JetSym - Compilerdirektiven und Konfigurationen

Einleitung

JetSym verwendet nur einen Programmquelltext für alle Steuerungstypen.
Im Gegensatz zu DOS-SYMPAS gibt es somit keine steuerungsspezifischen Dateiendungen mehr, wie z.B. *.pnb für NANO-B oder *.ppd für DELTA.
Um Programmvarianten komfortabel verwaltet zu können, bietet JetSym eine verbesserte Unterstützung von Compilerdirektiven und die Möglichkeit, diese in Konfigurationen zusammenzufassen.

Möglichkeiten mit Compilerdirektiven

- Sie können Ihr Programm in mehrere Funktionsblöcke aufteilen und so Ihr Projekt übersichtlicher gestalten. Darüber hinaus können Sie Funktionsbibliotheken erstellen und diese leicht in neue Projekte übernehmen.
- Funktionen, die nur auf einem bestimmten Steuerungstyp verfügbar sind oder dort in einer besonderen Weise programmiert werden sollen, können durch Compilerdirektiven aktiviert werden.
- Falls Sie mehrere Varianten oder Ausbaustufen Ihrer Maschine bzw. Anlage anbieten wollen, so können Sie leicht die entsprechenden Programmteile einbinden.
- Sie können sich spezielle Programmversionen zum Testen und Einrichten Ihrer Maschine bzw. Anlage erstellen, die Ihnen z.B. zusätzliche Bildschirmausgaben, spezielle Prüfroutinen oder ähnliches bieten.

Beim Einsatz von Compilerdirektiven in Symboldateien ergeben sich noch zusätzliche Möglichkeiten, wie z.B.:

- Sie können einfach länderspezifische Bedienerführungen realisieren, indem Sie die vordefinierten Anzeigetexte für das Bedienpersonal mittels Compilerdirektiven austauschen.

Vorteil für Sie: Sie brauchen nur einen Programmquelltext zu erstellen und zu pflegen.

Welche Direktiven gibt es und wie werden sie eingegeben?

- **Direktive: #include**
  Um Programme übersichtlicher zu gestalten, können Programmteile und Teile der Symboldatei in eigenständige Dateien ausgelagert werden. An der Stelle, an der der Inhalt dieser ausgelagerten Dateien eingefügt werden soll, muß im Programmquelltext oder in der Symboldatei der Befehl

  `#include` <Dateiname>

  eingefügt werden.

  - Den Befehl im Programmquelltext einfügen:

    Halten Sie zum vorangehenden Befehl mindestens ein Leerzeichen Abstand oder beginnen Sie eine Programmzeile.
    Drücken Sie dann Taste **#**, gefolgt von der Taste ***i*** .
    Es öffnet sich ein Parameterdialog, in dem Sie den Pfad und den Namen der Datei eingeben können, die eingebunden werden soll. Wenn Sie die Schaltfläche rechts neben dem

Alle Rechte vorbehalten.
Die Firma Jetter AG behält sich das Recht vor, Änderungen an ihren Produkten vorzunehmen, die der technischen Weiterentwicklung dienen. Diese Änderungen werden nicht notwendigerweise in jedem Einzelfall dokumentiert.

Diese Application Note und die darin enthaltenen Informationen wurden mit der gebotenen Sorgfalt zusammengestellt. Die Firma Jetter AG übernimmt jedoch keine Gewähr für Druckfehler oder andere Fehler oder daraus entstehende Schäden.

Die hier genannten Marken und Produktnamen sind Warenzeichen oder eingetragene Warenzeichen der jeweiligen Titelhalter.

Jetter AG
Gräterstrasse 2
D-71642 Ludwigsburg
Germany

Telefon - Zentrale: 07141/2550-0
Telefon - Vertrieb: 07141/2550-530
Telefon - Technische Hotline: 07141/2550-444

Telefax: 07141/2550-425
E-Mail - Vertrieb: sales@jetter.de
E-Mail - Technische Hotline: hotline@jetter.de
Internetadresse: http://www.jetter.de

Eingabefeld betätigen, dann öffnet sich ein weiteres Dialogfenster, in dem Sie die gewünschte Datei mit der Maus auswählen können. Schließen Sie die Dialogfenster danach mittels der entsprechenden Schaltflächen.

Der Dateiname muß in Anführungsstrichen stehen, wie in Abbildung 1 zu sehen.

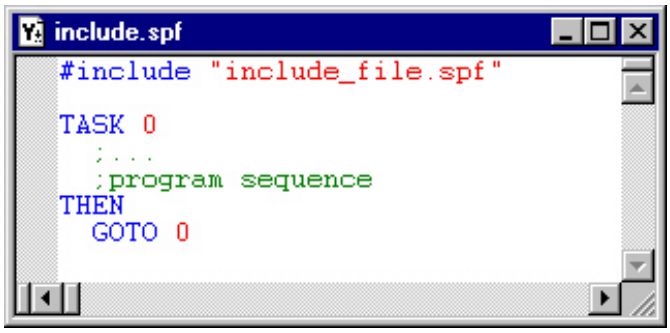


Abbildung 1

- Befehl in die Symboldatei einfügen:

Klicken sie mit der Maus auf die gewünschte Stelle in der Symboldatei und öffnen Sie durch Klicken mit der rechten Maustaste das Kontextmenü. Wählen Sie dann den Menüeintrag *Insert Directive* .

Geben Sie jetzt in der ersten Spalte der neu eingefügten Zeile die Direktive `#include` ein. Doppelklicken Sie mit der Maus in die zweite Spalte dieser Zeile. Es öffnet sich ein Dialogfenster (vgl. Abbildung 2), den Sie den Dateinamen der einzubindenden Datei direkt eingeben können. Über die Schaltfläche rechts neben dem Textfeld können Sie aber auch ein Auswahlfenster öffnen, in dem Sie die gewünschte Datei mit der Maus auswählen können.

Auch hier muß der Dateiname in Anführungsstrichen stehen, wie in Abbildung 3 zu sehen.

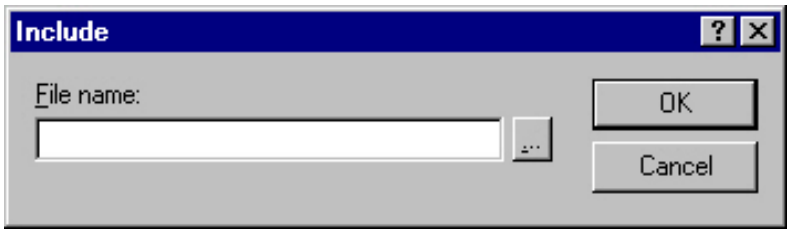


Abbildung 2

Abbildung 3

- **Direktive: #ifdef**

Verwendung:

```
#ifdef <Compilerkonstante>

  Block mit Quelltext oder Symbolen

#endif
```

Bedeutung:

Ein Programmteil oder Teil einer Symboldatei kann bedingt auskommentiert werden.

`#ifdef` ist wahr, wenn die <Compilerkonstante> definiert wurde und damit gültig ist. Definiert werden kann eine Compilerkonstante durch eine Direktive `#define <Compilerkonstante>` am Anfang der Symboldatei oder über die Projektkonfigurationen. Nähere Informationen dazu finden Sie weiter unten im Text.
Die Direktive `#ifdef` kann sowohl im Symboleditor als auch im Programmeditor verwendet werden. Ist die Bedingung erfüllt (Compilerkonstante ist definiert), so ist der Quelltext bis zur nächsten `#endif` Direktive gültig, d.h. der Quelltext wird vom Compiler bearbeitet.

Sie können nicht mehrere Compilerkonstanten in einer `#ifdef`-Abfrage verknüpfen.

- Den Befehl im Programmquelltext einfügen:

  Halten Sie zum vorangehenden Befehl mindestens ein Leerzeichen Abstand oder beginnen Sie eine Programmzeile.
  Drücken Sie dann Taste **#**, gefolgt von der Taste ***f***.
  Es öffnet sich ein Parameterdialog, in dem Sie die Compilerkonstante eingeben können. Durch Eingabe eines Fragezeichens können Sie eine Liste aller Compilerkonstanten, die JetSym zu diesem Zeitpunkt bekannt sind, aufrufen. Sie können auch zuerst ein paar Anfangsbuchstaben der Compilerkonstanten eingeben, gefolgt mit einem Fragezeichen, und Sie erhalten eine Liste mit allen Compilerdirektiven angezeigt, die mit diesen Anfangsbuchstaben beginnen. Klicken Sie dann auch *OK*.

  Die neue Direktive wird in den Quelltext eingefügt (vgl. Abbildung 4).

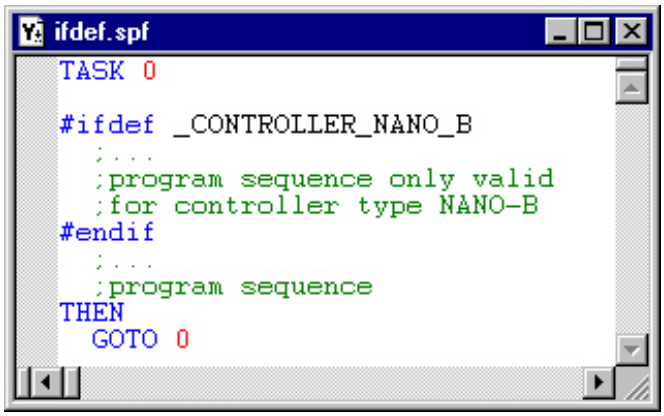


Abbildung 4

- Befehl in die Symboldatei einfügen:

  Klicken sie mit der Maus auf die gewünschte Stelle in der Symboldatei und öffnen Sie durch Klicken mit der rechten Maustaste das Kontextmenü. Wählen Sie dann den Menüeintrag *Insert Directive* .

  Geben Sie jetzt in der ersten Spalte der neu eingefügten Zeile die Direktive `#ifdef` ein. Wechseln Sie in die zweite Spalte dieser Zeile und geben Sie dort die Compilerkonstante ein. Die entsprechenden Zeilen in der Symboldatei sehen dann ähnlich wie in Abbildung 5 aus.

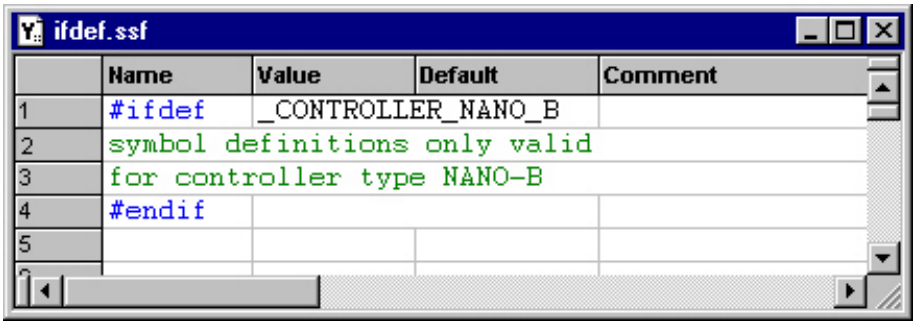


Abbildung 5

- **Direktive: #ifndef**

Verwendung:

```
#ifndef <Compilerkonstante>

  Block mit Quelltext oder Symbolen

#endif
```

Bedeutung:

Ein Programmteil oder Teil einer Symboldatei kann bedingt auskommentiert werden.

`#ifndef` ist wahr, wenn die <Compilerkonstante> nicht definiert wurde und damit ungültig ist. Definiert werden kann eine Compilerkonstante durch eine Direktive `#define <Compilerkonstante>` am Anfang der Symboldatei oder über die Projektkonfigurationen. Nähere Informationen dazu finden Sie weiter unten im Text.
Die Direktive `#ifndef` kann sowohl im Symboleditor als auch im Programmeditor verwendet werden. Ist die Bedingung erfüllt (Compilerkonstante ist nicht definiert), so ist der Quelltext bis zur nächsten `#endif` Direktive gültig, d.h. der Quelltext wird vom Compiler bearbeitet.

Sie können nicht mehrere Compilerkonstanten in einer `#ifndef`-Abfrage verknüpfen.

- Den Befehl im Programmquelltext einfügen:

  Halten Sie zum vorangehenden Befehl mindestens ein Leerzeichen Abstand oder beginnen Sie eine Programmzeile.
  Drücken Sie dann Taste **#**, gefolgt von der Taste ***n***.
  Es öffnet sich ein Parameterdialog, in dem Sie die Compilerkonstante eingeben können. Durch Eingabe eines Fragezeichens können Sie eine Liste aller Compilerkonstanten, die JetSym zu diesem Zeitpunkt bekannt sind, aufrufen. Sie können auch zuerst ein paar Anfangsbuchstaben der Compilerkonstanten eingeben, gefolgt mit einem Fragezeichen, und

Sie erhalten eine Liste mit allen Compilerdirektiven angezeigt, die mit diesen Anfangsbuchstaben beginnen. Klicken Sie dann auf *OK*.

Die neue Direktive wird in den Quelltext eingefügt (vgl. Abbildung 6).

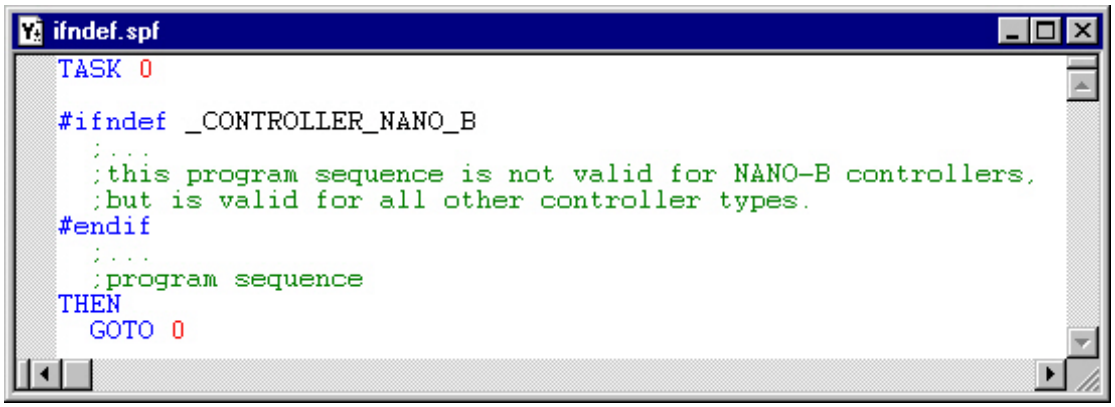


Abbildung 6

- Befehl in die Symboldatei einfügen:

  Klicken sie mit der Maus auf die gewünschte Stelle in der Symboldatei und öffnen Sie durch Klicken mit der rechten Maustaste das Kontextmenü. Wählen Sie dann den Menüeintrag *Insert Directive* .

  Geben Sie jetzt in der ersten Spalte der neu eingefügten Zeile die Direktive `#ifndef` ein. Wechseln Sie in die zweite Spalte dieser Zeile und geben Sie dort die Compilerkonstante ein. Die entsprechenden Zeilen in der Symboldatei sehen dann ähnlich wie in Abbildung 7 aus.

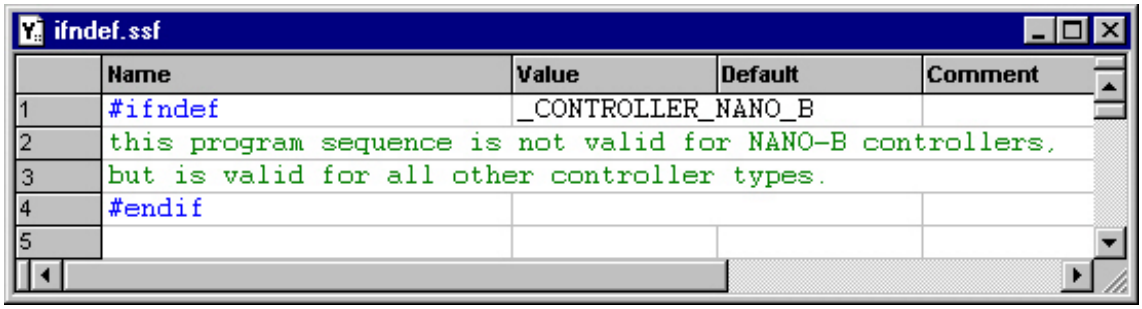


Abbildung 7

- **Direktive: #endif**

Verwendung:

```
#ifdef/#ifndef <Compilerkonstante>

  Block mit Quelltext oder Symbolen

#endif
```

Bedeutung:
Mit `#endif` werden die Direktiven `#ifdef` und `#ifndef` abgeschlossen.

- Den Befehl im Programmquelltext einfügen:

  Halten Sie zum vorangehenden Befehl mindestens ein Leerzeichen Abstand oder beginnen Sie eine Programmzeile.
  Drücken Sie dann Taste **#**, gefolgt von der Taste ***e***.
  Die neue Direktive wird in den Quelltext eingefügt.

- Befehl in die Symboldatei einfügen:

  Klicken sie mit der Maus auf die gewünschte Stelle in der Symboldatei und öffnen Sie durch Klicken mit der rechten Maustaste das Kontextmenü. Wählen Sie dann den Menüeintrag *Insert Directive* .

  Geben Sie jetzt in der ersten Spalte der neu eingefügten Zeile die Direktive `#endif` ein.

- **Direktive: #define**
  Die Direktive `#define` ist nur in Symboldateien anwendbar.

  Verwendung:

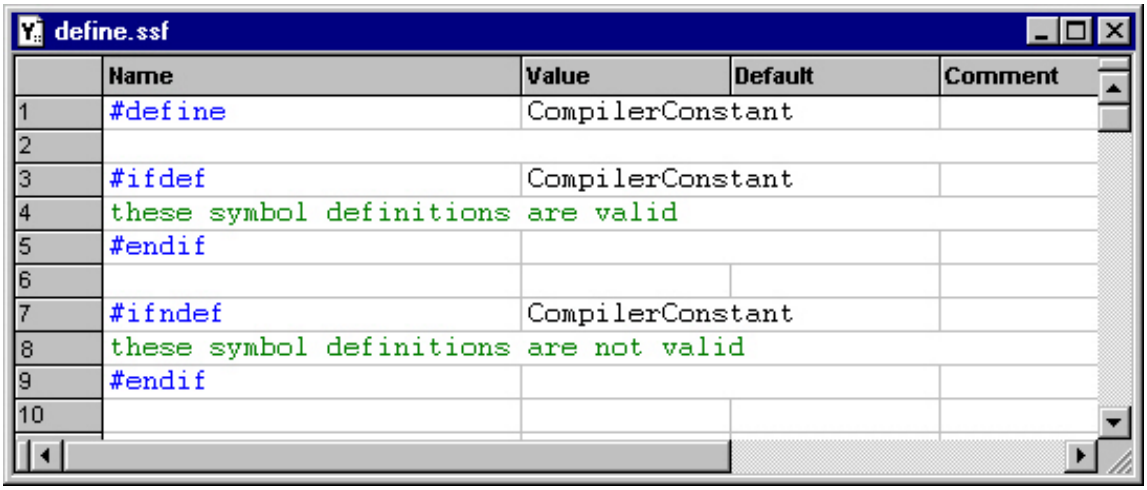
  ```
  #define <Compilerkonstante>
  ```

  Bedeutung:
  Die Direktive `#define` <Compilerkonstante> definiert ein Schlüsselwort, das mit `#ifdef` bzw. `#ifndef` abgefragt werden kann.

  - Befehl in die Symboldatei einfügen:

    Klicken sie mit der Maus auf die gewünschte Stelle in der Symboldatei und öffnen Sie durch Klicken mit der rechten Maustaste das Kontextmenü. Wählen Sie dann den Menüeintrag *Insert Directive* .

    Geben Sie jetzt in der ersten Spalte der neu eingefügten Zeile die Direktive `#define` ein. Wechseln Sie in die zweite Spalte dieser Zeile und geben Sie dort die Compilerkonstante ein. Die entsprechenden Zeilen in der Symboldatei sehen dann ähnlich wie in Abbildung 8 aus.



define.ssf

	Name	Value	Default	Comment
1	#define	CompilerConstant		
2				
3	#ifdef	CompilerConstant		
4	these symbol definitions are valid			
5	#endif			
6				
7	#ifndef	CompilerConstant		
8	these symbol definitions are not valid			
9	#endif			
10				

Abbildung 8

Vordefinierte Compilerkonstanten

Wenn Sie JetSym neu starten und ein neues Projekt anlegen, dann werden z. Zt. von JetSym automatisch diese Compilerkonstanten definiert:

`_DEBUG`	Ist definiert, wenn die Konfiguration "Debug" als aktive Konfiguration gesetzt wurde.
`NDEBUG`	Ist definiert, wenn die Konfiguration "Release" als aktive Konfiguration gesetzt wurde.
`_CONTROLLER_D_CPU`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "D-CPU" ausgewählt wurde.
`_CONTROLLER_D_CPU2`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "D-CPU2" ausgewählt wurde.
`_CONTROLLER_DELTA`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "D-CPU" oder "D-CPU2" ausgewählt wurde.

`_CONTROLLER_JC_241`
`_CONTROLLER_JC_243`
`_CONTROLLER_JC_246`
`_CONTROLLER_JC_24x`
	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "JC-24x" ausgewählt wurde.

`_CONTROLLER_NANO`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "NANO-A", "NANO-B", "NANO-C", "NANO-D" oder "PCPPLC" ausgewählt wurde.
`_CONTROLLER_NANO_A`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "NANO-A" ausgewählt wurde.
`_CONTROLLER_NANO_B`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "NANO-B" ausgewählt wurde.
`_CONTROLLER_NANO_C`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "NANO-C" ausgewählt wurde.
`_CONTROLLER_NANO_D`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "NANO-D" ausgewählt wurde.

`_CONTROLLER_PASE_E`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "PASE-E+" ausgewählt wurde.
`_CONTROLLER_PCPPLC`	Ist definiert, wenn in den "Project Settings" als Steuerungstyp "PCPPLC" ausgewählt wurde.

Die "Project Settings" sind über den Menüeintrag *Project* und dann *Settings...* erreichbar.

Verwaltung von Direktiven

Manuell: Liste mit #define anlegen

Wie schon von DOS-SYMPAS bekannt, kann eine Liste von `#define` Direktiven am Anfang einer Symboldatei angelegt und verwaltet werden, die dann jeweils manuell gesetzt bzw. auskommentiert werden.

- Nachteil: es muß immer aufs Neue überlegt werden, welche Direktiven zu setzen bzw. zu löschen sind -> mögliche Fehlerquelle

Automatisch: Konfigurationen anlegen

JetSym bietet die Möglichkeit mehrere Konfigurationen zu definieren und einfach zu verwalten.

- Vorteil: Die Konfigurationen werden einmal definiert und können dann unverändert bleiben. Für neue Anforderungen werden einfach neue Konfigurationen definiert, die bisherigen Konfigurationen sind aber weiterhin abrufbar -> weniger Fehlermöglichkeiten.

Konfigurationen

Konfigurationen anlegen

Über den Menüeintrag *Build* / *Configurations...* ist ein Dialog mit dem Titel "Configurations" (siehe Abbildung 9) erreichbar, in dem alle Konfigurationen des geöffneten Workspace bearbeitet werden können:

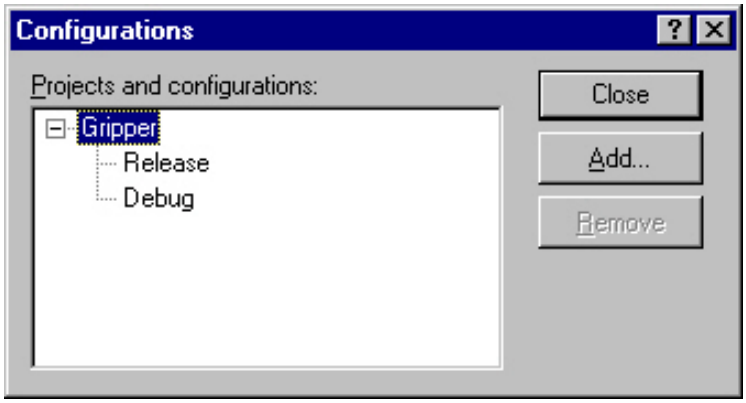


Abbildung 9

- Wollen Sie eine neue Konfiguration erstellen, dann wählen Sie das entsprechende Projekt aus und drücken Sie die Schaltfläche *Add...* .
- Es öffnet sich ein weiterer Dialog mit dem Titel "Add Project Configuration" (siehe Abbildung 10), in dem sie den Namen der neuen Konfiguration eingeben können. Hier können sie auch festlegen, welche

schon bestehende Konfiguration dieses Projekts hierzu kopiert werden soll. Bestätigen Sie dann, indem Sie die Schaltfläche *OK* klicken.

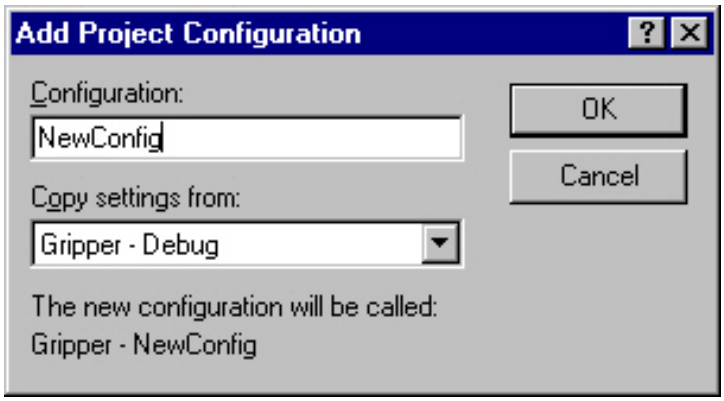


Abbildung 10

- Wurden alle gewünschten Konfigurationen angelegt, dann schließen Sie den "Configurations"-Dialog, indem Sie auf die Schaltfläche *Close* klicken.

Vordefinierte Konfigurationen

Wenn Sie JetSym neu starten und ein neues Projekt anlegen, dann werden z. Zt. von JetSym automatisch diese Konfigurationen angelegt:

- Debug: Die Direktive `_DEBUG` ist definiert.
- Release: Die Direktive `NDEBUG` ist definiert.

Konfigurationen aktivieren

- Über den Menüeintrag *Build* / *Set Active Configuration...* ist ein Dialog mit dem Titel "Set Active Project Configurations" (siehe Abbildung 11) erreichbar, in dem alle Konfigurationen des geöffneten Workspace aufgelistet sind.

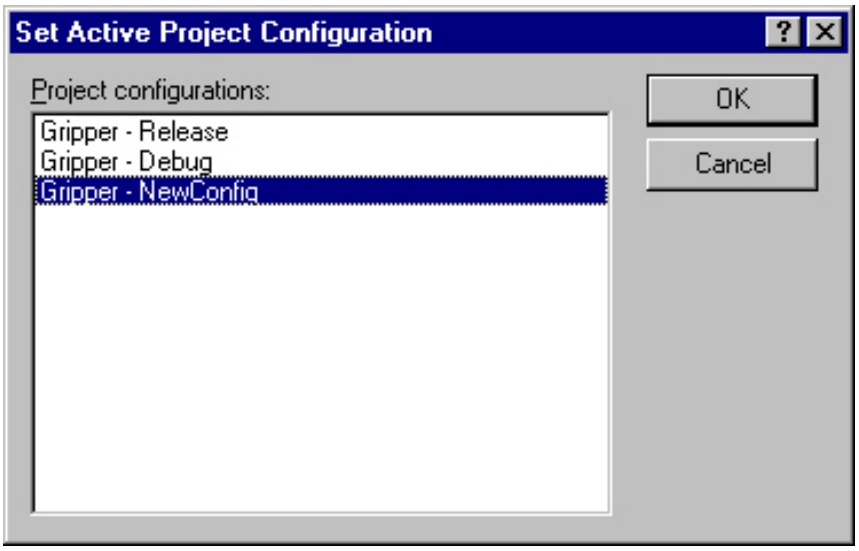


Abbildung 11

- Wählen Sie eine der aufgelisteten Konfigurationen aus und klicken Sie auf die Schaltfläche *OK*. Die gewählte Konfiguration wird daraufhin aktiv gesetzt und ggf. das entsprechende Projekt zum aktiven Projekt gemacht.

Direktiven für Konfigurationen setzen

Das Dialogfenster der *Project Settings* kann, wie in Abbildung 12 gezeigt, über den Menüeintrag *Project / Settings* (Tastenkombination *Alt + F7*) oder über das Kontextmenü des Projekts im *FileView*-Fenster geöffnet werden.

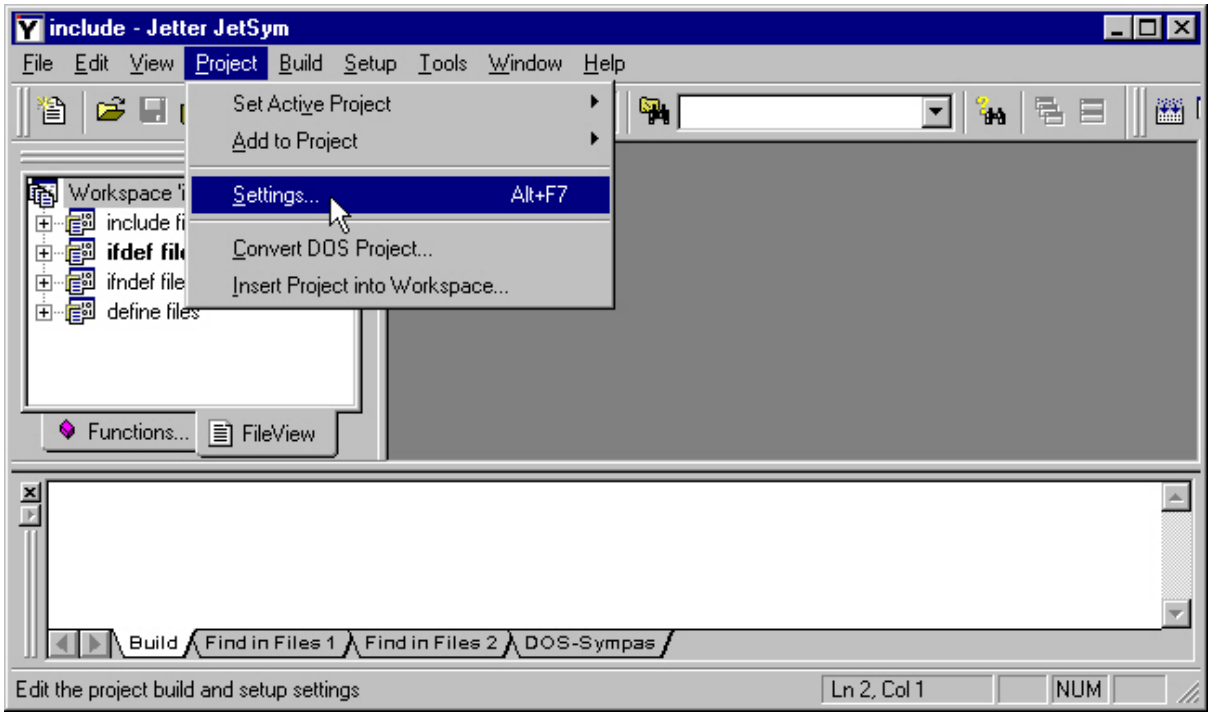


Abbildung 12

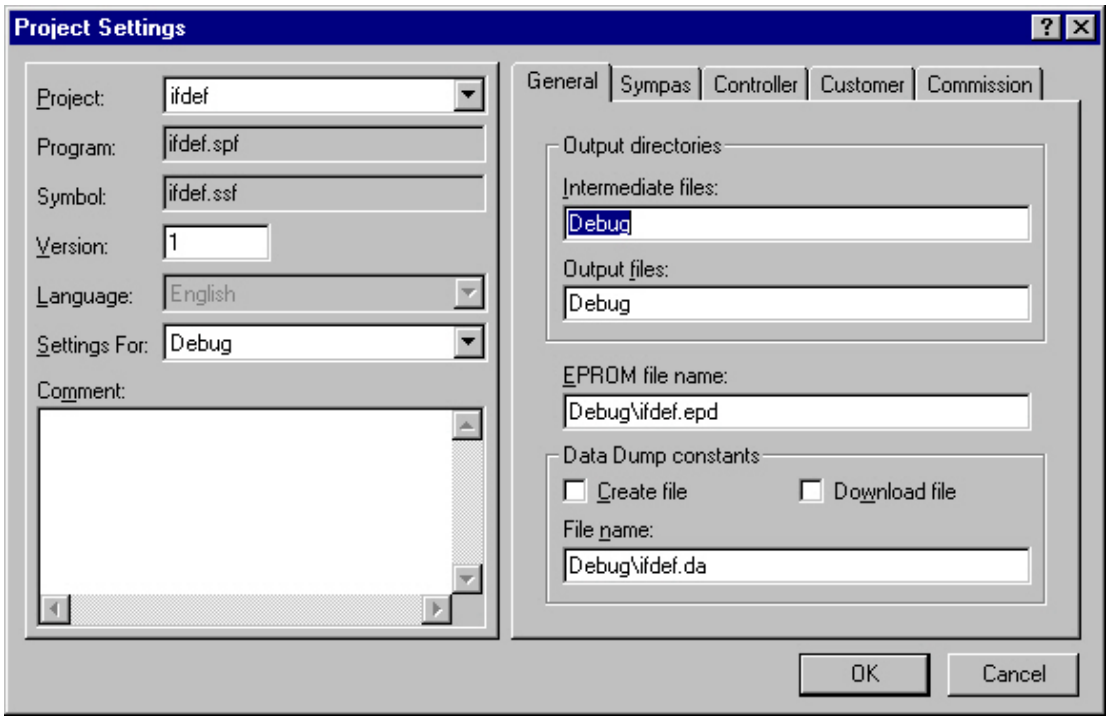


Abbildung 13

Das Dialogfenster der *Project Settings*, wie es sich in Abbildung 13 zeigt, teilt sich in zwei Hälften:

- In der linken Dialoghälfte ist eine Übersicht über das gewählte Projekt dargestellt. Die einzelnen Elemente dieser Übersicht sind (von oben nach unten):
    - Der Name des Projekts, dessen Einstellungen momentan bearbeitet werden.
    - Die Dateinamen der Programmtext- und Symbol-Hauptdatei ("ifdef.spf" bzw. "ifdef.ssf").
    - Die Versionsnummer des Projekts.
    - Die aktuelle Entwicklungssprache (zur Zeit unveränderbar auf "English" eingestellt).
    - Der Name der Konfiguration, deren Einstellungen auf der rechten Hälfte des Dialogfensters bearbeitet werden kann. Eine Beschreibung der einzelnen Einstellmöglichkeiten für Konfigurationen folgt weiter unten. (Abbildung 13 zeigt, dass für das Beispielprojekt im Eingabefeld *Settings For* die Konfiguration *Debug* eingestellt ist.)
    - Ein Textfeld für einen kurzen Kommentar zum Projekt. Dieser Kommentar gilt für alle Konfigurationen.
- Auf der rechten Hälfte des Dialogfensters sind konfigurationsspezifische Einstellungen angezeigt, wobei sich die Angaben wiederum auf fünf Registerkarten verteilen:
    - Registerkarte *General* (siehe Abbildung 13):
      Die vom Compiler erzeugten Ausgabedateien können abhängig von der Konfiguration automatisch in verschiedene Verzeichnisse abgelegt werden.
      Einstellbar sind unabhängig voneinander die Verzeichnisse für temporäre (*Intermediate files*) und endgültige (*Output files*) Dateien. Zudem können noch das Zielverzeichnis und der Dateiname der EPROM-Datei (das ist die Programmdatei, die später in die Steuerung geladen wird) und einer ggf. zu erstellenden Konstantendatei (*Data Dump constants*), d.h. die Datei mit den Register-Vorgabewerten, festgelegt werden.
    - Registerkarte *Sympas* (siehe Abbildung 14):
      Hier können Compiler-Direktiven für die aktuelle Konfiguration gesetzt werden.

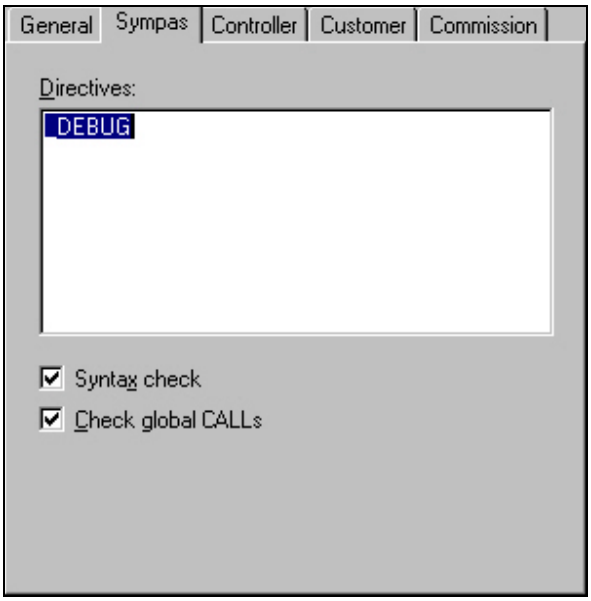


Abbildung 14

- Registerkarte *Controller* (siehe Abbildung 15):
  Die Einstellungen in dieser Registerkarte beziehen sich auf das angeschlossene Steuerungssystem, für die das Projekt übersetzt werden soll.
  Ebenfalls sind hier die Einstellungen der Schnittstelle und die Übertragungsparameter für den Programm-Download veränderbar.

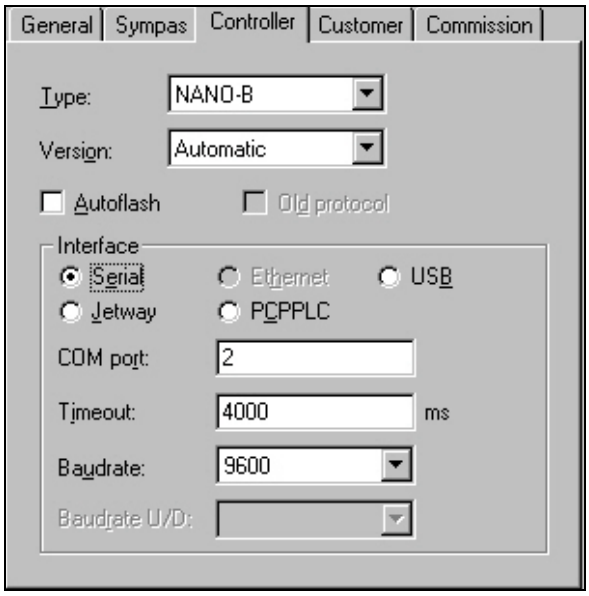


Abbildung 15

- Registerkarten *Customer* (Abbildung 16) und *Commission* (Abbildung 17):
  In diese beiden Registerkarten können bei Bedarf noch Informationen (Adresse, Telefon, Ansprechperson etc.) von Kunden eingetragen werden.

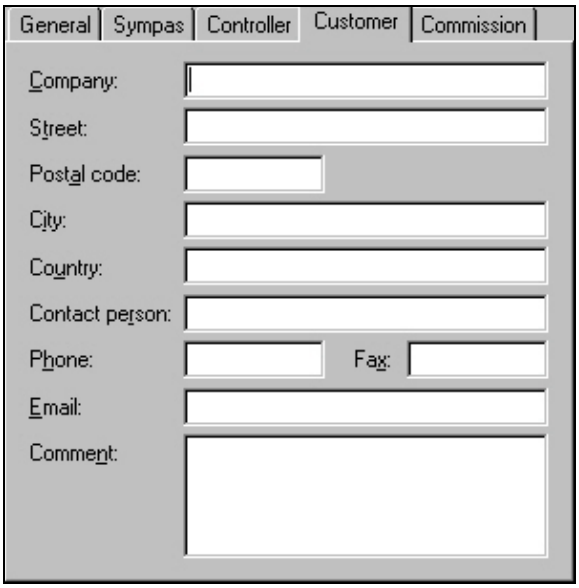


Abbildung 16

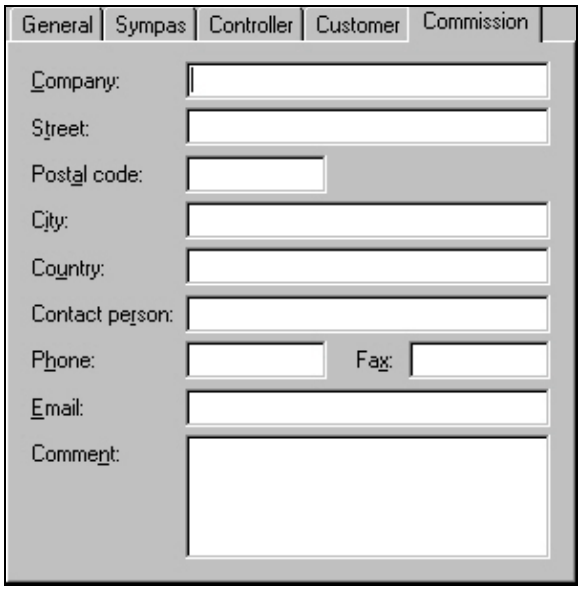


Abbildung 17

Nachdem Sie alle Einstellungen vorgenommen haben, können Sie das Dialogfenster der *Project Settings* durch Klicken auf die Schaltfläche *OK* (siehe Abbildung 13) schließen.

Merkregel

Wichtig !

In einem Workspace gibt es immer nur ein aktives Projekt mit einer aktiven Konfiguration!

Dieses Projekt mit dieser Konfiguration dient als Grundlage für den Compiler und den Download. Nur die Einstellungen, die in dieser Konfiguration gemacht sind, werden von JetSym beachtet.