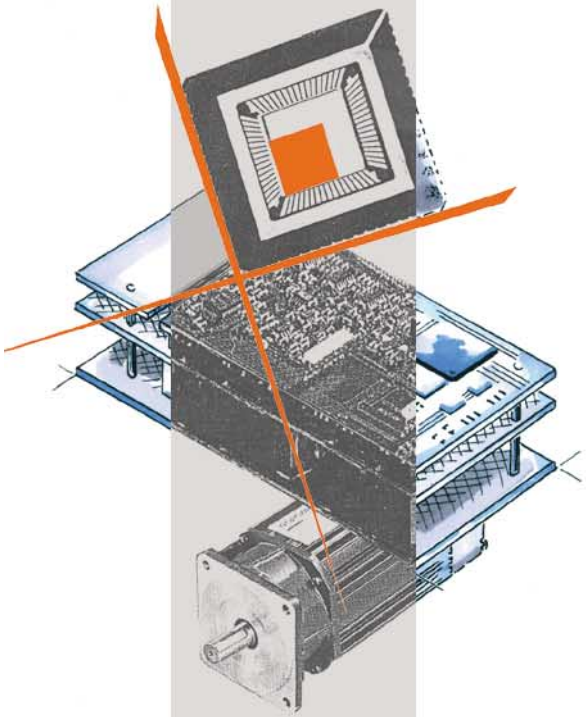
# Manual Additional Functions ND31 and ND32

**Version 4/2004**




NOVOTRON

für Dynamik und Bewegung

**Please read the following information about the symbols used in the manual:**

**Danger! Voltages may cause serious or fatal injury!**

**Noncompliance with instructions can endanger the life and sanity of persons!**

**Caution !** ☛ **Caution! Make sure to handle the device correctly!**

**Noncompliance with instructions can lead to the destruction or can cause malfunction of the device or the entire equipment!**

☞ **Link or recommendation**
Link to other sections of the text or recommendation for practical usage

**1 2** **Sequencing of an instruction**

**Menu *Limit values*** **Designation of a menu or submenu**

**Command** *Channel1* **Designation of a command or function**

[ ], [enter] **Designation of a key or key combination**

**Graphical representation of registers**

| **SwVersion** | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
|---|---|---|---|---|---|---|---|---|
| **0xFF3D** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

| | |
|---|---|
| **SwVersion** | Designation |
| **0xFF3D** | Address |
| R/W | Read/Write |
| R | ReadOnly |

**1 General Information ..... 4 - 1**

1.1 About this manual ..... 4 - 1

1.2 After-sales service ..... 4 - 1

1.3 Designations ..... 4 - 2

1.4 Industrial property rights ..... 4 - 2

**2. Safety Instructions ..... 4 - 3**

**3. General Notes ..... 4 - 5**

**4. NOVODRIVE Positioning Control (NM31-40) ..... 4 - 7**

4.1 Homing (Home Switch Search) ..... 4 - 7

4.2 Setting An Actual Value ..... 4 - 7

4.3 Zero Reference Point Search ..... 4 - 7

4.4 Positioning Procedure ..... 4 - 8

**5. Homing (Standard) ..... 4 - 9**

5.1 Function ..... 4 - 9

5.2 Register ..... 4 - 9

5.3 Initialization Of Homing Procedure ..... 4 - 10

5.4 Execution Of Homing Procedure ..... 4 - 11

5.5 End Of Homing Procedure ..... 4 - 11

**6. Setting An Actual Value (Standard) ..... 4 - 13**

6.1 Function ..... 4 - 13

6.2 Register ..... 4 - 13

6.3 Procedure ..... 4 - 13

**7. Zero Reference Point Search (Standard) ..... 4 - 15**

7.1 Function ..... 4 - 15

7.2 Register ..... 4 - 15

7.3 Initialization Of Zero Reference Point Search ..... 4 - 16

7.4 Execution Of Zero Reference Point Search ..... 4 - 16

7.5 End Of Zero Reference Point Search ..... 4 - 17

7.6 Notes ..... 4 - 17

**8. Relative Positioning (optional) ..... 4 - 19**

8.1 Function ..... 4 - 19
8.2 Register ..... 4 - 19
8.3 Initialization Of Relative Positioning ..... 4 - 21
8.4 Execution Of Relative Positioning ..... 4 - 22
8.5 End Of Relative Positioning ..... 4 - 22

**9. Absolute Positioning (optional) ..... 4 - 23**

9.1 Function ..... 4 - 23
9.2 Register ..... 4 - 23
9.3 Initialization Of Absolute Positioning ..... 4 - 25
9.4 Execution Of Absolute Positioning ..... 4 - 25
9.5 End Of Absolute Positioning ..... 4 - 25

**10. Online Positioning (optional) ..... 4 - 27**

10.1 Function ..... 4 - 27
10.2 Register ..... 4 - 27
10.3 Initialization Of Online Positioning ..... 4 - 29
10.4 Execution Of Positioning ..... 4 - 30
10.5 End Of Positioning ..... 4 - 30

**11 Error Sources / Troubleshooting With Positioning Control ..... 4 - 31**

**12 Table Interpolation (optional) ..... 4 - 33**

12.1 Function ..... 4 - 33
12.1.1 Table interpolation without override (starting with H8 Version 2.03) ..... 4 - 33
12.1.2 Trace function (starting with H8 Version 2.03) ..... 4 - 33
12.1.3 Time override (starting with H8 Version 3.00) ..... 4 - 33
12.1.4 Speed override (starting with H8 Version 3.00) ..... 4 - 33
12.2 Register ..... 4 - 33
12.3 Initialization Of Table Interpolation ..... 4 - 35
12.4 Execution Of Table Interpolation ..... 4 - 36
12.5 End Of Table Interpolation ..... 4 - 36
12.6 Operating Mode ..... 4 - 36
12.7 Table ..... 4 - 36
12.8 Trace Function ..... 4 - 37
12.9 Override Stretching Factor ..... 4 - 37
12.10 Error Sources / Troubleshooting ..... 4 - 38

**13 Cam Disk Function (Standard) ..... 4 - 39**

13.1 Function ..... 4 - 39
13.2 Register ..... 4 - 39
13.3 Initialization Of Cam Disk Function ..... 4 - 41
13.4 Table ..... 4 - 41
13.5 Example ..... 4 - 42
13.6 Error Sources / Troubleshooting ..... 4 - 42

**14 Controller Synchronization ..... 4 - 43**

14.1 Register ..... 4 - 43
14.2 Wiring ..... 4 - 44

**15 Ablaufsteuerung (Software Extension) ..... 4 - 45**

15.1 Function ..... 4 - 45
15.2 Installation ..... 4 - 46
15.3 The DOS program ND31ABL.EXE ..... 4 - 47
15.3.1 Installation ..... 4 - 47
15.3.2 Operation ..... 4 - 47
15.4 Execution ..... 4 - 48
15.4.1 Selection of commands ..... 4 - 48
15.4.2 Feedback over digital outputs ..... 4 - 49
15.4.3 Functional parameters ..... 4 - 50
15.5 Description Of Functions ..... 4 - 50
15.5.1 Homing ..... 4 - 50
15.5.2 Positioning ..... 4 - 51
15.5.3 Jogging ..... 4 - 51
15.5.4 Clear error ..... 4 - 52
15.5.5 Profile ..... 4 - 52
15.5.6 Teach-in ..... 4 - 52
15.5.7 Curve ..... 4 - 52
15.5.8 Analog-mode setpoint setting ..... 4 - 53
15.5.8 Step/direction setpoint setting ..... 4 - 53
15.5.10 Torque setting over Analog Input 2 ..... 4 - 53
15.5.11 Setting an actual value ..... 4 - 53
15.5.12 Zero reference point search ..... 4 - 54
15.5.13 Auto adjustment ..... 4 - 54
15.5.14 Autokomm (starting with 02/18/1999) ..... 4 - 54
15.6 Error Messages ..... 4 - 55
15.7 Error Sources / Troubleshooting ..... 4 - 55

NOVOTRON

# 1 General Information

## 1.1 About this manual

The entire documentation of NOVODRIVE comprises 7 parts:

**1** Manual **Basic Device ND31 and ND32**
Standard

**2** Manual **Bus Functions ND31 and ND32**
On demand

**3** Manual **Basic Functions ND31 and ND32**
On demand

**4** Manual **Additional Functions ND31 and ND32**
On demand *)

**5** Reserved

**6** Manual **Start-up ND31 and ND32**
Standard

**7** **Instructions for installation/exchange of ND31 and ND32**
Standard (leaflet)

**The symbols used in the manuals are listed and explained on the inside front cover.**

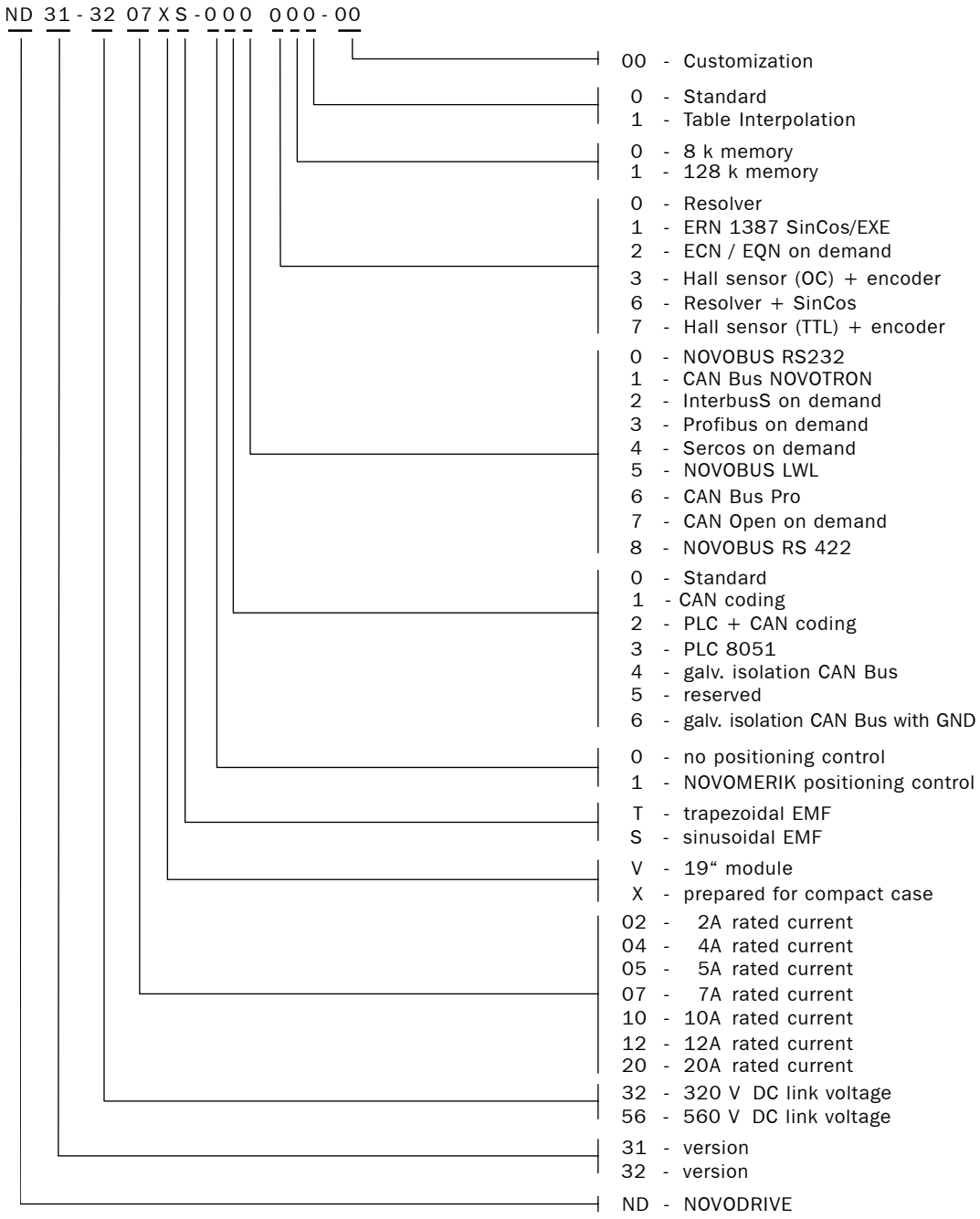
*) This manual

## 1.2 After-sales service

NOVOTRON GmbH
Mauserstrasse 31
71640 Ludwigsburg
Germany

Phone: +49 - (0)71 41 - 29 69 - 0
Fax: +49 - (0)71 41 - 29 69 - 22

## 1.3 Designations




## 1.4 Industrial property rights

IBM is a registered trademark of IBM Corporation.

## 2. Safety Instructions

**NOVODRIVE contains voltages that can be fatal!**

***Wiring*** Before switching on NOVODRIVE, carefully check the wiring. Make sure all plugs are properly connected and the device is properly grounded.

***Protection*** Make sure no voltage-carrying parts may be accidentally touched and NOVODRIVE safety components are in place and properly connected.

***Emergency power-off*** Provide an emergency power-off by which the motor can be stopped at any time.

***Discharge time and contact voltage*** After being switched off the electrolytic capacitors require at least five (5) minutes to discharge. That means: After being switched off the device still contains dangerous voltage for up to five minutes. During this time, do not touch the device or disconnect any plug.

In case the motor is still turning after the supply voltage has been switched off, hazardous contact voltage may be present in the device until its standstill. Discharge of the capacitors then begins after the standstill.

***Inrush current limitation*** Frequent switching of the supply voltage should be avoided, since thereby the inrush current limiter of NOVODRIVE may be overcharged, which may lead to the destruction of the inrush current limiting resistor. Wait one minute between switching on and switching off again.

***Switching on/off sequence*** When switching on, first apply the 24 VDC supply voltage for the NOVODRIVE control section before connecting with the power supply. When switching off, proceed vice versa.

☞ Please read:
„Basic Device ND31 and ND32“ manual,
Chapter 2, „Safety Instructions“

NOVOTRON

## 3. General Notes

The functions described here can be used by reading out and writing to registers over NOVOBUS or CAN Bus protocol NOVOTRON (see manual „Bus Functions ND31 and ND32“). This does not apply to the Ablaufsteuerung (sequencing control), which is programmed over the PC and controlled over digital inputs and outputs.

Dealing with this manual requires that the user is already familiar with the basic functions of NOVODRIVE. That is why aspects and procedures thoroughly described in the „Basic Functions“ manual will not be repeated here in detail. Among them are:

- enable, start, stop
  (see manual „Basic Functions“, Chapter 3.4.1)

- setpoint setting
  (see manual „Basic Functions“, Chapter 3.6.5)

- switching on/off position controllers
  (see manual „Basic Functions“, Chapter 3.6.5)

NOVOTRON

# 4. NOVODRIVE Positioning Control (NM31-40)

If requested, NOVODRIVE can contain a software module for single-axis positioning. The positioning control has been set up without additional hardware. It can be controlled by a central computer (e.g. PC) over NOVOBUS or CAN Bus.

The central computer can transmit the values for a new distance or a new target position by means of the bus, and it can start and monitor the positioning procedure.

## 4.1 Homing (Home Switch Search)

Both with resolver and encoder as feedback system, the absolute position is unknown after power-on. If a resolver is used, the angular position is determined, but not the number of revolutions. In order to be able to determine the actual position in its entirety, the zero reference point must be determined by means of a homing procedure.

Furthermore, it can be specified whether the drive is to move to the resolver zero point after the home switch has been found, and from which direction this is to be done.

## 4.2 Setting An Actual Value

By means of this function, the absolute position at the home switch can be determined.

## 4.3 Zero Reference Point Search

Sometimes encoders and linear measuring systems are equipped with their own zero pulse. This zero pulse allows to determine the position much more exactly than this would be possible using a home switch.

With the zero reference point search procedure the zero pulse is searched in one direction at a given speed. At the zero reference point the actual position value is set automatically.

## 4.4 Positioning Procedure

There are three types of positioning:

- relative positioning,
- absolute positioning,
- online positioning.

A new positioning procedure starts with NOVODRIVE receiving the values for a new distance or target position from a bus system. After that the position calculation can be started. In case of relative and absolute positioning, the calculation can be made only if the motor stands still. This condition is not required for online positioning.

As soon as NOVODRIVE has finalized this internal calculation, the positioning procedure can be started. The positioning works with a trapezoidal speed curve. Acceleration (ramp) and speed are programmable.

By using NOVOBUS it can be verified, if and when the target position has been reached.

# 5. Homing (Standard)

## 5.1 Function

The zero reference (home) point is searched in the direction programmed. If the search has been successful, the drive either stops at the position of the home switch or at the resolver zero point.

## 5.2 Register

| | |
|---|---|
| **Register** | **?512us** |
| **Address** | 0xFEAE |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Value range** | Name / Address / Notes<br>**@Dummy** 0x01EA Homing deactivated<br>**@Referenz** 0x01E4 Homing activated |

| | |
|---|---|
| **Register** | **RefV1** |
| **Address** | 0xFEDE |
| **Size** | 16-bit signed |
| **Access** | R/W |
| **Function** | Homing speed 1 |
| **Scaling** | See manual Basic Functions Chapter 3.6.3.3 |

| | |
|---|---|
| **Register** | **RefV2** |
| **Address** | 0xFEE0 |
| **Size** | 16-bit signed |
| **Access** | R/W |
| **Function** | Homing speed 2 |
| **Scaling** | See manual Basic Functions Chapter 3.6.3.3 |

| | |
|---|---|
| **Register** | **InternSoll** |
| **Address** | 0xFE4C |
| **Size** | 16-bit signed |
| **Access** | R |
| **Function** | The speed setpoint specified by the homing procedure |
| **Scaling** | See manual Basic Functions Chapter 3.6.3.3 |

| RefStatus | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
|---|---|---|---|---|---|---|---|---|---|
| 0xFF79 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit |
| **Start** | 1 | x | x | x | x | 0 | 0 | 0 | Start homing proced. |
| **State** | | | | | | | | | |
| (read only) | 1 | x | x | x | x | 0 | 0 | 1 | Move at **RefV1** |
| | 1 | x | x | x | x | 0 | 1 | 0 | Move at **RefV2** |
| | 1 | x | x | x | x | 0 | 1 | 1 | Creep in same direction as**RefV1** |
| | 1 | x | x | x | x | 1 | 0 | 0 | Delay |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Canceled, because e.g. start is missing |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Move to home switch |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | Move to resolver zero point |
| **Options for start** | | | | | | | | | |
| | | 0 | 0 | | | | | | Search for home switch only |
| | | 1 | x | | | | | | Move to resolver zero point only |
| | | 0 | 1 | | | | | | Move to home switch and resolver zero point |
| | | | | x | 0 | | | | Move to resolver zero point on shortest way |
| | | | | 0 | 1 | | | | Move to resolver zero point in positive direction |
| | | | | 1 | 1 | | | | Move to resolver zero point in negative direction |

## 5.3 Initialization Of Homing Procedure

1. NOVODRIVE must be in stop state (see manual Basic Function, Chapter 3.4.2).
2. Activate homing procedure by setting **?512us** to **@Referenz**.
3. Switch on position controller by setting **?nsoll** to **nsoll2**.
4. Define setpoint over **InternSoll** (**?Sollwert** = **InternSoll**).
5. Set homing speed **RefV1** and **RefV2**. **RefV2** must be significantly lower than **RefV1,** and it must have the opposite sign.
6. Connect home switch to GPIn4 (normaly-open).

## 5.4 Execution Of Homing Procedure

**7** Enable and start NOVODRIVE, otherwise the homing procedure is canceled.
**8** Set **RefStatus** to 1xxx x000, with xxx x representing the desired operating mode.
**9** Move at **RefV1**, as long as GPIN4 is on 0 V.
**10** Switch GPIN4 to 24 V. The motor gets stopped until its standstill.
**11** The motor moves at **RefV2** in the opposite direction, as long as GPIN4 is on 24 V.
**12** Switch GPIN4 to 0 V. The motor gets stopped again.
**13** The motor creeps in the direction it first moved to, until you switch GPIN4 to 24 V again.
**14** If requested, the drive moves to the resolver zero point.
**15** The motor stands still and remains in a controlled state.

## 5.5 End Of Homing Procedure

**16** The homing procedure is completed when **RefStatus** takes the values 0x01 (home switch only) or 0x41 (home switch and resolver zero point). The register can be read out by means of the **NB_ReadByte** function.

# 6. Setting An Actual Value (Standard)

## 6.1 Function

An actual position can be assigned with a new value. Thereby it is possible to shift the zero reference point after a homing procedure. However, the function can also be activated at any other point in time.

## 6.2 Register

| | |
|---|---|
| **Register** | **?512usB** |
| **Address** | 0xFECA |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Value range** | Name Address Notes<br>**@Dummy** 0x01EA Function deactivated<br>**@IstwertSetzen** 0x01EE Function activated |

Reference position (see also manual Basic Function, Chapter 3.6.3.5):

| Register | Size | Access | Resolution | | Revolutions | Position(0...360°) |
|---|---|---|---|---|---|---|
| **RefUmdr, RefLage** | 32 bit | R/W | 0,00549° | | 16 bit | 16 bit |
| | | | Address | | 0xFEDC | 0xFEDA |
| **RefUmdrH** | 8 bit | R/W | | 8 bit | | |
| | | | Address | 0xFF2B | | |

## 6.3 Procedure

1 Enter the value for the desired rotor position in **RefLage** and the value for the desired speed in **RefUmdr** and **RefUmdrH**, respectively.
2 Enter the address of the function **@IstwertSetzen** in **?512usB** to execute the function.
3 **?512usB** is automatically reset to **@Dummy** after the execution of the function.
4 The actual position is now identical with the reference position set.

# 7. Zero Reference Point Search (Standard)

## 7.1 Function

If sine encoders or linear measuring systems are used, the zero reference point search allows you to determine the absolute position with reference to the zero pulse of the measuring system. Basically, this function is identical with the homing procedure. The only difference is that the zero pulse is evaluated instead of the home switch. If rotary encoders are used, the absolute position can be determined only within one revolution.

At the zero reference point not only the absolute position but also the commutation position is redetermined. By this it is possible to improve the commutation position in case of inaccurate commutation tracks (see manual Basic Functions, Chapter 3.9.3.9).

**If a sine encoder with commutation track or analog Hall signals are used, the parameters PhiPO and KommSpurOffset must be determined first. If for determining the initial position the Autokomm function has been used instead of a commutation track, set Bit 2 in register Freigabe0. Otherwise the motor may lose its commutation position.**

## 7.2 Register

| **Register** | **?512us** |
|---|---|
| **Address** | 0xFEAE |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Value range** | Name Address Notes<br>**@Dummy** 0x01EA Search deactivated<br>**@NullSuche** 0x01F2 Search activated |

| **Register** | **RefV3** |
|---|---|
| **Address** | 0xFED6 |
| **Size** | 16-bit signed |
| **Access** | R/W |
| **Function** | Searching speed |
| **Scaling** | See manual Basic Function, Chapter 3.6.3.3 |

| Register | NPIOffs |
|---|---|
| Address | 0xFED0 |
| Size | 16-bit signed |
| Access | R/W |
| Function | Shifting the zero reference point |
| Scaling | 0,00549 ° / increment |

Reference position (see also manual Basic Function, Chapter 3.6.3.5):

| Register | Size | Access | Resolution | | Revolutions | Position(0...360°) |
|---|---|---|---|---|---|---|
| **RefUmdr, RefLage** | 32 bit | R/W | 0,00549 ° | | 16 bit | 16 bit |
| | | | Address | | 0xFEDC | 0xFEDA |
| **RefUmdrH** | 8 bit | R/W | | 8 bit | | |
| | | | Address | 0xFF2B | | |

| **RefStatus** | R | R | R | R | R | R | R | R | |
|---|---|---|---|---|---|---|---|---|---|
| **0xFF79** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit |
| | x | 0 | x | x | x | x | x | x | Zero point search active |
| | x | 1 | x | x | x | x | x | x | successfully completed |

## 7.3 Initialization Of Zero Reference Point Search

**1** The speed of the search is set in **RefV3**. Note that the speed must be low enough to let the zero pulse be active for at least 1 ms.

## 7.4 Execution Of Zero Reference Point Search

**2** Activate the zero reference point search by setting **?512us** to **@Nullsuche**.

**3** As soon as enable and start are activated, the function starts. The operation is immediately canceled if enable or start are deactivated, if any of the two limit switches responds or if an error occurs (see manual Basic Functions, Chapter 3.4.2).

**4** When the motor reaches the zero reference point, it stops. The actual position at the zero reference point is set to the reference position value. The commutation position at the zero reference point is identical with **PhiPO**.

**5** All pointers are automatically reset.

## 7.5 End Of Zero Reference Point Search

**6** The zero reference point search is completed when **?512us** points to **@Dummy** again. If the search has been successful, this is indicated in Bit 6 of **RefStatus**.

## 7.6 Notes

The zero reference point of the measuring system is determined as follows:

- The zero pulse, track A and track B must be positive at the same time. Thereby a scope of ¼ period of the measuring system is marked.
- The nearest period zero point is then taken as the zero reference point.

If several search attempts lead to the result that the zero reference point jumps back and forth by one period length, the zero pulse cannot be assigned unambiguously to a period of the measuring system. This difficulty can be overcome by means of **NPIOffs**. This parameter „shifts“ the position of the zero pulse when the zero reference point is being determined.

NOVOTRON

# 8. Relative Positioning (optional)

## 8.1 Function

A relative distance is written to NOVODRIVE, then the calculation is started. When the calculation is completed, the positioning procedure can be executed. If the target position has been reached within a given tolerance, this will be displayed in a status register.

## 8.2 Register

| | |
|---|---|
| **Register** | **?512us** |
| **Address** | 0xFEAE |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Value range** | Name / Address / Notes<br>**@Dummy** 0x01EA Relative positioning deactivated<br>**@PSrelativ** 0x01E2 Relative positioning activated |

| | |
|---|---|
| **Register** | **ps-v0** |
| **Address** | 0xFEE6 |
| **Size** | 16-bit signed |
| **Access** | R/W |
| **Function** | Positioning speed |
| **Scaling** | See manual Basic Functions, Chapter 3.6.3.3 |

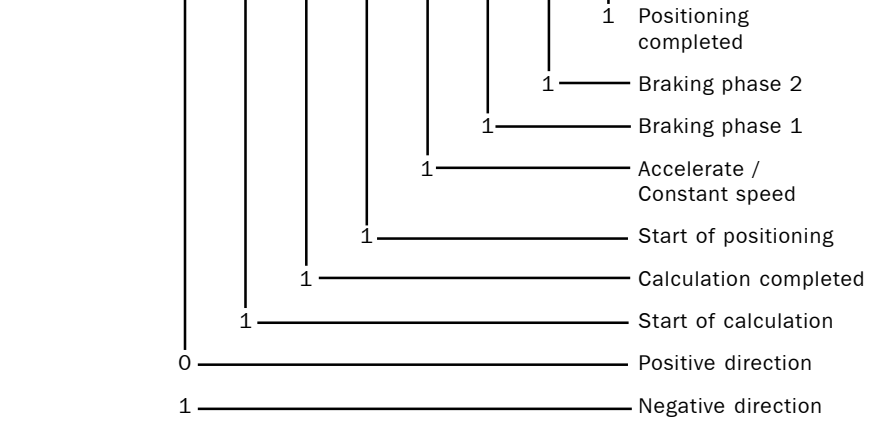
| | |
|---|---|
| **Register** | **ps-impuls** |
| **Address** | 0xFE2C |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Lower 16 bits of relative distance |
| **Scaling** | 0,00549 ° / increment |

| Register | **ps-umdrehung** |
|---|---|
| **Address** | 0xFE2A |
| **Size** | 16-bit signed |
| **Access** | R/W |
| **Function** | Upper 16 bits of relative distance |
| **Scaling** | 1 revolution / increment |

| Register | **Window** |
|---|---|
| **Address** | 0xFED8 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | ± Tolerance for „In Position“ message |
| **Scaling** | 0,00549 ° / increment |

| Register | **InternSoll** |
|---|---|
| **Address** | 0xFE4C |
| **Size** | 16-bit signed |
| **Access** | R |
| **Function** | The speed setpoint specified by the positioning control |
| **Scaling** | See manual Basic Functions, Chapter 3.6.3.3 |

| ps-status | R/W | R/W | R | R/W | R | R | R | R | |
|---|---|---|---|---|---|---|---|---|---|
| 0xFF73 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit |




## 8.3 Initialization Of Relative Positioning

1. NOVODRIVE must be in the stop state (see manual Basic Functions, Chapter 3.4.2).
2. Activate relative positioning by setting **?512us** to **@PSrelativ**.
3. The positioning control can only work correctly if the position controller is active. For this to be the case, enter the address of **nsoll2** in **?nsoll**.
4. Define setpoint by means of **InternSoll** (**?Sollwert** = **InternSoll**).
5. Make sure the ramp parameters are configured correctly. Enter the value for the acceleration/braking ramp in **Rampe+**. For the ramp only values are allowed for which the lower 6 bits are 0. The ramp pointers **?Rampe+** and **?Rampe-** must indicate the address of **Rampe+**. Whereas in the case of newer software versions this configuration is corrected automatically if it has not been set correctly by the user, in older software versions an error is generated.
6. Enter the value for the positioning speed in **ps-v0**.

## 8.4 Execution Of Relative Positioning

**7** Enable and start. Set **ps-status** to 0.
**8** The relative distance is a 32-bit value that is split to **ps-umdrehung** and **ps-impulse**. Enter the value for the direction of the positioning procedure in Bit 7 of **ps-status**.
**9** To start the calculation, set Bit 6 in **ps-status.** Then wait until Bit 5 switches to 1.
**10** Set Bit 4 in **ps-status** to start the positioning procedure.
**11** During the time of the positioning, the bits indicating the single phases are displayed in **ps-status**.

## 8.5 End Of Relative Positioning

**12** If the positioning procedure is canceled, **ps-status** takes the value 0x00.
**13** Bit 0 of **ps-status** is set to 1, if the absolute-value deviation between the actual position and the target position is smaller than indicated in **Window**.

# 9. Absolute Positioning (optional)

## 9.1 Function

An absolute target position is written to NOVODRIVE, then the calculation is started. When the calculation is completed, the positioning procedure can be executed. If the target position has been reached within a given tolerance, this will be displayed in a status register.

## 9.2 Register

| | |
|---|---|
| **Register** | **?512us** |
| **Address** | 0xFEAE |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Value range** | Name Address Notes<br>**@Dummy** 0x01EA Absolute positioning deactivated<br>**@PSabsolut** 0x01E6 Absolute positioning activated |

| | |
|---|---|
| **Register** | **ps-v0** |
| **Address** | 0xFEE6 |
| **Size** | 16-bit signed |
| **Access** | R/W |
| **Function** | Positioning speed |
| **Scaling** | See manual Basic Functions, Chapter 3.6.3.3 |

| | |
|---|---|
| **Register** | **psa_positionL** |
| **Address** | 0xFE50 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Lower 16 bits of target position |
| **Scaling** | 0,00549 ° / increment |

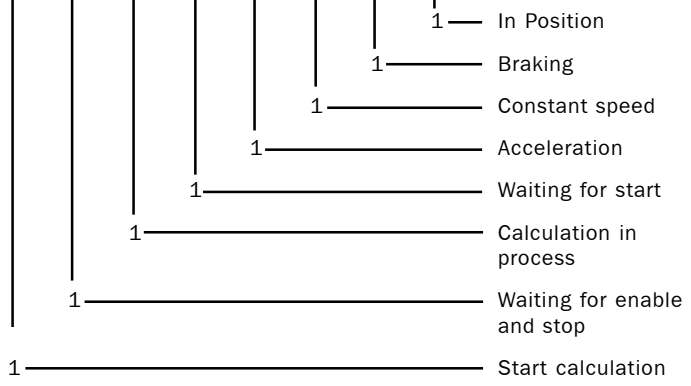
| | |
|---|---|
| **Register** | **psa_positionH** |
| **Address** | 0xFE4E |
| **Size** | 16-bit signed |
| **Access** | R/W |
| **Function** | Upper 16 bits of target position |
| **Scaling** | 1 revolution / increment |

| Register | Window |
|---|---|
| Address | 0xFED8 |
| Size | 16-bit unsigned |
| Access | R/W |
| Function | ± Tolerance for „In Position“ message |
| Scaling | 0,00549 ° / increment |

| Register | InternSoll |
|---|---|
| Address | 0xFE4C |
| Size | 16-bit signed |
| Access | R |
| Function | The speed setpoint specified by the positioning control |
| Scaling | See manual Basic Functions, Chapter 3.6.3.3 |

| psa-status | R/W | R | R | R | R | R | R | R |
|---|---|---|---|---|---|---|---|---|
| 0xFE52 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Bit



## 9.3 Initialization Of Absolute Positioning

**1** NOVODRIVE must be in the stop state (see manual Basic Functions, Chapter 3.4.2).
**2** Activate absolute positioning by setting **?512us** to **@PSabsolut**.
**3** The positioning control can work correctly only if the position controller is active. For this to be the case, enter the address of **nsoll2** in **?nsoll**.
**4** Define setpoint by means of **InternSoll** (**?Sollwert** = **InternSoll**).
**5** Make sure the ramp parameters are configured correctly. Enter the value for the acceleration/braking ramp in **Rampe+**. For the ramp only values are allowed for which the lower 6 bits are 0. The ramp pointers **?Rampe+** and **?Rampe-** must indicate the address of **Rampe+**. Whereas in the case of newer software versions this configuration is corrected automatically if it has not been set correctly by the user, in older software versions an error is generated.
**6** Enter the value for the positioning speed in **ps-v0**.

## 9.4 Execution Of Absolute Positioning

**7** The target position is a signed 32-bit value that is entered in **psa_positionL** and **psa_positionH**.
**8** Enter 0x80 in **psa_status**.
**9** NOVODRIVE checks enable and stop.
**10** Then the offline calculation is started. This is indicated in Bit 5 of **psa_status**.
**11** When Bit 4 of **psa_status** is set, the calculation is completed. NOVODRIVE now waits for start.
**12** After the start the motor moves to the target position. The bits indicating the single phases are displayed in **psa_status**.

## 9.5 End Of Absolute Positioning

**13** If the positioning procedure is canceled, **psa_status** takes the values 0xFE or 0xFF.
**14** After the last setpoint has been calculated, **psa_status** takes the value 0x00.
**15** Since this procedure is always accompanied by a tracking error, the motor has not yet reached the target position. Only if the absolute-value deviation between the actual position and the target position is smaller than indicated in **Window**, Bit 0 of **psa_status** is set to 1.

NOVOTRON

# 10. Online Positioning (optional)

Starting with H8 Version V3.03

## 10.1 Function

If the online positioning control is activated, NOVODRIVE moves to the given target position without delay. During the time NOVODRIVE moves to the target position, the values for speed and the ramps can be modified arbitrarily. Unlike in the case of relative and absolute positioning, the ramps can be chosen asymmetrically.

## 10.2 Register

| | |
|---|---|
| **Register** | **?512us** |
| **Address** | 0xFEAE |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Value range** | Name / Address / Note<br>**@Dummy** / 0x01EA / Online positioning deactivated<br>**@PSonline** / 0x01F0 / Online positioning activated |

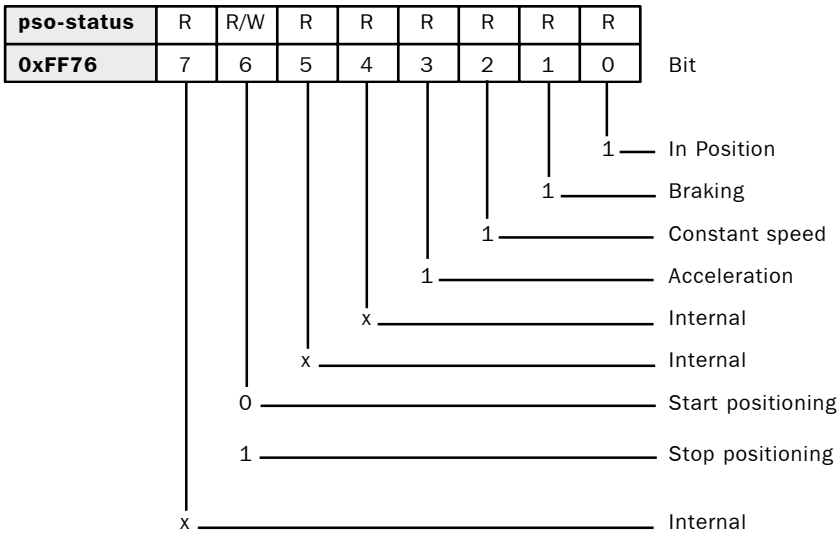
| | |
|---|---|
| **Register** | **?Anout2** |
| **Address** | 0xFF12 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Pointer to positioning speed |
| **Value range** | Name / Address<br>**ps-v0** / 0xFEE6<br>**CANinput1** / 0xFE3C<br>**CANinput2** / 0xFE3E<br>**CANinput3** / 0xFE40<br>**CANinput4** / 0xFE42 |

| Register | **?FiLage** |
|---|---|
| **Address** | 0xFEB0 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Pointer to the lower 16 bits of target position |
| **Value range** | Name Address<br>**psa_positionL** 0xFE50<br>**CANinput1** 0xFE3C<br>**CANinput2** 0xFE3E<br>**CANinput3** 0xFE40<br>**CANinput4** 0xFE42 |

| Register | **?FiUmdr** |
|---|---|
| **Address** | 0xFECC |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Pointer to the upper 16 bits of target position |
| **Value range** | Name Address<br>**psa_positionH** 0xFE4E<br>**CANinput1** 0xFE3C<br>**CANinput2** 0xFE3E<br>**CANinput3** 0xFE40<br>**CANinput4** 0xFE42 |

| Register | **Window** |
|---|---|
| **Address** | 0xFED8 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | ± Tolerance for „In Position“ message |
| **Scaling** | 0,00549 ° / increment |

| Register | **InternSoll** |
|---|---|
| **Address** | 0xFE4C |
| **Size** | 16-bit signed |
| **Access** | R |
| **Function** | The speed setpoint specified by the positioning control |
| **Scaling** | See manual Basic Functions, Chapter 3.6.3.3 |




## 10.3 Initialization Of Online Positioning

1. NOVODRIVE must be in the stop state (see manual Basic Functions, Chapter 6.2).
2. Activate online positioning by setting **?512us** to **@PSonline**.
3. The positioning control can work correctly only if the position controller is active. For this to be the case, enter the address of **nsoll2** in **?nsoll**.
4. Define setpoint by means of **InternSoll** (**?Sollwert** = **InternSoll**).
5. There are no restrictions with regard to the ramp values. The ramp values can be modified also during the motor's movement. The acceleration ramp is selected by means of **?Rampe+** and the braking ramp is selected by means of **?Rampe-**. For configuration of the ramp values see manual Basic Functions, Chapter 3.6.3.4.
6. The positioning speed is selected by means of **?AnOut2**. The speed may be changed during the motor's movement.
7. The target position is set by means of **?FILage** and **?FIUmdr**. The target position can be changed at any time.
8. Set Bit 6 of **pso_status** to 0.

***Example 1***

The pointers are set as follows:
**?512us = @PSonline**
**?nsoll = nsoll2**
**?Sollwert = InternSoll**
**?Rampe+ = Rampe+**
**?Rampe- = Rampe+**
**?AnOut2 = ps_v0**
**?FILage = psa_positionL**
**?FIUmdr = psa_positionH**

In this example, the online positioning procedure is similar to the absolute positioning procedure. The values for the ramps are entered in **Rampe+**, the value for speed is entered in **ps-v0**, and the values for the target position are entered in **psa_positionL** and **psa_positionH**.

Either the new target position must be entered as a 32-bit value in **psa_positionH** or NOVODRIVE must be stopped during the change of the target position. Otherwise an undesired target position may occur temporarily during the time between the transmission of the lower 16 bits and the transmission of the upper 16 bits.

***Example 2***

The pointers are set as follows:
**?512us = @PSonline**
**?nsoll = nsoll2**
**?Sollwert = InternSoll**
**?Rampe+ = Rampe+**
**?Rampe- = Rampe-**
**?AnOut2 = CANinput1**
**?FILage = CANinput2**
**?FIUmdr = CANinput3**

The ramp values for acceleration and braking can be set independently of one another in **Rampe+** and **Rampe-**. Both the positioning speed and the target position are updated cyclically by means of process-data write-telegrams over the CAN Bus.

## 10.4 Execution Of Positioning

**9** Stop NOVODRIVE.
**10** The target position is written to the selected registers (e.g. **psa_positionL** and **psa_positionH**).
**11** Start NOVODRIVE.
**12** NOVODRIVE moves to target position.

## 10.5 End Of Positioning

**13** Since this procedure is always accompanied by a tracking error, Bit 0 of **pso_status** is not set until the absolute-value deviation between the actual position and the target position is smaller than indicated in **Window**.

# 11 Error Sources / Troubleshooting With Positioning Control

- The NM31-40 positioning control, H8 Version of 04/04/1996 can only perform relative positioning.
- If NOVODRIVE is disabled, no positioning calculation is executed.
- If you work with a H8 Version older than V3.00 and if the positioning procedure is likely to take longer than 26 seconds, Error 600 is generated. To prevent this, you may shorten the distance or increase the positioning speed.
- If the homing procedure sometimes requires one additional revolution of the motor shaft, the home switch must be newly adjusted.
- If a positioning procedure is started although no positioning control is in place, Error 620 is generated.
- Basically, only **one** program for setpoint calculation may be active. If at the same time the table interpolation or the fine interpolation is activated over pointer ***µ*512usA and** the positioning control or the homing procedure is activated over pointer **?512us,** malfunctions are likely to occur.

☞ If you use a H8 Version V3.03 to V3.06, the positioning control and the table interpolation must not be active at the same time.

☞ Starting with H8 Version V3.03, the fine interpolation and the positioning control must not be active at the same time.

- If the setpoint is constantly set to -1 in **InternSoll** although the positioning procedure is in process, the parallel interface may be active. The parallel interface is deactivated by resetting Bit 6 of **SwVersion**.
- Malfunction may occur because the override function of the table interpolation is still active. The overrride factor is computed also when the actual table interpolation has not been activated over pointer ***µ*512usA**. In this case, the value for the positioning speed in **ps-v0** is overwritten. To switch off the override function, set Bit 2 and Bit 3 in **NB_Init** to 1.
- If the online positioning control is used, **pso-status** must be set to 0, otherwise the function will not be executed.

# 12 Table Interpolation (optional)

## 12.1 Function

### 12.1.1 Table interpolation without override (starting with H8 Version 2.03)

Every 1,024 ms the value of a relative distance is read out from a table and the setpoint for speed is calculated so that the new position setpoint has been reached by the time the next value is read out.

### 12.1.2 Trace function (starting with H8 Version 2.03)

It is possible to record a 16-bit value during the time the table is being worked through.

### 12.1.3 Time override (starting with H8 Version 3.00)

Normally, every 1,024 ms the value of a new distance is read out from the table.

By stretching the time segments by the factor 1,0 ... 256,0, the time for a motion cycle can be adjusted variably.

The speed is reduced to the same extent, i.e. the same distance is being covered.

The motor moves also if the largest stretching factor has been chosen. A standstill can be achieved only by means of a stop of the drive.

### 12.1.4 Speed override (starting with H8 Version 3.00)

The speed can be reduced by factor 1,0... 0,00392. The time is stretched so that the same distance is being covered.

## 12.2 Register

| | |
|---|---|
| **Register** | **?512usA** |
| **Address** | 0xFEC8 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Activation of table interpolation |
| **Value range** | Name / Address / Notes<br>**@Dummy** 0x01EA Table interpolation deactivated<br>**@Tabelle** 0x01E0 Table interpolation activated |

| Register | ?Tabelle |
|---|---|
| Address | 0xFE3A |
| Size | 16-bit unsigned |
| Access | R/W |
| Function | Pointer to next value in table; must be set before table interpolation is started |
| Value range | 0x4000...0xFFFF (external memory) |

| Register | FISoll |
|---|---|
| Address | 0xFE1A |
| Size | 16-bit signed |
| Access | R |
| Function | The setpoint for speed specified for the controller by the table interpolation (override active) |
| Scaling | See manual Basic Functions, Chapter 3.6.3.3 |

| Register | InternSoll |
|---|---|
| Address | 0xFE4C |
| Size | 16-bit signed |
| Access | R |
| Function | The setpoint for speed specified for the controller by the table interpolation (override not active) |
| Scaling | See manual Basic Functions, Chapter 3.6.3.3 |

| Register | ps_impuls |
|---|---|
| Address | 0xFE2C |
| Size | 16-bit unsigned |
| Access | R/W |
| Function | Trace offset; value for trace saved in **?Tabelle + ps_impuls** |
| Value range | 0x0000...0xFFFF (external memory) |

| Register | ps_umdrehung |
|---|---|
| Address | 0xFE2A |
| Size | 16-bit signed |
| Access | R/W |
| Function | Pointer to trace value |
| Value range | 0xFD80...0xFF7F (integrated memory) |

| Register | ps_k0 |
|---|---|
| Address | 0xFE30 |
| Size | 16-bit unsigned |
| Access | R/W |
| Function | Pointer to next table segment. |
| Value range | 0x4000...0xFFFF (external memory) |

| NB_Init | R/W | R | R | R | R | R | R | R | |
|---|---|---|---|---|---|---|---|---|---|
| 0xFF04 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit |
| | x | x | x | x | x | x | 0 | x | Speed override |
| | x | x | x | x | x | x | 1 | x | Time override |
| | x | x | x | x | 2 = 3 | | x | x | No override |
| | x | x | x | x | 2 ≠ 3 | | x | x | With override |

## 12.3 Initialization Of Table Interpolation

1. NOVODRIVE must be in the stop state (see manual Basic Functions, Chapter 3.4.2).
2. Switch on position controller by setting **?nsoll** to the address of **nsoll2**.
3. If override is not active, define setpoint by means of **InternSoll** (**?Sollwert** = **InternSoll**). If override is active, define setpoint by means of **FISoll** (**?Sollwert** = **FISoll**).
4. Activate function by setting **?512usA** to **@Tabelle**.
5. Set **?Tabelle** to the address at the beginning of the table.
6. Set parameters for trace and override.

☞ It is recommended that the acceleration ramp and the braking ramp be switched off (see manual Basic Functions, Chapter 3.6.3.4). Otherwise the target position gets falsified if the setpoints exceed the ramp values.

## 12.4 Execution Of Table Interpolation

**7** Start NOVODRIVE to start the table interpolation.
**8** The table is being worked through and the trace value is recorded.
**9** The function is canceled immediately, if enable or start are deactivated, if any of the two limit switches responds, or if an error is generated.

## 12.5 End Of Table Interpolation

**10** When the table's end is reached or when the procedure is canceled, the setpoint and register **?Tabelle** is set to 0.

## 12.6 Operating Mode

The operating mode is specified by means of Bit 1, 2 and 3 in **NB_Init**.

## 12.7 Table

The table must be stored in the external memory within the address range 0x4000...0xFFFF. It is possible to make up to 49152 entries. In connection with various complementary programs the available address range is reduced to 0x8000...0xFFFF.

**Various software extensions use parts of the table memory up to address 0x8000. To avoid that such complementary programs are being overwritten, do not use the address range 0x4000...0x8000 for tables or the trace function when software extensions are involved.**

The table can be written to the memory by means of **NB_WriteWordX** and read out from the memory by means of **NB_ReadWordX**.
Entries to the table consist of signed 16-bit values representing the relative distance increments to be covered within the next time period (without override 1,024 ms).
The distance depends on the speed range. Within the speed range of 0...6000 rpm, the increments are identical. Within the speed range of 0...18000 rpm, one increment in the table moves the motor by four increments.

**Special values for table entries:**

| | |
|---|---|
| 0x8000 | End of table |
| 0x8001 | The following value is the address with which the table is to be continued (for interrelating tables or creating ring tables). |
| 0x8002 | Starting with H8 V3.0:<br>Register **ps_k0** contains the address of the next table segment. |

## 12.8 Trace Function

Every time a table value has been read out, a 16-bit value, whose address is specified by means of the trace pointer **ps_umdrehung**, is read out and saved in **?Tabelle + ps_impuls**.

The trace values are stored in the same memory space that is used for storing the table.

**If the trace function is used incorrectly, the table may be overwritten!**

**Deactivation**: No trace value is saved, if any of the following conditions applies:

- The trace pointer in **ps_umdrehung** is 0.
- The storage address **?Tabelle + ps_impuls** lies beyond the limits of the address range 0x4000...0xFFFF.

## 12.9 Override Stretching Factor

The stretching factor is specified by means of **?FISoll**. To do so, enter the address of a 16-bit register over which override is to be executed in **?FISoll**:

| | |
|---|---|
| **Register** | **?FILage** |
| **Address** | 0xFEB0 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Pointer to override stretching factor |
| **Value range** | Name / Address / Notes<br>**Sollwert** 0xFE60 Override over service channel<br>**AnInput1** 0xFE5C Override over Analog Input 1<br>**CANinput1** 0xFE3C Override over CAN Bus process data<br>**CANinput2** 0xFE3D<br>**CANinput3** 0xFE40<br>**CANinput4** 0xFE42 |

For the selected value applies:

| | |
|---|---|
| **Register** | **Sollwert, AnInput1, CANinput1, CANinput2, CANinput3, CANinput4** |
| **Address** | See above |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Override stretching factor |
| **Value range** | Speed override<br>0x0100 0,390625 %<br>0xFFFF 100,0 %<br>Cycle time override<br>0x0100 1,024 ms<br>0xFFFF 262,14 ms<br>Values can never fall below 0x0100.<br>A standstill can be achieved only by switching to the stop state. |

The stretching factor is read out and evaluated frequently, but not at regular intervals.

## 12.10 Error Sources / Troubleshooting

- The table interpolation with override function **and** the positioning control or the fine interpolation must not be active at the same time. There are some registers that are used by both functions.

- In contrast to table interpolation without override function, table interpolation including override function works only if the motor runs, since the value of the actual position is required for the calculation.

- If the override function is active, a setpoint is computed both in **InternSoll** and in **FISoll**. However, the value computed by **InternSoll** is „distorted“ and must not be used (if **?Sollwert** is not set to **FISoll** but to **InternSoll**, the distance to be covered is stretched).

- Bits 1 ... 3 in **NB_Init** must be set to 1, if the positioning control or the fine interpolation is used.

# 13 Cam Disk Function (Standard)

Starting with H8 Version V3.00

## 13.1 Function

An input value can be transformed to an output value over a cyclic table. The table must be stored in the external memory within the address range 0x8000...0xFFFF. The address of an output value is computed as follows:

(input value (16 Bit) + offset) / 2 + 0x8000

## 13.2 Register

| | |
|---|---|
| **Register** | **?512usA** |
| **Address** | 0xFEC8 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Activation of cam disk function |
| **Value range** | Name Address Notes<br>**@Dummy** 0x01EA Cam disk function deactivated<br>**@Kurve** 0x01FA Cam disk function activated |

| | |
|---|---|
| **Register** | **?AnOut2** |
| **Address** | 0xFF12 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Pointer to cam disk input value |
| **Value range** | Name Address Notes<br>**_ROD** 0xC402 Encoder input<br>**STROD** 0xFE48 Encoder input<br>**AnInput1** 0xFE5C Analog Input 1<br>**AnIn_msoll** 0xFF66 Analog Input 2<br>**CANinput1** 0xFE3C Setpoints from CAN Bus<br>**CANinput2** 0xFE3E<br>**CANinput3** 0xFE40<br>**CANinput4** 0xFE42 |

| Register | ?CANout |
|---|---|
| Address | 0xFEAC |
| Size | 16-bit unsigned |
| Access | R/W |
| Function | Offset of cam disk input value;<br>Attention: this is no pointer ! |
| Scaling | 0,00549 ° / increment at 16 bit / revolution |

| Register | KurveFakt |
|---|---|
| Address | 0xFEC0 |
| Size | 16-bit unsigned |
| Access | R/W |
| Function | Scaling of cam disk output value |
| Scaling | 0x0000 0 %<br>0x0100 100 %<br>0xFFFF 255 % |

| Register | KurveOut |
|---|---|
| Address | 0xFDF6 |
| Size | 16-bit unsigned |
| Access | R/W |
| Function | Scaled cam disk output value |
| Scaling | 0,00549 ° / increment |

| MotConfig | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
|---|---|---|---|---|---|---|---|---|---|
| 0xFF21 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit |
| | 0 | x | x | x | x | x | x | x | 12-bit input value |
| | 1 | x | x | x | x | x | x | x | 16-bit input value |

## 13.3 Initialization Of Cam Disk Function

1. NOVODRIVE must be in the stop state (see manual Basic Functions, Chapter 6.2).
2. Create a table and load it to the external memory.
3. For activation enter the address of the function **@Kurve** in **?512usA**.
4. Specify the input value (12-bit or 16-bit) by means of **?AnOut2**.
5. Specify the value range of the input value by means of Bit 7 of **Motconfig**.
6. Set the input offset value or 0 in **?CANOut**. Enter the offset value directly in **?CANOut**.

☞ **?CANOut** is not used as a pointer here!

7. The scaling of the output value is done by means of **KurveFakt**.

In order to use the output value as a position setpoint, execute the following steps:

8. Switch on the position controller by setting **?nsoll** to the address of **nsoll2**.
9. Switch the mode for position setpoint determination by setting **?LageSollExt** to the address of **KurveOut**.
10. Switch the mode for speed setpoint determination by setting **?Sollwert** to the address of **Lage-nsoll**.

☞ It is recommended that the acceleration ramp and the braking ramp be switched off (see manual Basic Functions, Chapter 3.6.3.4). Otherwise the target position gets falsified if the setpoints exceed the ramp values.

11. Enable and start NOVODRIVE. After that the cam disk is active.

## 13.4 Table

The table uses the address range 0x8000...0xFFFF in the external memory. It contains 32768 entries for the output value. The input values have a resolution of 0,010986°. 12-bit input values are internally rescaled to 16-bit values, so in this case only every 16th table value is addressed. The output value of 0x0000...0xFFFF (= 0...360°) can be increased by means of the scaling factor, yet its resolution gets poorer accordingly.

The table can be written to the memory over NOVOBUS or CAN Bus by means of **NB_WriteWordX** and read out from the memory by means of **NB_ReadWordX**.

## 13.5 Example

As setpoint source the encoder input is used (**?Anout2** = **_ROD**). If the number of pulses has been set correctly, one revolution of the encoder shaft generates a movement of the motor by 0...1 revolutions, depending on the table values. By means of a scaling factor unequal to 0x0100, the range can be increased or decreased.

## 13.6 Error Sources / Troubleshooting

- The scaling factor cannot be set arbitrarily high. The maximum is exceeded when the speed setpoint for NOVODRIVE reaches 6000 rpm.
- If higher scaling factors are used, the setpoint resolution gets increasingly coarser.
- If the scaling value or offset value is changed while the controller is enabled and started, position jumps occur which may generate Error 977.

# 14 Controller Synchronization

It is possible to synchronize the controller timing of several NOVODRIVEs. To do so, one NOVODRIVE is configured as a master and all others as slaves. The master sends one synchronization signal during every controller cycle by which the slaves adjust their controller timing.

The synchronization provides for a clock-synchronous behavior of several drives, e.g. when the table interpolation is used. The hardware start signal is then evaluated synchronously. Even after several minutes of working through the tables the process remains synchronous.

## 14.1 Register

All modifications are immediately in effect.

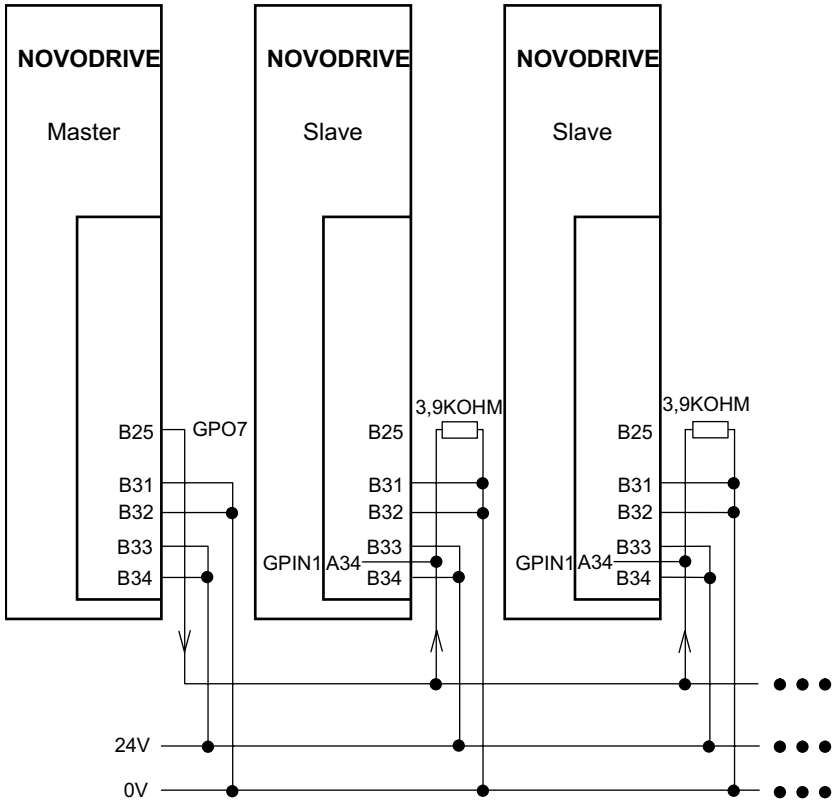
| **SwVersion** | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
|---|---|---|---|---|---|---|---|---|---|
| **0xFF3D** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit |
| | - | - | 0 | - | - | - | - | - | Synchronization off |
| | - | - | 1 | - | - | - | - | - | Synchronization on |

| **Steuerbits** | R/W | ... | R/W | R/W | R/W | R/W | R/W | R/W | |
|---|---|---|---|---|---|---|---|---|---|
| **0xFEA0** | 15 | ... | 5 | 4 | 3 | 2 | 1 | 0 | Bit |
| | - | ... | - | - | 0 | - | - | - | Master mode |
| | - | ... | - | - | 1 | - | - | - | Slave mode |

| **Register** | **Sync** |
|---|---|
| **Address** | 0xFE66 |
| **Size** | 16-bit unsigned |
| **Access** | R |
| **Function** | Measuring the synchronicity; a stable value of 0x0960±1 indicates that the slave runs synchronously. |

**After the synchronization is deactivated on the slave side, an immediate reset is necessary to initialize the hardware again. Otherwise NOVODRIVE does not respond any longer after some time.**

## 14.2 Wiring




**Make sure the wiring is noise-proof! Use shielded cables only!**

**The controller clock synchronization cannot be used until H8 Version 3.11 in connection with the table interpolation and the override function.**

**Since the controller clock synchronization interferes with the resolver converter, using this function is recommended only in connection with a sine encoder.**

# 15 Ablaufsteuerung (Software Extension)

## 15.1 Function

The Ablaufsteuerung is a software extension for sequencing control that is loaded to the external memory. This requires a 128 k memory and H8 Version V2.02 or newer. If you want to execute positioning procedures as well, NOVODRIVE must be equipped with the NOVOMERIK positioning control.

The Ablaufsteuerung allows you to use various additional functions of NOVODRIVE without needing a bus connection. This is made possible by programming up to 256 commands which can be activated over a combination of digital inputs.
By this it is possible among other things to move to various preprogrammed positions or to switch the setpoint source. For linear motors without commutation signals, an Autokomm function is provided by which the commutation position can be determined after power-on. Besides, homing procedures can be performed as well as zero reference point search for incremental measuring systems.

The functions of the Ablaufsteuerung are:

- homing,
- select command,
- teach-in,
- positioning,
- jogging right,
- jogging left,
- clear error,
- enable,
- start,
- profile,
- curve,
- set actual value,
- zero reference point search,
- auto adjustment of commutation position.

Every function comprises parameters such as functional mode, speed, position, resolution, controller parameters, ramps. These functional parameters are entered separately for every command, except for the gear factor parameter, which applies for all commands together.

☞ The gear factor parameter is used only during programming.

In the start-up software ND30Cfg, you can find the gear factor parameter listed under motor configuration. It is used there also to indicate the actual position.

There are two ways to select and execute the functions:

- selection over digital inputs,
- selection in the command.

If you choose the second option, every command indicates the number of the next command to be executed. A command is activated automatically after the preceding command has been concluded or by means of the hardware starting signal.

## 15.2 Installation

To install the Ablaufsteuerung, you need the DOS program ND31ABL.EXE and the file SPSPOS.HEX. Make sure SPSPOS.HEX is stored in the same directory which ND31ABL.EXE is started from.

Under Windows 2000 or Windows XP the handling of the Ablaufsteuerung is integrated in the start-up software ND30Cfg. The Ablaufsteuerung can easily be installed and switched on/off.

The software is switched on and off by means of **Bank** and **?SPS**.

| **Register** | **Bank** | |
|---|---|---|
| **Address** | 0xFF07 | |
| **Size** | 8-bit unsigned | |
| **Access** | R/W | |
| **Function** | Memory bank selection | |
| **Value range** | 0x00<br>0x40 | Standard configuration<br>Memory bank for Ablaufsteuerung |

| **Register** | **?SPS** | | |
|---|---|---|---|
| **Address** | 0xFEA2 | | |
| **Size** | 16-bit unsigned | | |
| **Access** | R/W | | |
| **Function** | Pointer for activation | | |
| **Value range** | Name<br>**@Dummy**<br>**@XRAM D000** | Address<br>0x01EA<br>0xD000 | Notes<br>Ablaufsteuerung deactivated<br>Ablaufsteuerung activated |

## 15.3 The DOS program ND31ABL.EXE

Used for

- installation of Ablaufsteuerung on NOVODRIVE,
- saving and loading commands,
- uploading and downloading commands from/to NOVODRIVE,
- editing and creating commands,
- editing and creating assignment lists for digital inputs and commands.

ND31ABL.EXE can be used also without NOVODRIVE.

### 15.3.1 Installation

**1** Connect NOVODRIVE to a COM interface of the PC.
**2** Switch on 24 V supply voltage of NOVODRIVE.
**3** Start ND31ABL.EXE.
**4** Tell the program which interface NOVODRIVE is connected to.
**5** Choose „Install“ by means of arrow keys. Press Enter.
**6** The Ablaufsteuerung is now loaded and installed on NOVODRIVE.

### 15.3.2 Operation

After the interface has been selected, the main menu opens up and the command bar is active. By means of the F10 key you may switch between the editing menus and the command bar. Then select the desired function by means of the arrow keys (right/left) and activate it with Enter.

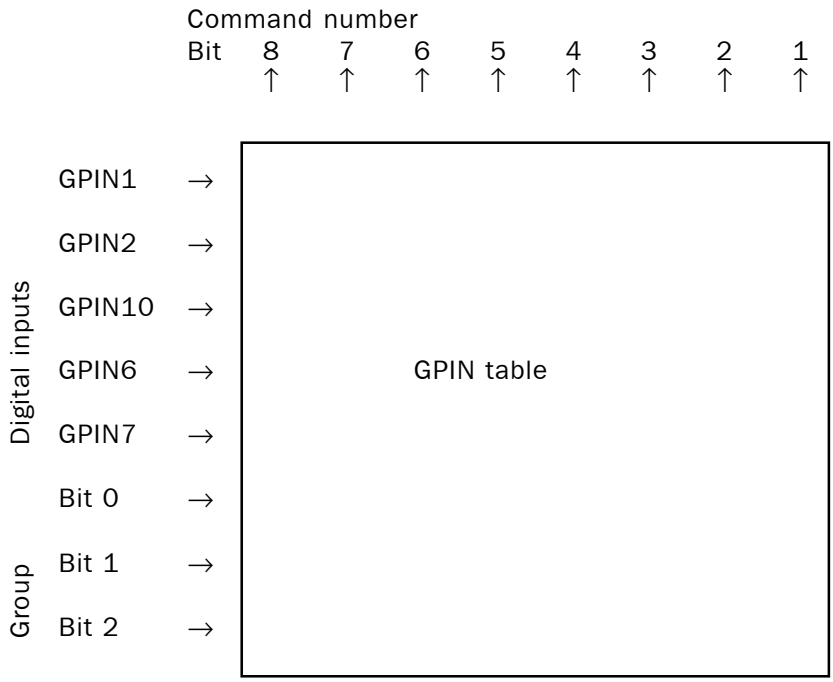
- **Upload**:
  Loads the commands and the assignment list from NOVODRIVE to the PC.
- **Download**:
  Loads the commands and the assignment list from the PC to NOVODRIVE.
- **Load**:
  Loads the commands and the assignment list from a floppy disk to the PC.
- **Save**:
  Saves the commands and the assignment list on floppy disk.
- **Satzedit**:
  Editing of a command. After you have entered the number of the command, the complete command appears on the screen. Select the parameter to be modified by means of the arrow keys. Activate the parameter selected by means of the Enter key. By means of the arrow keys you may change options. You can enter new values by means of the Enter key and the numeric keys.
- **EditGPin**:
  Editing of the list indicating the assignment of digital inputs (GPIN) and commands. Select individual assignments with arrow keys (up/down). To enter the number of the command, press Enter or right arrow key.

## 15.4 Execution

### 15.4.1 Selection of commands

A command is selected either over digital inputs (GPIN) or over the parameter „Continue“. If „Continue“ takes a value unequal to 0, the command with the number indicated in „Continue“ is activated next.

If „Continue“ is set to 0, the command is selected over the digital inputs. The selection is made indirectly over a table. Thereby always one command can be assigned arbitrarily to each GPIN bit pattern. It is also possible to assign the same command to several bit patterns. Since only five GPINs are available for selecting commands, only 32 different commands can be selected. However, all the 256 commands can be selected by parameter „Group“.




After NOVODRIVE has been reset, the first command to be executed is selected from the group 0 and the GPINs. The execution of the command starts with the low-high edge of the hardware starting signal (GPIN3).

If an error occurs while a command is edited or if the hardware starting signal is taken away, the Ablaufsteuerung switches to the same state as it does after reset.

After a command has been executed, you may start the next command with or without the hardware starting signal. If the hardware starting signal is used, the next command is started by means of a low-high edge. At this point, also the bit pattern from the GPINs is read in.

If the hardware starting signal is not used, the editing of the next command starts immediately. Thereby sequences of commands can be defined which run independently of an external control.

### 15.4.2 Feedback over digital outputs

☞ See also manual Basic Device, Chapters 5.3.3 and 5.3.5

**Outputs with fixed assignment:**

| Function | Connection | Characteristics |
|---|---|---|
| In Position | GPO1 | Open Collector |
| Ready-to-op. | Ready-to-op. contact | potential-free contact, normally open |
| Cancellation | GPO10 | Open Emitter |
| Brake | GPO3 | Open Emitter |

**Freely programmable outputs:**

When a command has been completed, GPOs can be set or reset to inform the control that the command has been executed. The bit pattern in parameter GPOB is transfered to the outputs after the command has been executed. In case of functions that consist of two phases, the completion of the first phase is indicated by the output of the bit pattern in parameter GPOA (in case of positioning commands after completion of the calculation).

| GPOA / GPOB | Connection | Characteristics |
|---|---|---|
| Bit 0 | GPO2 | Open Collector |
| Bit 1 | GPO4 | Open Emitter |
| Bit 2 | GPO5 | Open Emitter |
| Bit 3 | GPO6 | Open Emitter |
| Bit 4 | GPO7 | Open Emitter |
| Bit 5 | GPO8 | Open Emitter |
| Bit 6 | GPO9 | Open Emitter |

### 15.4.3 Functional parameters

Every command contains a parameter by which the function can be selected. Apart from positioning procedures, other functions can be executed as well. Some functions can be additionally specified by means of the mode parameter.

For each function, the control parameters **LKd, LKp, nKd, nKp, nKi** and **mmax** can be specified pairwisely:

- **LKd, LKp**
- **nKd, nKp**
- **nKi, mmax**

If both values of a pair are set to 0, the control parameters are not modified. Each modification is permanent, i.e. the control parameters are not reset after the command has been executed.

The other parameters of the command have various meanings depending on the specific function.

## 15.5 Description Of Functions

### 15.5.1 Homing

**Code of function** 0

**Modes**

- search for home switch only
- search for home switch and move to resolver zero point
- move to resolver zero point only
- move to resolver zero point using the shortest way
- move to resolver zero point always in positive direction
- move to resolver zero point always in negative direction

**Description** See also Chapter 5.
Search for home switch. The search direction is defined by the sign of the searching speed. The creeping speed is 1/8 of the searching speed.

As soon as the homing procedure has been concluded, the bit pattern in parameter GPOB is transfered to the outputs.

### 15.5.2 Positioning

**Code of function** 1

**Modes**
- absolute positioning
- relative positioning
- auto start
- start with a low-high edge at GPIN3

**Description** See also chapters 8 and 9.
Execution of a relative or an absolute positioning procedure with symmetrical, linear speed ramps. The positioning procedure consists of two phases:

**1** calculation of the trajectory curve and output of GPOA,
**2** execution of positioning and output of GPOB.

If „Autostart“ has been selected, the positioning procedure is automatically started after phase **1**. Otherwise a new low-high edge at the start input (GPIN3) is needed to trigger the positioning procedure.

### 15.5.3 Jogging

**Code of function** 2

**Modes**
- negative direction,
- positive direction

**Description** As long as the command is selected, the motor moves at the given speed.

**Note** By defining several commands that have different jogging speeds, you may select a fixed speed over GPINs.

**Extension** Starting with 02/18/1999:

The function is stopped, if the motor is disabled for at least 1 second (2 seconds at the most).

If you use a linear motor, you may combine this command with the zero point search. The zero pulse should be close to the end of the track, which is moved to by jogging.

### 15.5.4 Clear error

**Code of function** 3

**Modes**
- none

**Description** A NOVODRIVE error is cleared. If the cause for the error has not been eliminated, NOVODRIVE again takes the error state after the error has been registered again.

### 15.5.5 Profile

**Code of function** 4

**Modes**
- auto start
- start with a low-high edge at the start input (GPIN3)

**Description** In order to be able to follow a given trajectory curve, the respective trajectory curve must be loaded to the external memory over CAN Bus or NOVOBUS.

Enter the trajectory table's start address as a hex value.

For the table see Chapter 12 (table interpolation without trace and override function). The table contains relative distances that are covered every 1,024 ms. Therefore it must be calculated in advance what distance NOVODRIVE is able to cover in 1,024. If larger distances are set, an error is generated.

### 15.5.6 Teach-in

**Code of function** 5

**Modes**
- none

**Description** A current position is saved and entered in the command as a target position next time an absolute positioning procedure is activated.

### 15.5.7 Curve

**Code of function** 6

**Modes**
- none

**Description** not implemented

### 15.5.8 Analog-mode setpoint setting

**Code of function** 7

**Modes**
- none

**Description** Switch to analog-mode setpoint setting over Analog Input 1. The ramps are active. The scaling of the setpoints is defined separately for each command.

### 15.5.8 Step/direction setpoint setting

**Code of function** 8

**Modes**
- encoder
- step/direction

**Description** A setpoint for position is set over the encoder input or by step/direction setting. The function cannot be applied if an encoder or a sine encoder is used as feedback system.

The counter is reset every time the zero pulse occurs. After each revolution, the counter is rounded.

### 15.5.10 Torque setting over Analog Input 2

**Code of function** 9

**Modes**
- none

**Description** not implemented

### 15.5.11 Setting an actual value

**Code of function** 10

**Modes**
- none

**Description** Sets the value of the actual position to the value for the parameter Position.

### 15.5.12 Zero reference point search

**Code of function** 11

**Modes**
- none

**Description** Search for the zero reference point of an incremental measuring system (e.g. ERN 1387). The search direction is defined by the sign of the searching speed. At the zero reference point, the actual position is set to the value of the parameter Position.

### 15.5.13 Auto adjustment

**Code of function** 12

**Modes**
- none

**Description** Determination of the measuring system's commutation position.

**During this procedure the motor moves jerkily!**

### 15.5.14 Autokomm (starting with 02/18/1999)

**Code of function** 13

**Modes**
- none

**Description** Determination of the measuring system's commutation position. The Autokomm function is a refinement of the auto adjustment function.

**Benefits from using the Autokomm function (particularly in connection with linear motors):**

- The motor makes only a slight movement.
- The function can be canceled.
- Constant influential forces and friction are compensated.
- It is possible to use vertical axes (in connection with a brake that is controlled by NOVODRIVE's braking function).
- The Autokomm function can be used as an operating function.

In the parameter set, set Bit 2 of **Freigabe0** to 1 in order to internally disable NOVODRIVE after power-on. The disable state is deactivated only after Autokomm or Auto adjustment have been executed successfully. Thereby it is made sure that the motor is not moved while a wrong commutation position is set.

## 15.6 Error Messages

The Ablaufsteuerung can generate the following additional error messages:

| Error number | Error | Description |
|---|---|---|
| 879 | Autokomm | Autokomm function was not completed successfully. To proceed, reset NOVODRIVE. |
| 880 | AK_nichtbereit | Autokomm function cannot be started due to an error. |
| 881 | noAutokomm | A command has been started before complete execution of Autokomm function. |
| 882 | Pos_Sperre | A positioning procedure has been started while controller is disabled. |
| 883 | FR_Feedback | Step/direction setpoint setting only possible with resolver as feedback system. |

## 15.7 Error Sources / Troubleshooting

- Error 802:
  If a parameter set is loaded in which the Ablaufsteuerung has been activated (**?SPS** = 0xD000 and **Bank**=0x40) although no Ablaufsteuerung is installed, Error 802 is generated (pointer error). Due to the error, the Ablaufsteuerung gets deactivated automatically after the next reset. To avoid this error, first install the Ablaufsteuerung, then load the parameter set and the program.

☞ It is possible to order NOVODRIVE with preinstalled Ablaufsteuerung.

- Reinstallation of Ablaufsteuerung :
  If the Ablaufsteuerung is reinstalled, the program is deleted. It must then be reloaded.

- A linear motor moves uncontrolledly after start.
  See description for Autokomm function.

NOVOTRON

# 16 CAN Profile (Software Extension)

## 16.1 Function

CAN Profile is a software extension that is loaded to the external memory. This requires a 128 k memory.

This extension serves to simplify the operability of NOVODRIVE over the CAN Bus. Basically, however, it can be used over any bus system. CAN Profile allows you to use additional functions, such as the positioning control, without the need to deal with details, such as switching on the position controller, for example.

The following functions are implemented:

- Autokomm,
- zero reference point search,
- positioning,
- homing,
- fixed setpoint,
- table interpolation,
- set actual value.

The functions and the respective operating modes are selected by means of a register. The control of NOVODRIVE is done by means of a control register. For providing status information, a status register and an error register are available. All functions are controlled in the same way.

The selection of the functions and their parameterization is done over the commands of the service channel. The process data channel cannot be used for this purpose.

## 16.2 Register

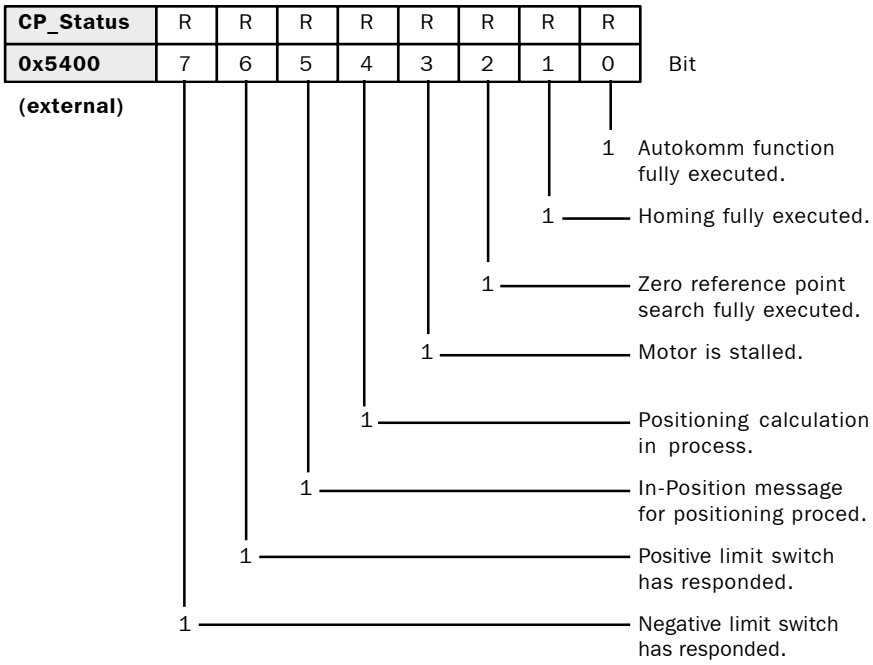
***Attention!***

**The control registers and status registers of CAN Profile are stored in the external memory. They can be read and written only by means of the commands ReadWordX and WriteWordX.**
**The parameters for the functions are stored in the integrated memory as variables. They can be read and written by means of the commands ReadByte, WriteByte, ReadWord and WriteWord.**

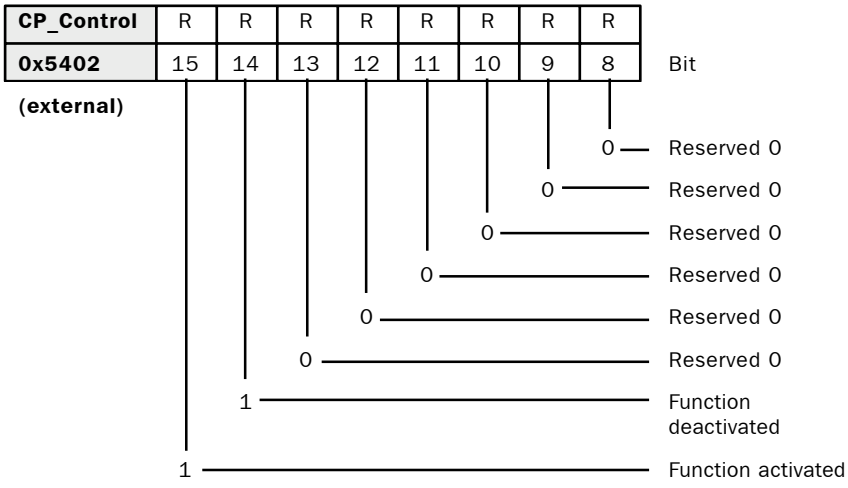
| CP_Status | R | R | R | R | R | R | R | R |
|---|---|---|---|---|---|---|---|---|
| 0x5400 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| (external) | | | | | | | | |

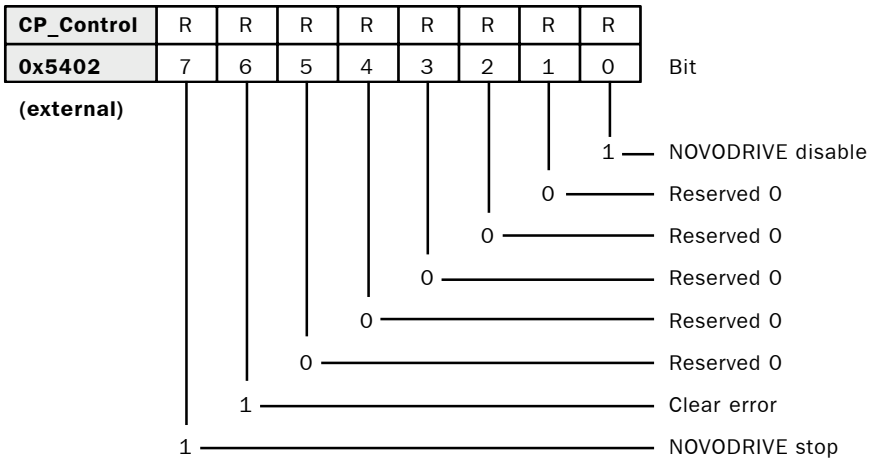
Bit

- 8: 1 — NOVODRIVE is stopped.
- 9: 1 — NOVODRIVE is disabled.
- 10: 1 — An error has occured.
- 11: 1 — Function was canceled by the user (e.g. by means of control word, stop or disable).
- 12: 1 — Function was canceled due to an error.
- 13: 1 — Function was fully executed.
- 14: 1 — Function is in process.
- 15: 1 — Function is initialized and can be started.




| Register | CP_Error |
|---|---|
| Address | 0x5401 (external) |
| Size | 16-bit unsigned |
| Access | R |
| Function | Error code |
| Value range | See manual Basic Functions, Chapter 3.5 |






| **Register** | **CP_Funktion** |
|---|---|
| **Address** | 0x5403 (external) LSB |
| **Size** | 8-bit unsigned |
| **Access** | R/W |
| **Function** | Select function |
| **Value range** | 0 Autokomm<br>1 Zero reference point search<br>2 Positioning<br>3 Homing<br>4 Fixed setpoint<br>5 Table interpolation<br>6 Set actual value |

| **Register** | **CP_Mode** |
|---|---|
| **Address** | 0x5403 (external) MSB |
| **Size** | 8-bit unsigned |
| **Access** | R/W |
| **Function** | Select mode |
| **Value range** | See description of functions |

## 16.3 Installation

Under DOS: Connect NOVODRIVE over the serial interface and activate the XCAN.EXE program. XCAN installs SPSPOS.HEX on NOVODRIVE and activates CAN Profile.

The Windows software ND30Cfg contains one page for software installation. Select the file SPSPOS.HEX and set 0x40 in **Bank**.

After the software has been loaded, set registers **Bank** and **?SPS** to activate the software.

| **Register** | **Bank** |
|---|---|
| **Address** | 0xFF07 |
| **Size** | 8-bit unsigned |
| **Access** | R/W |
| **Function** | Select memory bank |
| **Value range** | 0x00 Standard configuration<br>0xA0 Memory bank for CAN Profile |

| **Register** | **?SPS** |
|---|---|
| **Address** | 0xFEA2 |
| **Size** | 16-bit unsigned |
| **Access** | R/W |
| **Function** | Pointer for activation of function |
| **Value range** | Name Address Notes<br>**@Dummy** 0x01EA CAN Profile deactivated<br>**@XRAM D014** 0xD014 CAN Profile activated |

## 16.4 Usage

- The function and the operating mode are selected by entering the respective value in registers **CP_Funktion** and **CP_Mode**. By this, all necessary pre-settings have been specified.
- By **CP_Status** Bit 15 = 1, NOVODRIVE signals that the function has been initialized.
- As soon as the parameters (e.g. target position, ramp value, speed) required for the function are set, the function can be activated by setting **CP_Control** Bit 15 = 1. Except for the function for setting an actual value, the functions can only be executed if enable and start are activated.
- By setting **CP_Control** Bit 14 = 1, a function can be canceled at any time.
- By **CP_Status** Bit 14 = 1, NOVODRIVE signals that the function has been activated.
- If the function has been executed correctly, this is indicated in Bit 13 = 1. A cancellation of the function by the occurrence of an error or by the user is indicated in Bits 12 and 11. When the function is activated again, Bits 13, 12 and 11 in **CP_Status** are reset.
- Over the CAN Bus, stop and disable can be handled over Bits 7 and 0 in **CP_Control**. If an error occurs, this is indicated in Bit 12 of **CP_Status**.
- The error can be cleared by means of **CP_Control** Bit 6 = 1. After the error has been cleared, disable NOVODRIVE before putting it into operation again.
- If a limit switch has responded, this is indicated in Bits 7 or 6 of **CP_Status**. These bits are not reset when the limit switch is left. A reset is done by the clear error command.

## 16.5 Description Of Functions

### 16.5.1 Autokomm

**Number of function** 0

**CP_Mode**
- Bit 7...0 0 = reserved

**Description** Automatic determination of commutation position for linear motors.

**Note** In case of linear motors without commutation signals, set Bit 2 of **Freigabe0** to 1. By this you can prevent any enabling of hardware and software without the Autokomm function being executed beforehand.

**Other parameters:**

| Name | Address | Size [Bit] | See manual ..., Chapter | Description |
|---|---|---|---|---|
| **Freigabe0** | 0xFF3C | 8 | Basic Functions 3.6.2.2 | Start state of NOVODRIVE |

## 16.5.2 Zero reference point search

**Number of function** 1

**CP_Mode**

- Bit 1...0 0 = reserved
- Bit 2 1 = monitoring of distance active
- Bit 3...6 0 = reserved
- Bit 7 0 = positive direction
  1 = negative direction

**Description** Search for the zero reference point of an incremental measuring system. The search direction is defined over **CP_Mode**. If a limit switch responds before the zero pulse has been found, an inversion of the search direction is made.

If monitoring of distance is requested, this can be activated by means of Bit 2 in **CP_Mode**. The maximum distance is specified in **ps_impulse** and **ps_umdrehung**. If the maximum distance is exceeded, Error 884 is generated. At the end of the zero reference point search, an actual value is set.

**Other parameters:**

| Name | Address | Size [Bit] | See manual ..., Chapter | Description |
|---|---|---|---|---|
| **ps_umdrehung** | 0xFE2A | 16 | Additional Functions 8 | Upper 16 bits of maximum distance |
| **ps_impulse** | 0xFE2C | 16 | Additional Functions 8 | Lower 16 bits of maximum distance |
| **Rampe+** | 0xFEFA | 16 | Basic Functions 3.6.3.4 | Acceleration and braking ramp |
| **RefUmdr** | 0xFEDC | 16 | Additional Functions 5 + 6 | Upper 16 bits of position of zero pulse |
| **RefLage** | 0xFEDA | 16 | Additional Functions 5 + 6 | Lower 16 bits of position of zero pulse |
| **RefV3** | 0xFED6 | 16 | Additional Functions 6 | Speed of zero reference point search |

### 16.5.3 Positioning

**Number of function** 2

**CP_Mode**

- Bit 0 — 1 = absolute<br>0 = relative
- Bit 1 — 1 = auto start<br>0 = starting signal required
- Bit 2...6 — 0 = reserved

- Bit 7 — absolute positioning<br>0 = reserved

relative positioning<br>0 = positive direction<br>1 = negative direction

**Description** See also Chapters 8 and 9.

The positioning can be absolute or relative. The positioning can be executed immediately after the positioning calculation has been executed (auto start) or after a new starting signal has been given.

If you want to execute a relative positioning procedure, specify the direction in **CP_Mode** Bit 7.

In the positioning mode, **CP_Status** Bit 5 indicates the state of the positioning procedure (0 = not yet in position, 1 = in position).

In case of an absolute positioning procedure, the target position consists of **psa_positionH** and **psa_positionL**. The distance of a relative positioning procedure consists of **ps_umdrehung** and **ps_impulse**.

**Other parameters (to be modified only if required):**

| Name | Address | Size [Bit] | See manual ..., Chapter | Description |
|---|---|---|---|---|
| **psa_positionH** | 0xFE4E | 16 | Additional Functions 9 | Upper 16 bits of absolute target position |
| **psa_positionL** | 0xFE50 | 16 | Additional Functions 9 | Lower 16 bits of absolute target position |
| **ps_umdrehung** | 0xFE2A | 16 | Additional Functions 8 | Upper 16 bits of relative distance |
| **ps_impulse** | 0xFE2C | 16 | Additional Functions 8 | Lower 16 bits of relative distance |
| **ps_v0** | 0xFEE6 | 16 | Additional Functions 8 + 9 | Speed of positioning procedure |
| **Window** | 0xFED8 | 16 | Additional Functions 8 + 9 | „In Position“ tolerance |
| **Rampe+** | 0xFEFA | 16 | Basic Functions 3.6.3.4 | Acceleration and braking ramp |

### 16.5.4 Homing

**Number of function** 3

**CP_Mode**

- Bit 1...0 0 = reserved
- Bit 2 1 = monitoring of distance active
- Bit 4...3 move to resolver zero point
  0 0 = using shortest way
  0 1 = always in positive direction
  1 0 = reserved
  1 1 = always in negative direction

- Bit 6...5 specify homing procedure
  0 0 = search for home switch only
  0 1 = search for home switch and move to resolver zero point
  1 0 = move to resolver zero point only
  1 1 = reserved

- Bit 7 0 = positive direction
  1 = negative direction

**Description** See also Chapter 5.

The homing procedure includes searching for the home switch or moving to the resolver zero point or both.

The homing procedure does not involve setting an actual value. If this is required, activate the respective function afterwards.

**Other parameters:**

| Name | Address | Size [Bit] | See manual ..., Chapter | Description |
|---|---|---|---|---|
| **RefV1** | 0xFEDE | 16 | Additional Functions 5 | First homing speed<br>(move to home switch at high speed) |
| **RefV2** | 0xFEE0 | 16 | Additional Functions 5 | Second homing speed<br>(low speed until home switch opens again) |
| **Rampe+** | 0xFEFA | 16 | Basic Functions 3.6.3.4 | Acceleration and braking ramp |

### 16.5.5 Fixed setpoint

**Number of function** 4

**CP_Mode**
- Bit 7...0 0 = reserved

**Description** Setpoint setting for speed over service channel.

Write desired setpoint for speed to **Sollwert**.

**Other parameters:**

| Name | Address | Size [Bit] | See manual ..., Chapter | Description |
|---|---|---|---|---|
| **Sollwert** | 0xFE60 | 16 | Basic Functions 3.6.5 | Setpoint |
| **Rampe+** | 0xFEFA | 16 | Basic Functions 3.6.3.4 | Acceleration and braking ramp |

### 16.5.6 Table interpolation

**Number of function** 5

**CP_Mode**
- Bit 7...0 0 = reserved

**Description** See Chapter 12 (table interpolation without override function).

Activation of a motion cycle table from the external memory.

Write the first address of the table in **?Tabelle** and start the function.

**Other parameters:**

| Name | Address | Size [Bit] | See manual ..., Chapter | Description |
|---|---|---|---|---|
| **?tabelle** | 0xFE3A | 16 | Additional Functions 12 | Pointer to beginning of table |
| **ps_impulse** | 0xFE2C | 16 | Additional Functions 12 | Trace offset |
| **ps_umdrehung** | 0xFE2A | 16 | Additional Functions 12 | Pointer to the value to be traced |

### 16.5.7 Setting an actual value

**Number of function** 6

**CP_Mode**
- Bit 7...0 0 = reserved

**Description** See Chapter 6

Setting the actual position to a predefined value.

**Other parameters:**

| Name | Address | Size [Bit] | See manual ..., Chapter | Description |
|---|---|---|---|---|
| **RefUmdr** | 0xFEDC | 16 | Additional Functions 6 | Upper 16 bits of new actual position |
| **RefLage** | 0xFEDA | 16 | Additional Functions 6 | Lower 16 bits of new actual position |

## 16.6 Error Sources / Troubleshooting

- The limit switches are always evaluated, regardless of the other settings. If no limit switches are connected, every function gets canceled.
- The register **CANControl** is internally used by the program and mapped to the external memory by **?CANControl**. Therefore reset **?CANControl** to 0xFFF3 after the program is deactivated.
- The pointers **?512us, ?512usA, ?512usB, ?Sollwert, ?nsoll, ?Rampe+** and **?Rampe-** are constantly manipulated by the software. After the software has been deactivated, the parameters may need to be corrected.

NOVOTRON

# 17 Use Of Several Software Extensions

You may load several software extensions to the memory, however, due to the organization of the external memory, the programs cannot be active at the same time.

When switching between several programs, certain rules must be followed in order to avoid undesired reactions. To display the memory bank in which the program is stored, use the parameter **Bank**. This parameter, like the function pointers, is checked frequently and switched internally. This serves the purpose to avoid invalid settings.

The switching procedure is as follows:

1. To disable the functions, reset the pointer registers **?512us, ?512usA, ?512us, ?SPS** and **?Init** to **@Dummy** (0x01EA), insofar as they contain a value between 0xD000 and 0xDFFF.

☞ Software extensions are always stored within the address range 0xD000 to 0xDFFF.

☞ Pointers that are not used can be expected to contain the correct value.

2. Wait 100 ms, until the variables have been switched internally.
3. Write the new value to register **Bank**.
4. Wait 100 ms, until the new memory bank is displayed.
5. Set the required pointer registers **?512us, ?512usA, ?512us, ?SPS** or **?Init** to the start address of the new program.

☞ See manual Basic Functions, Chapter 3.6.7

# 18 Error History

Starting with H8 Version 2.03

☞ The error history can be read out by means of the start-up software ND30Cfg.

Errors that occurred are saved in the error history together with the time-stamp of the active time. The errors are saved in a ring buffer with 63 entries. The error history can be read out from the external memory by means of the **ReadWordX** command.

| Address (external) | Word | Entry (Index) | Assignment of 16-bit value | |
|---|---|---|---|---|
| | | | MSB | LSB |
| 0x0200 | 0 | 1 | Operating hour | |
| 0x0201 | 1 | | Error code (BCD) | |
| 0x0202 | 2 | 2 | Operating hour | |
| 0x0203 | 3 | | Error code (BCD) | |
| 0x0204 | 4 | 3 | Operating hour | |
| 0x0205 | 5 | | Error code (BCD) | |
| ... | ... | ... | ... | |
| 0x027C | 124 | 63 | Operating hour | |
| 0x027D | 125 | | Error code (BCD) | |
| 0x027E | 126 | Counter | Error offset | Error counter |
| 0x027F | 127 | - | - | |

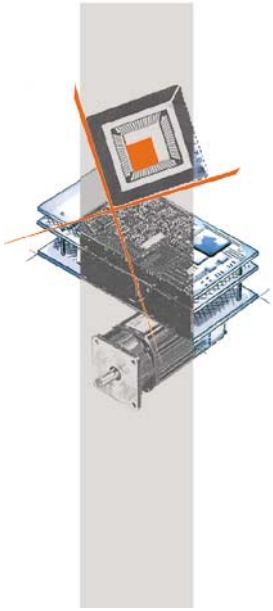
The error offset is the byte offset of the next free entry. The index of the last error is computed as folows:

```
if (error offset == 0)
        if (error counter == 0)
                -> error history is empty
        else
                LastErrorIndex = 63
else
        LastErrorIndex = (error offset / 4)
```

You may obtain the older errors by running reversely through the index. If you have reached 0, jump back to 63. The maximum number of entries is 63.

The total number of errors occurred is computed as follows:

*Total number = 63 * error counter + (error offset / 4)*



NOVOTRON

für Dynamik und Bewegung


***N O V O T R O N***

Industrie - Automation GmbH
Mauserstrasse 31
D - 71640 Ludwigsburg
Telefon 07141/2969 - 0
Telefax 07141/2969 - 22
e-mail: info@novotron-online.com
http: //www.novotron-online.com