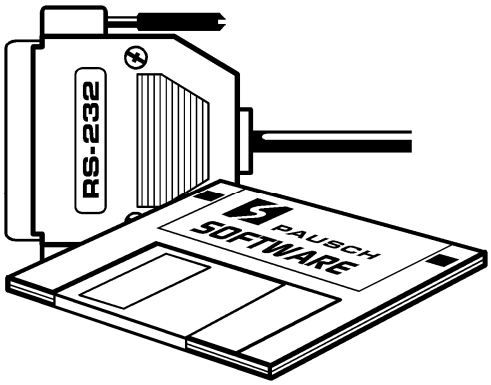
NIVCOMP Sprachbeschreibung



Compiler für den frei programmierbaren Niveauregler UNINIV

In diesem Dokument erfahren Sie, wie Sie eigene Programme für das UNINIV erstellen können. Hier steht, wie das UNINIV arbeitet, was es kann, wie die Programmiersprache aussieht und wie man den Compiler bedient. Ganz hinten finden Sie dann noch Programmbeispiele.

Wichtig:

Um das Prinzip der UNINIV-Sprache zu verstehen, lesen Sie sich bitte genau den Teil „Der Runde Tisch“ durch.

PAUSCH GmbH, A-2441 Mitterndorf ad Fischa, Moosgasse 10, T: +43/2234/738 66–0, F: –8, info@pausch.at, www.pausch.at

technische Daten des UNINIV

Eingänge

Es können **4 Sonden** angeschlossen werden, die Wasser oder Luft melden. Wenn die +12V Klemmen bestückt sind, können neben der Sonde NIVGAL (NIVTH, NIVFILM) noch die Sonden NIVOPT und NIVCAP angeschlossen werden. Wenn der Sondeneingang offen ist bzw. mehr als 4V anliegen, interpretiert das UNINIV als Luft. Wenn der Eingang auf GND oder Erde liegt, beziehungsweise unter 3.5V anliegen, sieht das UNINIV Wasser. Sie können daher auch einen Schwimmerschalter usw. anschließen. Die Logik (was nun als Luft bzw. Wasser gesehen wird) ist natürlich frei programmierbar.

Ausgänge

Es stehen **3 Relais** mit Wechselkontakten zur Verfügung. Die Wechselkontakte sind potentialfrei auf die Klemmen geführt. So ist eine maximale Flexibilität gegeben. Die maximale Belastbarkeit ist 800VA bei 250V. Wenn Sie eine größere Last schalten müssen, verwenden Sie einen Schütz !

Programm

Das UNINIV ist mit einer einfachen Sprache frei programmierbar. Es stehen Timer (Stoppuhren), Abfragen, Sondeneingänge, Relais, programmierbare Fehlerabfragen, usw. zur Verfügung:

Programmgröße

Anzahl Programme: *2*
Schreibschutz: *für Programm Nr. 2*

Anmerkung: Im UNINIV können gleichzeitig 2 Programme gespeichert werden. Über den „Programm-Auswahl-Jumper“ können Sie aussuchen, welches Programm UNINIV ausgeführt. Mit dem „Schreibschutz-Jumper“ können Sie das Programm Nr. 2 vor Veränderungen schützen.

Max. Programmzeilen pro Programm: *338 Zeilen* Effektivcode.

(3 Byte / Zeile, insges 1 kByte / Prg – 10 Byte für Header = 1024-10 / 3 = 338 Zeilen Effektivcode).

Ein-Ausgänge und Fehlermeldungen

Anzahl Sonden: *4*
Anzahl Relais: *3*
Max. Anzahl Fehler: *5 (+ ein interner Fehler)*

Timer

Max. Anzahl Delay-Timer: *4*
Max. Anzahl Was-Timer: *8*
Max. Fehler bei WAS und DEL Timer: *-1 Sek*, *+0.05 Sek / Programmzeile*

Zeiterfassung

Max. Zeitschritte: *254 Zeiteinheiten*
Max. Zeitauflösung: *1 Sek*. (TIME_RESOLUTION = 1)
Min. Zeitauflösung: *254 Sek*. (TIME_RESOLUTION = 255)

Max. darstellbare Zeit = ca. *18 Stunden*.
254 Zeiteinheiten * 254 Sek Auflösung = 64516 Sek
Min. Sonden-Software-Verzögerung: *0.0 Sek* (SOND_INTERIA=0.0)
Max. Sonden-Software-Verzögerung: *25.4 Sek* (SOND_INTERIA=25.4)
Max. Fehler Sonden-Software-Verzögerung: *-0.1 Sek*
Sonden-Hardware-Verzögerung: ca. *0.2 Sek*

Bedienelemente auf der Frontblende

Taster:

Wenn kein Fehler (Fehler-LED ist aus):

- Kurzer Druck schaltet das Gerät aus bzw. ein. Der eingestellte Zusand bleibt auch nach Stromausfall erhalten.
- 3 Sekunden halten, das Gerät schaltet in den Testmodus. Kurzes Drücken im Testmodus schaltet das zu testende Relais weiter. Wenn das letzte Relais eingeschaltet wurde, beendet ein weiteres kurzes Drücken den Testmodus wieder.

Wenn ein (programmierter) Fehler (Fehler-LED blinkt):

- Kurzer Druck quittiert den Fehler (wenn Ihr UNINIV-Programm das zuläßt).
- 3 Sekunden halten, schaltet das Gerät aus.

Grüne Betriebs LED:

- Flackern: Das UNINIV läuft (das Gerät arbeitet das Programm ab).
- Aus: Das UNINIV ist abgeschaltet (alle Relais und LED´s sind aus).
- Dauerleuchten: Das UNINIV ist im Testmodus (nacheinander werden die Relais eingeschaltet). Wenn im Testmodus nicht manuell zum nächsten Relais weitergeschaltet wird, erfolgt eine automatische Weiterschaltung alle 4 Minuten. Das ist eine Sicherheit, falls vergessen wird, den Testmodus zu beenden.

Rote Fehler LED:

- Aus: Es liegt kein Fehler an.
- Blinken: Das Programm hat einen Fehler erkannt. Die Fehlernummer ist ablesbar, wenn man die Blinker nach der 2 Sek. Pause zählt. Beispiel: kurz-kurz-kurz-lang .....Fehler Nr. 3

Orange Sonden LED´s:

Wenn Sie Pausch-Sonden anschließen, leuchtet die entsprechende LED, wenn die Sonde Wasserkontakt hat (siehe Punkt „Eingänge“ in diesem Kapitel)

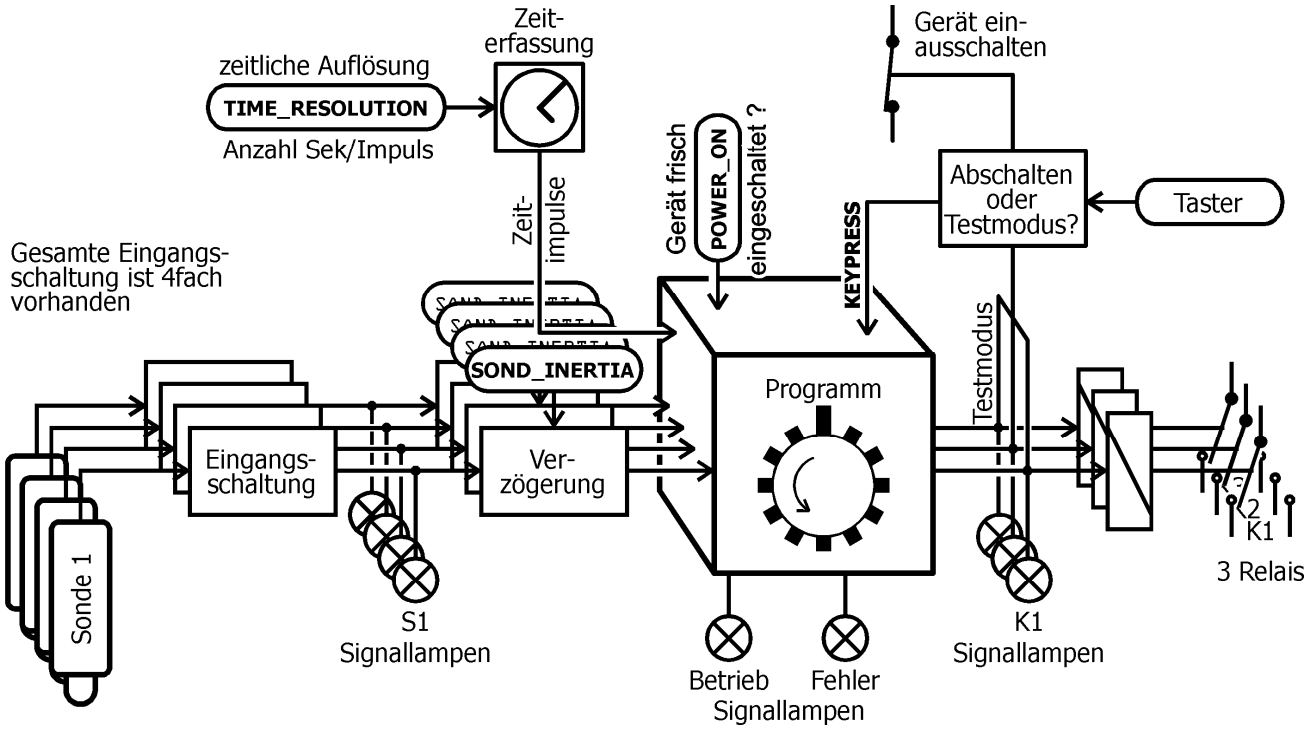
Relais-LED:

Wenn das Relais einschaltet (auf den Arbeitskontakt umschaltet), leuchtet die entsprechende LED auf.

Die UNINIV Hardware

Herzstück des UNINIV ist das Programm. Stellen Sie es sich noch als eine „Black Box“ vor. Diese Box hat Eingänge (Sonden, Taster, die Zeit) und Ausgänge (Signallampen, Relais). Was das UNINIV mit den Ausgängen in Abhängigkeit der Eingänge tut, hängt also nur vom Programm in der „Black Box“ ab.
Das Programm können Sie selber erstellen !

So sieht das Innere des UNINIV symbolisch dargestellt aus:



1. Die **Sonden** erfassen Wasser oder Luft. Es können Schwimmerschalter, Druckschalter, galvanische Sonden (NIVGAL, Stabelektroden), optische Sonden (NIVOPT) und kapazitive Sonden (NIVCAP) angeschlossen werden.

2. Die **Eingangsschaltung** verstärkt das Sondensignal und schützt vor eingespeisten Überspannungen, damit es weiterverarbeitet werden kann.

3. Die **Sonden-Signallampen** zeigen an, welche Signale die Sonden liefern (Wasserkontakt oder nicht).

4. Die **Verzögerung** dient zum Herausfiltern von Wellen. Die Verzögerungsdauer kann mit der variablen SOND_INERTIA eingestellt werden (in 1/10 Sek. Schritten).

5. Ihr **Programm** bestimmt, wie die Ausgänge (Relais 1 bis 3 und Fehler Signallampe) auf die Eingänge (Sonden 1 bis 4, Taste auf der Front, POWER_ON Flag) reagieren. Wie das Programm abgearbeitet wird, können sie unter „Der runde Tisch“ nachlesen.

6. Der **Taster** auf der Frontblende hat verschiedene Funktionen. Wenn Ihr Programm keinen Fehler ausgibt, führt ein kurzes Drücken zum Ein- bzw. Auschalten des UNINIV. Wenn die Taste gehalten wird, wird der Testmodus aktiviert. Jedes Relais wird kurz eingeschaltet. Das ist zum Überprüfen der Installation sehr praktisch. Wenn Ihr Programm einen Fehler ausgibt (die Fehler-Signallampe blinkt), kann Ihr Programm die Taste abfragen zum Quittieren des Fehlers

7. Die **Relais-Signallampen** zeigen an, ob die Relais ein- oder ausgeschaltet sind.

8. Die **Relais** haben potentialfreie Wechselkontakte. Sie können Motor- und Magnetventile, Pumpen, usw. schalten.

9. **POWER_ON** wird von der UNINIV-Hardware beim ersten Durchlauf nach dem Einschalten automatisch auf 1 gesetzt. Danach bleibt es immer 0. Ihre Software kann das POWER_ON – Flag dazu verwenden, um definierte Relais - Zustände nach dem Einschalten der Anlage zu gewährleisten.

10. Die **TIME_RESOLUTION** bestimmt, wie rasch die innere Uhr des UNINIV läuft. Alle Zeitangaben, die Sie im Programm machen (In den Timer Befehlen DEL und WAS), beziehen sich auf Zeitimpulse. Wenn Sie im Programm TIME_RESOLUTION auf 1 setzen, ist ein Zeitimpuls eine Sekunde lang. Wenn Sie ihn auf 60 setzen, ist ein Zeitimpuls eine Minute lang.

Die Software

Wozu das denn ?

Das UNINIV ist eine Speicher-Programmierbare-Steuerung (SPS). Damit es etwas Sinnvolles macht, muß es mit einer Reihe von Befehlen (=Programm) gefüttert werden. Diese Reihe von Befehlen bezeichnet man als Software bzw. Programm. Die Software wird am PC erstellt, und in das UNINIV geladen. Dort bleibt es dann für immer gespeichert. Wenn Sie das Programm ändern wollen, können Sie es, so oft Sie wollen, neu einspeichern.

UNINIV Befehle

Die UNINIV Sprache besteht aus nur drei Befehlen. Trotzdem können auch komplexe Aufgaben gelöst werden. Es gibt Befehle, mit denen ein Zustand abgefragt werden kann (z.B.: „ist die Sonde 1 in der Luft ?). Es gibt Befehle, mit denen man Zeit erfassen kann (z.B.: „War die Sonde 2 mehr als 30 Sekunden im Wasser ?“ oder „Schalte Relais K3 10 Sekunden lang ein, wenn S3 Wasserkontakt meldet“).
Hier die Befehle mit einer kurzen Beschreibung, wofür sie verwendet werden:

- IF(): Dient für Abfragen. Das englische Wort IF bedeutet auf deutsch WENN.
  Hier ein Beispiel: Sie wollen, daß Relais K1 abschaltet, wenn die Sonde S3 Wasserkontakt meldet. Die Befehlszeile, die das macht, würde so aussehen:
  ```
  K1_OFF = IF(S3=Wasser)
  ```

- WAS(): Dient zur Zeiterfassung. Das englische Wort WAS bedeutet auf deutsch WAR.
  Hier ein Beispiel: Sie wollen, daß Relais K1 abschaltet, wenn die Sonde S3 mindestens 10 Sekunden lang Wasserkontakt gemeldet hat. Die Befehlszeile, die das macht, würde so aussehen:
  ```
  K1_OFF = WAS(S3=Wasser, 10)
  ```

- DEL(): Dient zum Warten. Das englische Wort DELAY bedeutet auf deutsch VERZÖGERUNG.
  Hier ein Beispiel: Sie wollen, daß Relais K1 10 Sekunden lang abschaltet, ab dem Zeitpunkt, die Sonde S3

mit Wasser in Berührung kommt. Die Befehlszeile, die das macht, würde so aussehen:

```
K1_OFF = DEL(S3->Wasser, 10)
```

Zuweisungen

Wie Sie bei den oberen Bespielen schon gesehen haben, kann man die Befehle nur im Zusammenhang mit Zuweisungen verwenden. Der Befehl liefert ein Ergebnis, das dann mit dem ‚=' – Zeichen z.B. einem Relais zugewiesen wird.

Sehen wir uns das anhand eines Beispiels an:

```
K1_OFF = IF(S3=Wasser)
```

Der Befehl IF(S3=Wasser) liefert ein Ergebnis, das dem temporären Schalter K1_OFF zugewiesen wird. Wenn die Abfrage zutrifft, das heißt wenn S3 im Wasser ist, ist das Ergebnis der IF-Abfrage 1, sonst 0. Wenn K1_OFF den Wert 1 hat, schaltet es das Relais ab. Wenn es den Wert 0 hat, schaltet es das Relais nicht ab.

Es gibt nur 0 und 1

Intern arbeitet das UNINIV nur mit 0 und 1. Wenn eine Sonde in der Luft ist, liefert sie dem UNINIV den Wert 0. Ist sie im Wasser, liefert sie den Wert 1. Man kann also sagen, daß Luft der 0 und Wasser der 1 entspricht. Ähnlich verhält es sich mit Fehlern. Wenn z.B. der Fehler 3 angezeigt werden soll, muß ihm eine 1 zugewiesen werden. Möchte man dann mit einer IF-Abfrage prüfen, ob der Fehler 5 aktiv ist, fragt man einfach, ob Fehler 5 = 1 ist. Man kann also sagen, daß 1 wahr und 0 unwahr entspricht.

Hier einige Entsprechungen:

Für 0	**Für 1**
Luft, Falsch, Nein, Aus	Wasser, Wahr, Ja, Ein

Präprozessor

Damit das Programm gut lesbar ist, können Sie mit der ‚#define' – Anweisung Text automatisch ersetzen lassen: #define Wasser 1
Wenn Sie dann in Ihrem Programm das Wort ‚Wasser' verwenden, wird es automatisch durch die 0 ersetzt. Was Sie noch alles mit der ‚#define' Direktive machen können, steht weiter unten.

UND-Verknüpfungen

Mit der UND-Verknüpfung können mehrere Bedingungen so verknüpft werden, daß die Zuweisung nur dann stattfindet, wenn alle Bedingungen wahr sind. Man kann z.B. formulieren, daß das Relais K1 nur dann einschalten soll, wenn die Sonde S2 UND S3 in der Luft ist.

Sehen wir uns das anhand eines Beispiels an:

```
K1_ON   = IF(S2 = LUFT)      // Einschalten, wenn beide -
```

```
K1_ON &= IF(S3 = LUFT)      // Sonden in der Luft sind
```

Nur wenn S2 UND S3 in der Luft sind, wird K1_ON auf 1 gesetzt (schaltet das Relais 1 ein).

Anmerkung: Die UND-Verknüpfungen müssen untereinander stehen (es darf keine Zeile dazwischen mit einer anderen Anweisung stehen). Dem Ergebnis der UND-Verknüpfung kann nur an eine temporäre Variable zugewiesen werden. „K1_ON = IF(S2 = LUFT)
K2_ON &= IF(S3 = LUFT)“ ist also falsch, da einmal K1_ON, dann K2_ON angegeben ist.

Programmieren einer Hysterese mit UND-Verknüpfungen

Hysterese bedeutet, daß in einem gewissen Bereich nicht auf Änderungen reagiert werden soll. Nur wenn die Änderung größer als der Bereich ist, soll eine Reaktion erfolgen. Bei einer Niveauregelung (Nachfüllen, wenn zu wenig Wasser da ist), ist eine Hysterese sinnvoll. Wenn ein unterer Füllstand unterschritten wird, soll ein Ventil öffnen. Es soll aber nicht gleich wieder schließen, wenn der untere Füllstand nicht mehr unterschritten ist, sondern erst dann, wenn ein oberer Füllstand erreicht ist. Dadurch muß das Ventil nicht ständig kurz aufmachen (wenn Sie bei Ihrem Auto Scheibenwischflüssigkeit nachfüllen, füllen Sie den Vorratsbehälter ja auch voll, um nicht gleich wieder nachfüllen zu müssen).

Eine Hysterese kann so realisiert werden:

```
K1_ON   = IF(ObereFüll_S2 = LUFT)       // Einschalten, wenn beide -
K1_ON &= IF(UntereFüll_S3 =LUFT)        // Sonden in der Luft sind

K1_OFF   = IF(ObereFüll_S2=WASSER)      // Wieder ausschalten, wenn -
K1_OFF &= IF(UntereFüll_S3=WASSER)      // beide Sonden im Wasser sind
```

Der Bereich mit richtigem Füllstand erstreckt sich von Sonde S2 (die untere) bis zur Sonde S3 (die obere). Wenn der Wasserstand zwischen S2 und S3 liegt, wird das Ventil (mit K1) weder ein- noch ausgeschaltet – es bleibt wie es ist. Nur wenn der Wasserstand unter S2 liegt, wird das Ventil geöffnet. Es schließt erst dann wieder, wenn der Wasserstand bis S3 gestiegen ist.

Temporäre Schalter

Einen Schalter können Sie ein- oder ausschalten. Sie können ihn aber auch so lassen, wie er ist (ein oder aus). Es gibt temporäre Schalter, mit denen etwas ein-, andere, mit denen etwas ausgeschaltet werden kann. K1_OFF kann das Relais Nr. 1 ausschalten. K1_ON kann es einschalten.

Betrachten wir folgendes Beispiel: `K1_OFF = IF(S3=Wasser)`
Wenn die IF - Abfrage zutrifft (S3 also im Wasser ist), liefert sie 1. Dem temporären Schalter K1_OFF wird also der Wert 1 zugewiesen. Wenn am Ende des Programms K1_OFF noch immer 1 ist, wird das Relais tatsächlich physikalisch abgeschaltet (siehe der runde Tisch).

Liste mit allen temporären Schaltern

Dient für	Temp. Einschalter	Temp. Ausschalter	Bemerkung
K1	**K1_ON**	**K1_OFF**	Zum Einschalten bzw. Ausschalten des Relais K1
K2	**K2_ON**	**K2_OFF**	Zum Einschalten bzw. Ausschalten des Relais K2
K3	**K3_ON**	**K3_OFF**	Zum Einschalten bzw. Ausschalten des Relais K3
ERROR_1	**ERROR1_ON**	**ERROR1_OFF**	Zum Setzen oder Löschen des Fehlers Nummer 1
…	…	..	Bis
ERROR_5	**ERROR5_ON**	**ERROR5_OFF**	Zum Setzen oder Löschen des Fehlers Nummer 5

Der Runde Tisch

Wichtig:
Hier ist das Prinzip der UNINIV-Sprache erklärt. Nur wenn Sie es verstanden haben, können Sie die Sprache richtig anwenden !

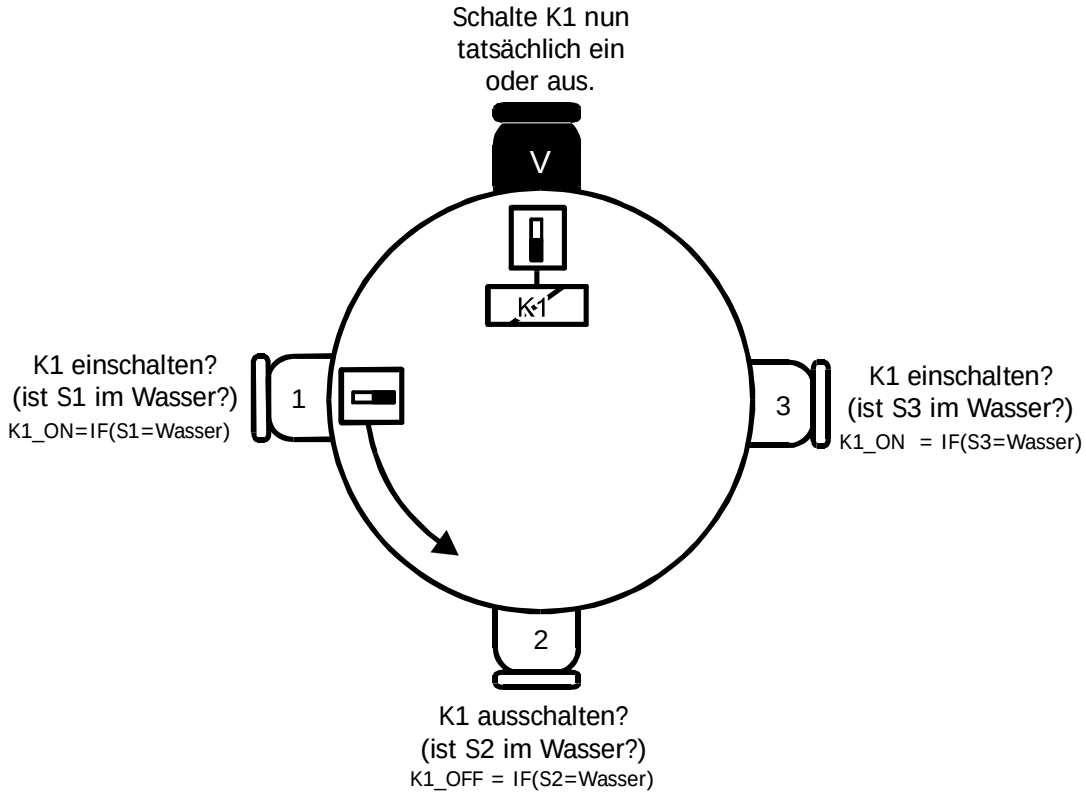
Beispiel 1:

Um die Funktionsweise der temporären Schalter zu verstehen, betrachten wir mal dieses Beispiel:

```
K1_ON  = IF(S1=Wasser)
K1_OFF = IF(S2=Wasser)
K1_ON  = IF(S3=Wasser)
```

Dieses Miniprogramm, bestehend aus 3 Zeilen, und macht folgendes: Wenn S3 im Wasser ist, schaltet K1 auf jeden Fall ein. Wenn nicht, schaltet es ein, wenn S1 im Wasser ist und S2 nicht im Wasser ist.

Wie funktioniert das nun genau ?



Stellen Sie sich einen runden Tisch vor. Am Tisch sitzen 4 Leute. Sie repräsentieren die 3 Programmzeilen und einen Vorsitzenden. Der Vorsitzende kann das Relais K1 ein- und ausschalten. Die 3 anderen können nur einen temporären Schalter, der herumgereicht wird, beeinflussen. Das Programm beginnt in der 1. Zeile, also beim 1. Sessel. Dort sitzt jemand, der den temporären Schalter dann auf 1 stellt, wenn die Sonde S1 im Wasser ist (siehe 1. Programmzeile). Er kann den Schalter also einschalten oder so lassen wie er war. Wenn er fertig ist, gibt er den temporären Schalter an seinen Sitznachbarn weiter (die Zeile 2). Er schaltet ihn aus, wenn S2 im Wasser ist. Wenn nicht, gibt er ihn ohne abzuschalten weiter an seinen Sitznachbarn (Zeile 3). Dieser prüft ob S3 im Wasser ist. Wenn ja, schaltet er den Schalter ein – wenn nicht, gibt er an seinen Sitznachbarn weiter ohne zu schalten. Sein Sitznachbar ist der Vorsitzende. Er schaut sich den temporären Schalter an und schaltet das Relais K1 ein bzw. aus, je nachdem, ob der temporäre Schalter ein- oder ausgeschaltet war. Danach gibt er den temporären Schalter an seinen Sitznachbarn weiter (Zeile 1). Das Spiel beginnt von neuem.

Anmerkung:
Bei jedem Weiterreichen (bei jeder neuen Programmzeile) wird die grüne Statuslampe umgeschaltet. Daher flackert sie, wenn das Programm läuft.

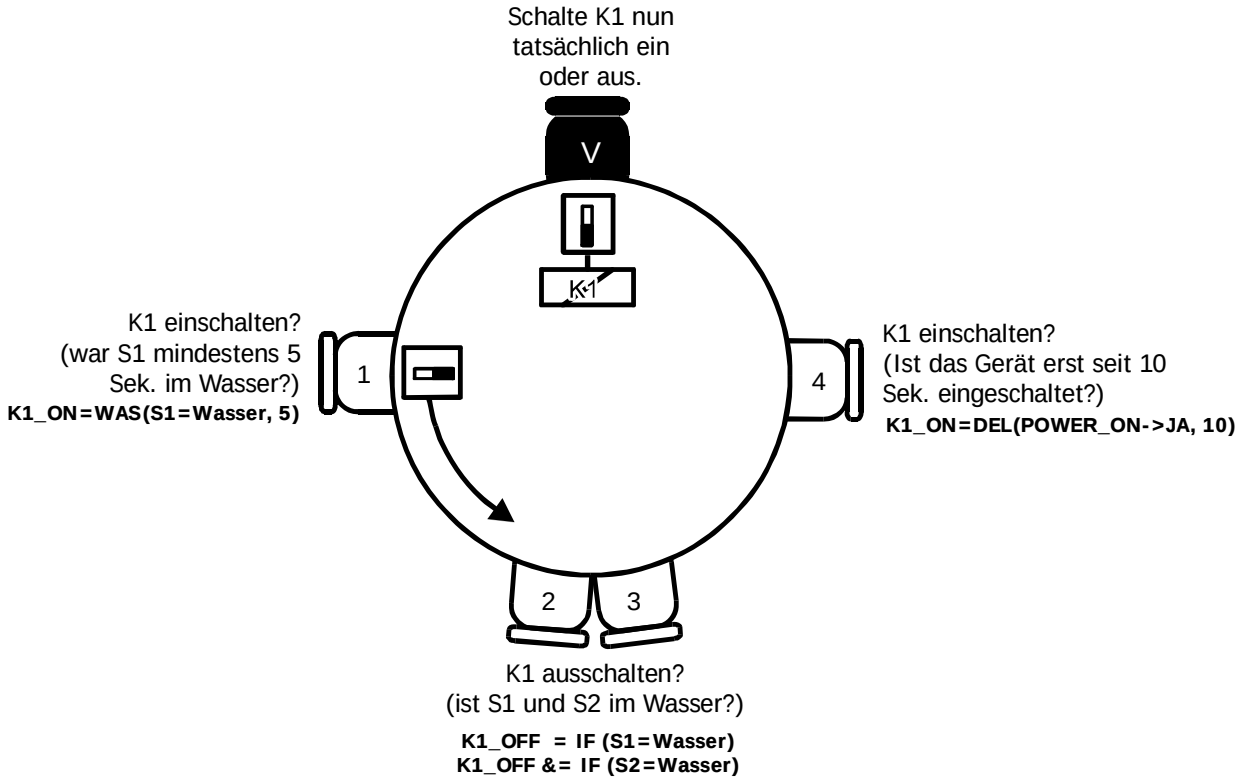
Beispiel 2:

Betrachten wir nun ein Beispiel mit Timer – Anweisungen

```
K1_ON   = WAS(S1=Wasser, 5)
K1_OFF  = IF (S1=Wasser)
K1_OFF &= IF (S2=Wasser)
K1_ON   = DEL(POWER_ON->JA, 10)
```

Dieses Miniprogramm, besteht aus 4 Zeilen und macht folgendes: Wenn das Gerät eingeschaltet wird, ist K1 auf jeden Fall für 10 Sekunden eingeschaltet (letzte Zeile). Danach schaltet es aus, sollten beide Sonden S1 und S2 im Wasser sein (Zeile 2 und 3). Wenn nicht, bleibt es eingeschaltet. Wenn es ausgeschaltet wurde, und nur S1 im Wasser ist und das mindestens für 5 Sekunden ununterbrochen, wird K1 eingeschaltet (Zeile 1).

Wie funktioniert das nun genau ?



Solange das UNINIV im Automatikbetrieb läuft, werden die temporären Schalter im Kreis gereicht. Auch bei einer DELay – Anweisung bleibt das Programm nicht stehen. Bildlich gesprochen passiert mit der Zeile: K1_ON = DEL (POWER_ON->JA, 10) folgendes: War im letzten Durchlauf S3 noch in der Luft und jetzt im Wasser ? Wenn ja, startet die Stoppuhr und schaltet den temporären Schalter für K1 aus. Beim nächsten Durchlauf wird überprüft, ob die Stoppuhr schon 10 Sekunden erfaßt hat. Wenn nicht, wird K1_OFF noch immer auf 1 gesetzt.

Schlußfolgerung aus dem runden Tisch

1. Die Reihenfolge der Programmzeilen spielt eine große Rolle. Je weiter hinten eine Anweisung im Quelltext steht, desto höher ist ihre Priorität. Schauen Sie sich das obere Beispiel an: Egal ob S2 im Wasser ist oder nicht (Zeile 2). Wenn S3 im Wasser ist, wird K1 eingeschaltet (Zeile 3).

2. Tatsächlich gibt es nicht nur einen, sondern viele temporäre Schalter. Relais K1 bis K3 und Fehler Nummer 1 bis 5 haben eigene temporäre Schalter.

Bestandteile eines UNINIV Programms

Ein UNINIV Programm besteht aus mehreren Teilen:
1. Setzen spezieller Parameter
2. Preprozessor Definitionen
3. Das eigentliche Programm.

Sehen Sie sich die Beispielprogramme an, um zu sehen, wie man die Teile praktisch verwendet.

Anmerkungen

Alles, was hinter zwei Schrägstrichen „//“ bis zum Ende der Zeile steht, wird ignoriert. Sie können also Anmerkungen, Erklärungen usw. überall in Ihrem Quelltext eintragen.
Beispiel: K1_OFF = IF(S1=LUFT) // Trockenlaufschutz

Die Teile genau betrachtet

1. Spezielle Parameter setzten:

   1.1. SOND_INERTIA (auf Deutsch Sonden Trägheit): Damit Wellen nicht stören können, wird hier angegeben, wie groß die Sondenträgheit (in Sekunden) sein soll. Erst wenn ein Wert länger als die angegebene Zeit ständig angelegen ist, wird der Wert dem Programm zur Verfügung gestellt. Die Sonden-Trägheit kann zwischen 0.0 und 25.4 Sekunden angegeben werden. Sie bezieht sich immer auf alle 4 Eingänge. Die fixe Verzögerung der Hardware von ca. einer Sekunde ist in der SOND_INERTIA noch nicht enthalten.
   Beispiel: SOND_INERTIA=5.0

   1.2. TIME_RESOLUTION (auf deutsch zeitliche Auflösung): Hier wird die interne zeitliche Auflösung eingestellt. Sie definiert, in welchen Sekunden - Schritten die Zeit gemessen wird. Das UNINIV kann Zeiten bis maximal 255 Zeitschritten erfassen. Wenn als zeitliche Auflösung z.B. 1 (eine Sekunde) eingegeben wird, kann im Programm mindestens eine Sekunde, höchstens 255 Sekunden (= 4 Minuten, 15 Sekunden) erfaßt werden. Wenn Sie jedoch als zeitliche Auflösung z.B. 240 eingeben (=

4 Minuten), kann im Programm mindestens 4 Minuten, höchstens 255 * 4 Minuten (= 17 Stunden) erfaßt werden.
Beispiel: TIME_RESOLUTION=60

2. Der Preprozessor:

Der Preprozessor ist eine Text – Ersetzfunktion, die der Compiler vor dem eigentlichen Kompilieren durchführt. Das Wort hinter dem Preprozessor-Befehl „#define“ wird durch den Text nach diesem Wort ersetzt.

Beispiel:

```
#deine EIN 1                      // Jedes Mal, wenn das Wort „EIN“ im Quelltext erscheint, wird es
                                     durch „1“ ersetzt.
#deine ZwangsEin_Löschen K1_0FF   // Jedes Mal, wenn das Wort „ZwangsEin_Löschen“ im Quelltext
                                     erscheint, wird es durch „1“ ersetzt.
```

3. Das eigentliche Programm:

Das Programm besteht aus einer Reihe von Zuweisungen. Pro Zeile darf nur eine Zuweisung stehen. Sie können nur temporären Schaltern etwas zuweisen. Was temporäre Schalter sind, können Sie weiter oben unter „Der runde Tisch lesen“

Beispielprogramme:

Übung 1: Dosierpumpen Steuerung I

Dieses Programmbeispiel illustriert die Verwendung von Timern die zum zyklischen Ein-Ausschalten verwendet werden.

Was soll die Dosierpumpen Steuerung machen ?

Solange die Sonde S4 „Trockenlauf_Sonde“ im Wasser ist, soll das „Dosierrelais“ alle 10 Sekunden eine Sekunde lang einschalten.

Und so wird´s gemacht:

Das zyklische Ein - Ausschalten wird mit nur 2 WAS – Timern realisiert.
Der eine mißt, wie lange das Relais bereits eingeschaltet ist. Wenn es länger als eine Sekunde bereits ein war, schaltet es dieser WAS-Timer aus. Der andere mißt, wie lange das Relais bereits ausgeschaltet ist. Wenn es länger als 10 Sekunden aus war, wird es wieder eingeschaltet.
Da das Programm ja ständig wiederholt wird, wird das Relais ständig alle 10 Sekunden für eine Sekunde eingeschaltet solange die Trockenlauf_Sonde nicht Luft meldet.

Die letzte Zeile definiert, wie das UNINIV beginnen soll, wenn es eingeschaltet wird.

So sieht das Programm aus, das im UNINIV läuft:

```
SOND_INERTIA =     3.00                    // Alle Sonden haben eine Trägheit von 3,00 Sekunden.
TIME_RESOLUTION =  1                       // Die kleinstmögliche Zeit, die erkannt werden kann ist,
                                           1 Sek.

// Preprozessor: Der Einfachheit halber sind in diesem Beispiel die #define – Anweisungen nicht
                                           abgedruckt.

// Impulse auf am DosierRelais erzeugen
DosierRelais_Aus = WAS(DosierRelais=Ein, 1)     //  Wenn DosierRelais schon 1 Sek. lang ein war, schalte
                                                es ab.
DosierRelais_Ein = WAS(DosierRelais=Aus, 20)    //  Wenn DosierRelais schon 20 Sek. aus war, schalte es
                                                ein.
DosierRelais_Aus = IF(Trockenlauf_Sonde=Luft)   //  DosierRelais bleibt ausgeschaltet, wenn zu wenig
                                                Flüssigkeit da ist.

// 2.4. Power On Zustand: Was soll unmittelbar nach dem Einschalten der Anlage passieren ?
```

```
DosierRelais_Aus = IF(POWER_ON=Ja)          // Beginne mit dem Aus-Zustand, falls gerade
                                            eingeschaltet wurde
```

Übung 2: Dosierpumpen Steuerung II

Dieses Programmbeispiel baut auf dem letzten Beispiel auf und zeigt, wie weitere Timer für das selbe Relais genutzt werden können.

Eine etwas kompliziertere Dosierpumpen-Steuerung:

Das Beispiel von vorhin soll etwas erweitert werden. Je mehr Wasser im Behälter ist, desto mehr soll die Dosierpumpe dosieren. Es wurde bisher nur die „Trockenlauf_Sonde“ verwendet. Die restlichen 3 Sonden sollen messen, wie viel im Behälter drinnen ist. Ich habe ihnen mit dem #define Befehl folgende Namen gegeben: „untere_Sonde“, mittlere_Sonde“ und „obere_Sonde“.

Wenn wenig Wasser im Behälter ist, also die „untere_Sonde“ Luft meldet, soll „SehrWenig_Dosieren“ ausgegeben werden.

So sieht das Programm aus:

Übung 3: Schwallwassersteuerung

Dieses Programm läuft in unserer Schallwassersteuerung NIVPOOL. Es ist recht komplex und nützt alle Aspekte der UNINIV-Sprache aus (Preprozessor, UND-Verknüpfungen, IF-Abfragen, DEL- und WAS-Timer).

Da die gesamte Dokumentation dieses Programms dieses Skript sprengen würde, ist es in der eigenen Dokumentation namens „NIVPOOL Steuerung für Schwallwasserbehälter“ untergebracht. Sie müßte auch im NIVCOMP- bzw. NIVCOMPdemo- Pakt dabei sein.

Siehe „**NIVPOOL Steuerung für Schwallwasserbehälter**“

NIVCOMP

NIVCOMP ist der Compiler für das UNINIV. Er erzeugt aus Ihrem Quelltext einen vom UNINIV ausführbaren Code und sendet ihn auf Wunsch auch gleich zum UNINIV.

Systemanforderungen, damit UNINIV funktioniert

Kurz gesagt: Läuft auf praktisch jedem PC mit DOS oder Windows.

Der Compiler ist ein DOS-Programm und stellt kaum Anforderungen an die Hardware. Er läuft auf altertümlichen XT´s genauso wie auf modernsten PC´s. Das Betriebssystem muß DOS zur Verfügung stellen. Es kann also DOS, Windows3.x oder Windows95 verwendet werden (auf WindowsNT haben wir den Compiler noch nicht getestet). Zum Senden des Programms zum UNINIV benötigt man einen freien COM-Port (RS232). Üblicherweise ist COM1 durch die Maus belegt.

Installation des UNINIV Compilers

Kopieren Sie einfach den gesamten Inhalt der Diskette (inklusive Unterverzeichnissen) auf die Festplatte in einen Ordner Ihrer Wahl. Das NIVCOMP beeinflußt keinerlei Systemdateien (INI, Registry, Config.sys, ...).

Im Unterordner „ Icons“ sind Icons (Symbole) für UNINIV-Programm-Dateien (Quelltext-, List- und Preprozessor Dateien) enthalten. Wenn Sie die Anweisungen im File „_ReadMe.txt“ befolgen, sehen Sie die im Windows95 Explorer, die UNINIV-Programm-Dateien mit den Icons. Das ist praktisch und sieht gut aus.

Deinstallation des UNINIV Compilers

Wenn Sie den Compiler nicht mehr brauchen, können Sie einfach das gesamte Verzeichnis löschen.

Bedienung des UNINIV Compilers

Der Compiler hat den Namen „NIVCOMP.exe“ (bzw. „NIVCOMPd.exe“ für das Demo).
Zum Aufruf geben Sie einfach den Namen des UNINIV-Programms in der Aufrufzeile an. Als zweites Argument können Sie dann noch zusätzlich den COM-Port angeben. Wenn Sie den COM-Port nicht beim Aufruf angegeben haben, fragt Sie NIVCOMP danach.

Beispiel: „nivcomp schwallw.niv 2“. Ergebnis: Compiliert und sendet das Programm über die COM2.

Möglichkeiten zum Aufruf des Compilers

Es gibt verschiedene Möglichkeiten unter Windows(95) den Compiler aufzurufen.
Hier nun erklärt am Beispiel des UNINIV-Programms „Schwallw.niv“ (Schwallwassersteuerung):

1. Unter DOS: Der Compiler und „Schwallw.niv“ muß sich im selben Ordner befinden. Wechseln Sie mit dem CD-Befehl in den Ordner. Tippen sie dann: „nivcomp.exe Schwallw.niv“.

2. Mit dem Windows95 Explorer: Ziehen Sie mit der Maus die Datei „Schwallw.niv“ auf die Datei „Nivcomp.exe“.
   Anmerkung: „Nivcomp.exe“ kann auch in einem ganz anderen Ordner stehen als „Schwallw.niv“. Die erzeugten Dateien schreibt der Compiler in den Ordner des „Schwallw.niv“.

3. Auch nur unter Windows95: Wenn Sie Files mit der Dateiendung .NIV mit dem Compiler verbunden haben, brauchen Sie nurmehr auf ein Quelltextfile zu klicken.
   Anmerkung: NIV-Files so mit dem Compiler verknüpfen: Einmal ein NIV-File z.B „Schwallw.niv“ anklicken und die Datei „Nivcomp.exe“ angeben bzw. siehe File „_ReadMe.txt“).

Von NIVCOMP erzeugte Dateien

Wenn ein UNINIV-Programm (Dateiendung NIV) an den Compiler übergeben wird, erzeugt dieser 3 Dateien:

1. HEX-Datei: Ist der vom UNINIV ausführbare Code, der mit dem Compiler zum UNINIV gesendet wird.

2. LST-Datei: Wenn der Compiler in Ihrem Programm einen Fehler gefunden hat, gibt er Ihnen in diesem File nähere Informationen.

3. PRE-Datei: Hier sehen Sie, wie der Compiler Ihr Programm sieht, nachdem der Preprozessor (#define) den Quelltext bearbeitet hat. Wenn Sie mit Fehlermeldungen kämpfen, schauen Sie mal in diese Datei hinein.

Vermeidung von Problemen

Bitte, lesen Sie sich die Punkte durch, um möglichen Problemen aus dem Weg zu gehen.

1. Geben Sie Ihren UNINIV-Programmen immer die Dateiendung NIV.

2. Schreiben Sie Ihr UNINIV-Programm mit einem Editor (z.B. den, der bei Windows95 dabei ist) und nicht mit einem Textverarbeitungsprogramm. Textverarbeitungsprogramme fügen auch spezielle Steuerzeichen bei, die den Compiler verwirren (außer wenn Sie als Textdatei speichern).

3. Führen Sie den Compiler immer als Vollbild und nicht im Fenster aus. Wenn er im Fenster ausgeführt wird, drücken Sie einfach die Alt+Enter – Taste, gleichzeitig.

4. Wenn im Hintergrund Faxsoftware oder Modemtreiber laufen, kann es zu Problemen beim Übertragen mit der COM-Schnittstelle kommen. Starten Sie in so einem Fall den Computer neu. Beenden Sie alle anderen Programme. Wenn das Problem immer noch auftritt, erstellen Sie ein Hardwareprofil, in dem nur die wichtigsten Treiber enthalten sind, und starten Sie den Computer unter dem erstellten Hardwareprofil.

Schlußbemerkungen: