Introduction

This application note describes the operating principle of JetSym and how to use it by means of a simple project. The application note does not comprise an introduction to the programming language.

We assume that the user is familiar with the instruction set of the controllers as well as with the programming procedure.

The project serving as example is the program for the gripper already introduced in the training of JetSym basics.

Task

A simple „pick and place“ gripper is to be programmed, see fig. 1.
The controller for this project is a NANO-B with a digital input module of the type N-ID8.

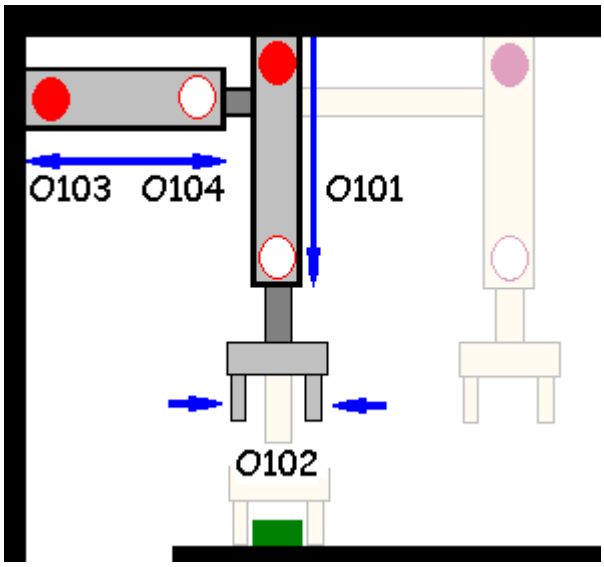


Figure 1

Jetter AG
Gräterstrasse 2
D-71642 Ludwigsburg
Germany

Phone - Switchboard: +49 7141 2550-0
Phone - Sales: +49 7141 2550-530
Phone – Technical Hotline: +49 7141 2550-444

Telefax: +49 7141 2550-425
E-Mail - Sales: sales@jetter.de
E-Mail - Technical Hotline: hotline@jetter.de
Internet address: http://www.jetter.de

Desired Program Flow

1. **Start**: when the base position has been reached, wait for the “START” operating switch.
2. **Picking up the item**:
    1. The vertical axis is to move downwards.
    2. When the vertical axis has reached the lowest position, the gripper is to close.
    3. After 0.5 seconds, the vertical axis is to move upwards again.
3. **Placing the item**:
    1. When the vertical axis has reached the highest point, the horizontal axis is to move.
    2. When the horizontal axis has reached the working position, the axis is to move downwards vertically in working position.
    3. When the vertical axis has reached the lowest position, the gripper is to open.
    4. After 0.5 seconds, the vertical axis is to move upwards again.
4. **Base position:** when the vertical axis has reached the base position, the horizontal axis is also to return to its working position.
5. **Repetition:** this process is to be repeated as long as the switch is on “START” position. When the switch is not on „START“ position anymore, the ongoing cycle is to be completed.

Assignment of Inputs and Outputs

Inputs
101 START/STOP operating switch
205 Base position vertical (top)
206 Working position vertical (bottom)
207 Base position horizontal (rear)
208 Working position horizontal (front)

Outputs
101 Move downward in vertical direction
102 Close gripper
103 Move in backward direction horizontally
104 Move in forward direction horizontally

Programming

Creating a Project

- After having started JetSym, an empty development environment appears, see fig. 2.

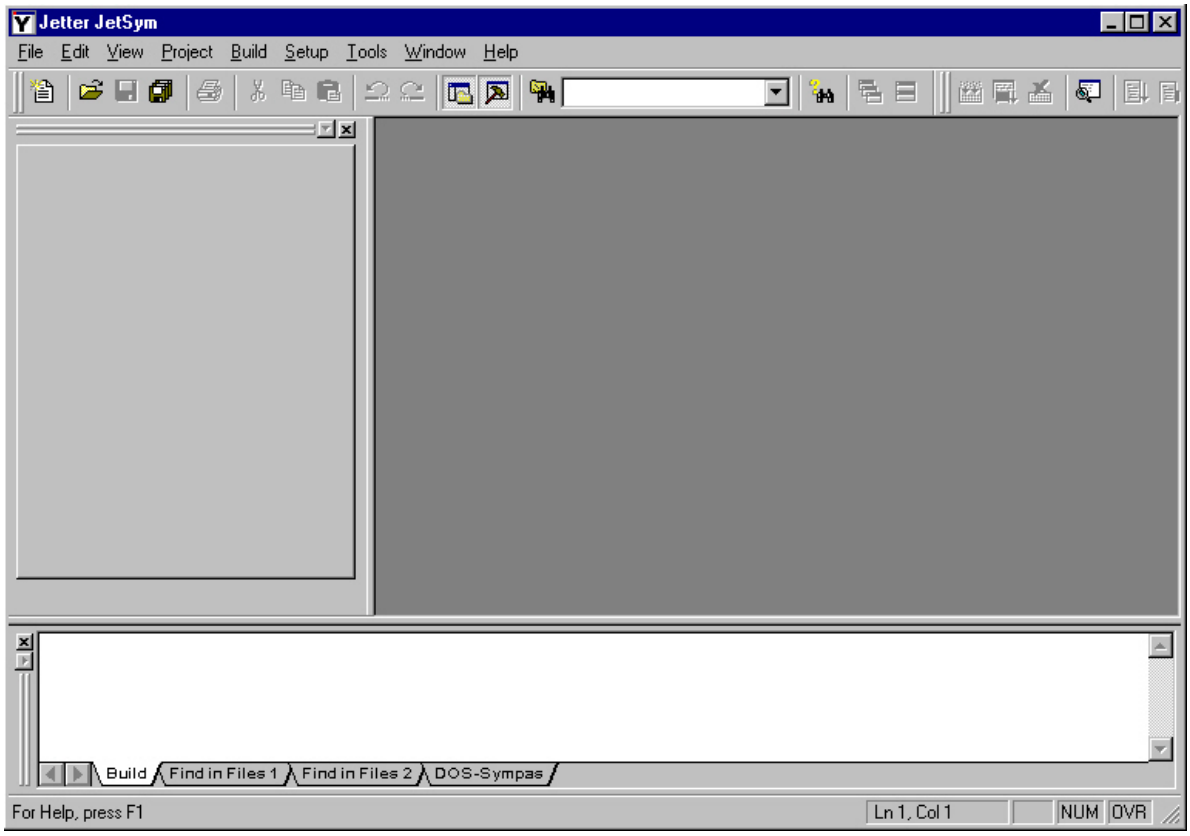


Figure 2

- Select *File / New* from the menu or press the *Ctrl-N* key combination. A dialog window opens as shown in fig. 3. In the input field for the *Project name*, enter `Gripper` for this exemplary project.

- If necessary, you can modify the path for the new project in the *Location* input field. A mouse-click on the button next to the input field opens a pick list to select the desired drive and directory.

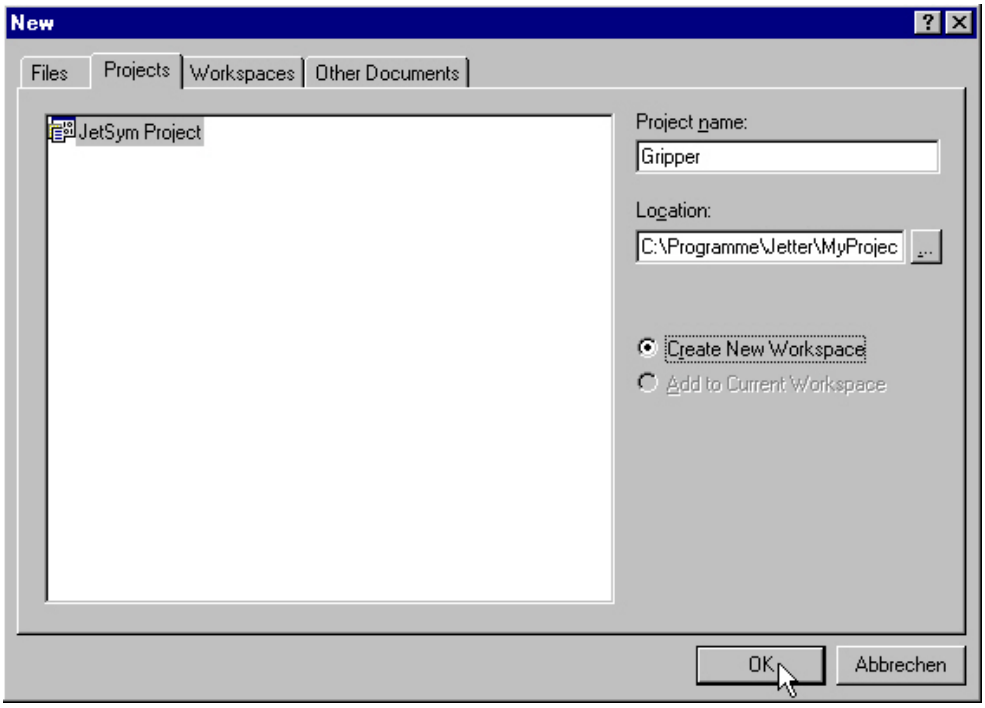


Figure 3

- JetSym creates a new workspace named *Gripper* in the *FileView* window of the development environment (see fig. 4). A workspace is comparable to a desktop where several files might be located. The files correspond to JetSym projects.
- In the new workspace, JetSym has created a project with the name *Gripper*.
- A project consists of an arbitrary number of various files stored in specified project folders.
    - Program source texts are stored in the *Program* folder. A program always consists of a master file. JetSym has already created a master file named *Gripper.spf*. Apart from the master file, any number of include files can be created which can be linked in the master file.
    - Symbol definitions are stored in the *Symbol* folder. Symbol definitions also consist of a master file and an arbitrary number of include files that can be nested. The name of the master symbol file for this exemplary project is *Gripper.ssf.*
    - „Setup screens“ can be defined in the *Setup* folder to display online values of the connected controller.
    - The *Dump* folder is ignored in this application note. Values from entire register areas of the controller can be stored in a data dump file.
    - The *Oscilloscope* folder is ignored in this application note. Definitions for scope recordings (graphic representation of modified register contents covering a defined period of time) can be stored in this folder.
    - Other files, such as Microsoft Office files, can be stored in the *Others* folder. Thus it is possible to assign any documentation directly to the project.

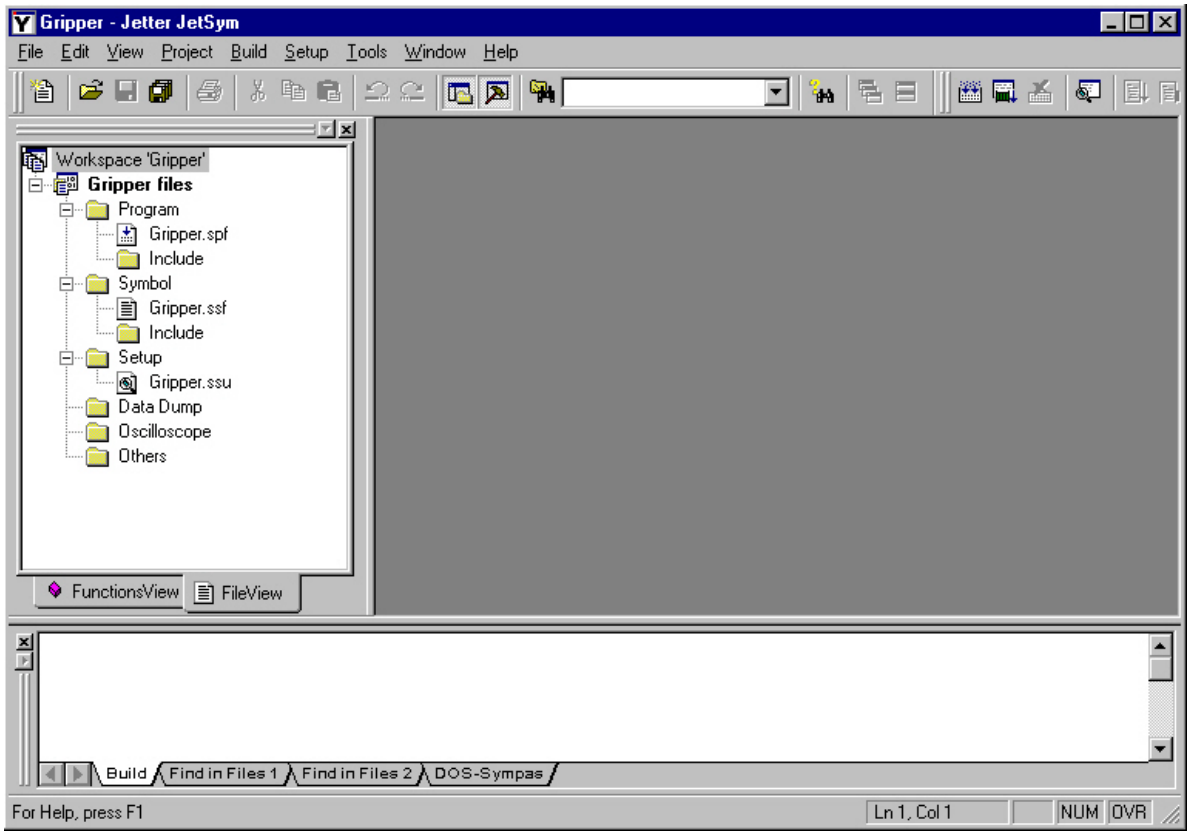


Figure 4

Creating the Symbol File

- Open the *Gripper.ssf* symbol file by double-clicking the file name in the *FileView* window.
- This opens a symbol editor (see figure 5) which is roughly similar to use like a spreadsheet. By default, the symbol editor offers four columns for entries:
    - The *Name* column to enter symbol texts.
    - The *Value* column contains the content of the symbol. This might be a register number or an input/output address.
    - If a specific value is to be pre-assigned to the symbol, for example as constant, this default value can be entered in the *Default* column.
    - The *Comment* column provides space for additional comments.
- Besides, the symbol editor allows to insert mere comment lines. Click any location in the symbol editor with the right mouse button as shown in fig. 6 (or press the key combination *Shift - F10*) to open a context menu. From the context menu, select *Insert Comment*.

- Important key combinations for the symbol editor:
    - *Tab key*: to go to the next column. The cursor goes to the next line when the end of the line is reached.
    - *Enter key*: to go to the next line
    - Press *Ctrl-Z* or select *Edit / Undo* from the menu to undo the last modification.
    - Press *Ctrl-Tab* to toggle between open windows (program editor, symbol editor, etc.).

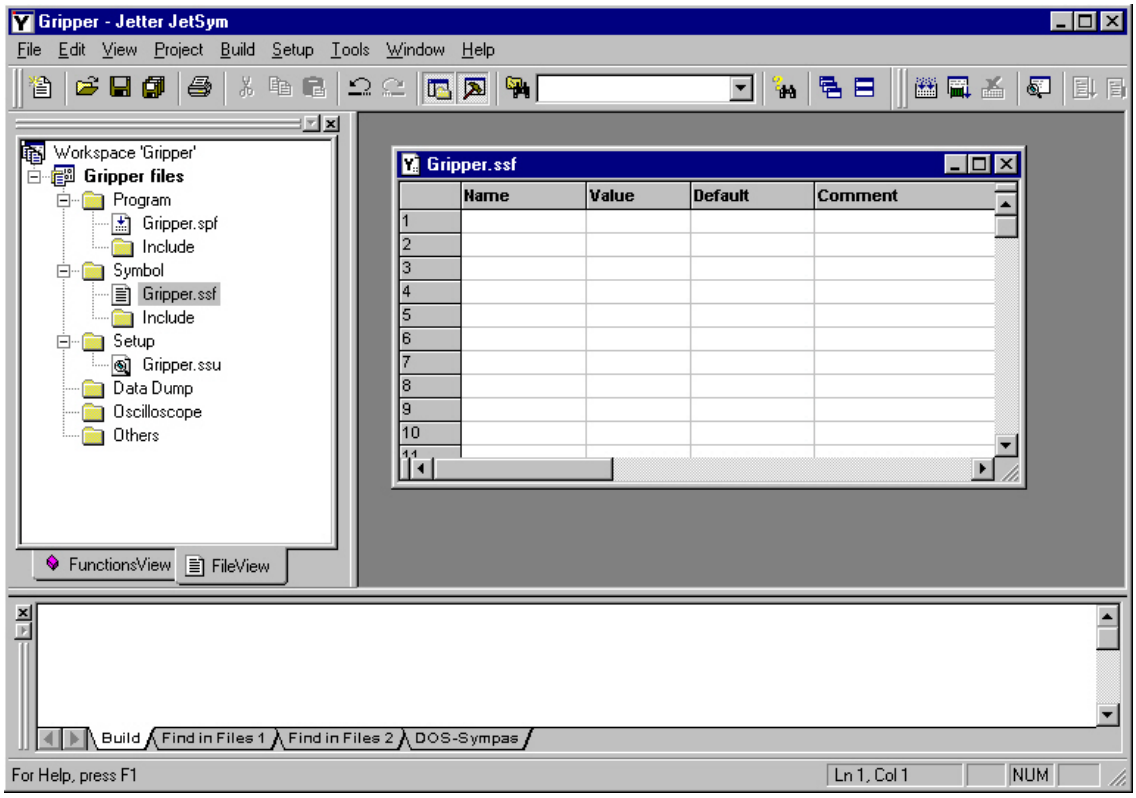


Figure 5

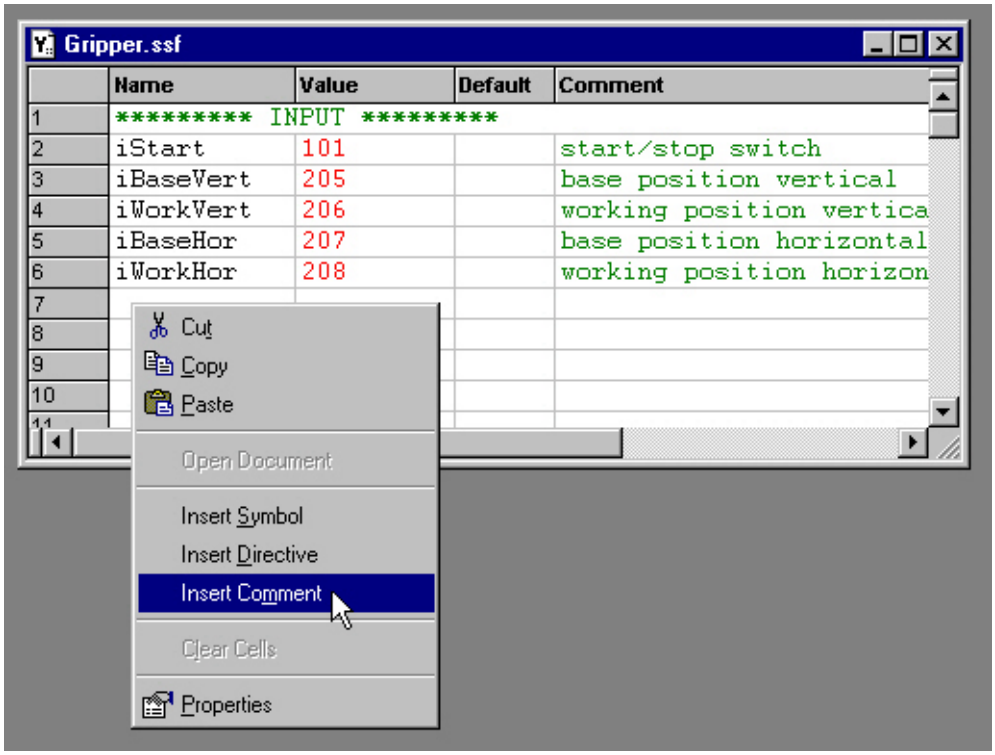
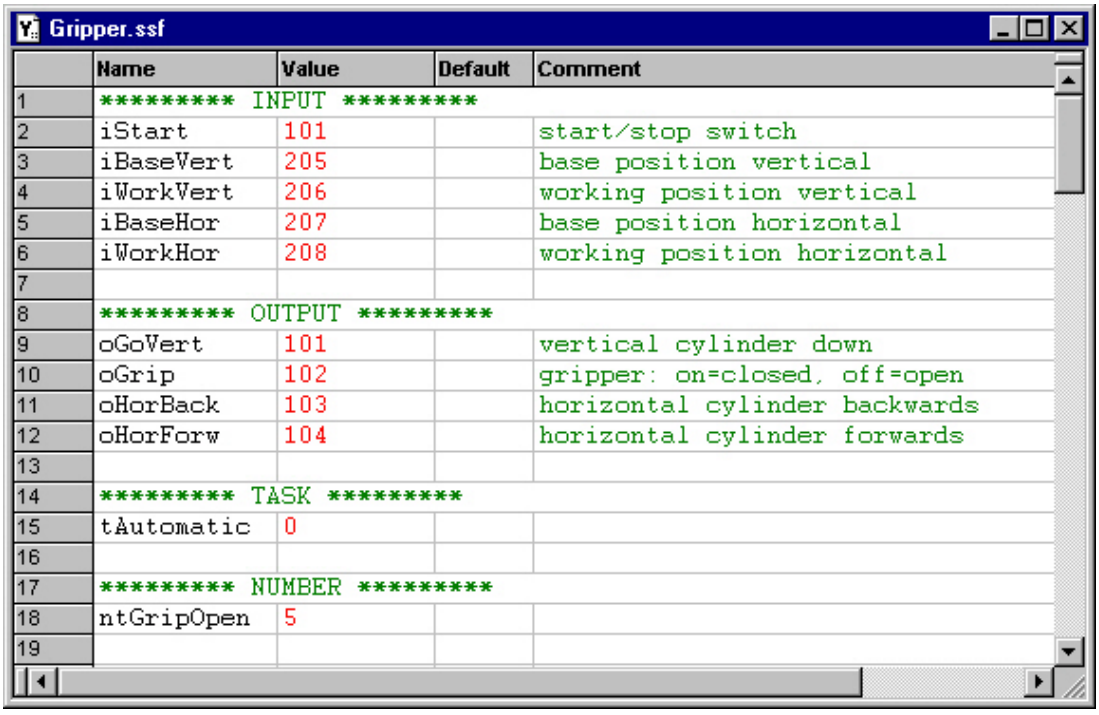


Figure 6

- After having defined all symbols as shown in fig. 7, save the symbol file. To do so select *File / Save* from the menu or press *Ctrl-S.*



	Name	Value	Default	Comment
1	********** INPUT **********			
2	iStart	101		start/stop switch
3	iBaseVert	205		base position vertical
4	iWorkVert	206		working position vertical
5	iBaseHor	207		base position horizontal
6	iWorkHor	208		working position horizontal
7				
8	********** OUTPUT **********			
9	oGoVert	101		vertical cylinder down
10	oGrip	102		gripper: on=closed, off=open
11	oHorBack	103		horizontal cylinder backwards
12	oHorForw	104		horizontal cylinder forwards
13				
14	********** TASK **********			
15	tAutomatic	0		
16				
17	********** NUMBER **********			
18	ntGripOpen	5		
19				

Figure 7

Generating Program Source Text

- Open the program source text file *Gripper.spf* by double-clicking the file name in the *FileView* window of the development environment. This opens an empty program editor window.

- You can enter control commands in the program editor by means of key shortcuts which usually start with the first letter of the corresponding command. Upon entering the corresponding key shortcut, the input dialog as known from the old DOS-SYMPAS opens and instruction parameters can be entered.

- Comments are indicated in the source text by a leading semicolon (;) or double slashes (//).

Important

In this case, operation seems to be like under DOS-SYMPAS. However, the program generated by the input dialogs is a mere text file, i.e. unlike DOS-SYMPAS, it is not possible to return to the input dialogs by pressing the enter key.

Note

Since the program is a mere text file, JetSym offers various options to edit and format text such as

- cut / copy / paste according to the Microsoft Windows standard
- extensive undo /redo possibilities
- several commands are possible in a line of text
- unlimited length of comment lines
- symbol and function names can have a length of up to 127 characters

Gripper.spf

```
TASK tAutomatic

IF OUT oGrip THEN       ;release item and go to base position
    -OUT oGrip
    DELAY ntGripOpen

THEN -OUT oGoVert       ;move vertical axis to base position

;if vertical axis is in base position then move hoizontal axis backwards to base position
WHEN IN iBaseVert THEN
-OUT oHorForw
OUT oHorBack

WHEN IN iBaseHor THEN

WHEN IN iStart THEN     ;if START/STOP operating switch is set then get item
OUT oGoVert

;if axis is in working position then close gripper and move up
WHEN IN iWorkVert THEN
OUT oGrip
DELAY ntGripOpen
-OUT oGoVert

WHEN IN iBaseVert THEN  ;move horizontal axis forward to working position
-OUT oHorBack
OUT oHorForw

WHEN IN iWorkHor THEN   ;lay down item at working position
OUT oGoVert

WHEN IN iWorkVert THEN
GOTO tAutomatic
```

Figure 8

- After having entered the program as shown in fig. 8, save the source text file containing the program. To do so select *File / Save* from the menu or press *Ctrl-S*.

Creating a Configuration

Important

Unlike the old DOS-SYMPAS which knows file suffixes like *.ppd for DELTA programs, JetSym does not know controller-specific program source texts and symbol files.
Controller-specific program parts can be distinguished by defining different configurations.

Note

The support of compiler instructions was extended considerably in JetSym.
If a program is to apply to several controllers (e.g. Delta and NANO), operator languages or machine variants, then the specific program parts can be activated by compiler instructions if necessary.

The use of compiler instructions is described in a separate application note.

- Open the *Project Settings* dialog window by selecting *Project / Settings* from the menu (key combination *Alt + F7*) or via context menu of the project in the *FileView* window, see fig. 9.

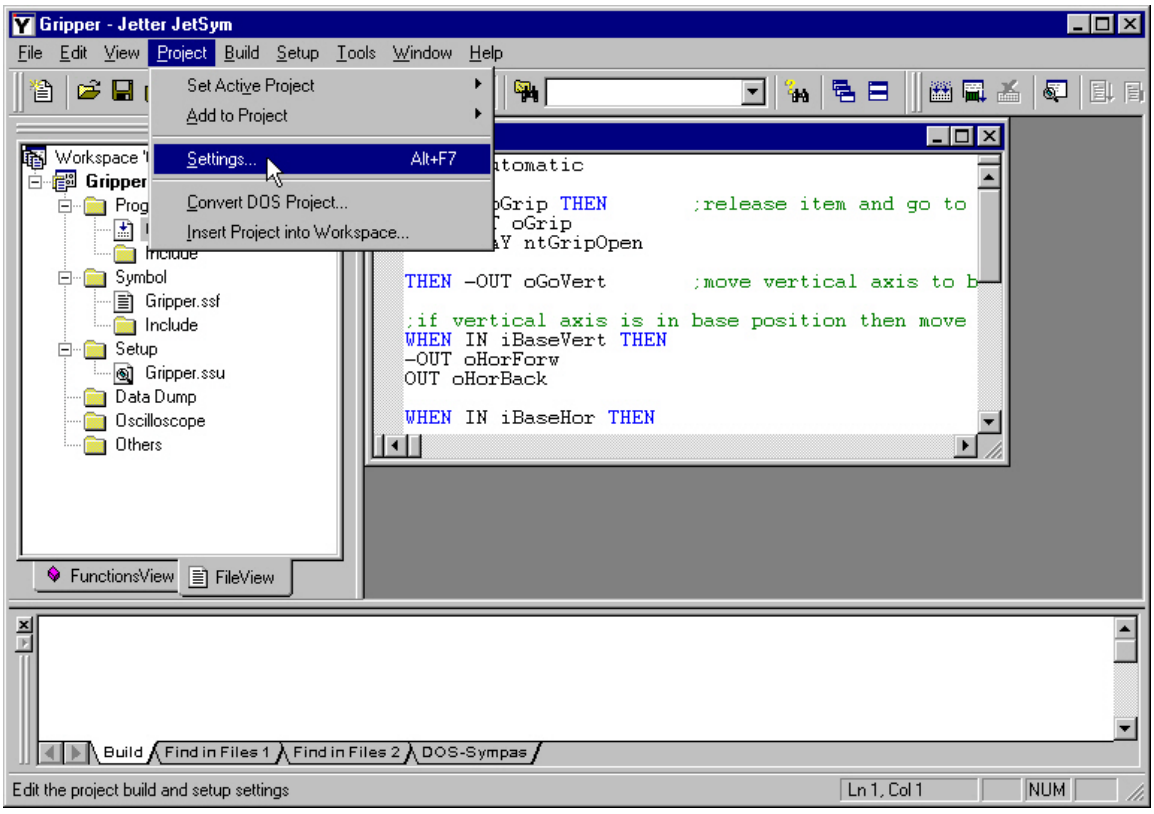


Figure 9

- The *Project Settings* dialog window as shown in fig. 10 consists of two parts:
    - The left part provides an overview of the current project, i.e. information provided in the overview:
        - file names of the master files of source text and symbol file (Gripper.spf and Gripper.ssf)
        - version number of the program
        - language used (currently English is the default)
        - current configuration (fig. 10 shows that *Debug* is indicated in the *Settings For* input field as configuration for our exemplary project).
    - The right part of the dialog window indicates configuration-specific settings which are categorized in the five following tabs:
        - *General* tab (see fig. 10):
          The output files generated by the compiler can be automatically stored in various directories depending on the configuration.
          The directories for temporary (*Intermediate files*) and final (*Output files*) files can be set individually. Besides, the destination directory and the file name of the EPROM file (a program file that is loaded to the controller later) can be specified as well as a constant file (*Data Dump constants*), i.e. the file with the default values for the registers, if necessary.
        - *Sympas* tab:
          This tab serves to set compiler directives for the current configuration. (The use of compiler instructions is described in a separate application note.)
        - *Controller* tab (see fig. 11):
          The settings on this tab refer to the connected controller for which the project is to be compiled.
          The interface settings as well as transmission parameters for the program download can be modified here.
        - *Customer* and *Commission* tabs:
          If necessary, further information of customers (address, telephone, contact person, etc.) can be entered here.
- For our exemplary project “pick and place” gripper, no settings need to be made in these tabs. The standard settings are the correct settings for the task described in this application note. You can now close the *Project Settings* window.

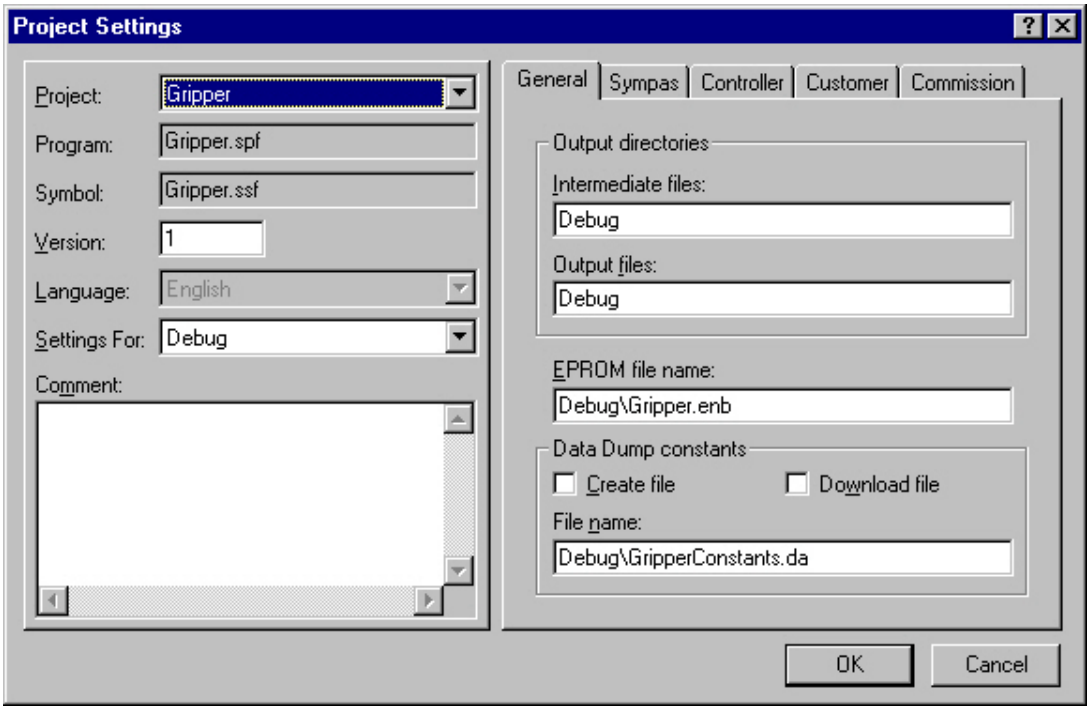


Figure 10

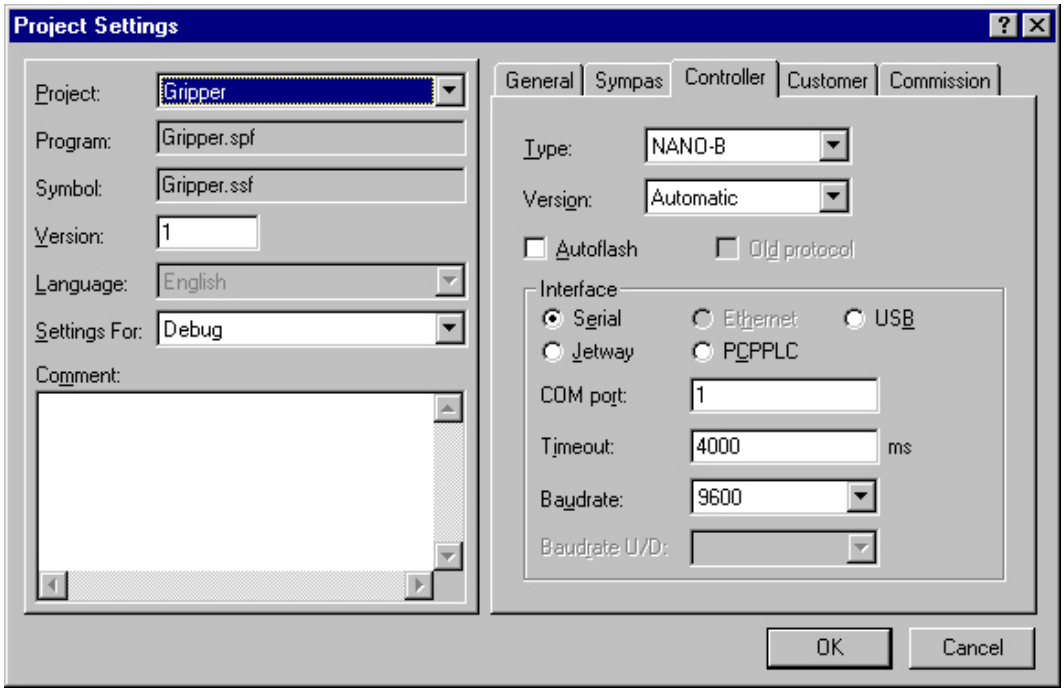


Figure 11

Compiling and Downloading

- The exemplary project need to be compiled in order to be loaded to the controller as executable program. To do so, select *Build / Build* from the menu (key *F7*) (see fig. 12).

Note

Build checks whether source text files (program or symbols) have been modified after the last compilation and compiles only the modified data again.

Rebuild All compiles all source text files no matter whether they have been modified or not.

Clean clears all files of previous compilations. A *Build* carried out after a *Clean* has the same effect as a *Rebuild All*.

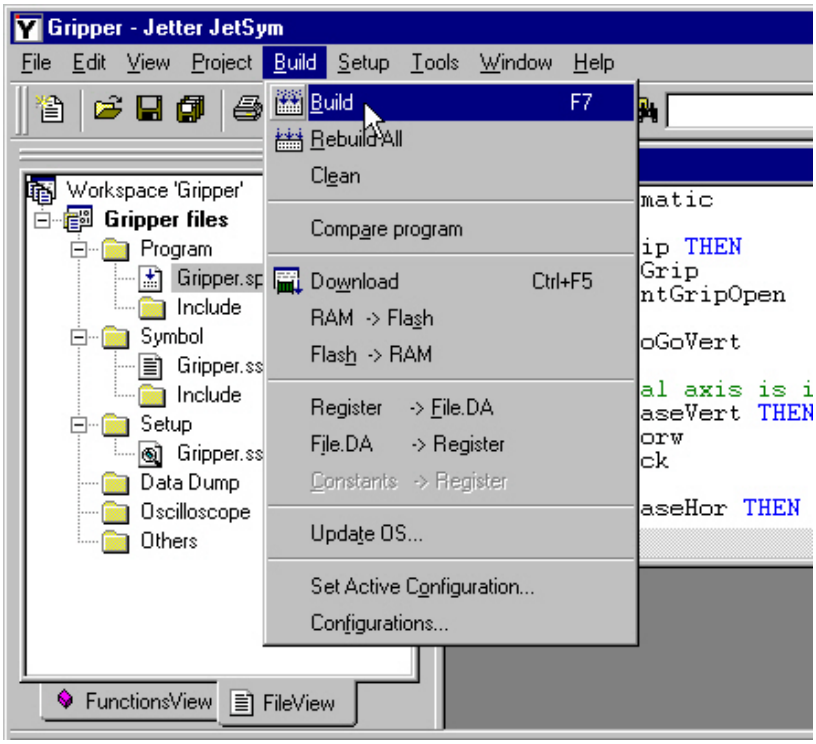


Figure 12

- The result of the compilation should look similar to what is shown in fig. 13. The compiler should not indicate any errors or warnings. If the result differs from fig. 13, please check whether all previous steps were carried out exactly as described in this application note.

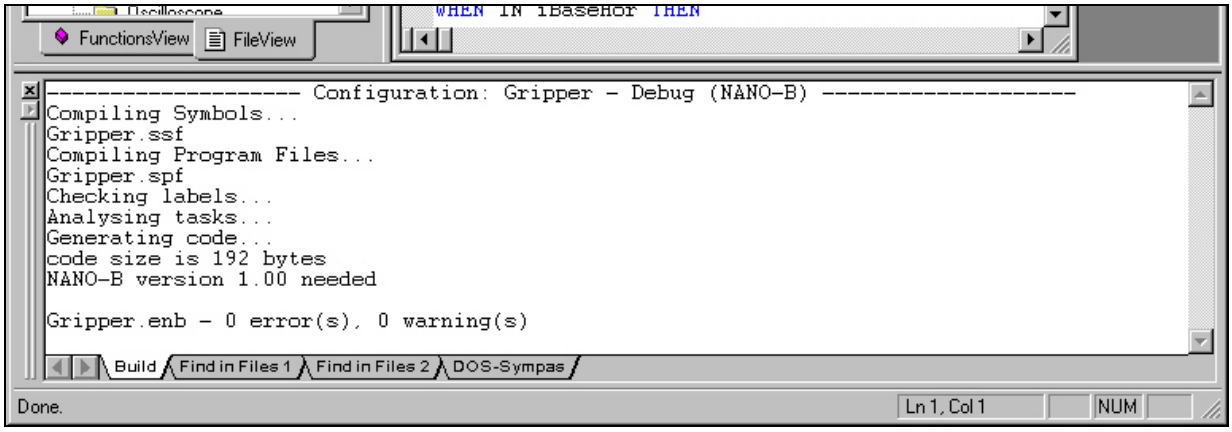


Figure 13

- Load the compilated program to the connected controller by selecting *Build / Download* (*Ctrl+F5* keys) from the menu (see fig. 14).

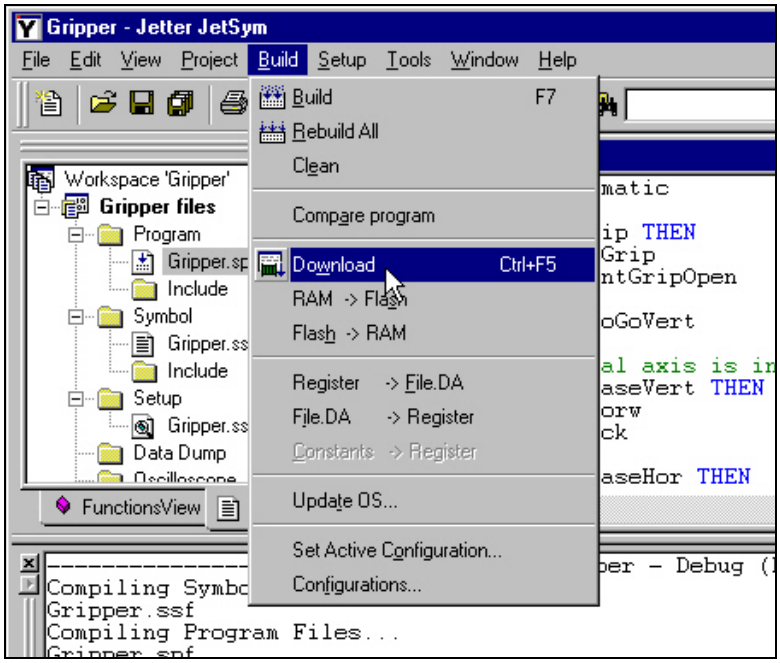


Figure 14

- If the download has been completed successfully, JetSym will display a report similar to that shown in fig. 15. If errors occurred during the download, please eliminate them by means of the error message. Typical error causes are:

    - The EM-PK programming cable between computer and controller is not plugged.
    - The controller is not supplied with power.
    - The incorrect port of the computer is used. Check the setting in the *Controller* tab, see “Creating a Configuration“.
    - The incorrect controller type or the wrong baud rate is used. Check the setting in the *Controller* tab, see „Creating a Configuration“.

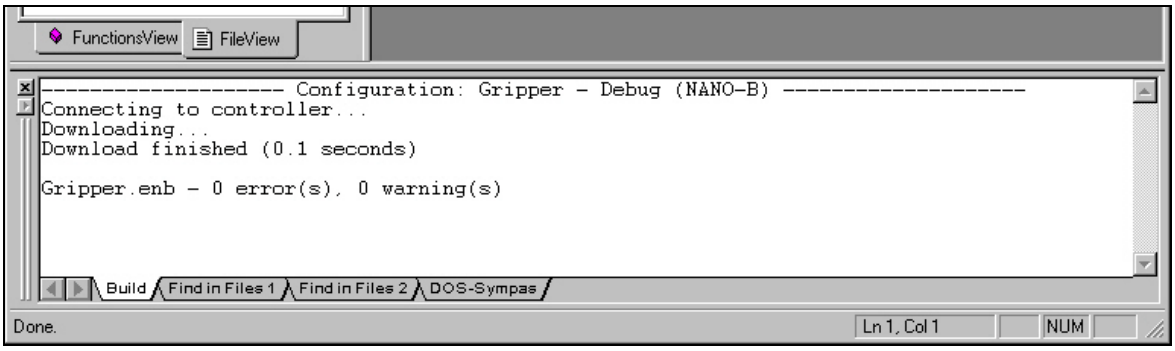


Figure 15

You have just successfully completed your first project in JetSym. Congratulations !
You can now test the functions of your programs and extend them according to your needs.