Motoren abgeschaltet, sobald diese Bedingung erfüllt ist, E1 also 1 wird (z. B. indem ein Notaus-Schalter, der am Interface angeschlossen ist, gedrückt wird). Das Programm läuft unterdessen weiter.
Die Motoren werden wieder eingeschaltet, sobald die Bedingung nicht mehr zutrifft, E1 also wieder 0 wird (z. B. indem man den Taster loslässt).

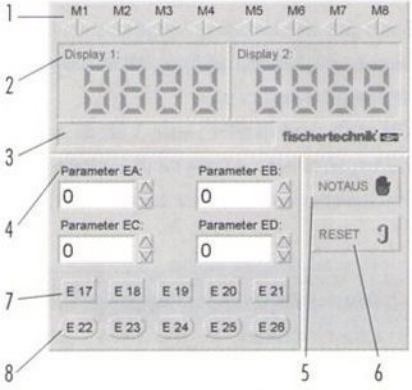
Als Bedingung kann im Dialog auch eine mathematische Formel eingegeben werden, z. B. Var5 = 6. In der Eingabeunterstützung des Dialogs können die Vergleichsoperatoren, die logischen Verknüpfungen und die Klammern direkt angeklickt werden. Die ganze Formel kann aber auch komplett über die Tastatur eingetippt werden. Die maximale Länge der Formel beträgt 40 Zeichen. Längere Formeln können nicht bearbeitet werden. Bereits vor Erreichen der maximalen Länge kann die Formel nicht mehr komplett im Anzeigefenster des Bausteins abgebildet werden. Sie wird einfach abgeschnitten, aber dennoch richtig berechnet. Im Dialog, der bei bereits eingefügten Bausteinen über die rechte Maustaste aktiviert wird, ist die komplette Formel zu sehen.
Gibt man in eine Formel einen Ausdruck ein, der mathematisch nicht korrekt bzw. nicht erlaubt ist, erscheint eine Fehlermeldung sobald man den Dialog mit OK bestätigen will. Um detaillierte Hilfe zu erhalten, kann man dann durch Drücken der Funktionstaste F1 auf der Tastatur zu dem entsprechenden Thema in der Online-Hilfe gelangen.

> Hinweis: Die Verwendung von Formeln wird im Kapitel 6 ausführlicher besprochen. Auf jeden Fall lesen! Das hilft Fehler zu vermeiden.

7.9 Terminal

Der Terminal-Baustein dient während des Programmablaufs dazu, bestimmte Werte anzuzeigen und einzugeben.

Auch das Terminal wird ohne Verbindungslinien zu anderen Bausteinen auf der Programmieroberfläche platziert.
Ganz oben im Baustein wird angezeigt, welche der Interface-Ausgänge M1-M8 eingeschaltet sind und in welche Richtung sich die Motoren, die am Interface angeschlossen sind, drehen (1).



Darunter befinden sich zwei Displays (2), in denen aktuelle Werte von Variablen, Analogeingängen, Parametern EA-ED oder Zahlen angezeigt werden können. Damit die Werte angezeigt werden können, muss man einen Baustein "Display", in dem die Werte festgelegt werden, in den Ablauf einfügen.
Im Feld darunter (3) kann ein Text mit einer Länge von bis zu 17 Zeichen eingeblendet werden. Der Text wird im Baustein "Meldung" festgelegt, der in einen Ablauf eingefügt wird und sobald er aktiv ist, die eingegebene Meldung in diesem Feld einblendet.

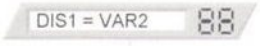
Die 4 Parameter EA-ED (4) sind Platzhalter für Werte, die man jederzeit verändern kann, auch während das Programm läuft (außer natürlich im Download-Betrieb). So z. B. kann man im Baustein Position festlegen, dass ein Motor immer bis zum Endwert "EA" fahren soll. Den Wert EA stellt man dann im Terminal ein und kann ihn während das Programm läuft beliebig oft korrigieren.

Mit den Buttons Notaus (5) und Reset (6) können diese beiden Funktionen per Mausclick ausgelöst werden, aber natürlich nur, wenn das Programm im Online-Modus (gestartet mit Run – Start) läuft. Läuft das Projekt im Downloadbetrieb auf dem Intelligent Interface (gestartet mit Run – Download), muss man die entsprechenden Bausteine Notaus und Reset verwenden.

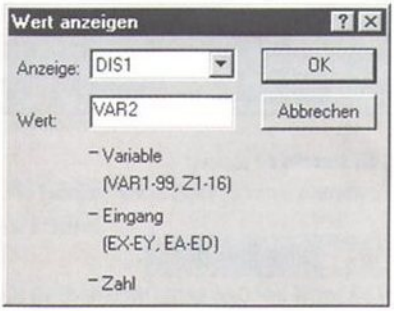
Das Terminal besitzt 10 digitale Eingänge E17-E26, die mit der linken Maustaste bedient werden können. Die Eingänge E17-E21 (7) sind als Schalter ausgelegt und bleiben nach Loslassen der Maustaste gedrückt. Um sie wieder auszuschalten muss man erneut mit der Maus darauf klicken. Die Eingänge E22-E26 funktionieren als Taster, die nach Loslassen der Maustaste sofort in ihren Ausgangszustand zurückspringen. Die Eingänge E17-E21 können vor dem Programmstart in Run - Init voreingestellt werden (gedrückt oder nicht gedrückt). Die Taster E22-E26 hingegen können nur im Online-Modus betätigt werden.

Die Parameter und Schalter des Terminals können im "Bearbeiten"-Modus über die rechte Maustaste aktiviert und geändert werden.

7.10 Display

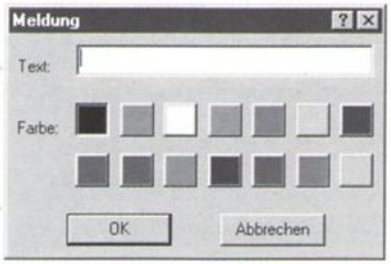


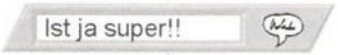
Der Baustein Display dient dazu, einen Wert, eine Variable oder einen der Eingänge EX – EY bzw. EA – ED auf einem der beiden Displays des Terminals anzuzeigen. Beim Einfügen des Bausteins wählt man im Bausteindialog aus, welches der beiden Displays DIS1 oder DIS2 man verwenden will und welcher Wert dort angezeigt werden soll.



7.11 Meldung

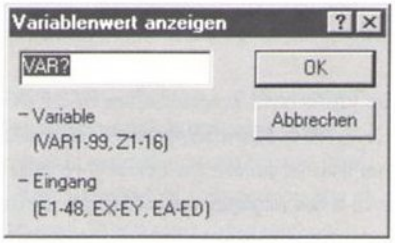
Der Baustein Meldung kann einen Text mit einer Länge von maximal 17 Zeichen im Textfeld des Terminals anzeigen. Beim Einfügen des Bausteins "Meldung" gibt man im Dialogfeld den gewünschen Text ein und kann die Farbe, in der die Meldung angezeigt wird, festlegen.





7.12 Werte anzeigen

Dieser Baustein zeigt, während das Programm im Online-Betrieb läuft, stets den aktuellen Wert einer Variablen an. Der Baustein wird ohne Verbindungen zu anderen Bausteinen auf der Programmieroberfläche eingefügt.

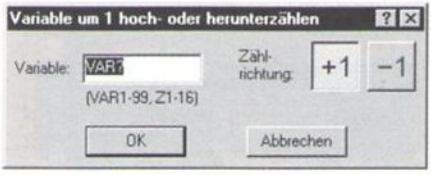


Im Dialogfeld wird eingegeben, welche Variable (VAR1-VAR99, Z1-Z16) angezeigt werden soll. Außer Variablenwerten können auch Werte digitaler und analoger Eingänge, sowie die aktuellen Werte der Parameter EA-ED ausgegeben werden.

Es besteht zudem die Möglichkeit, auch während das Programm läuft eine andere Variable in dem Baustein anzuzeigen. Dazu ruft man mit der linken Maustaste den Dialog auf und ändert die Variable.

7.13 Variable +/-1

Mit diesem Baustein kann man den Wert einer Variablen um eins erhöhen oder um eins verringern. Im Dialogfeld gibt man an, welche Variable bearbeitet werden soll. Zur Verfügung stehen die Variablen Var1-Var99 sowie die Standardzählvariablen Z1-Z16 des Bausteins Position. Außerdem wählt man aus, ob der Wert um eins erhöht oder um eins verringert werden soll. Diese Zählrichtung wird Symbol des Bausteins mit angezeigt.

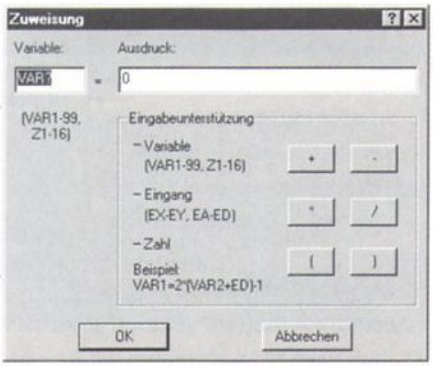


Variable1 um 1 erhöhen

Variable1 um 1 verringern

7.14 Zuweisung

Mit dem Baustein Zuweisung kann einer Variablen VAR1-Var99 oder einer Zählvariablen Z1-Z16 ein bestimmter Wert zugewiesen werden.



Im Dialogfeld trägt man links unter "Variable" die Variable ein, der ein Wert zugewiesen werden soll. Rechts unter "Ausdruck" legt man den Wert fest, den die Variable annehmen soll. Dieser Wert kann einfach eine Zahl sein, z. B. Var1=0.

Es besteht jedoch auch die Möglichkeit den Wert als mathematische Formel auszudrücken, z. B.

Var1=(Var1+3 * Var2)/5

Alle verfügbaren mathematischen Operationen (+ , -, *, /,) können in der Eingabeunterstützung des Dialogfelds per Mausklick eingefügt werden, ebenso das Setzen von Klammern.
Die Länge des Ausdrucks darf maximal 34 Zeichen betragen. Im Anzeigefenster des Bausteins werden zu lange Formeln abgeschnitten. Im Dialogfenster ist die ganze Formel sichtbar.
Ist in der Formel ein Fehler enthalten, erscheint beim Bestätigen mit OK eine Fehlermeldung. Durch Drücken der Funktionstaste F1 gelangt man in der Online-Hilfe auf die Seite, auf der ebenfalls beschrieben ist, welche Eingaben erlaubt sind.

Hinweis: Die Verwendung von Formeln wird im Kapitel 6 ausführlicher besprochen. Auf jeden Fall lesen! Das hilft Fehler zu vermeiden.

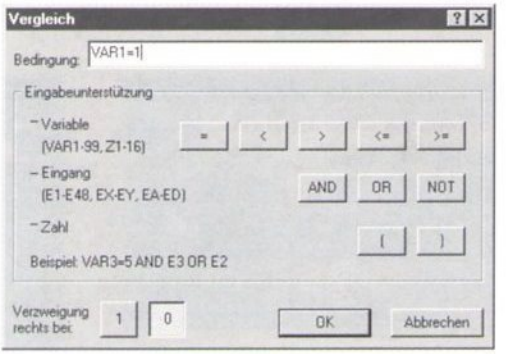
7.15 Vergleich

Im Baustein Vergleich trägt man eine Bedingung ein. Abhängig davon ob die Bedingung erfüllt ist oder nicht verzweigt der Ablauf nach rechts oder wird am unteren Ausgang des Bausteins fortgesetzt. Die Zahlen 0 und 1 an den Ausgängen stehen für "Bedingung erfüllt" (1) bzw. "Bedingung nicht erfüllt" (0). Ob bei 1 oder 0 nach rechts verzweigt wird, kann im Dialog des Bausteins festgelegt werden.

Die Bedingung wird im Dialog als Formel eingegeben,
z. B. Var1=1.

In der Eingabeunterstützung des Dialogs können die Vergleichsoperatoren, die logischen Verknüpfungen und die Klammern direkt angeklickt werden. Die ganze Formel kann aber auch komplett über die Tastatur eingetippt werden.

Die maximale Länge der Formel beträgt 40 Zeichen. Längere Formeln können nicht bearbeitet werden. Bereits vor Erreichen der maximalen Länge kann die Formel nicht mehr komplett im Anzeigefenster des Bausteins abgebildet werden. Sie wird einfach abgeschnitten, aber dennoch richtig berechnet. Im Dialog, der bei bereits eingefügten Bausteinen über die rechte Maustaste aktiviert wird, ist die komplette Formel zu sehen.

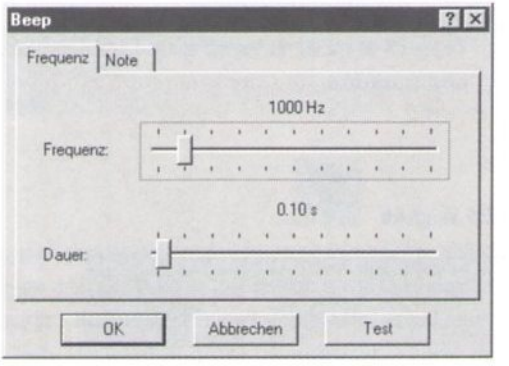


Gibt man in eine Formel einen Ausdruck ein, der mathematisch nicht korrekt bzw. nicht erlaubt ist, erscheint eine Fehlermeldung sobald man den Dialog mit OK bestätigen will. Um detaillierte Hilfe zu erhalten, kann man dann durch Drücken der Funktionstaste F1 auf der Tastatur zu dem entsprechenden Thema in der Online-Hilfe gelangen.

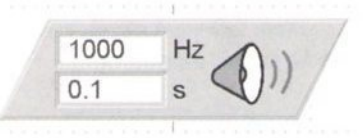
Hinweis: Die Verwendung von Formeln wird im Kapitel 6 ausführlicher besprochen. Auf jeden Fall lesen! Das hilft Fehler zu vermeiden.

7.16 Beep

Der Baustein Beep gibt einen Signalton über den Lautsprecher des PCs aus. Im Dialog kann die Tonhöhe und Tondauer eingestellt werden. Die Tonhöhe kann entweder als Frequenz (50-10000 Hz) oder für Musiker auch als Note (c-h in 5 Oktaven) eingegeben werden. Die Tondauer kann 0-5 Sekunden betragen.

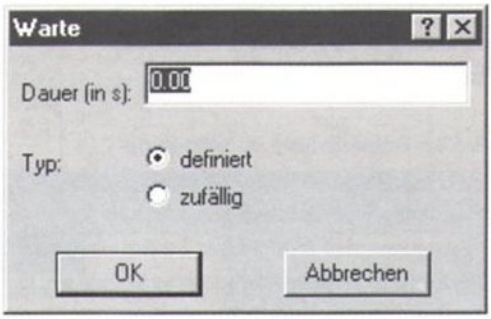


Das Spielen von Tönen und gleichzeitiges Einlesen von Analogwerten am parallelen Interface Art.-Nr. 30520 funktioniert nicht.
Beim Intelligent Interface Art.-Nr. 30402 gibt es dabei keine Schwierigkeiten.
Da das Intelligent Interface keinen eigenen Lautsprecher besitzt, können im Download-Betrieb keine Töne erzeugt werden.



7.17 Warte

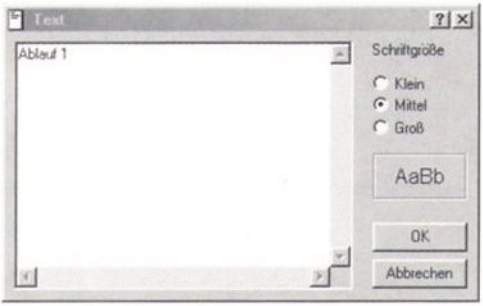
Mit dem Baustein "Warte" kann man eine Wartezeit in einem Ablauf programmieren. Die Wartezeit beginnt wenn der Baustein im Ablauf an der Reihe ist. Ist die eingegebene Wartezeit zu Ende, wird der Ablauf fortgesetzt.



Im Dialog des Bausteins wird die gewünschte Wartezeit in Sekunden eingegeben (maximal 999,99 s). Außerdem kann man wählen zwischen "definiert",
dann wartet der Baustein exakt die eingegebene Zeit, bevor der Ablauf fortgesetzt wird,
und "zufällig",
dann wartet er ein zufällig lange Zeitspanne, die aber maximal so lang ist wie die eingegebene Zeit. Diese zufällige Zeitspanne kann z. B. zur Programmierung von Reaktionstests verwendet werden.

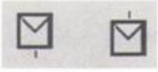
7.18 Text

Der hier eingegebene Text kann als Kommentartext an einer beliebigen Stelle auf der Programmieroberfläche platziert werden, z. B. zur Programmdokumentation.

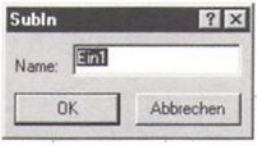


Im Dialog kann außerdem die Schriftgröße eingestellt werden. Ein bereits eingefügter Text kann geändert werden. Man klickt dazu mit der rechten Maustaste auf den Text und aktiviert damit den Dialog, in welchem Änderungen vorgenommen werden können.

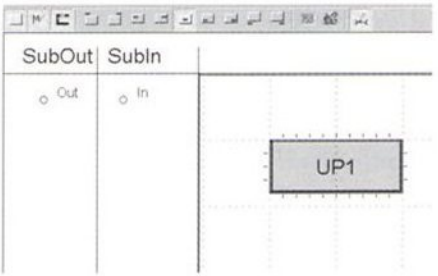
7.19 SubIn, SubOut



Diese beiden Bausteine sind im Bausteinfenster nur sichtbar, wenn man ein Unterprogramm bearbeitet. Sie stellen die Verbindung zwischen dem Hauptprogramm und dem Beginn des Unterprogramms (SubIn) bzw. dem Ende des Unterprogramms und dem Hauptprogramm (SubOut) her. Beim Einfügen wird für jeden dieser



Bausteine automatisch ein Name vorgeschlagen, den man auch ändern kann. Der selbe Name darf in einem Unterprogramm nicht mehrmals vorkommen, da sonst die verschiedenen SubIn- und SubOut-Bausteine nicht mehr eindeutig zugeordnet werden können.



Nach dem Einfügen von SubIn- und SubOut-Bausteinen in ein Unterprogramm müssen diese auch noch in Unterprogramm Design als Kreise auf dem Rahmen des Unterprogrammsymbols eingefügt werden. Erst dann können zwischen dem Ablauf des Hauptprogramms und einem Unterprogrammsymbol Verbindungslinien gezogen werden (siehe auch Kapitel 4, Arbeiten mit Unterprogrammen).

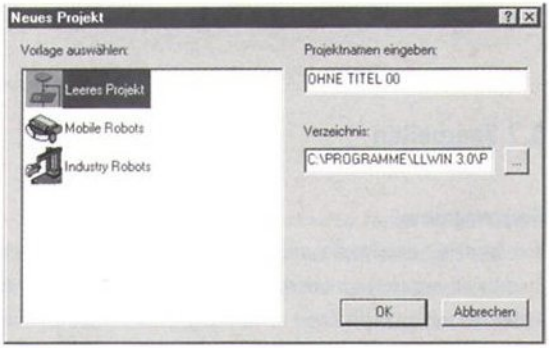
8. Menübefehle

8.1 Projekt

Wenn noch kein Projekt geöffnet ist, sind nur die Befehle NEU, ÖFFNEN und BEENDEN verfügbar.

Neu

Mit diesem Befehl wird ein neues Projekt angelegt. Zuerst erscheint ein Dialog, in dem man eine Vorlage für das Projekt auswählen kann.



In den Projektvorlagen sind bereits Unterprogramme für verschiedene fischertechnik-Modelltypen vorhanden (z. B. für Mobile Robots oder Industry Robots). Will man keine der vorhandenen Vorlagen nutzen wählt man "Leeres Projekt". Es kann gleich zu Beginn ein Projektname eingegeben werden. Will man das nicht, erhält das neue Projekt automatisch den Namen "Ohne Titel". Des weiteren kann man angeben, in welchem Verzeichnis das neue Projekt abgespeichert werden soll. Hierzu kann der Verzeichnispfad direkt eingegeben oder über die Schaltfläche ... ein Verzeichnis ausgewählt werden. Bestätigt man mit OK, wird das neue Projekt angelegt und geöffnet.

Öffnen

Mit diesem Befehl wird ein bereits vorhandenes Projekt geöffnet.

Speichern

Mit diesem Befehl speichert man ein Projekt ab. Hat dieses Projekt noch keinen Namen, muss man spätestens jetzt einen eingeben. LLWin-Projekte erhalten automatisch die Dateiendung .mdl.

Speichern unter

Will man ein vorhandenes Projekt unter einem neuen Namen speichern, z. B. weil man es geändert hat, die alte Version aber noch behalten möchte, wählt man diesen Befehl aus.

Schließen

Mit diesem Befehl wird das gerade geöffnete Projekt geschlossen. Wurde es seit dem letzten Abspeichern geändert, fragt LLWin, ob die Änderungen gespeichert werden sollen.

Seite drucken

Das Haupt- oder Unterprogramm, das gerade auf dem Bildschirm sichtbar ist, kann damit ausgedruckt werden. Die Größe der Bausteine auf dem Ausdruck wird mit dem Menübefehl OPTIONEN-ARBEITSBLATT eingestellt.

Beenden
Damit beendet man LLWin. Ist noch ein Projekt geöffnet erscheint die Abfrage ob es gespeichert werden soll.

Unter dem Befehl Beenden befindet sich eine Liste mit den maximal 5 zuletzt geöffneten Projekten, die durch anklicken mit der linken Maustaste direkt geöffnet werden können.

8.2 Bearbeiten

Hauptprogramm
Nach dem Öffnen eines Projektes wird dieser Befehl automatisch aufgerufen. Im Hauptprogramm fügt man die Software-Bausteine ein, verbindet sie mit Linien und erstellt so den Programmablauf. Man kann hier ebenso Bausteine und Linien löschen. Hat man ein Projekt einmal im Menü RUN gestartet und will, nachdem man das Programm gestoppt hat, Änderungen daran vornehmen, muss man zuerst wieder ins Hauptprogramm zurückkehren. Dass man sich wieder im Hauptprogramm befindet, erkennt man an dem Gitternetz auf der Programmieroberfläche.

Unterprogramm
Mit diesem Befehl kann man ein Unterprogramm erstellen oder ein bestehendes Unterprogramm bearbeiten. Wie man genau ein Unterprogramm erstellt, ist in Kapitel 4, Arbeiten mit Unterprogrammen, erklärt. Grundsätzlich sind beim Erstellen eines Unterprogramms die gleichen Funktionen verfügbar wie beim Erstellen eines Hauptprogramms.

Bausteine einfügen
Mit diesem Befehl wechselt man in den Modus, in dem aus dem Bausteinfenster Bausteine in ein Projekt eingefügt, verschoben und geändert sowie Linien verschoben werden können.
Zum Einfügen eines Bausteins klickt man im Bausteinfenster mit der linken Maustaste auf den gewünschten Baustein und zieht diesen bei gedrückter Maustaste an die gewünschte Stelle auf der Programmieroberfläche. Anschließend lässt man die Maustaste los, worauf der Dialog des Bausteins erscheint. Nachdem man die gewünschten Werte eingegeben hat, wird der Dialog mit OK geschlossen und der Baustein ist endgültig auf der Oberfläche platziert.

Will man anstelle eines Bausteins ein bestehendes Unterprogramm einfügen, muss man im Bausteinfenster auf den Reiter "Unterprogramme" wechseln.
Zunächst wird aus der Liste das gewünschte Unterprogramm ausgewählt. Der zugehörige Unterprogrammbaustein erscheint im oberen Feld und kann mit der Maus wie jeder andere Baustein von dort auf die Programmieroberfläche gezogen werden.

Durch einen Klick mit der linken Maustaste auf einen Baustein oder ein Unterprogramm kann man diese bei gedrückter Maustaste an eine andere Stelle verschieben. Die Verbindungslinien bleiben dabei erhalten. Mit eben beschriebenem Verfahren lassen sich aber auch die Verbindungslinien verschieben. Einfach Verbindungslinie anklicken und bei gedrückter Maustaste an die gewünschte Position verschieben.

Auch können die Werte der Bausteine geändert werden, indem man mit der rechten Maustaste auf den Baustein klickt und im darauf erscheinenden Dialog die Werte ändert.

Die genaue Vorgehensweise zum Einfügen, Verschieben und Ändern von Bausteinen ist im Kapitel 3.3 ausführlich beschrieben. Einfach dort nachlesen!

Bausteine löschen

Mit diesem Befehl aktiviert man den Modus, in dem Bausteine aus einem Ablauf gelöscht werden können. Das Bausteinfenster verschwindet in diesem Modus, denn es wird ja nicht benötigt. Der Cursor verändert sein Aussehen und wird zum Hammer. Man löscht einen Baustein auf dem sich der Hammer befindet durch einen Click mit der linken Maustaste. Die angrenzenden Verbindungslinien zu anderen Bausteinen werden ebenfalls gelöscht.

Achtung! Die Aktion "Löschen eines Bausteins" kann nicht rückgängig gemacht werden.

Bausteine ersetzen

In diesem Modus kann ein bereits in einem Ablauf eingefügter Baustein durch einen anderen ersetzt werden. Dazu zieht man mit der linken Maustaste aus der Toolbox einen Baustein und platziert ihn über dem zu ersetzenden Baustein. Besitzt der neue Baustein die gleichen Ein- und Ausgänge wie der alte, werden auch die Verbindungslinien automatisch übernommen.

Linien zeichnen

Aktiviert man diesen Befehl, wird aus dem Cursor ein Stift. Zwischen zwei Bausteinen wird eine Linie gezogen, indem man zuerst mit der linken Maustaste auf den Ausgang des ersten Bausteins klickt. Lässt man die Maustaste wieder los, wird aus dem Cursor ein Fadenkreuz. Daran erkennt man, dass der Ausgang "getroffen" wurde. Danach klickt man auf den Eingang des zweiten Bausteins, die Linie wird gezogen. Den Verlauf von Linienteilstücken oder des gesamten Linienzuges kann man auch selbst bestimmen. Durch Klick auf einen Eingang, Ausgang oder eine Linie wird das Zeichnen begonnen. Bewegt man dann das Fadenkreuz genau senkrecht oder waagerecht zum ersten Punkt und betätigt erneut die linke Maustaste, wird die erste Teillinie gezeichnet. In gleicher Weise können weitere Teilstücke gezeichnet werden, bis man den Zielpunkt erreicht. Referenzpunkt für die nächste Zeichenoperation ist immer der Endpunkt der zuletzt gezeichneten Teillinie. Möchte man den restlichen Linienzug automatisch zeichnen lassen, klickt man auf den Zielpunkt. Durch STRG + linke Maustaste können nach dem Zeichnen einzelne Stücke der Linie verschoben werden.

Linien löschen

Mit diesem Befehl wird Cursor zu einem Radiergummi der zum Löschen einer Linie verwendet wird, nicht anders als beim Radieren auf dem Papier auch. Klickt man mit der linken Maustaste an irgendeiner Stelle auf eine Verbindungslinie, so wird

nur diese gelöscht. Fehlende Teilstücke an anderen angrenzenden Linien werden vervollständigt. Betätigt man die rechte Maustaste wird die getroffene Linie inklusive aller angrenzenden Linien gelöscht.

Rückgängig

Rückgängig gemacht werden kann jeweils die letzte Aktion, bei der Bausteine oder Verbindungen verschoben wurden. Das Löschen oder Einfügen von Bausteinen kann leider noch nicht rückgängig gemacht werden (wir arbeiten daran).

Alles markieren

Einzelne oder mehrere Bausteine können markiert werden, um sie danach zusammen auszuschneiden, zu kopieren, zu löschen oder zu verschieben. Markierte Bausteine werden mit einem gestrichelten Rechteck versehen. Mit dem Befehl "Alles markieren" werden alle Bausteine des Haupt- oder Unterprogramms, in dem man sich gerade befindet, markiert. Will man nur einzelne Bausteine markieren, so klickt man diese bei gedrückter STRG-Taste mit der linken Maustaste an. Eine ausführliche Beschreibung hierzu ist in Kapitel 3.6 zu finden.

Markierung aufheben

Wie sich wahrscheinlich jeder denken kann, verschwindet durch diesen Befehl die Markierung bei allen bis dahin markierten Bausteinen.

Ausschneiden

Mit diesem Befehl werden alle markierten Bausteine (auch ganze Abläufe) aus dem Projekt ausgeschnitten und in der Zwischenablage gespeichert. Deshalb ist der Befehl nur aktiv, wenn markierte Bausteine vorhanden sind.

Kopieren

Alle markierten Bausteine werden in die Zwischenablage kopiert. Auch dieser Befehl ist nur verfügbar, wenn Bausteine markiert sind. Zusätzlich wird beim Kopieren ein Bitmap der kopierten Bausteine erzeugt, das in andere Windows-Programmme, z. B. Word oder Paint, eingelesen werden kann.

Einfügen

Die ausgeschnittenen oder in die Zwischenablage kopierten Bausteine können entweder in das selbe Haupt- oder Unterprogramm aus dem sie stammen, in ein anderes Unterprogramm des selben Projekts oder in ein anderes Projekt eingefügt werden. Dabei bleiben die Verbindungslinien zwischen den eingefügten Bausteinen erhalten, die angrenzenden Linien werden gelöscht. Was man beim Einfügen insbesondere von Unterprogrammsymbolen in andere Projekte beachten muss, siehe Kapitel 4.4.

Löschen

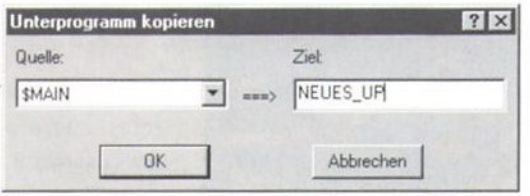
Damit kann man alle markierten Bausteine auf einen Schlag löschen.

Achtung! Diese Aktion kann nicht mehr rückgängig gemacht werden.

8.3 Unterprogramm

Kopieren

Dieser Befehl kopiert ein vorhandenes Unterprogramm in ein neues Unterprogramm des selben Projekts. Der Name des neuen Unterprogramms wird im folgenden Dialog im Feld "Ziel" eingegeben:

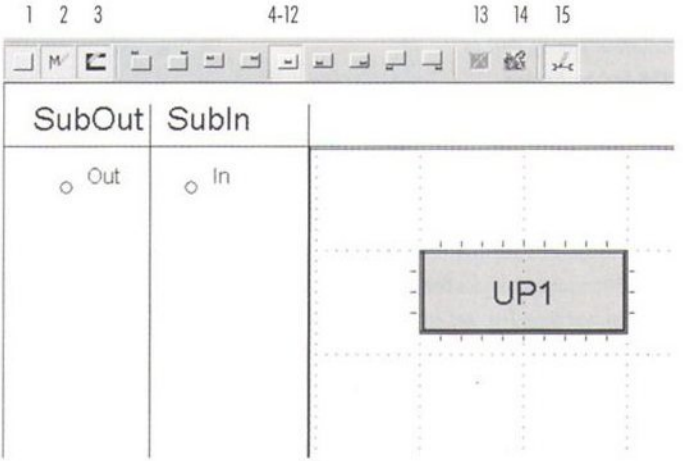


Der Name des Unterprogramms darf maximal 14 Zeichen lang sein, Leerzeichen sind nicht zulässig. Gibt man als Quelle "$MAIN" an, werden alle Abläufe des Hauptprogramms in ein Unterprogramm kopiert.

Wie man ein Unterprogramm in ein anderes Projekt kopiert, siehe Kapitel 4.4.

Design

Mit Hilfe des Befehls Design kann das Aussehen eines Unterprogrammbausteins verändert werden.



Die Größe des Bausteins lässt sich ändern, indem mit der linken Maustaste auf die rechte, untere Rahmenecke geklickt und bei gedrückter Maustaste der Rahmen auf die gewünschte Größe gezogen wird. Dann lassen sich auch längere Unterprogrammnamen gut im Rahmen unterbringen. Wurden beim Bearbeiten von Unterprogrammen SubIn- bzw. SubOut-Bausteine eingefügt, müssen diese auf dem Rahmen des Unterprogrammbausteins plaziert werden. Sie sind in der links angeordneten Tabelle als Kreise dargestellt. Mit der Maus greift man die Kreise (Maustaste gedrückt halten) und setzt sie auf dem Unterprogrammrahmen an der gewünschten Position. Die Kreise stellen die Anschlüsse dar, die den Unterprogrammbaustein mit den anderen Bausteinen im Hauptprogramm verbinden.

Schließlich können über die zusätzlichen Schaltflächen unter der Menüleiste weitere Eigenschaften des Unterprogrammnamens eingestellt werden. Die folgende Erklärung bezieht sich auf die Nummerierung in der Abbildung):

1: Darstellung des Unterprogrammbausteins als graue Schaltfläche ein- bzw. ausschalten.

2: Anzeige des Unterprogramm-Namens ein- bzw. ausschalten.

3: Rahmen für Baustein ein- bzw. ausschalten.

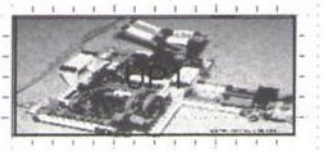
4-12: Position des Unterprogramm-Namens.

13: Ist die Darstellung als graue Schaltfläche deaktiviert (Icon 1) kann man dem Unterprogrammbaustein ein Bitmap zuweisen. Es erscheint folgender Dialog:

Man wählt entweder eines der zur Verfügung stehenden Bilder aus oder kopiert mit "Importieren" ein Bitmap aus einem anderen Verzeichnis in das Verzeichnis \PROJEKTE DEUTSCH\BITMAP. Das importierte Bitmap kann dann aus der Liste ausgewählt werden.

Wichtig: Bei dem importierten Bitmap darf es sich maximal um ein 256-Farben-Bitmap handeln, Bilder mit mehr Farben werden nicht dargestellt.

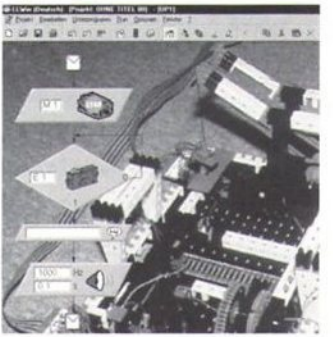
Soll das Bitmap in der Originalgröße dargestellt werden, aktiviert man "Baustein an Größe der Bitmap anpassen". Der Unterprogrammbaustein wird dann entsprechend vergrößert oder verkleinert. (Diese Funktion ist nur verfügbar, wenn das Unterprogramm nicht in andere Unterprogramme oder das Hauptprogramm eingefügt wurde.) Deaktiviert man dieses Feld, wird das Bild an die Größe des Bausteins angepasst und entsprechend verzerrt. Das Ergebnis sieht dann z. B. so aus: ▶



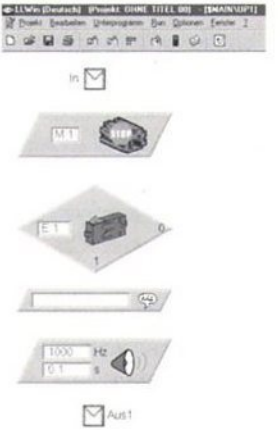
14: Hier kann man ein Bitmap auf den Hintergrund der Programmieroberfläche des Unterprogramms setzen. Die Bausteine können dann auf dem Hintergrundbitmap platziert werden. Das Bild kann wie bei Icon 13 beschrieben eingefügt oder importiert werden und ist in seiner Orginalgröße sichtbar.

Wichtig: Bei dem importierten Bitmap darf es sich maximal um ein 256-Farben-Bitmap handeln, Bilder mit mehr Farben werden nicht dargestellt.

Man kann auch ein Bitmap auf dem Hintergrund des Hauptprogramms platzieren. Man wählt dazu beim Aufrufen des Befehls "Design" in der Auswahlbox der verfügbaren Unterprogramme das Unterprogramm "$MAIN" aus.



15: Mit diesem Icon kann man die Verbindungslinien im Ablauf des Unterprogramms verstecken. Sie werden dann im Init-Modus (RUN-INIT) nicht mehr angezeigt. Diese Funktion kann man z. B. dann nutzen, wenn man ein Projekt präsentiert, das so viele Verbindungslinien besitzt, dass sie mehr verwirren als den Ablauf zu verdeutlichen.



Löschen

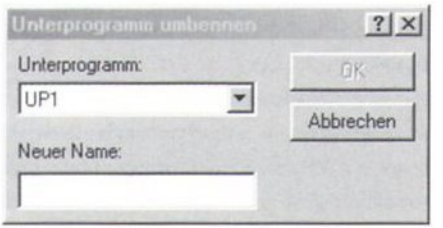
Um ein Unterprogramm löschen zu können, muss zunächst im Menü FENSTER der Befehl ALLE SCHLIESSEN ausgeführt werden. Dann lässt sich der Befehl löschen ausführen. Es erscheint das Fenster mit den im Projekt enthaltenen Unterprogrammen. Durch Auswahl des gewünschten Unterprogramms und der Bestätigung mit OK wird das Unterprogramm gelöscht.

> Hinweis: Mit dem Befehl BAUSTEINE LÖSCHEN im Menü BEARBEITEN können einzelne Bausteine und Unterprogrammaufrufe gelöscht werden, nicht das Unterprogramm selber. Dies geschieht mit obigem Befehl löschen. Mit dem Unterprogramm werden auch alle Bausteine dieses Unterprogramms im gesamten Projekt gelöscht.

Umbenennen

Mit diesem Befehl können sehr einfach die Namen der im Projekt enthaltenen Unterprogramme geändert werden.

Zunächst wird das gewünschte Unterprogramm ausgewählt und anschließend im Feld "Neuer Name" dieser eingegeben. Der Name des neuen Unterprogramms darf maximal 14 Zeichen lang sein, Leerzeichen sind nicht zulässig. Ein Klick auf OK ändert den Namen des Unterprogramms entsprechend ab.



8.4 Run

Das Menü Run dient zum Übersetzen, Testen oder Starten des erstellten Steuerungsprogramms. Das Einfügen oder Löschen von Bausteinen bzw. Änderungen an Verbindungen sind in dieser Betriebsart nicht möglich. Befindet man sich in diesem Modus, ist auf der Programmieroberfläche kein Gitternetz zu sehen. Zurück in den Editiermodus, in dem man Bausteine einfügen und Verbindungslinien zeichnen kann, gelangt man durch den Befehl HAUPTPROGRAMM im Menü BEARBEITEN.

Init

Nach Aufruf von Init wird geprüft, ob alle Bausteine miteinander verbunden sind. Nicht verbundene Bausteine werden violett angezeigt. Außerdem können im Init-Modus die Nummern und Zahlenwerte in den Bausteinen verändert werden. Sind im Hauptprogramm Unterprogrammbausteine enthalten, genügt ein Klick mit der linken Maustaste auf den Unterprogrammbaustein, um in die nächst tiefere Ebene (das Unterprogramm selber) zu gelangen. Ein Klick mit der rechten Maustaste führt zurück in die nächst höhere Ebene.
Mit jedem Mausklick auf den Befehl Init wird ein neues Fenster geöffnet. Mit den Funktionen des Menüs Fenster können diese gleichzeitig am Bildschirm sichtbar gemacht werden. So können, während ein Programm im Online-Modus läuft, z. B. gleichzeitig das Hauptprogramm und mehrere Unterprogramme dargestellt werden.

Start

Mit diesem Befehl wird die Übersetzung des Projekts gestartet und dann das Steuerungsprogramm im Online-Betrieb abgearbeitet, das heißt, der PC steuert über das Interface das angeschlossene Modell. Bei sehr langsamen PCs und großen Projekten kann das Übersetzen etwas länger dauern. Während des Übersetzens erscheint in einem kleinen Fenster eine Balkenanzeige, die über den Status des Vorgangs informiert. Danach wird das Steuerungsprogramm gestartet. Die Bausteine, die gerade abgearbeitet werden, sind im Ablaufplan rot dargestellt. Einige Bausteine zeigen zusätzlich den Wert der Variablen oder des Eingangs an. Während das Steuerungsprogramm abgearbeitet wird, können die Schalter und Parameterfelder des TERMINALS (sofern verwendet) bedient werden. Darüber hinaus ist auch das bereits im Init-Modus beschriebene Wechseln in die verschiedenen Haupt- und Unterprogrammebenen möglich.

Stop

Hierzu ist verständlicher Weise nicht viel zu sagen. Das im Onlinemodus laufende Steuerungsprogramm wird unterbrochen und in den Init-Modus gewechselt.

Download

Wird das Intelligent Interface (Art.-Nr. 30402) verwendet, kann, im Gegensatz zum Online-Betrieb, das erstellte Steuerungsprogramm auch auf das Intelligent Interface übertragen werden. Dies geschieht mit dem Befehl Download. Anschließend kann das Interface vom PC getrennt und das angeschlossene Modell unabhängig vom PC betrieben werden. Interessant ist das z. B. bei mobilen Robotern (Baukasten: Mobile Robots), bei denen das Übertragungskabel zum PC sehr hinderlich wäre.

Hinweis: Wird das ältere parallele Interface (Art.-Nr. 30520) verwendet, ist die Möglichkeit des Downloads deaktiviert. Wie das jeweils verwendete Interface eingestellt wird, kann bei Optionen-Interfaceeinstellungen nachgelesen werden.

Wichtig: Um nach einem Download die Verbindung zwischen dem Intelligent Interface und dem PC wieder herzustellen muss man die Stromversorgung des Interface für ca. 3-4 Sekunden unterbrechen. Das auf dem Interface laufende Programm wird dadurch gelöscht, das Interface kehrt in den Online-Modus zurück und kann wieder Verbindung zum PC aufnehmen.
Andernfalls erscheint beim Aufrufen der Interfacediagnose oder beim erneuten Programmstart die Meldung: "Keine Verbindung zum Interface".

8.5 Optionen

Über die ersten 3 Befehle dieses Menüs lassen sich **Symbolleiste**, **Statusleiste** und das **Bausteinfenster** (sofern aktiviert) ein- bzw. ausblenden. Dies kann manchmal sehr hilfreich sein, wenn man die maximal mögliche Fensterfläche ausnutzen möchte.

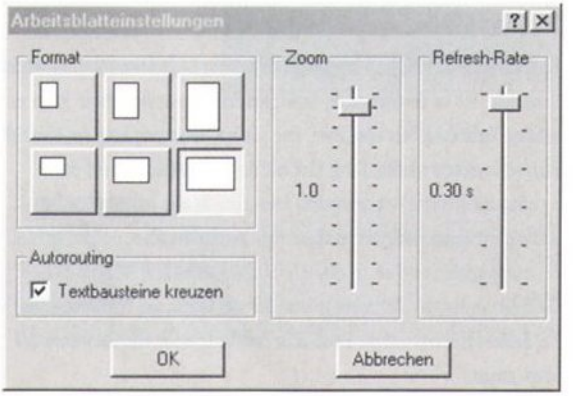
Sprache

Hier lässt sich die gewünschte Sprache, in der sich LLWin präsentiert, einstellen. Wechselt man die Sprache, muss LLWin neu gestartet werden.

Zur Zeit sind nur die Sprachen Deutsch und Englisch verfügbar. Weitere Sprachen sind in Arbeit – Bitte etwas Geduld.

Arbeitsblatt

Mit diesem Befehl können die Eigenschaften des Arbeitsblattes, auf dem die Bausteine plaziert werden, eingestellt werden.



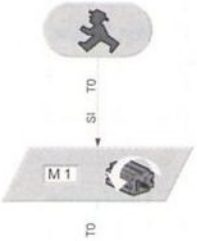
Bei Format kann die Größe des Arbeitsblattes in Hoch- oder Querformat eingestellt werden. Je umfangreicher das Projekt (Anzahl der verwendeten Bausteine), desto größer sollte das Format sein, damit auch alle Bausteine

auf das Arbeitsblatt passen. Beim Drucken wird das Arbeitsblatt auf die bei der Druckereinrichtung eingestellte Papiergröße angepasst (Arbeitsblatt wird wie beim Kopieren so verkleinert, dass alles beispielsweise auf ein A4-Papier passt). Die Option "Textbausteine kreuzen" bewirkt, dass beim Autorouting die Verbindungslinien von Bausteinen auch über Textfelder gelegt werden.

Damit auch umfangreichere Projekte auf dem Bildschirm dargestellt werden können, kann mit dem Schieberegler ein entsprechender Zoomfaktor eingestellt werden. Der Zoom lässt sich auch sehr schnell über die Taste "+" und "-" des Ziffernblocks der Tastatur ändern, was gerade bei großen Ablaufsteuerungen sehr hilfreich ist. Die aktuelle Zoomeinstellung wird beim Beenden von LLWin gespeichert und beim nächsten Aufruf voreingestellt.

Die ebenfalls einstellbare Refresh-Rate gibt den Zeitabstand in s an, mit der im Online-Betrieb die angezeigten Werte und Variablen im Ablaufplan auf dem Bildschirm aktualisiert werden.

Anschlüsse



Mit der Option Anschlüsse kann die Anzeige der Anschlussbezeichnungen an den Ein- und Ausgängen der Bausteine aktiviert werden. Bei Unterprogrammen mit mehreren SUBIN- und SUBOUT-Bausteinen, die sich nur in ihren Bezeichnungen unterscheiden, diese Option zur sehr hilfreich.

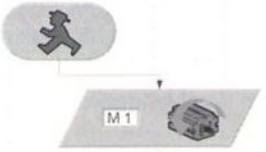
Cursorumschaltung

Im Editiermodus bewirkt die Cursorumschaltung, dass je nachdem, wo man sich mit der Maus im Ablaufplan befindet, der Cursor entsprechend der dort ausführbaren Aktionen sein Aussehen ändert. Befindet sich der Mauszeiger z.B. über einem Baustein, so wechselt der Cursor in ein Kreuz mit Pfeilen und zeigt damit an, dass der Baustein verschoben werden kann. Die ausführbaren Aktionen werden zusätzlich mit den evtl. notwendigen Tastenkombinationen in der Statuszeile angezeigt.

Autorouting

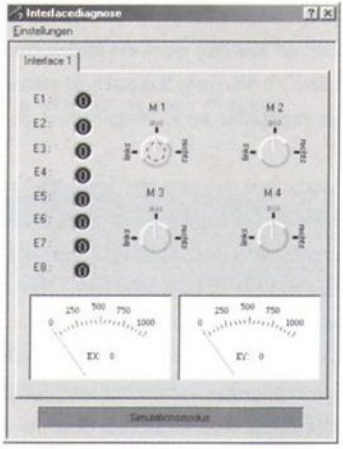
Ist die Funktion Autorouting aktiviert, so zeichnet der Computer beim Verbinden von zwei Bausteinen bzw. beim Verschieben eines Bausteins automatisch die Verbindungslinien. Bei abgeschaltetem Autorouting funktioniert das sog. "Auto Connect" (Kapitel 3.4) weiterhin. Findet das Autorouting keine rechtwinklige Verbindung außerhalb der gesperrten Bereiche anderer Bausteine (Textfelder) und Verbindungen, so wird die Verbindung als Linie ohne Knickpunkte zwischen den Anschlüssen dargestellt.

Diese dann als schräge Linie angezeigte Verbindung ist funktionsfähig, es werden jedoch keine Pfeile für die Richtung dargestellt. Die an der Verbindung beteiligten Bausteine sollte man so lange verschieben, bis das Autorouting eine rechtwinklige Verbindung herstellen kann. Eine schräge Verbindung kann nur unmittelbar am Eingang oder am Ausgang eines Bausteins gelöscht werden. Durch Klick mit der linken Maustaste auf einen Ausgang mit schräger Verbindungslinie kann diese in eine rechtwinklige Darstellung mit einem Knickpunkt umgewandelt werden.



Die Deaktivierung des Autoroutings kann bei sehr langsamen PCs sinnvoll sein, bei denen das Berechnen der Linien nach dem Verschieben von Bausteinen zu lange dauert.

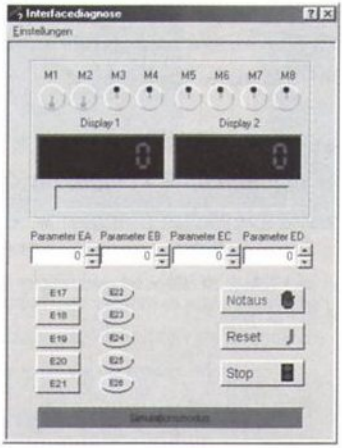
Interfacediagnose



Mit Hilfe des Befehls Interfacediagnose kann die Funktion des/der Interface/es getestet werden. Dazu gehört die Anzeige der digitalen bzw. analogen Eingänge und die Knöpfe für den Test der Ausgänge. Eine genaue Vorgehensweise hierzu ist in Kapitel 2.4 beschrieben.

Des weiteren verfügt die Interfacediagnose über einen Menüpunkt "Einstellungen". Deaktiviert man dort die Option IMMER IM VORDERGRUND, verschwindet das Fenster in den Hintergrund, sobald man mit der Maus ein anderes Fenster aktiviert. Über die Taskleiste oder die Symbolleiste von LLWin kann das Fenster wieder in den Vordergrund geholt werden. Der Befehl INTERFACEEINSTELLUNGEN hat die gleiche Wirkung wie der gleichlautende Befehl im Menü OPTIONEN (siehe dort). Mit BEENDEN wird das Diagnosefenster geschlossen.

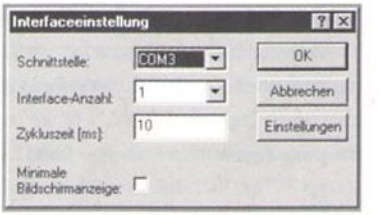
Startet man bei geöffneter Interfacediagnose ein Projekt mit RUN-START, wird aus dem Diagnosefenster ein Bedienterminal mit der gleichen Funktionalität wie der Baustein "Terminal" (siehe hierzu die Bausteinbeschreibung im Kapitel 7).



Während des Programmablaufs kann dieses Fenster auch im Menü über OPTIONEN-BEDIENTERMINAL oder das Diagnose-Icon in der Symbolleiste aufgerufen werden.
Ein Vorteil gegenüber dem Baustein Terminal ist, dass auf Schalterbetätigungen sehr viel schneller reagiert wird und eine flüssigere Programmsteuerung möglich ist.
Hält man das Projekt an, z. B. über den Button "Stop" im Bedienterminal erscheint wieder das ursprüngliche Diagnosefenster.
Während das Diagnosefenter mit dem X in der Titelzeile oder über den Menüpunkt EINSTELLUNGEN-BEENDEN ganz geschlossen werden kann, wird das Bedienterminal bei Betätigung des X oder Aufruf von Beenden nur minimiert, um Probleme im Programmablauf durch mehrmaliges Öffnen und Schließen des Fensters zu vermeiden.

Hinweis: Für reine Testzwecke kann die Interfacediagnose auch unabhängig von der Software LLWin über das entsprechende Programmsymbol (START-PROGRAMME-LLWIN-INTERFACEDIAGNOSE)gestartet werden. Etwaige Veränderungen in den Interfaceeinstellungen bleiben erhalten!!

Interfaceeinstellung



Damit die Verbindung von PC und Interface korrekt funktioniert, müssen mit Hilfe dieses Menüpunkts die entsprechenden Einstellungen vorgenommen werden. Hierzu erscheint folgendes Fenster:

Zunächst muss die verwendete "Schnittstelle" ausgewählt werden. Hierfür kann eine parallele Schnittstelle LPT1 bzw. LPT2 oder eine serielle Schnittstelle COM1 bis COM4 ausgewählt werden. Wird das ältere parallele Interface (Art.-Nr.: 30520) verwendet, muss LPT1 oder LPT2 eingestellt werden. Für das neue Intelligent Interface (Art.-Nr. 30402) muss die entsprechende Schnittselle COMx verwendet werden.

Über die Interface-Anzahl kann eingestellt werden, ob nur ein Interface angesteuert wird (Interface-Anzahl 1) oder ob ein Intelligent Interface mit Extension Module (Art.-Nr. 16554) bzw. zwei miteinander gekoppelte parallele Interfaces angesteuert werden (Interface-Anzahl 2). In diesem Fall zeigt das Interfacediagnosefenster zwei Interfaces zur Auswahl.

Die "Zykluszeit" ist die Zeit in ms, die von LLWin benötigt wird um einen Baustein zu durchlaufen. Sie ist auf 10 ms voreingestellt. Bei größeren Programmen und sehr langsamen PCs kann es erforderlich werden, die Zykluszeit zu vergrößern, da sonst die Windows-Oberfläche zu wenig Rechenzeit erhält. Muss man auf eine Tastatureingabe mehrere Sekunden warten, sollte man versuchen, dieses Programm mit einer größeren Zykluszeit zu betreiben. Allerdings kann es bei einer größeren Zykluszeit beim Zählen von Impulsen zu Fehlern kommen. Dann wird z. B. eine programmierte Position nicht genau angefahren.

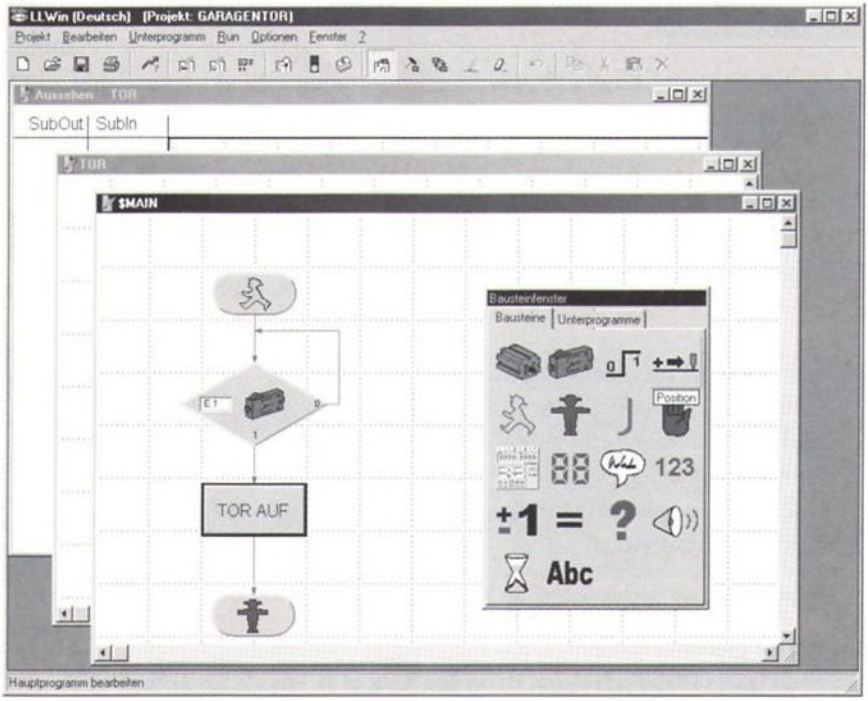
Nach dem Einschalten der "Minimale Bildschirmanzeige" werden im Online-Modus z. B. die Zustände der digitalen Eingänge nicht in den Bausteinen angezeigt. Damit ist mehr Rechenzeit für die Aktualisierung des Bildschirmes und für die Abfrage der Tastatur und der Maus verfügbar.

Ist COM1-COM4 eingestellt, öffnet sich über die Schaltfläche "Einstellungen" ein weiteres Dialogfeld, das die für die serielle Schnittstelle verwendeten Parameter zeigt. Darüber hinaus kann hier eingegeben werden, nach wieviel Abfragen die Analogwerte EX und EY des Intelligent Interface aktualisiert werden. Hier ist standardmäßig der Wert 10 eingetragen. Da das Einlesen der Analogwerte im Intelligent Interface sehr viel Rechenkapazität in Anspruch nimmt, kann die Aktualisierung bei jeder Abfrage zu Fehlern bei gleichzeitigem Zählen von Impulsen führen. Weitere Informationen dazu sind auch im Kapitel 10.2 zu finden.

Hinweis: Die Möglichkeit keine Schnittstelle auszuwählen, wurde für den Fall vorgesehen, dass kein Interface angeschlossen wird. Dann kann die Zykluszeit auf z.B. 1000 ms (= 1 Sekunde) gesetzt werden. Bei dieser Zykluszeit lässt sich besser beobachten, wann jeder einzelne Baustein aufgerufen wird. Ist "keine Schnittstelle" eingestellt, erscheint in der Interfacediagnose im Balken, der die Verbindung zum Interface anzeigt, die Meldung "Simulationsmodus".

8.6 Fenster

Im Editiermodus erzeugt LLWin für Hauptprogramm, Unterprogramme und Unterprogrammdesigns jeweils ein eigenes Fenster.



Mit dem Menü **Fenster** können die verschiedenen, geöffneten Fenster angeordnet, bzw. zwischen einzelnen Fenstern hin- und hergeschaltet werden. Sehr praktisch ist dies z.B. dann, wenn man verschiedene Abläufe in Haupt- und Unterprogrammen gleichzeitig nebeneinander darstellen, oder einzelne bzw. mehrere Bausteine hin und her kopieren möchte.

Hinweis: Im Init- bzw. Run-Modus sind die verschiedenen Bearbeitungs- oder Editierfenster geschlossen. Der Wechsel in ein Unterprogramm ist dann über einen Klick mit der linken Maustaste auf den Unterprogrammbaustein möglich. Zurück geht's mit der rechten Maustaste!!

Überlappend

Alle geöffneten Fenster werden so dargestellt, dass sie sich überlappen. Die Fensterüberschriften aller Fenster sind sichtbar. Obige Abbildung wurde mit Hilfe dieses Befehls erzeugt.

Nebeneinander

Alle geöffneten Fenster werden nebeneinander bzw. ab 4 Fenster neben- und untereinander angeordnet. Diese Darstellung ist sehr praktisch beim Kopieren von Bausteinen.

Symbole anordnen

Wurden mehrere Fenster mit der Schaltfläche "Fenster minimieren" zu Symbolen verkleinert, werden die Symbole mit diesem Befehl neu angeordnet.

Schließen

Damit lässt sich, wie könnte es anders sein, das aktuelle Fenster schließen.

Alle schließen

Soll ein Unterprogramm gelöscht werden, muss vorher der Befehl Alle schließen ausgeführt werden. Es werden alle geöffneten Bearbeiten-Fenster bzw. Run-Fenster geschlossen.

Hinweis: Mit diesem Befehl werden zwar alle geöffneten Fenster, nicht jedoch das Projekt selber geschlossen (der Projektname ist in der Titelzeile noch enthalten). Die einzelnen Fenster lassen sich wieder sehr schnell mit den Befehlen HAUPTPROGRAMM und UNTERPROGRAMM aus dem Menü BEARBEITEN bzw. den Tasten <F6> und <F7> öffnen.
Zum Schließen des gesamten Projekts muss dieses zunächst gespeichert und anschließend über PROJEKT-SCHLIEßEN beendet werden.

Auswahl der geöffneten Fenster (1, 2, 3 ....)

Sind mehrere Fenster als Vollbild geöffnet, dann kann im Menü Fenster mit der Maus oder über die Tastatur auf ein anderes Fenster umgeschaltet werden. Am schnellsten geht dies, wenn die Tasten <Alt+F> betätigt, und dann die Ziffer 1, 2, 3 ... gedrückt werden.

Exportieren in die Ablage

Bei Ausführung dieses Befehles wird der am Bildschirm sichtbare Inhalt des aktuellen Fensters in die Windows-Zwischenablage kopiert. Die Zwischenablage kann dann von Zeichen- oder Textverarbeitungs-programmen eingelesen werden.

8.7 ? oder die Hilfe

Index

Mit dem Befehl **Index** wird das Inhaltsverzeichnis der Online-Hilfe von LLWin aufgerufen.

Tastatur

Mit dem Befehl Tastatur wird das Kapitel Tastaturkombinationen der Online-Hilfe aufgerufen. Tastaturkombinationen erleichtern und beschleunigen die Bedienung von LLWin und sind darüber hinaus für einige Bedienhandlungen unerläßlich.

Befehle

Mit diesem Befehl können alle in LLWin verfügbaren Menübefehle gleichzeitig angezeigt werden. Durch Mausklick auf den jeweiligen Befehl wird die zugeordnete Online-Hilfe angezeigt.

Hilfe verwenden

Mit diesem Befehl wird die allgemeine Windows-Hilfe aufgerufen. Darin sind Hinweise zur prinzipiellen Funktion und Arbeitsweise der Online-Hilfe nachzulesen.

Info über..

Dieser Befehl öffnet ein Fenster, in dem die Adresse von fischertechnik und die Versionsnummer von LLWin angezeigt werden.

fischertechnik-Homepage

Mit diesem Befehl wird der Internet-Browser gestartet und direkt versucht, die fischertechnik-Homepage zu laden. Voraussetzung dafür ist natürlich, dass man über einen Internetzugang verfügt. Sollte die notwendige Online-Verbindung noch nicht bestehen, muss sie zunächst aufgebaut werden.

9. Die Symbolleiste und wichtige Tastaturkombinationen

Die wichtigsten Menübefehle können auch direkt durch Anklicken des entsprechenden Icons in der Symbolleiste ausgeführt werden.


Neues Projekt


Projekt öffnen


Projekt speichern


Seite drucken


Interface-Diagnose / Bedienterminal


Hauptprogramm


Unterprogramm


Programm-Design


Init


Start


Download


Bausteine einfügen


Bausteine löschen


Bausteine ersetzen


Linie zeichnen


Linie löschen


Rückgängig


Kopieren


Ausschneiden


Einfügen


Löschen


Aufwärts (Init-/Run-Modus)

Einfügen von Gitternetzlinien (Spalten- und Zeilenfunktion)

Beim Bearbeiten können im Modus "Bausteine einfügen" Spalten oder Zeilen in das Gitterraster des Arbeitsblattes eingefügt bzw. gelöscht werden. Dazu muss die Cursorumschaltung aktiv sein. Die Spalten- und Zeilenfunktion kann verwendet werden, um die Bausteine ab einer bestimmten Linie nach rechts bzw. nach unten zu verschieben.

- Spalten werden eingefügt, wenn man beim Bearbeiten die Taste <SHIFT> drückt und dann mit der linken Maustaste auf die gewünschte Stelle des Arbeitsblattes klickt. Zum Löschen einer Spalte muss anstelle der Taste <SHIFT> die Taste <Strg> gedrückt werden.
- Zeilen werden eingefügt, wenn man die Taste <SHIFT> drückt und dann mit der rechten Maustaste auf das Arbeitsblatt klickt. Zum Löschen einer Zeile muss anstelle der Taste <SHIFT> die Taste <STRG> gedrückt werden.

Hinweis: Spalten und Zeilen können nur eingefügt bzw. gelöscht werden, wenn sich auf der gedachten Trennlinie keine Bausteine befinden. Der Cursor zeigt an, ob das Einfügen einer Spalte bzw. Zeile möglich ist.

Weitere wichtige Tastaturkombinationen

Neben den Icons der Symbolleiste stehen dem Programmierer für die wichtigsten Menübefehle auch eine ganze Reihe von Tastaturkombinationen zur Verfügung:

1. Funktionstasten

F1	Direkter Aufruf der Online-Hilfe zu der Aktion, die im Moment ausgeführt wird!
STRG+F1	Während sich der Cursor über einem Baustein befindet: Direkter Aufruf der Online-Hilfe zu diesem Baustein.
F6	Wechseln ins Hauptprogramm zum Bearbeiten
F7	Wechseln in ein Unterprogramm zum Bearbeiten
F5	Starten des Steuerungsprogramms im Online-Modus
F8	Wechsel in Init-Modus
F9	Stoppen des Steuerungsprogramms
F10	Projekt speichern
SHIFT+F5	Fenster überlappend anordnen
SHIFT+F4	Fenster nebeneinander anordnen

2. Wechsel in die verschiedene Bearbeitungsmodi

Im Editiermodus kann mit den Ziffern 1–5 sehr schnell zwischen den verschiedenen Bearbeitungsmodi umgeschaltet werden:

1	Bausteine einfügen
2	Bausteine löschen
3	Bausteine ersetzen
4	Linie zeichnen
5	Linie löschen

3. Tastaturkombinationen beim Bearbeiten von Bausteinen und Verbindungen

STRG + li. Maustaste	Im Bearbeitungsmodus Bausteine einfügen lassen sich damit einzelne Bausteine markieren. Im Bearbeitungsmodus Linie zeichnen können damit Linien verschoben werden!
SHIFT + li. Maustaste	Gruppe von Bausteinen markieren
STRG + A	Alles markieren
STRG + U	Markierung aufheben
STRG + X	Markierte Bausteine ausschneiden
STRG + C	Markierte Bausteine kopieren
STRG + V	Kopierte Bausteine einfügen
ENTF	Markierte Bausteine löschen

4. Zoomen

Mit den Tasten + und - des Ziffernblocks lässt sich sehr schnell der Zoomfaktor verändern und damit die Darstellung vergrößern bzw. verkleinern.

5. Interfacediagnose

Mit linker bzw. rechter Maustaste lässt sich der Ausgang für die Dauer des Mausklicks einschalten. Wird zuerst die `STRG-Taste` gedrückt und dann auf den Ausgang geklickt, bleibt er eingeschaltet. Ausschalten lässt er sich mit einem erneuten Mausklick.

10. Hier spricht der Profi – Hinweise, Tipps und Tricks

10.1 Was bedeutet eigentlich "Zykluszeit"?

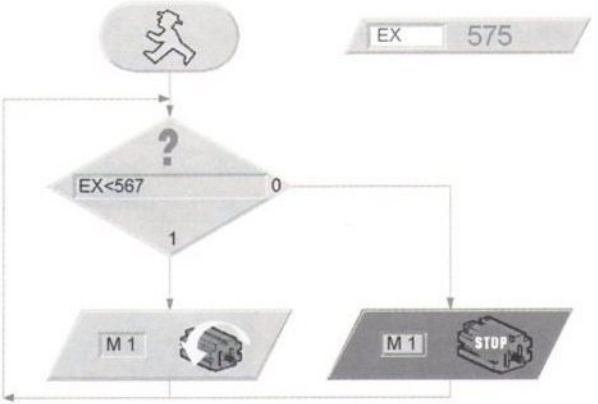
Die Zykluszeit ist die Zeit in ms, die von LLWin im "Online-Betrieb" benötigt wird, um einen Baustein zu durchlaufen. Im Normalfall ist sie auf 10 ms voreingestellt. Bei größeren Programmen und sehr langsamen PCs kann es erforderlich werden, die Zykluszeit zu vergrößern, da sonst die Windows-Oberfläche zu wenig Rechenzeit erhält. Muss man auf eine Tastatureingabe mehrere Sekunden warten, sollte man versuchen, dieses Programm mit einer größeren Zykluszeit zu betreiben. Bei Verwendung eines Interfaces kann die Zykluszeit auf maximal 100 ms eingestellt werden. Wird kein Interface angeschlossen, ist 1000 ms der maximale Wert. Allerdings muss beachtet werden, dass bei größeren Zykluszeiten der Ablaufplan langsamer abgearbeitet wird, was beim Zählen von schnellen Impulsen zu Fehlern führen kann. Dann wird z. B. eine programmierte Position nicht mehr genau angefahren (siehe auch Kapitel 8.5 `Optionen Interfaceeinstellungen`).

10.2 Was heißt "Aktualisierung der Analogwerte"?

Im Menü `Optionen Interfaceeinstellungen` kann, bei Verwendung des Intelligent Interface, unter "Einstellungen" der Parameter "Aktualisierung der Analogwerte nach x Abfragen" eingestellt werden.
Hintergrund hierzu ist folgender: Das Einlesen der Analogwerte EX und EY nimmt im Intelligent Interface sehr viel Rechenkapazität in Anspruch. Damit es beim gleichzeitigen Zählen von schnellen Impulsen nicht zu Fehlern kommt, lässt sich die Häufigkeit, mit der die Analogwerte aktualisiert werden, verändern. Standardmäßig steht dieser Wert auf 10. Das bedeutet, dass ein Analogwert, der an einer oder mehreren Stellen eines Ablaufs abgefragt wird, nur bei jeder 10. Abfrage aktualisiert wird.

Im Kapitel 5.3 wird für das Einlesen des analogen Temperaturwerts ein eigener Ablauf verwendet, in welchem der Analogwert einer Variablen Var1 zugewiesen wird. Diese Programmierweise gewährleistet, dass der Analogwert, auch wenn er nur jedes 10. Mal abgefragt wird, stets aktuell ist, ohne dass es im Ablauf, in dem er (bzw. Var1) verarbeitet wird, zu Fehlern kommt.

Das Einlesen des Temperaturwertes könnte auch innerhalb des Regelkreises selbst geschehen. Dann sähe der Ablaufplan folgendermaßen aus: ▶



Ist nun in den `INTERFACE-EINSTELLUNGEN` für die Analogwertaktualisierung "10" eingestellt, bedeutet dies, dass EX nur bei jedem 10. Durchlauf des Ablaufplans aktualisiert wird! Will man

erreichen, dass der Ablauf schneller auf eine Änderung des Analogwerts reagiert, stellt man die Analogwertaktualisierung auf 1. Dann erhält EX tatsächlich bei jedem Durchlauf den aktuellen Analogwert vom Interface. In diesem Beispiel wäre dies auch problemlos möglich. Müssten jedoch im Steuerungsprogramm gleichzeitig Impulse gezählt werden, könnte es passieren, dass Impulse "verschluckt" werden.

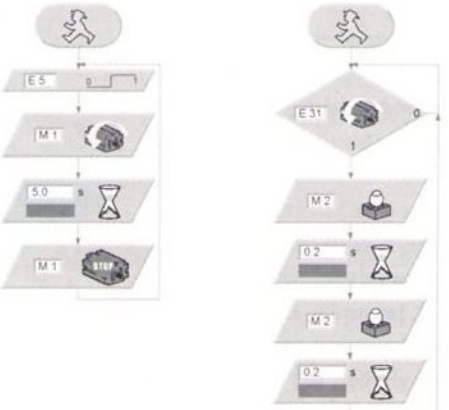
Deshalb gilt grundsätzlich:
Werden Analogwerte in einem Projekt nur sehr selten abgefragt (z. B. nur einmal in einem Ablauf, der insgesamt eine Minute dauert), dann kann man die Aktualisierungsrate getrost auf "1" stellen. Denn stünde der Wert auf "10", würde der Analogwert ja nur alle 10 Minuten aktualisiert.
Fragt man hingegen einen Analogwert ständig ab, sollte man den Wert höher stellen (z. B. auf 20), um das Interface zu entlasten.

Hinweis: Der Baustein "Werte anzeigen" fragt keine Analogwerte am Interface ab. Er zeigt lediglich den Analogwert an, den LLWin in diesem Moment verwendet.

10.3 Das Abfragen der Motorzustände

Im Kapitel 5 wurde das gegenseitige Steuern zweier Abläufe mit Hilfe von Variablen gezeigt. Eine weitere Möglichkeit, einen Ablauf in Abhängigkeit eines anderen zu steuern, besteht darin, im Baustein "Eingang" direkt die Drehrichtung der Motoren abzufragen.
In den Eingängen E31-E38 und E41-E48 speichert LLWin intern die Drehrichtung der Motoren, die am Interface betrieben werden. E31-E38 stehen für die linke Drehrichtung der Motoren M1-M8, E41-E48 für die rechte Drehrichtung von M1-M8. Dreht sich z. B. der Motor M1 nach links, dann ist der Eingang E31 gesetzt (1) E41 ist 0. Dreht sich M2 nach rechts, dann ist E42 gesetzt (1) usw. Im folgenden Beispiel soll eine Lampe, die an M2 angeschlossen ist, immer dann blinken, wenn sich der Motor an M1 nach links dreht. Das Projekt, bestehend aus zwei Abläufen, sieht dann so aus:

Im linken Ablauf wird der Motor durch Betätigen von E5 eingeschaltet, läuft 5 Sekunden lang und schaltet wieder ab. Im rechten Ablauf wird abgefragt, ob der Eingang E31 "1" ist, was immer der Fall ist, wenn sich M1 nach links dreht. Dann blinkt die Lampe an M2. Ist E31 "0", bleibt die Lampe aus.



10.4 Bausteine sparen – Effizienteres Programmieren in LLWin

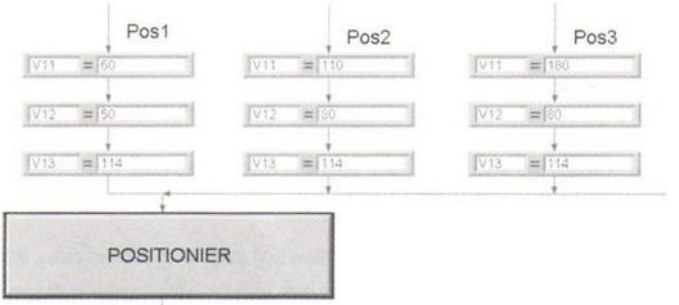
Erstellt man in LLWin sehr große Projekte, gelangt man, besonders wenn man das Programm im Download-Modus des Intelligent Interface betreiben möchte, irgendwann an die Grenzen des Systems. Entweder ist der Prozessor des Interfaces überlastet und wird langsamer, so dass z. B. nicht mehr alle Zählimpulse erfasst werden. Oder aber LLWin meldet, das Programm ist zu groß, der Speicher voll. Dann enthält das Projekt zu viele Bausteine. Zwar wurde der Speicher, den LLWin 3.0 im Online-Betrieb zur Verfügung hat, gegenüber der Version 2.10 verzehnfacht, die Kapazität auf dem Interface ist aber unverändert. Zunächst ärgert man sich vielleicht, dass die Software oder das Interface nicht mehr Kapazität zur Verfügung stellt. Allerdings kann man durch einen effizienteren Programmierstil auch komplexe Projekte so gestalten, dass sie selbst im Download-Modus des Interfaces funktionieren.

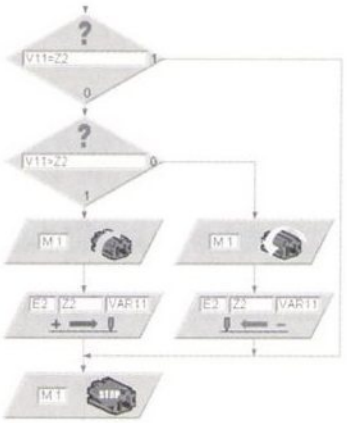
Dazu sollte man zunächst Folgendes wissen:
Verwendet man Unterprogramme, dann werden die darin enthaltenen Bausteine jedes Mal, wenn der Unterprogrammbaustein im Hauptprogramm eingefügt wird, ebenfalls intern in das LLWin-Programm eingefügt. Häufiges Einfügen sehr großer Unterprogramme führt also zu einer wahren Explosion der Bausteinanzahl in einem Projekt. Aus diesem Grund sollte das Projekt so gestaltet werden, dass jedes Unterprogramm möglichst nur einmal aufgerufen wird.

Dazu bedient man sich der sog. zustandsorientierten Programmierung. Dabei werden, abhängig vom Zustand verschiedener Variablen, bestimmte Aktionen ausgeführt.
Wie das aussehen kann, soll anhand des nächsten Beispiels erläutert werden.
Es handelt sich hierbei um ein Programm für das Modell "Rob4" aus dem Baukasten "Industry Robots". Dieses Programm wird auf der LLWin 3.0-CD mitgeliefert. Es befindet sich im Verzeichnis ...\Industry Robots\ und heißt rob4.mdl. Da das Projekt sehr umfangreich ist, wird es nicht komplett im Handbuch abgedruckt. Es werden hier nur die wichtigsten Ausschnitte dargestellt.

Programmbeschreibung:
Der Knickarm-Roboter Rob4 fährt zunächst in seine Ausgangsposition. Dann holt er eine gelbe Tonne von Position 1 und setzt sie auf Position 2 ab. Danach holt er die zweite Tonne von Position 3 und stapelt sie auf die erste Tonne (Position 4). Anschließend setzt er die Tonnen zurück auf die Positionen 1 und 3. Danach beginnt er wieder von vorne.

Das Programm ist so angelegt, dass das Unterprogramm "Positionier" die drei Achsen des Roboters (Motoren M1-M3) gleichzeitig in die jeweils gewünschte Position fährt. Die Sollpositionen für die Motoren M1-M3 werden durch die Variablen Var11-Var13 vorgegeben.



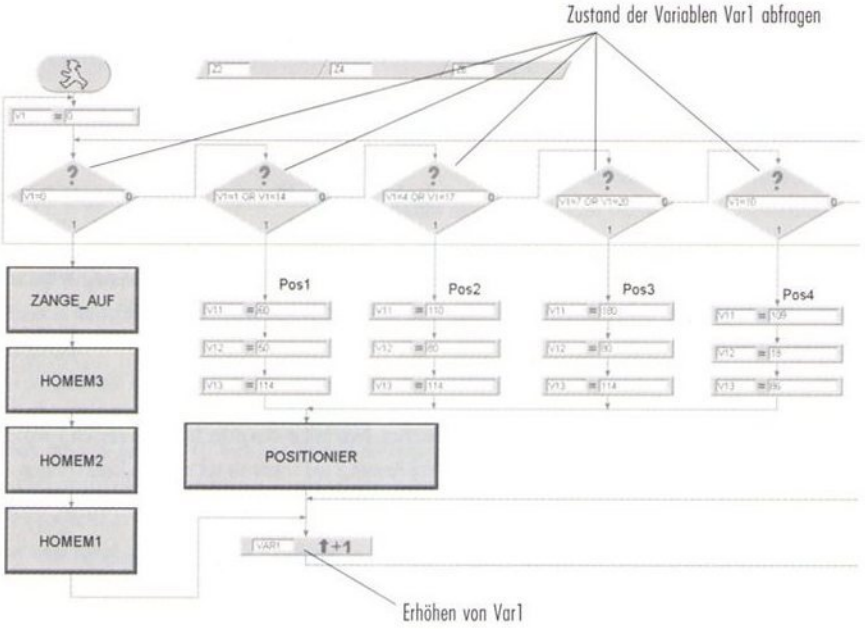


Im Unterprogramm "Positionier", das nur einmal im Hauptprogramm eingefügt ist, werden diese Variablen dann verarbeitet.

◀ Ausschnitt aus dem Unterprogramm Positionier, Ablauf für Motor 1:

Hier fährt z. B. der Motor M1 entweder nach links oder nach rechts, abhängig davon, ob die aktuelle Position des Motors (gespeichert in den Zählvariablen Z2) größer oder kleiner ist als die neue Sollposition Var11. Ist im Baustein Position der neue Endwert Var11 erreicht, stoppt der Motor.

Auf welche Position der Roboter fahren soll, hängt vom Zustand der Variablen Var1 im Hauptprogramm ab. Nach jeder erfolgten Positionierung wird Var1 um eins erhöht.



Ist Var1=0, fährt der Roboter in seine Ausgangsposition, ist Var1=1 fährt der Roboter zu Position 1 usw.
Es wird also immer nur die Variable Var1 abgefragt, für jede Position die Variablen Var11-Var13 definiert und dann das Unterprogramm Position ausgeführt.

Auf diese Weise erreicht man, dass dieses sehr umfangreiche Projekt trotzdem noch im Download-Modus des Intelligent Interface abgearbeitet werden kann. Würde man statt dessen in einem Ablauf für jede neue Position des Roboters das Unterprogramm Positionier erneut einfügen, wäre die Speichergrenze wahrscheinlich schnell erreicht.

Vielleicht ist diese Art der Programmierung am Anfang nicht ganz einfach nachzuvollziehen. Man muss es einfach ausprobieren. Und schließlich soll dieses Kapitel ja auch die Profis unter den LLWin-Programmierern ansprechen.

11. Was geschieht mit alten LLWin-Projekten?

Alte Projekte, die mit der Version 2.1 erstellt wurden, können in der Version 3.0 geladen und gestartet werden.

Hinweis: Speichert man sie jedoch unter LLWin 3.0 ab, lassen sie sich nicht mehr in LLWin 2.1 öffnen.

In LLWin 3.0 können Projekte aus der Version 2.1 bearbeitet werden, d. h. Bausteine lassen sich verschieben, kopieren, ausschneiden, einfügen. Neue LLWin 3.0-Bausteine aus dem Bausteinfenster können hinzugefügt werden. Zusätzliche alte Bausteine können nicht hinzugefügt werden. Dies wäre aber auch unnötig, da alle Funktionen der alten Bausteine in den neuen Bausteinen ebenfalls vorhanden sind.

Alte Bausteine können auch nicht in andere Projekte, die mit LLWin 3.0 erstellt wurden, kopiert werden.

12. Stichwortverzeichnis

Software LLWin 3.0

- Grafisches Programmiersystem für Windows 95, 98, 2000, NT
- 32-bit-Software
- Echtzeitverarbeitung
- frei programmierbar

Systemvoraussetzungen:

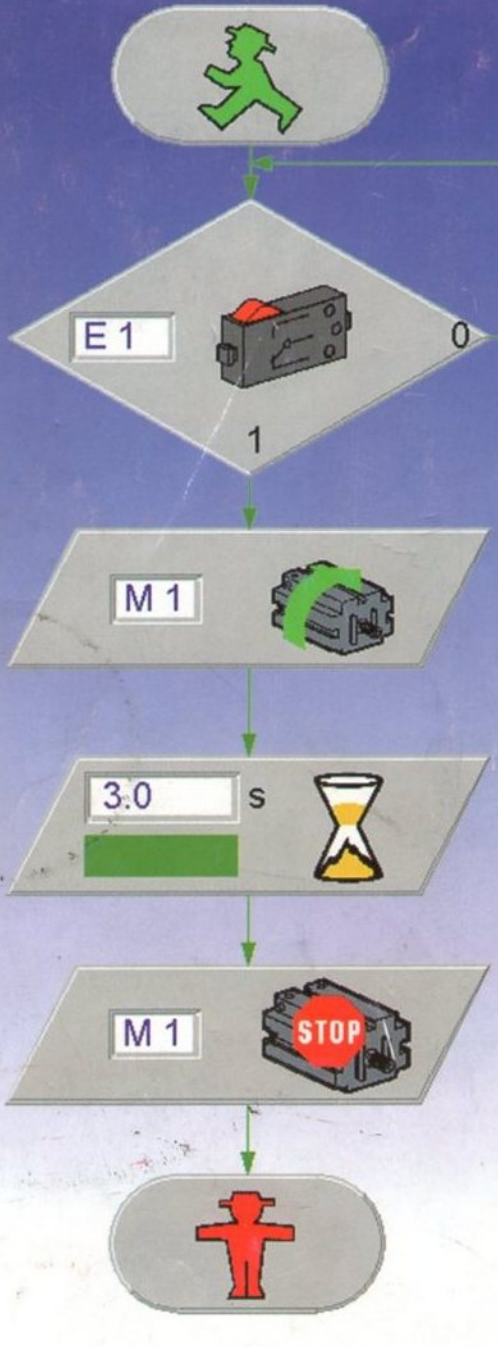
- IBM-kompatibler PC mit Pentium-Prozessor, ab 100MHz
- 32MB RAM
- 20 MB freie Speicherkapazität auf der Festplatte
- CD-ROM Laufwerk
- Freie serielle RS232-Schnittstelle COM1-COM4 für Intelligent Interface Art.-Nr. 30402 bzw. freie parallele Schnittstelle LPT1 oder LPT2 für paralleles Interface Art.-Nr. 30520
- Microsoft® Windows 95, 98, 2000, NT

Software LLWin 3.0

- Graphical programming system for Windows 95, 98, 2000, NT
- 32-bit software
- Real-time processing
- Freely programmable

System requirements:

- IBM-compatible PC with Pentium processor with at least 100MHz
- 32 MB RAM
- 20 MB free storage capacity on hard disk
- CD-ROM drive
- A free serial port COM1-COM4 for Intelligent Interface part no. 30402 or a parallel port LPT1 or LPT2 for parallel Interface part no. 30520
- Microsoft® Windows 95, 98, 2000, NT



Art. No. 30 407

4 006209 304075

fischerwerke Artur Fischer GmbH & Co. KG
Weinhalde 14-18, D-72178 Waldachtal
Telefon 0 74 43/12-43 69
Telefax 0 74 43/12-45 91
http://www.fischertechnik.de
email: fischertechnik-Service@fischerwerke.de

fischertechnik®

Printed in Germany