

# Programmierung einer Steuerelektronik mit Zustandsautomaten

Version 1.0  
28.12.2008  
Gerd Bartelt  
[www.sebulli.com](http://www.sebulli.com)

# Inhaltsverzeichnis

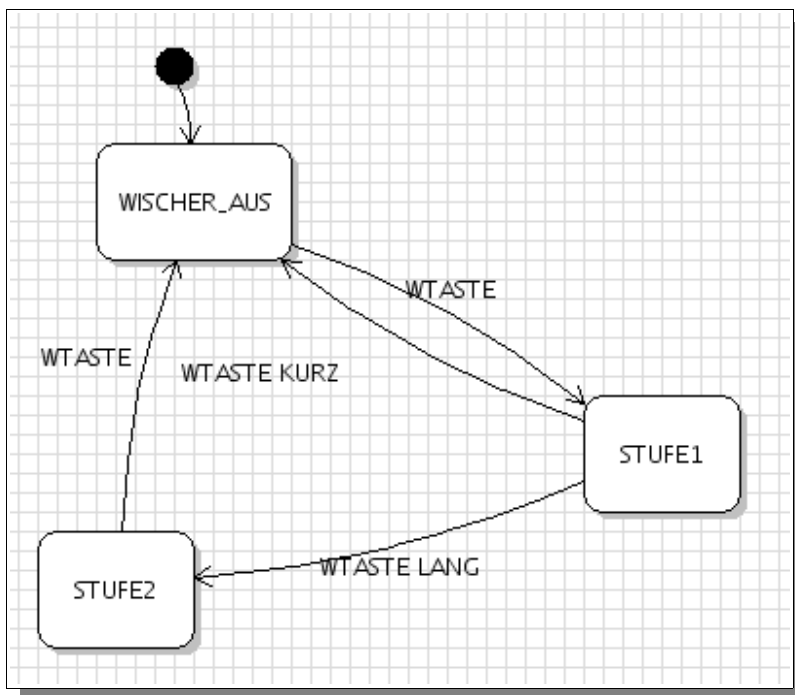
Einleitung.....	3
Violet zum Zeichnen von Zustandsdiagrammen.....	4
Installation.....	4
Das erste Dokument.....	4
Select.....	4
Objekte eines Zustandsdiagrammes.....	5
State.....	5
Scenario start.....	5
Scenario end.....	5
Note.....	5
Linked diagram.....	6
Event / Action.....	6
Node connector.....	6
Bedingungen.....	7
Negation.....	8
Operatoren.....	8
Zahlenwerte.....	9
Standardwerte.....	9
Verknüpfung von Bedingungen.....	10
UND Verknüpfung.....	10
ODER Verknüpfung.....	10
Tasten.....	11
Tastenzuordnung.....	11
Multifunktionsknopf.....	11
Handsender (Blaupunkt).....	11
Attribute der Tasten.....	12
Digitale Eingänge.....	13
Zuordnung.....	13
Attribute der digitalen Eingänge.....	13
Analoge Eingänge.....	14
Zuordnung.....	14
Wertebereich.....	14
Relaisausgänge.....	15
Zuordnung.....	15
Setzen von Relaisausgängen.....	15
Definitionen.....	16

## Einleitung

Dieses Dokument beschreibt, wie eine Steuerelektronik mithilfe von Zustandsautomaten frei programmiert werden kann.

Ein Zustandsautomat besteht aus einer Menge von Zuständen, die ein System annehmen kann. Zu jedem Zeitpunkt hat ein Zustandsautomat immer genau einen aktiven Zustand. Ein Wechsel zu einem neuen Zustand erfolgt, wenn bestimmte Bedingungen erfüllt sind.

Zustandsautomaten lassen sich in Zustandsdiagrammen beschreiben.



Violet ist ein grafischer Editor zum Zeichnen von Zustandsdiagrammen.

StateMachine ein Tool, um Zustandsdiagramme einzulesen, zu dekodieren, das Programm in die Steuerelektronik zu laden, oder es in einer Simulation zu testen.

# Violet zum Zeichnen von Zustandsdiagrammen

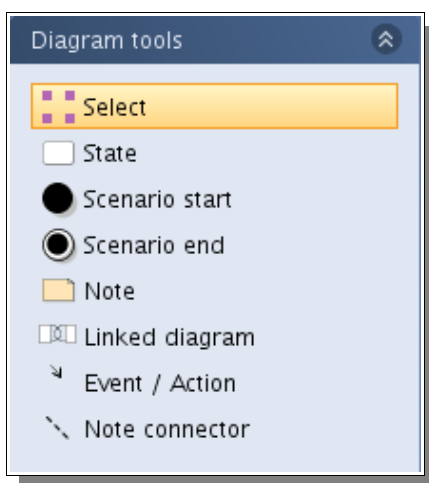
## ***Installation***

Der Violet Editor installiert sich im Startmenü in den Ordner *Statemachine*. Auf dem Rechner muss Java 6 installiert sein. [www.java.com](http://www.java.com)

## ***Das erste Dokument***

Nach dem Start ist ein neues Dokument vom Typ „state diagram“ anzulegen.

Ein Diagramm wird mit den Elementen aus der Toolbox *Diagram tools* gezeichnet.



## **Select**

Auswahlwerkzeug zum Verschieben und Bearbeiten von Elementen

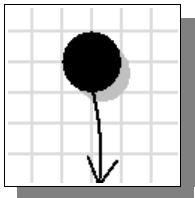
## Objekte eines Zustandsdiagrammes

### State



Zustand eines Zustandsautomaten. Doppelklick zum Ändern des Namens. Jeder Zustand muss einen eindeutigen Namen erhalten, der auch in keinem anderen verwendeten Dokument vorkommen darf. Jeder Name muss mit einem Buchstaben beginnen und ist in einer einzigen Zeile anzugeben.

### Scenario start

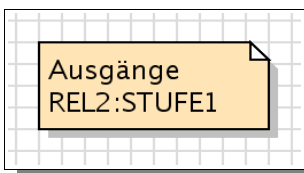


Startpunkt eines Zustandsautomaten. Jeder Zustandsautomat muss genau einen Startpunkt haben. Dieser zeigt auf den ersten Zustand, der nach dem Einschalten der Steuerelektronik eingenommen wird. Dazu ist ein Eventpfeil ohne eine Bedingung vom Scenario start-Punkt zum ersten Zustand zu ziehen.

### Scenario end

Endpunkt eines Zustandsautomaten. Dieses Objekt sollte nicht verwendet werden!

### Note

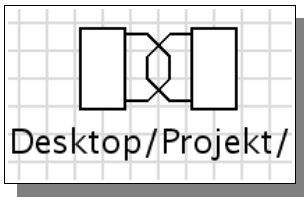


Notiz. Über Notizen kann das Diagramm beschriftet werden. Außerdem können Definitionen gesetzt, oder das Schaltverhalten der Relaisausgänge bestimmt werden.

Auf einem Notizzettel lassen sich mehrere Notizen unterbringen. Jede Zeile eine Notiz.

Befindet sich in einer Notiz(zeile) ein Doppelpunkt, wird diese ausgewertet. Ansonsten dient die Notiz dazu, das Diagramm zu beschriften.

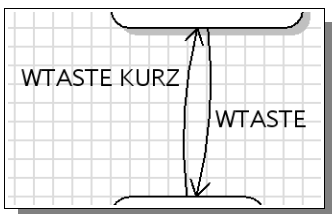
## Linked diagram



Einbinden eines weiteren Diagrammblattes. Die einzelnen Diagramme sollten übersichtlich gehalten werden. Deshalb bestehen größere Projekte aus mehreren Seiten. Diese sind über das Objekt *Linked diagram* einzubinden.

**Wichtig:** Der Dateiname wird in einem Dialog ausgewählt und durch einen Doppelklick auf die Datei übernommen. Nicht durch den OK-Knopf.

## Event / Action



Übergangspfeil von einem Zustand in den nächsten. Alle Zustände eines Zustandsautomaten sind über diese Pfeile miteinander zu verbinden. Ein Zustand wechselt dann in einen anderen, wenn die Bedingung erfüllt ist, die am Übergangspfeil steht. Ist keine Bedingung angegeben, wird der Zustand auf jeden Fall geändert.

## Node connector

Linie zum Verbinden von Zuständen. Dieses Objekt sollte nicht verwendet werden!

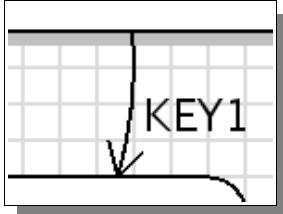
## Bedingungen

Bedingungen spielen eine zentrale Rolle beim Programmieren mit Zustandsautomaten.

Zustände wechseln, wenn die Bedingung erfüllt ist, die am Übergangspfeil angegeben ist.

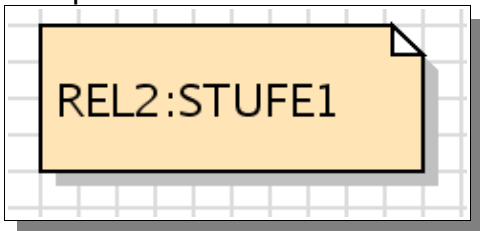
Relaisausgänge werden geschaltet, wenn die Bedingung erfüllt ist, die in einer Notiz zum Relaisausgang angegeben ist.

Beispiel 1:



Der Zustand wechselt, wenn Taste KEY1 gedrückt wird.

Beispiel 2:



Das Relais REL2 wird eingeschaltet, wenn Zustand STUFE1 aktiv ist.

## ***Negation***

Eine Bedingung wird negiert, wenn das Schlüsselwort NOT vorangestellt wird.  
Beispiel:

**NOT STUFE1**

Die Bedingung ist erfüllt, wenn Zustand STUFE1 NICHT aktiv ist.

## ***Operatoren***

Es stehen verschiedene Operatoren zur Verfügung, um Zahlenwerte zu vergleichen.

**DIN1 = 0**

Die Bedingung ist erfüllt, wenn Eingang DIN1 auf 0(=LowPegel ist)

**KEY1 > 1s**

Die Bedingung ist erfüllt, wenn Taste KEY1 länger als 1 Sekunde gedrückt wurde.

Folgende Operatoren können verwendet werden:

=	ist gleich
==	ist gleich
>	größer
<	kleiner
>=	größer oder gleich
<=	kleiner oder gleich
!=	ungleich
<>	ungleich

## **Zahlenwerte**

Bedingungen können aus Vergleichen mit Zahlenwerte bestehen. Es sind nur positive Zahlen zulässig.

Bei digitalen Eingängen 0 oder 1:

**DIN1 = 1**

Bei analogen Eingängen von 0 bis 100:

**AIN1 > 50**

Bei Zeitangaben endet der Zahlenwerte mit einem „s“ für Sekunde. Die kleinste Einheit ist 0.1s. Als Dezimaltrennzeichen wird ein Punkt verwendet.

Beispiel Vergleich mit 3.2 Sekunden:

**KEY1 > 3.2s**

## **Standardwerte**

Wird kein Vergleichsoperator angegeben, wird ein „=“ verwendet.

Bei Zeitangaben ein „>=“.

Wird kein Zahlenwert angegeben, wird eine „1“ verwendet.

Beispiel 1:

**DIN2 = 1**

kann auch geschrieben werden als:

**DIN2**

Beispiel 2:

**KEY1 >= 1.5s**

kann auch geschrieben werden als:

**KEY1 1.5s**

## ***Verknüpfung von Bedingungen***

### **UND Verknüpfung**

Mehrere einzelne Bedingungen können logisch verUNDet werden. Die gemeinsame Bedingung ist erst dann erfüllt, wenn alle Einzelbedingungen erfüllt sind. Die Einzelbedingungen werden durch ein kaufmännisches UND (&) miteinander verknüpft.

Beispiele:

**DIN2 = 1 & KEY1 1.5s**

Die gemeinsame Bedingung ist erst wahr, wenn DIN2 auf „1“ steht UND die Taste KEY1 1.5 Sekunden gedrückt wurde.

### **ODER Verknüpfung**

Mehrere einzelne Bedingungen oder UND-Verknüpfungen können logisch verODERT werden. Die gemeinsame Bedingung ist erst dann erfüllt, wenn eine der Einzelbedingungen erfüllt sind. Die Einzelbedingungen werden durch ein Komma „ , “ miteinander verknüpft.

Beispiele:

**KEY1 , KEY2**

Die gemeinsame Bedingung ist erst wahr, wenn KEY1 ODER KEY2 gedrückt wurde.

**WICHTIG:** UND-Verknüpfungen werden vor ODER-Verknüpfungen ausgeführt.

## Tasten

Tasten werden mit „KEY“ und einem nachfolgendem Index 0 bis 9 angesprochen.

Beispiel:

**KEY6**

Ohne weiteren Angaben wird nur der Klick ausgewertet.

## ***Attribute der Tasten***

Es können noch weitere Attribute angegeben werden:

### **KEY6 DOWN**

Ist dann wahr, solange die Taste gedrückt ist.

### **KEY6 UP**

Ist dann wahr, solange die Taste nicht gedrückt ist.

### **KEY6 PRESSED**

Ist dann wahr, wenn die Taste gedrückt wird (Übergang von nicht-gedrückt zu gedrückt)

### **KEY6 RELEASED**

Ist dann wahr, wenn die Taste wieder losgelassen wird (Übergang von gedrückt zu nicht-gedrückt)

**WICHTIG:** Unterschied zwischen KEYx und KEYx PRESSED: Eine Abfrage nach KEYx PRESSED erzeugt später auch ein KEYx RELEASED Ereignis. Eine Abfrage nach KEYx hingegen nicht.

## Digitale Eingänge

Digitale Eingänge werden mit „DIN“ und einem nachfolgendem Index 0 bis 2 angesprochen.

Beispiel:

**DIN2**

### *Zuordnung*

Die 3 digitalen Eingänge der Steuerelektronik sind zur Zeit folgenden Sensoren zugeordnet:

DIN0	Hallsensor links
DIN1	Hallsensor Mitte
DIN2	Hallsensor rechts

### *Attribute der digitalen Eingänge*

Es können noch weitere Attribute angegeben werden:

**DIN2 HIGH**

Ist dann wahr, solange der Eingang High Pegel hat.

**DIN2 LOW**

Ist dann wahr, solange der Eingang Low Pegel hat.

**DIN2 LH**

Ist dann wahr, beim Übergang Low auf High.

**DIN2 HL**

Ist dann wahr, beim Übergang High auf Low.

## **Analoge Eingänge**

Analoge Eingänge werden mit „AIN“ und dem nachfolgendem Index angesprochen.

Beispiel:

<b>AIN0</b>
-------------

### ***Zuordnung***

Der analoge Eingang der Steuerelektronik sind zur Zeit folgendem Sensor zugeordnet:

AIN0	Lichtsensord
------	--------------

### ***Wertebereich***

Der Wertebereich der analogen Eingänge erstreckt sich von 0 bis 100.

## Relaisausgänge

Relaisausgänge werden mit „REL“ und einem nachfolgendem Index 0 bis 15 angesprochen.

Beispiel:

**REL10**

### *Zuordnung*

Die Relaisausgänge sind folgendermaßen zugeordnet:

REL0	...
REL1	...
REL2	...
REL3	...
REL4	...
REL5	...
REL6	...
REL7	...
REL8	...
REL9	...
REL10	...
REL11	...
REL12	...
REL13	...
REL14	...
REL15	...

### *Setzen von Relaisausgängen*

Das Verhalten von Relaisausgängen wird in Notizen beschrieben. Dazu wird die Bezeichnung des Relais angegeben, ein Doppelpunkt und die Bedingung, bei der das Relais schalten soll.



REL2:STUFE1

## Definitionen

Um die Übersichtlichkeit eines Zustandsdiagramms zu erhöhen, können Schlüsselwörter durch gebräuchlichere Begriffe ersetzt werden.

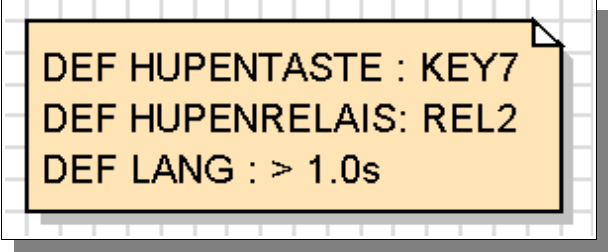
Eine Definition wird in Notizen beschrieben und ersetzt alle Begriffe durch einen neuen.

In einer Notizzeile ist dazu das Schlüsselwort DEF gefolgt von dem neuen Begriff, einem Doppelpunkt und dem zu ersetzenden Begriff anzugeben.

So lässt sich die Beschreibung von:  
„Relais2 ansteuern, wenn Taste KEY7 länger als 1 Sekunde gedrückt wurde“

**REL2: KEY7 >1.0s**

Mit den Definitionen:



```
DEF HUPENTASTE : KEY7
DEF HUPENRELAIS: REL2
DEF LANG : > 1.0s
```

Viel leserlicher schreiben:

**HUPENRELAIS: HUPENTASTE LANG**