Projektierung

ND21

1. Allgemeines

1.1. Angaben zu diesem Handbuch

Die vollständige Dokumentation Ihres Novodrives gliedert sich in 3 Teile:

- **Anwenderhandbuch ND21**.

Es richtet sich an Projektierer, Konstrukteure und Softwareentwickler. Darin finden Sie die Information, die Sie für die Konstruktion einer Anlage mit dem Novodrive benötigen.

- **Handbuch Inbetriebnahme und Parametereinstellungen von ND21**.

Es richtet sich an den Entwickler/Techniker, der die Inbetriebnahme und Konfiguration des ND21 durchführt. Wird auf Anfrage bei ND21-Lieferung mitgeschickt.

- **Anleitung zum Einbau/Austausch von ND21**.

Diese Anleitung richtet sich an den Techniker oder Elektriker, der den ND21 einbaut/austauscht. Dieses Faltblatt wird der ND21-Lieferung beigefügt.

In diesem Handbuch werden die unten aufgeführten Symbole verwendet. Diese Symbole sollen Ihnen das schnelle Auffinden wichtiger Informationen erleichtern.

Mit dem allgemeinen Gefahrensymbol sind Textstellen gekennzeichnet, die Sie unbedingt lesen und beachten müssen. Nichtbeachtung kann zur Gefährdung von Leben und Gesundheit von Personen führen.

ACHTUNG!

Mit ACHTUNG sind Textstellen gekennzeichnet, die Sie unbedingt lesen und beachten müssen. Nichtbeachtung kann zur Zerstörung oder Fehlfunktion des Novodrives oder der Anlage, in die der Novodrive eingebaut wird, führen.

1.2. Service / Kundendienst

NOVOTRON GmbH

Mauserstraße 31

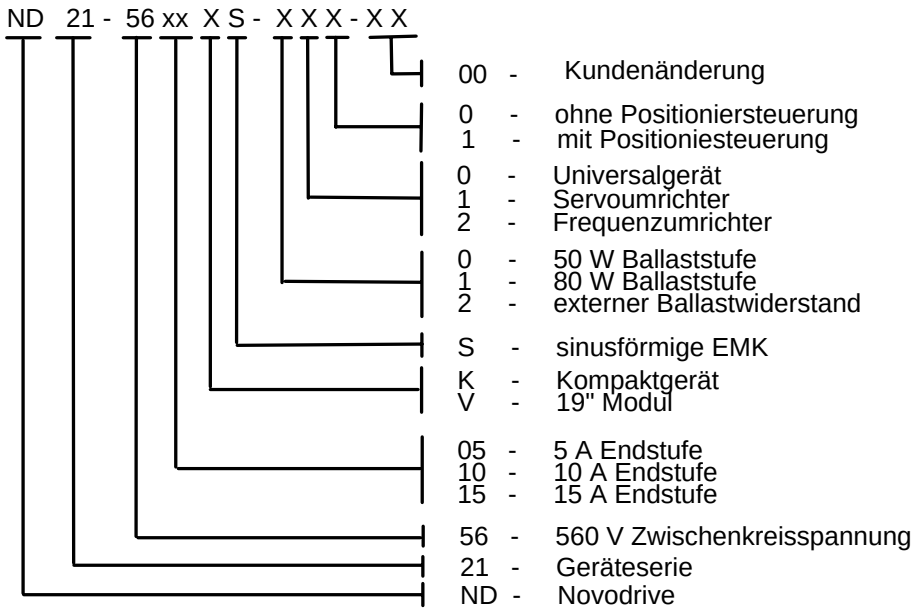
D - 71640 Ludwigsburg

Germany

Telefon: +49-71 41-29 69-0

Fax +49-71 41-29 69-22

1.3. Bezeichnung



Z.B. Ausführung:

ND21-5605**KS**-101-00: 5 A Universalgerät im Kompaktgehäuse mit Ballastwiderstand (80 W) und Positioniersteuerung

1.4. Schutzrechte

IBM ist eingetragenes Warenzeichen der IBM-Corporation.

2. Allgemeine Sicherheitshinweise

Im ND21 gibt es lebensgefährliche Betriebsspannungen!

Verdrahtung Deshalb ist vor dem Einschalten des ND21 die Verdrahtung zu kontrollieren. Überprüfen Sie, ob alle Stecker richtig gesteckt sind, und ob die Erdung richtig ausgeführt wurde.

Absicherung Stellen Sie sicher, daß keine spannungsführenden Teile versehentlich berührt werden können und die Absicherung des ND21 vorhanden und richtig angeschlossen ist.

Not-Aus Sehen Sie eine "Not-Aus"-Schaltung vor mit der Motor jederzeit stillgesetzt werden kann.

Entladezeit Nach dem Ausschalten beträgt die Entladezeit der Elkos ca. 1 Minute. Das bedeutet: Nach dem Ausschalten steht noch eine Minute lang eine gefährliche Berührspannung am Gerät an. Solange darf nichts berührt werden und kein Stecker gezogen werden.

Berührspannung Falls sich beim Ausschalten der Versorgungsspannung der Motor noch dreht, kann dieser die gefährliche Berührspannung noch bis zu seinem Stillstand aufrecht erhalten. Erst dann beginnt die Entladezeit der Elkos.

Ein- und Ausschalten Häufiges Ein- und Ausschalten der Versorgungsspannung in schneller Folge ist zu vermeiden, da dadurch die Einschalt-Strombegrenzung des ND21 überlastet werden kann. Diese Überlastung kann zur Zerstörung des Einschaltstrom-Begrenzungswiderstand führen.

2.1. Bestimmungsgemäße Verwendung

Novodrive ND21 Der Umrichter Novodrive ND21 ist ein Servo- und Frequenz-Umrichter zum Ansteuern von bürstenlosen Servomotoren und Asynchronmotoren. Er ist nach dem Stand der Technik gebaut. Ein anderer Einsatz als der beschriebene kann zu gesundheitlicher Gefährdung des Benutzers oder Dritter führen. Ferner können der Umrichter, der Antrieb oder andere Sachwerte beschädigt werden.

Den Umrichter nur in technisch, einwandfreiem Zustand, sowie bestimmungsgemäß, sicherheits- und gefahrenbewußt unter Beachtung dieses Handbuchs und der beiden anderen Anleitungen einsetzen.

Passender Antrieb

Nur bürstenlose Servomotoren oder Asynchronmotoren verwenden, deren technische Daten zum Umrichter passen und die den Vorschriften entsprechen.

Vorschriften

Den Umrichter nur entsprechend den länderspezifischen Vorschriften, Normen und Richtlinien in eine Anlage einbauen.

Umgebungs-bedingungen

Den Umrichter nicht in explosionsgefährdeten Bereichen oder im Medizingerätebereich, sowie in anderen Bereichen, die als gefährlich klassifiziert sind, einsetzen.

Ausnahme: Der Umrichter ist in für diese Zwecke zugelassene Gehäuse montiert und entsprechend den jeweiligen Vorschriften geprüft worden.

2.2. Organisatorische Maßnahmen

Sicherheits-vorschriften beachten

Als Hersteller und Betreiber einer Anlage, in der dieser Umrichter eingesetzt wird, sind Sie für die Einhaltung der geltenden Unfallverhütungs- und Sicherheitsvorschriften verantwortlich.

Qualifiziertes Personal

Stellen Sie sicher, daß die Installation und Wartung nur von einer Elektrofachkraft durchgeführt wird.

Stellen Sie sicher, daß die Inbetriebnahme nur durch eine dafür ausgebildete Elektrofachkraft durchgeführt wird. Bei der Inbetriebnahme sind die Sicherheitshinweise im Handbuch Inbetriebnahme und Parametereinstellung von ND21 zu beachten.

Handbücher

Der Konstrukteur/Entwickler einer Anlage, in die der Umrichter eingebaut wird, muß die Handbücher gelesen haben und die Sicherheitshinweise beachten.

Transport und Lagerung

Für den Transport und die Lagerung der Umrichter ist die dafür vorgesehene Originalverpackung zu verwenden.

2.3. Sicherheitseinrichtungen

Not-Aus

Falls durch bewegte Teile Gefahren für Personen oder Schäden an der Anlage entstehen können, Anlage mit einer Not-Aus-Einrichtung versehen. Führen Sie die Not-Aus-Einrichtung **entsprechend dem Kapitel 6. Einbau** aus.

2.4. Hochspannung

Der Umrichter arbeitet mit lebensgefährlicher Hochspannung. Deshalb folgende Punkte unbedingt beachten:

Stellen Sie sicher, daß keine spannungsführenden Teile berührt werden können.

- Absicherung **entsprechend dem Kapitel 6.2. Absicherung**.
- Not-Aus-Einrichtung, **wie in Kapitel 6.4. Not-Aus-Konzept ND21** beschrieben, installieren.
- Erdung richtig ausführen.
- Alle Anschlüsse **entsprechend Kapitel 5. Anschluß-belegung** ausführen.
- Das Gerät nicht zerlegen. Keine Veränderungen am Gerät vornehmen. Reparaturarbeiten dürfen nur vom Hersteller durchgeführt werden.

Bei der Inbetriebnahme die Sicherheitsvorschriften einhalten und überprüfen ob die Sicherheitseinrichtungen vorhanden sind.

2.5. Berührschutz Kompaktgerät

Forderung: Schutz gegen gefährliche Körperströme (Entwurf DIN VDE 160 Abschnitt 5.5)

Zur Erfüllung der obigen Forderung sind folgende Maßnahmen erforderlich:

- Ziehen oder Stecken der Steckverbinder des ND21 ist unzulässig wenn sich das Gerät unter Spannung befindet. Es darf nur dann Spannung an das ND21 angelegt werden, wenn alle Steckverbinder durch Verschraubung mit dem ND21 gegen Abgleiten gesichert sind. Arbeiten an Steckverbindern des ND21 sind unzulässig, wenn sich das ND21 unter Spannung befindet. Alle Anschlußarbeiten dürfen nur durch einen dafür ausgebildeten Elektronikfachmann ausgeführt werden.
- Das Öffnen des Gehäuses ist nicht erlaubt.
- Vor dem ersten Einschalten der Versorgungsspannung muß der sichere Sitz aller Kabel kontrolliert und geprüft werden, ob die Isolation an allen Aderenden in Ordnung ist.

- Dies ist bei den Wartungsintervallen der Anlage zu wiederholen. Eventuell lose Klemmen sind nachzuziehen. Kabel die zu beweglichen Teilen der Maschine führen, sind durch geeignete Zugentlastungen vor dem ND21 zu sichern.

  Schutz durch Isolierung der aktiven Teile (Entwurf DIN VDE 0160 Abschnitt 5.5.1.1.a)

- Es besteht mindestens Basisisolierung der aktiven Teile gegen das metallische Gehäuse. Das Gehäuse muß geerdet werden. Die Erdung erfolgt über eine am Gehäuse befindliche Erdungsschraube.

  Für die Erdung ist ein Leitungsquerschnitt ≥ 10 mm² CU erforderlich.

Peripheriesteckverbinder X3

Sichere Trennung aller Signale des Peripherie-Steckverbinders X3 von den aktiven Teilen durch doppelte Isolierung **(Entwurf DIN VDE 160 Abschnitt 5.6.3.2)** ist bereits auf dem ND21 gewährleistet. Das Kabelgehäuse für den Steckverbinder X3 muß eine nichtleitende Oberfläche besitzen.

Resolversteckverbinder X2

Zur sicheren Trennung aller Rückmeldesignale von den aktiven Teilen des ND21 ist es erforderlich alle Rückmeldesignale mit mindestens Basisisolierung für eine Bemessungsspannung von 300 V_{eff} zu versehen. Das Kabelgehäuse für den Steckverbinder X2 muß eine nichtleitende Oberfläche besitzen. Dadurch entsteht zusammen mit der Basisisolierung der Rückmeldesignale gegen die aktiven Teile des ND21 eine doppelte Isolierung nach **Entwurf DIN VDE 0160 Abschnitt 5.6.3.2.**

Leistungsanschluß X1

Die verwendeten Anschlußkabel und Leitungen müssen eine doppelte oder verstärkte Isolierung zwischen Ader und Oberfläche haben **(Entwurf DIN VDE 160 6.5.1.1).**

Der Anschlußbereich muß gegen versehentliches Berühren geschützt sein. Vor dem Einschalten der Versorgungsspannung muß der sichere Sitz aller Kabel kontrolliert und geprüft werden, ob die Isolation an allen Aderenden in Ordnung ist.

Für die Aderenden müssen Aderendhülsen mit Isolation, der entsprechenden Größe verwendet werden.

Busanschluß X4 und X5

Die Signale des Busses sind bereits durch doppelte Isolierung sicher von den aktiven Teilen des ND21 getrennt. Für die Anschlußkabel sind Kabelgehäuse mit einer nichtleitenden Oberfläche erforderlich.

2.6. Berührschutz 19"-Einschub-Gerät

Forderung: Schutz gegen gefährliche Körperströme (Entwurf DIN VDE 160 Abschnitt 5.5)

Zur Erfüllung der obigen Forderung sind folgende Maßnahmen erforderlich:

- Ziehen oder Stecken von Steckverbindern des ND21 ist unzulässig, wenn sich das Gerät unter Spannung befindet. Es darf nur dann Spannung an das ND21 angelegt werden, wenn alle Steckverbinder durch Verschraubung mit dem 19"-Rahmen oder dem ND21 gegen Abgleiten gesichert sind. Arbeiten an Steckverbindern des ND21 oder des 19"-Rahmens sind unzulässig wenn sich das ND21 unter Spannung befindet. Alle Anschlußarbeiten dürfen nur durch einen dafür ausgebildeten Elektronikfachmann ausgeführt werden.

- Das Herausziehen des ND21 aus dem 19"-Rahmen ist nur im spannungslosen Zustand und nur durch einen ausgebildeten Elektronikfachmann erlaubt. Vor dem ersten Einschalten der Versorgungsspannung muß der sichere Sitz aller Kabel kontrolliert werden, und geprüft werden, ob die Isolation an allen Aderenden in Ordnung ist. Dieses ist bei den Wartungsintervallen der Anlage zu wiederholen. Eventuelle lose Klemmen sind nachzuziehen.

- Der Anschlußbereich am Rückblech des 19"-Rahmens muß gegen versehentliches Berühren geschützt sein. Kabel die zu beweglichen Teilen der Maschine führen sind durch geeignete Zugentlastungen vor dem 19"-Rahmen zu sichern.
  Schutz durch Isolierung der aktiven Teile (Entwurf DIN VDE 160 (Abschnitt 5.5.1.1.a)

- Es besteht mindestens Basisisolierung der aktiven Teile gegen das metallische Gehäuse. Das 19"-Gehäuse muß geerdet werden. Die Erdung des Gehäuses erfolgt über eine am 19"-Gehäuse befindliche Erdungsschraube.

- Für die Erdung ist ein Leitungsquerschnitt ≥ 10 mm² CU erforderlich.

- Der 19"-Rahmen, die Seitenwangen des 19"-Rahmens, das Abdeckblech, der Lüftereinschub, sowie die Frontplatten sind durch doppelte Isolierung sicher von den aktiven Teilen des ND21 getrennt.

Peripheriesteckverbinder X3

(siehe Kapitel 2.5. Berührschutz Kompaktgerät)

Resolversteckverbinder X2

(siehe Kapitel 2.5. Berührschutz Kompaktgerät)

Leistungsanschluß X1

(siehe Kapitel 2.5. Berührschutz Kompaktgerät)

Busanschluß X4 und X5

(siehe Kapitel 2.5. Berührschutz Kompaktgerät)

3. Funktionsbeschreibung

3.1. Allgemeines

Servoumrichter

Beim ND21 handelt es sich um einen Servoumrichter für bürstenlose Synchronmotoren.

Ein- und Ausgänge

Der ND21 besitzt alle üblichen Steuer- und Melde- Ein- und -Ausgänge, wie Reglerfreigabe, Betriebsbereit, Endschalter usw.

Inbetriebnahme

Die Inbetriebnahme des ND21 erfolgt mit einem Laptop PC. Die im ND21 eingebaute Speicheroszilloskopfunktion bietet die Möglichkeit ohne meßtechnischen Aufwand alle Soll- und Istwertverläufe von Strom, Drehmoment und Rotorlage auf dem Laptopbildschirm grafisch darzustellen.

Dadurch wird die Justage des ND21 sehr einfach, da die Auswirkungen von Parameteränderungen sofort beurteilt werden können. Parameter werden im ND21 gespeichert und auf einer Diskette gesichert. Parametersätze können beliebig auf andere ND21 kopiert werden.

NOVOBUS

Über den Novobus können Antriebe mit einem Leitrechner vernetzt werden. Grundlage hierfür ist eine ringförmige Verdrahtung. Es werden keine weiteren Hardware-Komponenten benötigt. Das physikalische Übertragungs-medium ist die genormte, serielle Schnittstelle RS232 oder RS485.

Positionier-steuerung

Als Zusatz bietet Novotron eine Einachspositioniersteuerung im ND21 an. Damit läßt sich dann auch eine einfache Ablaufsteuerung im ND21 programmieren.

3.2. Leistungsteil

Netzanschluß

ND21 beinhaltet alle Komponenten die für eine Positionierachse erforderlich sind. Dazu gehört ein Netzteil für den direkten Anschluß des ND21 an das 400 V AC Netz. Schutzschaltungen begrenzen den Einschaltstoßstrom und schützen ND21 vor Netzüberspannungen und Transienten **(VDE 0160)**. Für die Eigenstromversorgung ist ein Schaltnetzteil integriert.

Zwischenkreis

Die Bremsenergie des Motors wird im Gleichspannung-Zwischenkreis aufgenommen. Falls die Kapazität der Zwischenkreis-Kondensatoren nicht ausreicht, wird der eingebaute Bremschopper aktiv und verhindert ein zu hohes Ansteigen der Zwischenkreisspannung.

Beim Einsatz von mehreren ND21 besteht zusätzlich die Möglichkeit, die Zwischenkreise zu koppeln. Somit kann die

Bremsenergie auf mehrere Zwischenkondensatoren verteilt werden. Der Zwischenkreis von ND 21 darf mit dem anderer Hersteller nicht parallel geschaltet werden.

Ballastschaltung Die interne Ballastschaltung ist elektronisch überwacht. Bei Überlast wird die Ballastschaltung abgeschaltet und ND21 geht in Überspannung.

Leistungsschalter Ein IGBT Umrichter versorgt den Motor mit Strom. Dabei ist jeder Schalter des Umrichters einzeln kurzschlußfest. Alle Netzstromkreise sind galvanisch von der Regelung getrennt.

3.3. Regelungsteil

Regelung Die Lage-, Drehzahl- und Stromregelung im ND21 erfolgt digital. Auch der Pulsweitenmodulator des ND21 arbeitet digital. Die Stromregelung und die Pulsweitenmodulation sind im von Novotron eigens für ND21 entwickelten ASIC NOVOCHIP untergebracht.

Rückmeldung Die übrigen regelungstechnischen Aufgaben übernimmt ein Hitachi H8 µController. ND21 benötigt als Rückmeldesystem im Motor einen Resolver. Der ND21 eignet sich auch für die Auswertung von Multiturnresolvern. Die Resolver/Digital-Wandlung erfolgt durch den Analog Devices Chip 2S82.

Die Auflösung der Rotorlagemessung beträgt bei Drehzahl bis 700 Upm 16 bit, bei Drehzahlen von 700 - 2700 Upm 14 bit und bei Drehzahlen größer als 2700 Upm 12 bit. Die Umschaltung erfolgt dynamisch im Betrieb.

Sollwerte Für die Sollwert stehen folgende Möglichkeiten zur Auswahl:

- Analog ± 10 V
- Analog ± 24 mA
- Frequenz und Richtungsimpulse sowie
- Digitale Sollwertvorgabe über den No Cost Sensor-Aktor-Bus "NOVOBUS"
- interne Positioniersteuerung

4. Technische Daten

4.1. Kenndaten

Normen

Überspannungsfestigkeit	EC 801-4 Klasse 2
Funkentstörung	EN55011 Klasse A wird in Verbindung mit Schaffner Netzfilter FN 351-8-29 eingehalten.
Isolationskonzept	2 Isolationsstrecken: Netzschaltkreise - Logik Logik - Peripherieankoppelung bzw. Logik - Busankoppelung **(DIN VDE 0160)**. **Anmerkung:** Resolversignale sind auf Logikpotential und haben Basisisolation gegen Netz. Für doppelte oder verstärkte Isolation gegen Netz ist deshalb ein isoliertes Kabel erforderlich **(siehe Kapitel 2. Allgemeine Sicherheitshinweise).**
Prüfspannung Isolation	2250 VDC
Schutzart	ND21 56XX VS IP00 (Schutzart wird vom 19"-Rahmen bestimmt) ND21 56XX KS IP20 nach DIN40050 und IEC144
Serielle Schnittstelle	RS232: ANSI/EIA232D RS485: EIA485

Signalverarbeitung

Modulation der Endstufe	Digitaler Pulsweitenmodulator
Modulationsverfahren	modifiziertes Sinus-Dreieck-Verfahren
Stromregelung	Digitaler PI-Regler Zykluszeit 54 µs
Auflösung Motorstrommessung	11 bit
Auflösung Drehfeldbildung	11 bit

Drehzahlregler	Digitaler PI-Regler mit Drehmomentvorsteuerung Zykluszeit 432 µs Auflösung 16 bit Anti-Wind-Up-Schaltung
Lageregler	Digitaler P-Regler mit Drehzahlvorsteuerung Zykluszeit 432 µs Auflösung 16 bit
Positioniersteuerung	Auflösung 4 Byte 40 Positionen im ND21 speicherbar Lineare Anfahr- und Bremsrampe
Lagerückmeldesystem	Resolver: dynamische Anpassung der Winkelauflösung. 16 Bit/Udr. bei Drehzahlen bis ca. 700 Upm 14 Bit/Udr. bei Drehzahlen von ca. 700 - 2700 Upm 12 Bit/Udr. bei Drehzahlen größer 2700 Upm
Absolut Weg-Meßsystem	Absolut-Resolversystem IMAS der Fa. Baumer electric wird unterstützt
Motortemperaturfühler	wahlweise: Öffner, NTC, PTC

Kommunikation

Serielle Datenkommunikation	Kommunikationsprotokoll: NOVOBUS
Analogschnittstelle	Spannung (±10 V) Strom (±20 mA)
Frequenz/Richtungsschnittstelle	Schrittmotorkommunikation mit maximaler Frequenz 150kHz
SPS-Ankopplung	**(siehe Kapitel 13.3.3. ND21 und Leitrechner und SPS)**

4.2. Elektrische Daten

4.2.1. Netz-Anschluß, Versorgungsspannung

Gerätetyp ND21-	5605	5610	5615
Nennanschluß-spannung	400 VAC		
Anschluß-spannungsbereich	150 ... 450 VAC		
Anschlußwert bei Nennstrom	3,5 kVA	7,5 kVA	10 kVA
Maximaler Einschaltstoßstrom (intern begrenzt)	9 A		
Absicherung	3 x 10 A träge	3 x 20 A träge	3 x 20 A träge
Eigenstrom-versorgung	eingebautes Schaltnetzteil		
Eigenleistungs-aufnahme	20 W		

4.2.2. Zwischenkreis und Endstufe

Gerätetyp ND21-	5605	5610	5615
Zwischenkreisspannung bei Nennanschlußspannung	560 V		
Abschaltschwelle bei Überspannung	800 V		
Abschaltschwelle bei Unterspannung	200 V		
Zwischenkreiskapazität	300 uF		
Verlustleistung der Endstufe bei Nennstrom In	50 W	100 W	100 W

Restspannungsabfall bei Nennstrom		4 V	
Taktfrequenz der Endstufe	9,26 kHz	4,6 kHz	4,6 kHz
Taktfrequenz am Motor	18,5 kHz	9,2 kHz	9,2 kHz
Nennmotorstrom Tu = 25°C	5 Aeff	10 Aeff	15 Aeff
Spitzenmotorstrom Tk = 25°C	10 Aeff	20 Aeff	20 Aeff

Maximaler Ausgangsstrom in Abhängigkeit von der Kühlkörpertemperatur:

Temp	**25**	**35**	**45**	**55**	**65**	**75**	**85**	**°C**
5605	10	8,7	8,2	7,1	6,1	5,3	4,4	Aeff
5610	20	17,4	16,2	14,1	12,3	10,6	8,8	Aeff
5615	20	20	20	20	18,4	15,9	13.2	Aeff

Diese Ströme können ohne zeitliche Begrenzung abgegeben werden, falls die Kühlkörpertemperatur auf dem angegebenen Wert gehalten wird.

Anmerkung: Bei guter Fremdbelüftung und einer Umgebungstemperatur von 40°C und 10 A_{eff} Motorstrom erreicht der Kühlkörper bei ND21 5610 ca. 75°C.

4.2.3. Ballastschaltung

Dauerverlustleistung	intern Ballastwiderstand, extern (Option)	50/100 W 5 kW
Impulsleistung Ballastschaltung		15,5 kW
Einschaltschwelle (Schwellenautomatik)		725 - 750 V

4.2.4. Lüfter Kompaktgehäuse

Lüftungsart	fremdbelüftet (eingebaut)
Anschluß	Spannung: 220 V_{AC} Stromaufnahme: < 200 mA
Absicherung	315 mA träge

4.2.5. Verwendete Stecker

Anschluß für Leistungsteil X1 5605,5610,5615	Phönix Combicon Front-GMSTB 2,5/12-STF
Anschluß Resolver und Motortemperaturfühler X2	High Density D-SUB 15pol (auf dem ND 21: Buchsenkontakte)
Anschluß Peripherie X3	High Density D-SUB 44pol (auf dem ND21: Buchsenkontakte)
Anschluß Busausgang X4	D-SUB 9pol (auf dem ND21: Stiftkontakte
Anschluß Buseingang X5	D-SUB 9pol (auf dem ND21: Buchsenkontakte)

4.2.6. Resolver

Resolver:	Sagem: 21RX360407, 15RX310107
	Litton: JSSBH-15 E-5, JSSBH-21-P4
	Siemen: V23401-H2001-B202
	Tamagawa: TS2018N321 E52, TS2112N21 E11

4.3. Mechanische Daten

4.3.1. Abmessungen

19"Modul	**5605 u 5610**	229 mm x 100 mm x 71 mm
	5615	229 mm x 100 mm x 142 mm
Kompaktgerät	**5605 u. 5610**	327 mm x 180 mm x 86 mm
	5615	327 mm x 180 mm x 170 mm

4.3.2. Gewicht

19"Modul	**5605 u. 5610**	1,15 kg
	5615	2 kg
Kompaktgerät	**5605 und 5610**	3,6 kg
	5615	5 kg

4.4. Umgebungsbedingungen

Lagerungs-temperatur	**max. Lagertemperatur**	-25°C bis +70°C
	max. Luftfeuchtigkeit	95 %
Betriebs-temperatur	**Betriebstemperatur**	0°C bis 70°C
	rel. Luftfeuchtigkeit	20 - 75 %
	Höhe über NN	ab 1000 m über NN muß mit Leistungs-minderung gerechnet werden.

5. Anschlußbelegung

Hochspannung! Lebensgefahr auch in ausgeschaltetem Zustand!
Solange der Motor dreht, ist der Motor ein Generator! Deshalb unkontrollierten Antrieb des Novodrives im Störungsfall gegebenenfalls durch den Einbau einer Bremse verhindern.

5.1. Netzanschluß, Motoranschluß und Ballastwiderstand (X1)

Stecker X1 ND21 5605, 5610 und 5615: 12 pol Combicon.

Hinweis: Nachfolgend ist die Belegung für Novotron Motoren angegeben. Abweichende Anschlußbelegungen für andere Motorfabrikate: **(siehe Motoranschluß-Tabelle)**

Anschlußbelegung

Kontaktbelegung	**Pin**	
externer Ballastwiderstand (Option)	1	-
Netzanschluß	2	L1
Netzanschluß	3	L2
Netzanschluß	4	L3
-	5	-
Zwischenkreisspannung minus	6	- ZKS
Zwischenkreisspannung plus	7	+ZKS
Motoranschluß	8	V
Motoranschluß	9	U
Motoranschluß	10	W
Netzanschluß	11	PE
Netzanschluß	12	PE

Querschnitt der Anschlußkabel

Gerätetype ND21-	**5605**	**5610**	**5615**	
Netzanschluß 4 x	1,5	2,5	2,5	mm²
Motoranschluß 4 x (ohne Bremse)	1,5	2,5	2,5	mm²
Zwischenkreis-Bus 2 x	1,5	2,5	2,5	mm²
Ballastwiderstand 2 x	1,5	2,5	2,5	mm²

Alle Kabel abgeschirmt.

Isolation

Die verwendeten Anschlußkabel und Leitungen müssen eine doppelte oder verstärkte Isolierung zwischen Ader und Oberfläche haben **(Entwurf DIN VDE 160 6.5.1.1)**.

Absicherung

Gerätetyp ND21-	**5605**	**5610**	**5615**
Sicherung	3 x 10 A träge	3 x 20 A träge	3 x 20 A träge

Falls mehrere Antriebe gemeinsam abgesichert werden, ist für die Gesamtabsicherung von der Summe der Einzelgeräte auszugehen.

Motoranschluß

Der Motor wird mit einem abgeschirmten Kabel an den Steckverbinder X1 von ND21 angeschlossen. Der Kabelschirm ist sowohl auf Motorseite als auch auf der ND21 Seite aufzulegen (beidseitig aufgelegter Schirm). Bei ND21 sind zum Auflegen des Schirmes die dafür vorgesehenen Kabelschellen zu verwenden.

Der Drahtquerschnitt kann auf die zu erwartenden Motorströme ausgelegt werden. **Es gilt VDE 0113, deutsche Ausgabe von EN 60 204**:

Strombelasbarkeit

Nennquerschnitt	0,75 mm²	1,00 mm²	1,50 mm²	2.50 mm²	4 mm²
Nennstrom	7,5 A_{eff}	10 A_{eff}	13 A_{eff}	18 A_{eff}	25 A_{eff}

Die verwendeten Anschlußkabel und Leitungen müssen eine doppelte oder verstärkte Isolierung zwischen Ader und Oberfläche haben **(Entwurf DIN VDE 160 6.5.1.1)**.

Für die Aderenden müssen Aderendhülsen mit Isolation, der entsprechenden Größe verwendet werden.

Der Anschlußbereich muß gegen versehentliches Berühren geschützt sein. Vor dem Einschalten der Versorgungsspannung muß der sichere Sitz aller Kabel kontrolliert und geprüft werden, ob die Isolation an allen Aderenden in Ordnung ist.

Der Erdanschluß erfolgt an dem dafür vorgesehenen Erdungsbolzen auf dem Gehäuse des Kompaktgerätes oder auf der Rückwand des 19" Racks.

Empfohlene Motorkabel:Lütze Silflex NSY
Lapp Ölflex - 400CP

Motor-Anschlußtabelle

Fabrikat	Bez.	Pole	u	v	w	dRESOLVER
Novotron	NHD55	4	9	8	10	4800h
	NHD70					
	NHD92					
	NHD115	6	9	8	10	F180h
	NHD142					
	NHD190					
AEG	MS34-62	6	10	8	9	0000h
Siemens	1FT6	6	10	8	9	0000h
	1FT6	8	10	8	9	2000h
Baldor	BSM80A	4	8	9	10	4000h
MOOG	D313	8	8	10	9	0FC0h
Georgii Kobold	KSY464	6	9	8	10	0900h

Hinweis: Bei dRESOLVER handelt es sich um einen ND21 Parameter mit dessen Hilfe es möglich ist, beliebige Resolvereinstellungen elektronisch an ND21 anzupassen.

Bei der Variante mit internem Ballastwiederstand darf kein externer Ballastwiederstand angeschlossen werden!

Ballastwiderstand

Der Wert des externen Ballastwiderstandes muß 33 Ohm betragen. Der externe Ballastwiderstand wird am Stecker X1 Klemmen 1 und 7 angeschlossen. Die Kabel zum externen Widerstand müssen geschirmt sein.

Die interne Ballastschaltung ist elektronisch überwacht. Bei Überlastung wird die Ballastschaltung abgeschaltet und ND21 geht in Überspannung. Nach dem Ausschalten der Versorgungsspannung und einer Wartezeit von ca. 5 Minuten kann ND21 erneut in Betrieb genommen werden. Die Ballastschaltung ist dann wieder voll funktionsfähig.

Die Einsatzschwelle der Ballastschaltung liegt bei 720V Zwischenkreisspannung. Die eingebaute Schwellenautmatik erlaubt es, die Zwischenkreise mehrerer ND21 parallel zu schalten. Da diejenige Ballastschaltung die gerade arbeitet ihre Ansprechschwelle um bis zu 30V anheben kann, ist gewährleistet, daß die Ballastenergie auf alle angeschlossenen Ballastschaltungen gleichmäßig verteilt wird.

5.2. Resolver-Anschluß (X2)

Gefahr durch unkontrolliert laufenden Antrieb!

Wenn der Resolver falsch angeschlossen ist, kann der Antrieb unkontrolliert hochlaufen.

Deshalb beim Anschluß eines Motors an das ND21 unbedingt auf den richtigen Anschluß des Resolvers achten

Resolveranschluß ND21

Stecker X2

Kabel: Adern paarweise verdrillt und abgeschirmt.

Hinweis: Nachfolgend ist die Belegung für Novotron Motoren angegeben. Abweichende Anschlußbelegungen für andere Motorfabrikate **(siehe Resolveranschluß-Tabelle).**

Stecker: 15 pol HD-DSUB - Buchsenkontakte geräteseitig, Stiftkontakte kabelseitig

Schirm: Kabelgehäuse D-Sub HD

Kontaktbelegung	**Pins**			
	6		-	
Temperaturfühler	1	11	-	
	7			
Temperaturfühler	2	12	-	
	8		R1	Rotor
-	3	13		
	9		S2	Stator
Schirm R1, R2	4	14	S3	Stator
	10		S4	Stator
Rotor R2	5	15	S1	Stator

Resolveranschluß

	Novotron Baldor Moog Georgii Kobold	**AEG**	**Siemens**
Resolver	**X2**	**X2**	**X2**
Temperaturfühler	1	1	1
Temperaturfühler	2	2	2
Rotor R2 / REF	5	5	5
Rotor R1 / REF	8	8	8
Stator S2 / SIN	9	10	15
Stator S4 / SIN	10	9	14
Stator S1 / COS	15	15	9
Stator S3 / COS	14	14	10

Resolverkabel und Resolverstecker

Zur sicheren Trennung aller Rückmeldesignale von den aktiven Teilen des ND21 ist es erforderlich alle Rückmeldesignale mit mindestens Basisisolierung für eine Bemessungsspannung von 300 V_{eff} zu versehen.

Das Kabelgehäuse für den Steckverbinder X2 muß eine nichtleitende Oberfläche besitzen. Dadurch entsteht zusammen mit der Basisisolierung der Rückmeldesignale gegen die aktiven Teile des ND21 eine doppelte Isolierung nach **Entwurf DIN VDE 0160 Abschnitt 5.6.3.2.**

Das Resolverkabel sollte paarweise verdrillt und abgeschirmt sein.

Empfehlung für das Resolverkabel

Lütze Superflex (C)Y-PUR-Kombi Best. Nr.:111094 (Schleppkettentauglich, Ölbeständig)

Lütze Electronic-LIY(C)Y-(C)Y-Kombi Best. Nr.:110652 (Ölbeständig)

5.3. Peripherieanschluß (X3)

Kabel

Die verwendeten Kabel müssen der **VDE 0113 Abschnitt 14** entsprechen. Für den Analog-Sollwert, die Frequenz-Richtungsvorgabe und die Enkoderemulation sind unbedingt abgeschirmte Kabel erforderlich. Es wird empfohlen auch die anderen Signale mit abgeschirmten Kabeln anzuschließen.

Stecker X3

44 pol HD-DSUB: Buchsenkontakte geräteseitig, Stiftkontakte kabelseitig

Kontaktbelegung	**Pins**		
	16		ext. Einspeisung -15V
Strom-Meßwiderstand -	1	31	GPO1
	17		ext. Einspeisung +15V
Strom-Meßwiderstand +	2	32	analog Ausgang +
	18		analog Eingang +
Endschalter N	3	33	analog Ausgang -
	19		analog Eingang -
Inkremental-	4	34	Endschalter P

ausgang A				
		20		Inkrementalausgang B
Inkremental-ausgang N	5		35	ext. Einspeisung Null
		21		GPIN1
GPIN2	6		36	ext. Einspeisung Null
		22		GPIN3 / Start
GPIN4	7		37	ext. Einspeisung Null
		23		GPO2
GPIN6	8		38	ext. Einspeisung Null
		24		GPIN5 / Freigabe
	9		39	ext. Einspeisung Null
		25		DIR1/GPIN8
FREQ/GPIN7	10		40	ext. Einspeisung 5-24V
		26		-
-	11		41	-
		27		-
-	12		42	-
		28		Betriebsbereit Kontakt 1
Betriebsbereit Kontakt 2	13		43	-
		29		-
-	14		44	int. Null (Option)
		30		int. +15 V (Option)
intern -15 V (Option)	15			

5.3.1. Analog Eingang

Auflösung: 11 bit + 1 Vorzeichenbit, automatische Offsetkompensation.

Analog Eingang +: Stecker X3 44pol HD-DSUB Pin 18.

± 10 V Analog Eingang für Sollwerte oder als Prozessignal-eingang.
Eingangswiderstand R_i = 20 kOhm.

Analog Eingang -: Stecker X3 44pol HD-DSUB Pin 19 ist mit externer Einspeisung-Null verbunden.

Strom-Meßwiderstand +: Stecker X3 44pol HD-DSUB Pin 2,
Strom-Meßwiderstand -: Stecker X3 44pol HD-DSUB Pin 1

Zwischen den beiden Anschlüssen befindet sich ein Meßwiderstand R = 410 Ohm, 0,3 W, 1 %. Der Widerstand dient zur Umsetztung von +/- 24 mA Signalen auf +- 10 V.

Hinweis: Drehzahlvorgabe über Analogeingang **(siehe Handbuch Inbetriebnahme und Parameter-einstellung von ND21, Kapitel 3).**

5.3.2. Analog Ausgang

Analog Ausgang +: Stecker X3 44pol HD-DSUB Pin 32

Analog Ausgang -: Stecker X3 44pol HD-DSUB Pin 33 mit externer Einspeisung-Null verbunden.

+/- 10 V Analog Ausgang für analoge Prozeßperipherie Ansteuerung, Belastbarkeit 5 mA, Auflösung 8 bit.

Funktionsweise: Ein 20 kHz PWM-Signal auf GPO1 wird gefiltert als Analogwert ausgegeben.

Hinweis: Betrieb des Analogausganges **(siehe Handbuch Inbetriebnahme und Parametereinstellung von ND21, Kapitel 4 Analog-Ausgang).**

Der Analogausgang ist nicht kurzschlußfest!

Hinweis: Für den Betrieb der analogen Signalein- und ausgänge ist eine externe Stromversorgung erforderlich:

ext. Einspeisung +15V +/-10%, 30mA: X3 44p HD-DSUB Pin7

ext. Einspeisung -15V +/-10%, 30mA: X3 44pol HD-DSUB Pin 16

ext. Einspeisung Null: Stecker X3 44pol HD-DSUB Pin 33

ext. Einspeisung 5-24V: Stecker X3 44pol HD-DSUB Pin 40

(für die externe Einspeisung 5-24V können auch die +15V verwendet werden).

5.3.3. Digitaleingänge: Gruppe 1

Signalpegel der Digitaleingänge: GPIN1 - 6, ESCHP, ESCHN

"0": < 2 V

"1": > 4,5 V (max 24 V +10 %)

Eingangswiderstand: 4,7 kOhm

Eingang	Funktion	Aktivpegel	Stecker X3 Pin
GPIN1	Mode Schalter	"1"	21
GPIN2	Mode Schalter	"1"	6
GPIN3	Start	"1"	22
GPIN4	Referenznocken	"1"	7
GPIN5	Freigabe	"1"	24
GPIN6	Mode Schalter	"1"	8
ESCHP	Endschalter positiv	programmierbar	34
ESCHN	Endschalter negativ	programmierbar	3

5.3.4. Digitaleingänge: Gruppe 2

"0": < 1,5 V

"1": > 2,5 V (max 24 V +10 %)

Eingangswiderstand: 4,7 kOhm (Pull-Down)

FREQ/GPIN7: Stecker X3 44pol HD-DSUB Pin 10
Frequenzeingang oder Mode Schalter
Maximale Frequenz: 500 kHz

DIR1/GPIN8: Stecker X3 44pol HD-DSUB Pin 25.
Richtungseingang bei Frequenzvorgabe oder zusätzlicher Mode Schalter.
Maximale Richtungsänderung-Frequenz: 2,5 kHz

Hinweis: Für Frequenz-/Richtungsvorgabe **(siehe Handbuch Inbetriebnahme und Parametereinstellung von ND21, Kaptiel 2).**

Hinweis: Die Nutzung des GPIN8 als Mode Schalter (**siehe Kapitel 13, Die ND21-Positioniersteuerung).**

Es können immer nur entweder der Analogeingang oder die Frequenz-Richtungs-Vorgabe verwendet werden.

Hinweis: Für den Betrieb der Digitaleingänge der Gruppe 2 ist eine externe Versorgungsspannung erforderlich:
ext. Einspeisung 5-24V: Stecker X3 44pol HD-DSUB 44pol Pin 40
ext. Einspeisung Null: Stecker X3 44pol HD-DSUB 44pol Pin 39

5.3.5. Digitalausgänge

(gilt nicht für Encoderemulation)

Ausgangspegel: 5 - 24 V (abhängig von der externen Versorgungsspannung).

Belastbarkeit: 100mA

Die Digitalausgänge sind nicht kurzschlußfest!

GPO1: Stecker X3 44pol HD-DSUB Pin 31
FET-Ausgang, Rdson (FET) = 7,5 Ohm, eingebauter Pullup-Widerstand 6,8 K zu ext. Einspeisung 5 - 24V, eingebaute Schutzdiode.

Funktion: Ablaufsteuerung : Auftrag durchgeführt
oder PWM oder frei programmierbar.
Falls GPO1 als Digitalausgang verwendet wird, ist kein Analog Ausgang möglich.

GPO2: Stecker X3 44pol HD-DSUB Pin 23
Bipolartransistor-Ausgang, eingebauter Pullup-Widerstand 6,8 kOhm zur externen Einspeisung 5 - 24 V, eingebaute Schutzdiode.

Funktion: Motor steht "1"
oder Position erreicht"1"
oder Strombegrenzung "1"
oder frei programmierbar

Hinweis: Programmierung der Digitalausgänge (**siehe Kapitel 14.3. Konfiguration).**

Hinweis: Encoderemulation **siehe Kapitel 5.3.6.**

Hinweis: Für den Betrieb der Digitalausgänge ist eine externe Versorgungsspannung erforderlich:
ext. Einspeisung 5-24V: Stecker X3 44pol HD-DSUB 44pol Pin 40
ext. Einspeisung Null: Stecker X3 44pol HD-DSUB 44pol Pin 39

Beispiel: Geeigneter Pullupwiderstand für GPO1 und GPO2 um einen 24V 10mA SPS Eingang mit einem Pegel 20 V zu treiben:

$$R = \frac{24V - 20V}{10mA} = 400Ohm$$

Leistung des Pull-Up-Widerstandes:

$$R = \frac{24V^2}{400Ohm} = 1{,}44W$$

Belastung der 24 V Versorgung durch den Pullupwiderstand:

$$R = \frac{24V}{400Ohm} = 60mA$$

5.3.6. Encoderemulation (ROD426)

Optokoppler-Ausgänge: Interne Pull-Up-Widerstände 18 kOhm sind mit der externen Versorgung 5-24V verbunden, die Ausgangsspannung ist durch Z-Dioden auf 15V begrenzt.
Um die Optokoppler voll auszunützen, sollte extern ein Pullupwiderstand nach 5-24V angebracht werden.

Größe des externen Widerstand:

$$R_{ext} = \frac{U_{ext} \cdot 18K}{(57V - U_{ext})}$$

Inkrementalausgang Kanal A:
Stecker X3 44pol HD-DSUB Pin 4
Inkrementalausgang Kanal B:
Stecker X3 44pol HD-DSUB Pin 20
Inkrementalausgang Kanal N:
Stecker X3 44pol HD-DSUB Pin 5

Hinweis: Für den Betrieb der Encoderemulation ist eine externe Versorgungsspannung erforderlich.

ext. Einspeisung 5-24V:
Stecker X3 44pol HD-DSUB 44pol Pin 40

ext. Einspeisung Null:
Stecker X3 44pol HD-DSUB 44pol Pin 39

Hinweis: Für den Betrieb der Enkoderemulation **siehe Handbuch Inbetriebnahme und Parametereinstellung von ND21, Kapitel 5 Encoderemulation.**

Ausgangspegel: 0 - 15 V (begrenzt durch Z-Dioden)

Die Enkodersignale sind nicht kurzschlußfest!

5.3.7. Betriebsbereitkontakt

Betriebsbereit Potentialfreier Kontakt (Schließer) Belastbarkeit: < 500mA, < 100V
Betriebsbereit Kontakt 1: Stecker X3 44pol HD-DSUB Pin 28
Betriebsbereit Kontakt 2: Stecker X3 44pol HD-DSUB Pin 13

5.4. Busankopplung

Das ND21 ist mit einer RS232 oder mit RS422/485 Schnittstelle ausgerüstet. RS232, RS422 und RS485 sind genormte elektrische Datenschnittstellen.

NOVOBUS hat eine Ringstruktur. Der Leitrechner sendet Daten zum Antrieb Nr. n, dieser zu Antrieb Nr. n-1 usw.. Antrieb Nr. 0 sendet zum Leitrechner zurück.

Daten von einem Antrieb zum Leitrechner werden ebenfalls von Antrieb zu Antrieb weitergereicht, bis sie beim Leitrechner ankommen.

Um die Daten von Antrieb Nr. 0 zum Leitrechner zurück zu leiten, ist auf X4 bei Antrieb Nr. 0 ein Abschlußstecker erforderlich, der die jeweiligen Brücken für RS232 oder RS422/RS485 enthält. Hin- und Rückleitung sind im selben Kabel geführt.

NOVOBUS muß geerdet sein! Normalerweise ist er automatisch über den Leitrechner geerdet. Sollte das nicht der Fall sein z.B. wenn zwischen Leitrechner und ND21 Lichtwellenleiter-Komponenten eingesetzt werden), muß alternativ am Abschlußstecker geerdet werden (Pin 5 bei RS232, Pin 3 bei RS422/RS485).

Die Signale des Busses sind bereits durch doppelte Isolierung sicher von den aktiven Teilen des ND21 getrennt. Für die Anschlußkabel sind Kabelgehäuse mit einer nichtleitenden Oberfläche erforderlich.

Für NOVOBUS sind abgeschirmte Kabel zu verwenden.

5.4.1. Serielle Schnittstelle - Ausgang Stecker X4

Kabel Standard RS232 oder RS485 Kabel abgeschirmt.

Stecker DSUB 9 polig Stiftkontakte am ND21 Buchsenkontakte am Kabel.

Kontaktbelegung

	RS232	RS485	Pins		RS232	RS485
	X5.1	/TD	1			
				6	X5.6	TD
	X5.2	X5.2	2			
				7	X5.7	X5.7
	TD	TGND	3			
				8	X5.8	X5.8
	X5.4	X5.4	4			
				9	-	-
	SG	-	5			

Hinweis: X5.a bedeutet: Interne Brücke zu Stecker X5 Pin a (z. B. X5.1 bedeutet: Interne Brücke zu Stecker X5 Pin 1).

5.4.2. Serielle Schnittstelle - Eingang Stecker X5

Kabel Standard RS232 oder RS485 Kabel abgeschirmt

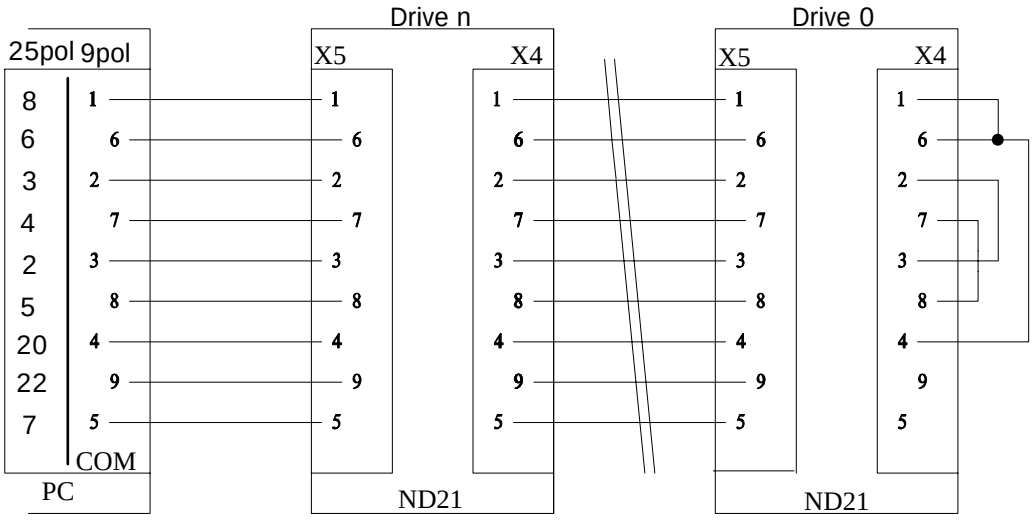
Stecker DSUB 9 polig Buchsenkontakte am ND21 Stiftkontakte am Kabel.

Kontaktbelegung

RS232	RS485	Pins		RS232	RS485
X4.1	/RD	1			
			6	X4.6	RD
X4.2	X4.2	2			
			7	X4.7	X4.7
RD	RGND	3			
			8	X4.8	X4.8
X4.4	X4.4	4			
			9	-	-
SG	-	5			

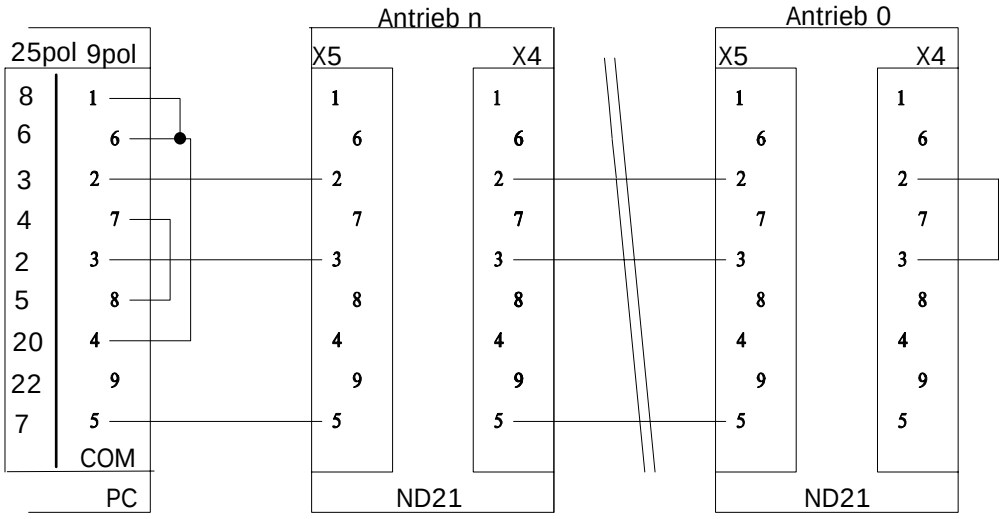
Hinweis: X4.a bedeutet: Interne Brücke zu Stecker X4 Pin a.

RS232 mit Verbindungskontrolle



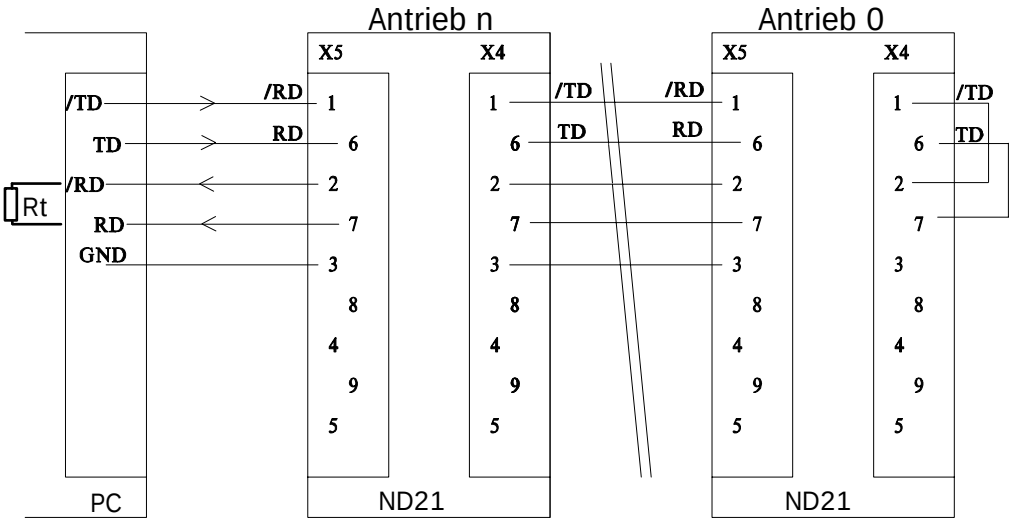
NOVOBUS mit RS232 und Verbindungskontrolle

RS232 ohne Verbindungskontrolle



NOVOBUS mit RS232 ohne Verbindungskontrolle

RS422, RS485



NOVOBUS mit RS422, RS485

Die Empfangsleitung im Leitrechner muß mit einem Widerstand R_t = 100 .. 120 Ohm abgeschlossen sein (bei ND21 ist der Abschlußwiderstand bereits integriert).

5.5. Lüfteranschluß

Am Kompaktgerät befindet sich ein 2 poliger Combicon-Stecker zur Stromversorgung des eingebauten Lüfters.

Anschlußspannung: 230 VAC

Stromaufnahme: < 200 mA

Absicherung: 315 mA träge

6. Einbau

6.1. Mechanischer Einbau

ACHTUNG!

Zerstörung des Novodrive!

Der Novodrive kann, wenn er in einer nicht geeigneten Umgebung betrieben wird, zerstört werden.

Lüftung

ND21 darf nur in vertikaler Lage eingebaut werden. Beim Kompaktgerät müssen dabei die Leistungs-Steckverbinder unten sein. Beim 19 " Rack muß der Lüftereinschub unterhalb der ND21 eingebaut sein. Der Bereich um die Lüftungsöffnungen von ND21 darf nicht verbaut werden. ND21 darf nicht oberhalb von wärmeerzeugenden Geräten montiert werden.

Umgebung

Die Montage darf nur an einem Ort erfolgen, der frei ist von Staub, Ruß, Metallspänen, korrodierenden oder metallischen Dämpfen, Gasen oder Flüssigkeiten. Kondensierung von Feuchtigkeit muß vermieden werden. Falls Kondensbildung nicht ausgeschlossen werden kann wenn ND21 außer Betrieb ist, so muß dafür gesorgt werden, daß die Kondensationsfeuchtigkeit vor der Inbetriebnahme entfernt wurde. Eventuell ist dazu am Einbauort ein geeigneter Heizer vorzusehen.

ND21 dürfen nicht in als gefährlich klassifizierten Bereichen betrieben werden, wenn sie nicht in zugelassenen Gehäusen montiert und geprüft worden sind.

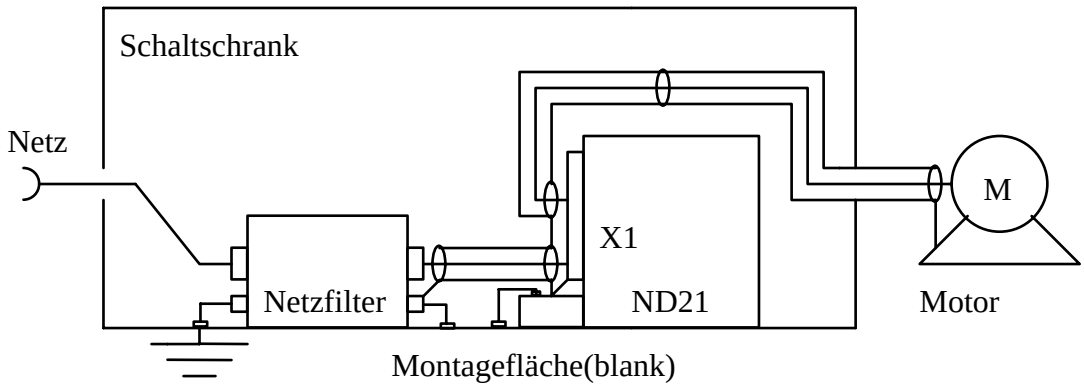
6.2. Absicherung

Gerätetyp ND21-	5605	5610	5615
Sicherung	3 x 10 A träge	3 x 20 A träge	3 x 20 A träge

Falls mehrere Antriebe gemeinsam abgesichert werden, ist für die Gesamtabsicherung von der Summe der Einzelgeräte auszugehen.

Vorsicht Hochspannung! Lebensgefahr!
Berührschutz ist nur gewährleistet, wenn Erdung, Netz- und Motoranschluß, wie in diesem Kapitel beschrieben, ausgeführt sind.

6.3. Erdung und Schirmung



Hinweis: Aus Gründen der Übersichtlichkeit sind im obigen Diagramm Schalter- und Sicherungselemente nicht eingezeichnet. Diese Elemente sind so einzufügen, daß sie den gezeichneten prinzipiellen Verlauf der Schirme und Erdungen möglichst wenig stören.

Um die **EMV Norm EN55011** einzuhalten, ist besonderes Augenmerk auf korrekte Schirmung und Erdung zu legen.

Erforderlich sind:

Abgeschirmte Motorleitungen.

Motorleitung Der Schirm ist auf der einen Seite am Motorgehäuse aufzulegen und auf der anderen Seite mit Kabelschelle am Kompaktgehäuse oder 19"-Rack des ND21 zu befestigen.

Abgeschirmte Versorgungsleitung zwischen Netzfilter und ND21!

Netzfilter Der Schirm ist auf der Filterseite mit dem entsprechenden Erdanschluß des Netzfilters zu verbinden und auf der ND21 Seite mit Kabelschelle am Kompaktgehäuse oder 19"-Rack des ND21 zu befestigen.

Für das Kompaktgerät wird der Netzfilter der Firma Schaffner FN 351-8-29 vorgeschlagen. Der Netzfilter muß, um voll wirksam zu werden, auf die blanke Montageplatte geschraubt werden. Zusätzlich sollte der Filter mit einer möglichst kurzen Verbindung 2,5 mm², zur Montageplatte hin, geerdet werden.

Schutzleiter Der Schutzleiteranschluß von ND21 (Pin 12 X3) ist mit der Erdungsschraube des Kompaktgehäuses oder beim 19" Rack mit den dafür vorgesehenen Schrauben zu verbinden. Verbindungsquerschnitt 2,5 mm² für 5605, 5610 und 5615 bzw 4 mm² für 5620.

Erdung des Kompaktgehäuses oder des 19"-Racks!

Erdung ND21 Wenn möglich, ist das Kompaktgehäuse auf die blanke Montageplatte zu schrauben. Zusätzlich möglichst kurze Erdverbindungen 10 mm² zwischen dem Erdungsbolzen des Kompaktgerätes oder des 19"-Racks zur Montageplatte oder Erdungsschiene verwenden. Die Montageplatte muß sehr gut geerdet sein.

Grundsätzlich gilt:

- Schirme beidseitig auflegen
- Erdungsverbindungen möglichst kurz und dick
- Schirme immer möglichst großflächig auflegen
- Die ungeschirmten Teile möglichst kurz halten
- Den Schaltschrank selbst gut erden
- Alle Leitungen immer möglichst kurz halten
- Signal- und Steuerleitungen immer räumlich getrennt von Leistungskabeln führen

EMV-Verträglichkeit nach EN 55011 wird nur gewährleistet, wenn:

- passender Netzfilter eingesetzt wird,
- Verbindungskabel zwischen Netzfilter und ND21 sowie Leistungskabel zwischen ND21 und Motor geschirmt ausgeführt sind,
- die Schirmung mit der Erdungsschraube des Netzfilters, dem Kompaktgehäuse oder 19''-Rack und dem Motorgehäuse verbunden ist.

6.4. Not-Aus-Konzept ND21

Wenn Gefahren für Personen oder Schäden an der Maschine entstehen können, müssen zu ihrer Verhinderung, durch Betätigen der Not-Aus-Einrichtung, gefährliche Teile der Maschine oder die ganze Maschine so schnell wie möglich stillgesetzt werden.

Verletzungsgefahr durch laufenden Antrieb!

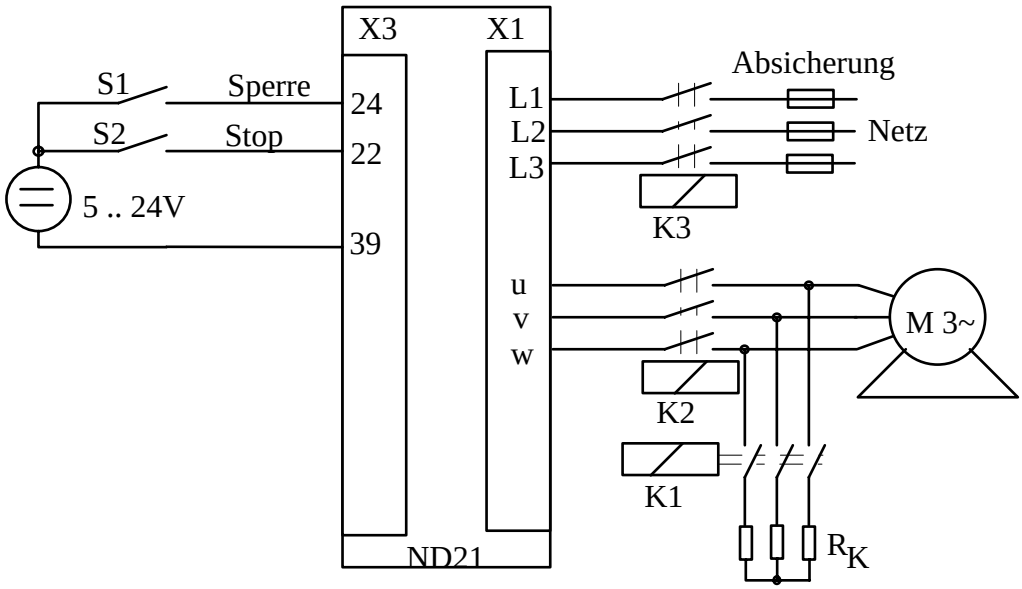
Durch bewegte Teile können Gefahren für Personen oder Schäden an der Anlage entstehen.

Deshalb muß die Anlage, in die der Novodrive eingebaut wird, mit einer Not-Aus Einrichtung versehen werden. Die Not-Aus-Einrichtung muß die Anlage so schnell wie möglich stillsetzen.

6.4.1. Allgemeines

In diesem Abschnitt werden zwei unterschiedliche Konzepte zur Not-Aus-Schaltung beschrieben:
Kurzschlußbremsung
Geregeltes Abbremsen

Wann, welches Not-Aus-Konzept anzuwenden ist, hängt von der Anlage ab. Abhängig von der Anwendung müssen Sie selbst das richtige Not-Aus-Konzept auswählen.



Bei der Not-Aus-Schaltung müssen Sie folgende Punkte beachten:

- Not-Aus-Schütz zwischen Novodrive und Motor muß einen Gleichstrom trennen können, der gleich dem Scheitelwert des Spitzenstroms Ihres Novodrives ist.

- Bei Kurzschlußbremsung muß der Bremswiderstand richtig dimensioniert sein.

6.4.1.1. Kurzschlußbremsung

Für die Kurzschlußbremsung der Servomotoren muß die Endstufe des ND21 gesperrt werden, in dem der Eingang 24 auf X3 stromlos gemacht wird. Gleichzeitig wird der Schütz K1 eingeschaltet.

Bemessung der Bremswiderstände R_K:

$$R_K = \frac{\text{Maximaldrehzahl (Upm)} \cdot \text{Spannungsradient (V/Upm)}}{\sqrt{3} \cdot \text{Spitzenstrom des Motors}}$$

6.4.1.2. Not-Aus mit geregeltem Abbremsen

Gefahr durch nach Not-Aus nachlaufenden Antrieb.

Antrieb wird bei Not-Aus geregelt abgebremst, deshalb ist auf folgendes zu achten:

- **Bei Not-Aus Endstufe nicht sperren! Geregeltes Abbremsen ist nur möglich, wenn die Endstufe freigegeben ist (weder Software- noch Hardwaresperre).**
- **Regelung muß korrekt eingestellt sein.**
- **Bei Not-Aus steht der Motor erst nach Abbau eines eventuell vorhandenen Schleppfehlers!**
- **Beachten Sie, daß bei einer mechanischen Störung ein Schleppfehler aufgebaut werden kann.**

ACHTUNG! **ND21 kann den Motor natürlich nur bei freigegebener Endstufe bremsen (weder Software-, noch Hardwaresperre).**

ACHTUNG! **Falls ND21 den Motor bremst, so erfolgt dies geregelt. Dies bedingt wiederum eine korrekt eingestellte Regelung. Falls z.B. die Regelung so eingestellt ist, daß sich im lagegeregelten Betrieb ein Schleppfehler aufgebaut hat, so wird dieser Schleppfehler noch abgebaut. Erst dann bleibt der Motor stehen.**

Die Bremsung erfolgt mit der vorprogrammierten Rampe. Anschließend muß die Endstufe gesperrt werden. Dazu ist der Freigabe - Eingang (Pin 24 auf X3) stromlos zu machen.

Anmerkung: Bei analoger Drehzahl-Sollwert-Vorgabe kann eine Bremsung auch dadurch ausgeführt werden, daß der analoge Sollwert auf 0 geschaltet wird.

Zusätzlich ist es erforderlich, den Motor vom Netz zu trennen. Hierfür gibt es ebenfalls zwei Möglichkeiten. Entweder kann mit Schütz K3 zwischen Netz und ND21 oder mit Schütz K2 zwischen ND21 und Motor getrennt werden.

Es ist nur einer der beiden Schütze erforderlich. Falls zwischen ND21 und Motor getrennt werden soll, ist zu beachten, daß vorher das ND21 gesperrt werden muß.

Gegebenenfalls muß das Abschalten des Schützes K2 durch geeignete Maßnahmen solange verzögert werden, bis eine Bremsung **nach Kapitel 6.4.1.2.** abgeschlossen und der Regler gesperrt ist. Grundvoraussetzung für die Zulässigkeit der Trennung zwischen ND21 und Motor ist, daß sich ND21 in einem abgeschlossenen Betriebsraum befindet, der gewährleistet, daß im Not-Aus-Fall eine Berührung stromführender Teile am ND21 ausgeschlossen ist.

ACHTUNG! **Der Schütz K2 muß geeignet sein einen Gleichstrom zu trennen, der dem Scheitelwert des Spitzenstromes von ND21 entspricht.**

7. Auslegung des Antriebs

In diesem Kapitel erfahren Sie, wie Sie den passenden Antrieb für ein gegebenes Antriebsproblem finden. Sie sehen an einem Berechnungsbeispiel, wie Sie Ihren Antrieb richtig auslegen.

7.1. Elektrische Auslegung

Fragestellung: Kann, mit einem für ein gefordertes Drehmoment ausgesuchten Motor, mit ND21, eine bestimmte Drehzahl mit diesem Drehmoment erreichen.

Beispiel: Eine Applikation erfordert, ein Drehmoment von 30 Nm zur Beschleunigung und es sollen mit diesem Drehmoment 2200 Upm erreicht werden. Es soll geprüft werden, ob eine derartige Beschleunigung mit ND21 5610 und einem Motor NHD142E6-180S möglich ist.

Der Motor NHD 142E6-180S hat ein Dauerstillstands-Moment von 16 Nm und kann zur kurzzeitigen Beschleunigung 5 fach überlastet werden. Es ist damit sicher die geforderte Beschleunigung möglich. Um herauszufinden, ob mit dem geforderten Moment bis zur geforderten Maximaldrehzahl beschleunigt werden kann folgende Berechnung:

Induktiver Spannungsabfall am Motor:

$U_L = n \; p \; L \; i \; 0{,}0453$

mit: NHD142E6-180S

n: Drehzahl [Umdrehungen pro Minute]	2200
p: Motorpolzahl	6
i: Beschleunigungsstrom effektiv	14,3 A
$i = \frac{M}{(3 \text{Drehmomentskonstante})}$	$\frac{30 \text{Nm}}{(3 \cdot 0{,}7 \frac{\text{Nm}}{\text{A}})}$
L: Motorinduktivität Phase - Phase [H]	0,022 H

Eingesetzt ergibt dies:

U_L = 188,3 V

Resistiver Spannungsabfall am Motor:

U_R = R i 0,866

mit

R: Wicklungswiderstand Phase - Phase [W] 1,9

damit :

U_R = 23,5 V

Gegen-EMK des Motors:

U_E = 0,5 $\sqrt{2}$ Vg n / 1000)

mit

Vg: Spannungsgradient Phase - Phase [V/1000] 180

damit U_E = 280 V

Spannungsbedarf für den Motor:

U = $\sqrt{(U_E + U_R)^2 + U_L{}^2}$ = 357,2 V

Erforderliche Zwischenkreisspannung:

U_{ZK} = $\sqrt{2}$ U = 505 V

Bei dreiphasigem 400 V Netzanschluß hat ND21 eine Zwischenkreisspannung von 565 V. Es steht also ausreichend Spannung zur Verfügung um die geforderte Drehzahl mit dem geforderten Drehmoment zu erreichen.

8. Novobus

In diesem Kapitel erfahren Sie, wie Sie den Novobus zur Steuerung Ihres Antriebs nutzen können.

8.1. Allgemeines

Das NOVOBUS-Protokoll und der NOVOBUS-Treiber für IBM-kompatible PCs sind geistiges Eigentum der Fa. Novotron GmbH.

NOVOBUS ist eine kostengünstige Lösung zur Vernetzung digitaler Antriebe und ermöglicht die schnelle Kommunikation zwischen einem Leitrechner (z.B. PC oder SPS) und den Antrieben:

- Soll-Istwert Austausch (z.B. Drehzahlsollwert und -istwert)
- Übertragung neue Position-Sollwerte für Positionierachsen
- Parametrierung des Antriebreglers (z.B. die Einstellung und On-line Änderung von Reglerstrukturen, Reglerparametern und den erlaubten Maximalwerten, usw.)
- Übertragung von Steuerbefehlen (Start, Stop, Reglersperre, ...)
- Abfrage wichtiger Informationen (z.B. Kühlkörper- und Motortemperatur, Endschalter-, Betriebsbereit-, In-Position-Signal, zusätzliche externe Signale als Prozeßinformation, integrierter Betriebsstundenzähler, Status des Antriebs, eventuelle Fehlermeldungen)
- Steuerung der programmierbaren analogen und digitalen Ausgänge der Antriebe (z.B. für Betätigung von Schützen oder Bremsen, Ausgeben von Warnsignalen usw.)

RS232/RS485 Das physikalische Übertragungsmedium für NOVOBUS ist die genormte serielle Schnittstelle RS232 oder RS485 - Standard bei allen PCs und modernen Steuerungen. Für die Kommunikation per NOVOBUS sind keine Hardware-Erweiterungen (wie z.B. Buskontroller Kommunikationskarte, Protokollchip, intelligente Busklemmen) nötig.

Treiber Alle digitalen Antriebe der Fa. Novotron GmbH enthalten standardmäßig die notwendige serielle Schnittstelle, sowie den Software-Treiber für das NOVOBUS-Protokoll. Für PC-Programme steht ein NOVOBUS-Treiber als Software-Bibliothek (NOVOBUS.LIB) kostenlos zur Verfügung.

Ringstruktur

Die einzige Voraussetzung für die Kommunikation mit NOVOBUS ist die ringförmige Verkabelung des Rechners und der Antriebe mit einem RS232- bzw. RS485-Kabel oder mit Lichtwellenleitern, sowie den entsprechenden optoelektronischen Umwandlern.

In einem Ring kann man bis zu 250 Achsen steuern. Die Antriebe sind durch ihre Positionen im Ring automatisch adressiert. Für höhere Übertragungsgeschwindigkeit oder um autarke Maschinenteile zusammenzufassen, können die Antriebe auf mehrere Ringe verteilt werden.

8.2. Leistungsmerkmale

Die Übertragung eines Byte dauert 286,46 µs bei 38400 bit/s Übertragungsgeschwindigkeit.

Die erforderliche Zeit zum kompletten Soll-Istwert Austausch bei drehzahlgeregelten Antrieben beträgt:

Antriebe pro Ring	Zeit
1	0.86 ms
2	2.0 ms
3	2.9 ms
4	3.7 ms
5	4.6 ms
6	5.4 ms

Die erforderliche Zeit zur Übertragung neuer Position-Sollwerte für Positionierachsen mit starten der PS-Rechnung beträgt:

Antriebe pro Ring	Zeit
1	4,01 ms
10	40,39 ms
100	401,33 ms
250	1002,79 ms

8.3. Physikalisches Übertragungsmedium

Wahlweise RS232 oder RS485.

Lichtwellenleiter sind in Verbindung mit Schnittstellen-umsetzern möglich.

Übertragungsgeschwindigkeit: standard 38400 bit/s

Die Übertragung erfolgt mit 8 Datenbits, 1 Paritybit (ungerade Parity) und 1 Stopbit.

8.4. Busstruktur

NOVOBUS hat eine ringförmige Struktur: Die Antriebe können auf einen oder mehrere Ringe verteilt sein.

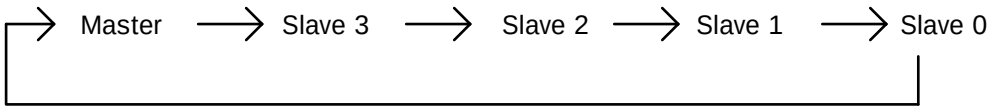
Pro Ring: 1 Bus Master (Leitrechner)
max. 250 Slaves (Antriebe)

Im Normalbetrieb dürfen die Slaves nur auf die Master-Telegramme antworten. Beim Time-Out Fehler dürfen auch die Slaves selbständig Fehler-Telegramme schicken.

8.5. Adressierung der Teilnehmer

In einem Ring können bis zu 250 Achsen gesteuert werden. Die Antriebe sind durch ihre Positionen im Ring automatisch adressiert. Die Nummerierung der Teilnehmer beginnt mit dem letzten im Ring und der Adresse '0'. Die Adresse des ersten Teilnehmers im Ring ist N-1, wobei N die Zahl der Slaves bedeutet.

z.B. bei 4 Achsen:



9. Bus-Definition

Der Master (Leitrechner) schickt kontinuierlich Telegramme. Die meisten Telegramme enthalten eine Addresse (Ausnahme: 'SYNC0' und 'PAUSE'). Die nicht angesprochenen Teilnehmer schicken die Telegramme weiter. Der angesprochene Teilnehmer antwortet auf das Telegramm. Die Länge der Antwort ist gleich der des Master-Telegramms.

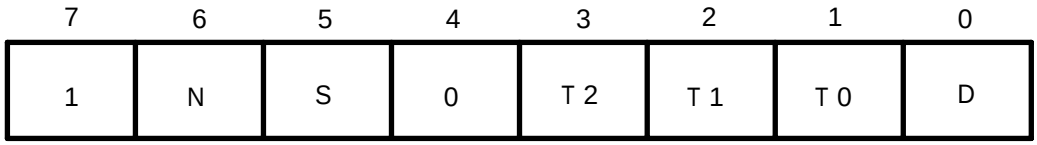
9.1. Telegramm-Aufbau

Synchronbyte	(obligatorisch)
Addressbyte	(optional)
Prozeß-Datenkanal	(2 Byte, optional)
Parameterkanal	(1-7 Byte, optional)

Alle Bytes werden mit ungerader Parity gesendet. Die Netto-Telegrammlänge (Prozeß-Datenkanal + Parameterkanal) kann maximal 7 Byte betragen. Bei einem Telegramm mit Prozeß-Datenkanal ist also der Parameterkanal max. 5 Byte lang.

9.2. Synchronbyte

Das Synchronbyte ist immer das erste Byte eines Telegramms. Darin ist kodiert die Telegrammlänge enthalten.



N = 1:	Next bei Short Address (sonst = 0)
S = 1:	Short Address
T2-T0:	Netto Telegrammlänge (ohne Synchronbyte und Addressbyte, 0...7)
D = 1:	Das Telegramm enthält Prozeßdaten-Informationen (2 Byte Datenkanal)

Kurzaddressierung

Bei S=1 und N=0 wird der gleiche Antrieb angesprochen, wie vorher.

Bei S=1 und N=1 wird der nächste Antrieb angesprochen. (Address = Address0+1)

Bei S=0 und N=0 folgt ein Addressbyte

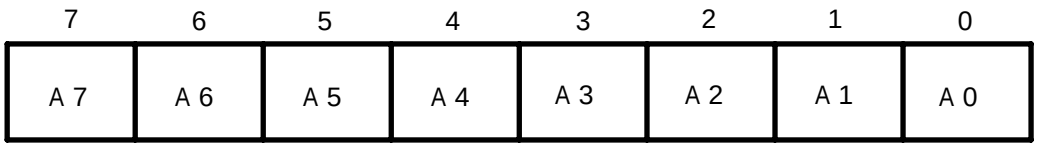
Falls kein Prozeß-Datenkanal oder Parameterkanal aktiv ist, wird das Synchronbyte 'SYNC0' gesendet, um den kontinuierlichen Datenstrom aufrechtzuhalten. 'SYNC0' wird von dem Empfänger unverändert weitergeschickt.

'SYNC0': H' 80

Um den Datenstrom aufzulockern, kann ein Synchronbyte 'PAUSE' gesendet werden. 'PAUSE' als Synchronbyte wird von dem Empfänger ignoriert (keine Antwort).

'PAUSE': H' 81

9.3. Addressbyte



Beim Master-Telegramm enthält A7-A0:

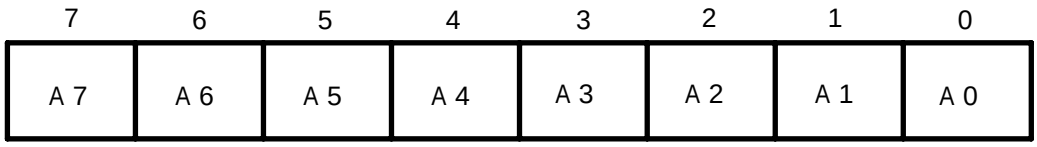
k-N

k = Teilnehmer-Adresse im Ring (s. 1.3)

N = Zahl der passiven Teilnehmer (Slaves).

Alle Teilnehmer erhöhen die Addresse um eins und schicken das Telegramm mit der neuen Adresse weiter. Ein Antrieb ist adressiert, wenn die Adresse nach der Inkrementierung gleich null ist. Die nicht adressierten Antriebe schicken das ganze Telegramm, ohne den Inhalt zu prüfen, bis auf das Addressbyte unverändert, weiter.

Beispiel: N=5 , der Leitrechner möchte den Teilnehmer Achse 2 ansprechen:(k-N = 2-5 = -3 = H' fd).



Master:	1	1	1	1	1	1	0	1	(H' fd=k-N)
Achse4:	1	1	1	1	1	1	1	0	(H' fe)
Achse3:	1	1	1	1	1	1	1	1	(H' ff)
Achse2:	0	0	0	0	0	0	0	0	(H' 00) <- !!!
Achse1:	0	0	0	0	0	0	0	1	(H' 01)
Achse0:	0	0	0	0	0	0	1	0	(H' 02 = k)

Die inkrementierte Adresse ist bei der adressierten Achse (Achse2) gleich Null. Der Master bekommt die Antwort auf sein Telegramm mit der Adresse 'k' zurück (In diesem Beispiel, in dem Achse2 angesprochen wurde: H' 02).

9.4. Prozeß-Datenkanal

Mit diesem Kanal kann schneller der Soll- und Istwert-Austausch realisiert werden. Dazu muß das niedrigste Bit im Synchronbyte gesetzt werden **(siehe Kapitel 9.2.).** Die Daten sind immer 2 Byte lang. Im Telegramm wird zuerst das Byte mit dem höheren Wert (MSB) und dann das mit dem niedrigen Wert (LSB) gesendet.

9.5. Parameterkanal

Mit Hilfe des Parameterkanals kann man die Antriebe mit Parametern versehen und Informationen von den Antrieben holen. Ein Befehl im Parameterkanal besteht aus einem Commandbyte, Datenbytes (0...4) und einem Kontrollbyte (Checksumme).

Mit einem Novobus-Telegramm können mehrere Befehle durch den Parameterkanal geschickt werden, bzw. ein Befehl im Parameterkanal kann in mehrere Telegramme verteilt werden. Die Antwort auf einen Befehl ist genauso lang wie der Befehl selbst (2 bis 6 Byte).

Das Commandbyte und das Kontrollbyte dürfen nie den Wert H' 00 haben.

9.6. Kontrollbyte im Parameterkanal

Der Master (Leitrechner) sendet alle Befehle im Parameterkanal mit einer Checksumme als Kontrollbyte (+Checksum). Diese Checksumme wird vom angesprochenen Antrieb kontrolliert, und der Befehl wird nur bei richtiger Checksumme ausgeführt, sonst meldet der Antrieb Fehler **(siehe Kapitel 9.7. Fehlerbehandlung).**

Die Checksumme ist die Summe der Bytes eines Befehls. Falls die Summe 0 ausgibt, muß die Checksumme korrigiert werden. Statt 0 ist 1 zu senden. (H' 00 ist reserviert für Fehlermeldungen). Der angesprochene Antrieb bildet aus seiner Antwort eine neue Checksumme und sendet deren Zweierkomplement (-Checksum) weiter.

Die Checksumme betrifft nur den Parameterkanal und wird nur aus den Bytes des Parameterkanal gebildet. Synchronbyte, Adressbyte und Prozessdaten werden nicht in der Checksumme berücksichtigt.

9.7. Fehlerbehandlung

Wenn der Antrieb einen Fehler in der Kommunikation bemerkt (Parityfehler, Framingfehler, falsches Synchronbyte, falscher Befehl im Parameterkanal, falscher Befehls-Parameter oder falsche Checksumme), geht er in Fehlerzustand.

Nach einem Fehler antwortet der Antrieb, der ihn zuerst erkennt mit H' 00 auf alle empfangene Bytes. Die folgende Achse kann den Fehler schnell erkennen, wenn sie adressiert wurde: falsches Synchron-, Command- oder Kontrollbyte (dürfen nie H' 00 sein), eventuell falsche Befehls-Parameter (falls an der Stelle keine H' 00 akzeptabel ist).

Ein nicht adressierter Antrieb prüft den Inhalt des Telegrammes nicht. Er kann den Fehler erst am Ausbleiben des nächsten Synchronbytes erkennen. Ein Telegramm kann maximal 9 Byte lang sein. Falls der Fehler im Adressbyte vorkommt, bemerken ihn die anderen Achsen, die seit diesem Fehler lauter H' 00 empfangen, erst mit dem 9. Byte, nach dem letzten Synchronbyte.

Ein fehlerfreies Telegramm kann nur in dem Addressbyte (1 Byte), im Datenprozeßkanal (2 Byte) und im Datenbereich des Parameterkanals (max. 4 Byte) H' 00 enthalten. Nacheinander kommt also höchstens 7mal H' 00. Die Antriebe im Fehlerzustand zählen deshalb, wieviele Nullbytes der vorgeschaltete Antrieb sendet.

Falls er ohne Unterbrechung 8mal H' 00 empfängt, bedeutet es für ihn, daß sich auch der vorherige Antrieb im Fehlerzustand befindet.

In diesem Fall prüft er die empfangenen Bytes, ob eine Checksequenz gesendet wird. Wenn er eine Checksequenz erkennt, geht er wieder in den Betriebszustand **(siehe Kapitel 9.8. Checksequenz).**

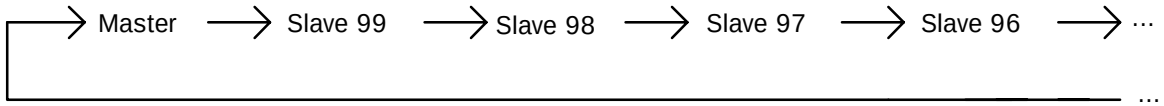
Damit auch der nächste Antrieb den Fehler mit Sicherheit erkennt, muß er mindestens 9mal H' 00 senden. Damit dieser Antrieb erkennt, daß der vorherige Antrieb auch im Fehlerzustand ist, muß er weitere 8 Nullbytes empfangen.

Deshalb senden alle Antriebe im Fehlerzustand zuerst 17mal H' 00. Falls sie erkannt haben, daß der vorherige Antrieb im Ring im Fehlerzustand ist, senden sie alle empfangenen Bytes inkrementiert weiter.

Der Master empfängt bei einem Fehler zuerst maximal 25mal H' 00, dann die Adresse des Antriebes der den Fehler bemerkt hat. Damit ist die einfache Lokalisierung des Fehlers gewährleistet.

Beispiel: 100 Achsen

Der Master möchte je ein Byte aus dem RAM der Antriebe No.95 bis No.99 lesen. Zwischen den Antrieben No.98 und No.97 wird ein Bit fehlerhaft übertragen. Der Antrieb No.97 bemerkt es als Parityfehler.



Das 1. Telegramm

1. Byte:	H' 88 =	Synchronbyte (folgt Addressbyte und 4 Bytes im Parameterkanal, keine Prozeßdaten)
2. Byte:	H' FB =	-5 = 95-100(Slave 95 ist angesprochen bei 100 Slaves im Ring)
3. Byte:	H' C0 =	Readbyte-Befehl für ND21
4. Byte:	H' 13 =	AddressL, LSB der Adresse H' FE13
5. Byte:	H' FE =	AddressM, MSB der Adresse H' FE13
6. Byte:	H' D1 =	Kontrollbyte (C0+13+FE = 1D1)

Die folgenden Telegramme unterscheiden sich nur im 2. Byte (96-100 = -4 = H' FC, 97-100 = -3 = H' FD, ...):

M	S	S	F	S	S	S	...	S
a	l	l	e	l	l	l		l
s	a	a	h	a	a	a		a
t	v	v	l	v	v	v		v
e	e	e	e	e	e	e		e
r	9	9	r	9	9	9		0
	9	8		7	6	5		0
88	88	88		88	88	88	...	88
FB	FC	FD		FE	FF	00		5F
C0	C0	C0		C0	C0	C0		C0
13	13	13		13	13	13		13
FE	FE	FE		FE	FE	>88		88
D1	D1	D1		D1	D1	+A5		A5
88	88	88		88	88	88		88
FC	FD	FE		FF	00	01		60
C0	C0	C0		C0	C0	C0		C0
13	13	13	12	!00	00	00		00
FE	FE	FE		00	!00	00		00
D1	D1	D1		00	00	00		00
88	88	88		00	00	!00		!00
FD	FE	FF		00	00	00		00
C0	C0	C0		00	00	00		00
13	13	13		00	00	00		00
FE	FE	FE		00	00	00		00
D1	D1	D1		00	*00	00		00

88	88	88	00	00	00		00
FE	FF	00	00	00	*00		*00
C0	C0	C0	00	00	00		00
13	13	13	00	00	00		00
FE	FE	88	00	00	00		00
D1	D1	A5	00	00	00		00
88	88	88	00	00	00		00
FF	00	01	#00	00	00		00
C0	C0	C0	00	#00	00		00
13	13	13	00	01	00		00
FE	88	88	00	01	#00		#00
D1	A5	A5	00	01	02	...	61
88	88	88	00	01	02		61
9C	9D	9e	00	01	02		61
.	.	.	.	.	.	.	.	.

Anmerkungen:

>:	Antwortbyte in einem Telegramm (Inhalt der Speicherzelle)
+:	Neue Checksumme
!:	Der Antrieb erkennt einen Fehler
***:**	Der Antrieb erkennt, daß der vorherige Antrieb im Fehlerzustand ist
#:	Das 17. gesendete Nullbyte nach der Fehlererkennung

Die Meldungen H' 00 zeigen dem Master den Übertragungsfehler, die Bytes H' 61=97 bedeuten, daß der Fehler vom Slave 97 bemerkt wurde.

9.8. Checksequenz

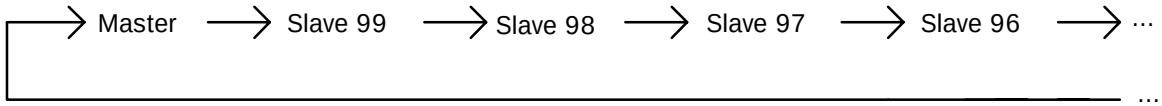
Falls der Master einen Fehler im Datenverkehr bemerkt, schickt er eine Checksequenz, um die Antriebe zurückzusetzen. Die ersten 17 Bytes vor der eigentlichen Checksequenz sind H' 00, damit auch die Antriebe in den Fehlerzustand gehen, die den Fehler noch nicht bemerkt haben. Nach der Checksequenz kann der Master die nicht abgeschlossenen Befehle wiederholen.

Die Cheksequenz lautet:

(17mal H' 00) H' ff H' 44 H' 72 H' 4c H' 41

Beispiel:

100 Achsen, die Achsen No.97 - No.0 sind im Fehlerzustand, die Achsen No.99 - No.98 haben keinen Fehler in der Kommunikation feststellen können. Der Master sendet gerade ein Telegramm, als er den Fehler durch ein unerwartetes Nullbyte erkennt. Nach der Erkennung des Fehlers sendet er 17mal H' 00, dann setzt er den Bus mit einer Checksequenz zurück.



	M	S	S	S	S	S	...	S
	a	l	l	l	l	l		l
	s	a	a	a	a	a		a
	t	v	v	v	v	v		v
	e	e	e	e	e	e		e
	r	9	9	9	9	9		0
		9	8	7	6	5		0
Angefangenes Telegramm	88	88	88	00	01	02	...	61
17 Nullbytes	00	01	02	00	01	02	...	61

00	00	00	00	01	02		61
00	00	00	00	01	02		61
00	00	00	00	01	02		61
00	!00	!00	00	01	02		61
00	00	00	00	01	02		61
00	00	00	00	01	02		61
00	00	00	00	01	02		61
00	00	00	*01	02	03		62
00	00	00	01	02	03		62
00	00	00	01	02	03		62
00	*00	*00	01	02	03		62
00	00	00	01	02	03		62
00	00	00	01	02	03		62
00	00	00	01	02	03		62
00	00	00	01	02	03		62
00	00	00	01	02	03		62

Checkcode

FF	@FF	@FF	@FF	@FF	@FF	...	@FF
44	44	44	44	44	44		44
72	72	72	72	72	72		72
4C	4C	4C	4C	4C	4C		4C
41	41	41	41	41	41		41

Wiederholung des Telegramms

88	88	88	88	88	88	...	88
FC	FD	FE	FF	00	01		60
C0	C0	C0	C0	C0	C0		C0

13	13	13	13	13	13	13
FE	FE	FE	FE	>88	88	88
D1	D1	D1	D1	+A5	A5	A5

Anmerkungen:

!:	Der Antrieb erkennt einen Fehler
***:**	Der Antrieb erkennt, daß der vorherige Antrieb im Fehlerzustand ist
@:	Der Antrieb erkennt das erste Byte der Checksequenz
>:	Antwortbyte in einem Telegramm (Inhalt der Speicherzelle)
+:	Neue Checksumme

Nach der Checksequenz ist der Bus wieder funktionsfähig.

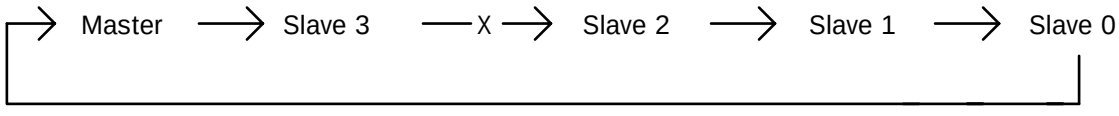
9.9. Time-out Fehler

Wenn der Antrieb mindestens 10 ms lang kein Byte empfängt, bedeutet es für ihn einen Time-out Fehler. Diese Funktion ist ausschaltbar **(siehe Kapitel 14.3.6. Das Byte EESelftest).**

Bei Time-out Fehler geht der Antrieb in den Fehlerzustand und wartet nach der Fehlererkennung weitere 10 ms, damit auch die anderen Antriebe den Time-out Fehler erkennen und dementsprechend reagieren können.

Nach dieser Wartezeit beginnt der Antrieb selbständig Nullbytes zu senden. Die anderen Antriebe erkennen es spätestens nach der achten 'H' 00' als Fehlermeldung und inkrementieren die empfangenen Bytes.

Beispiel: Unterbrechung zwischen Achse 3 und Achse 2.



Slave2: ... 00 00 00 00 00 00 00 00 00

Slave1:	...	00	00	00	*01	01	01	01	01	01
Slave0:	...	00	00	*01	02	02	02	02	02	02

***:** Der Antrieb erkennt, daß der vorherige Antrieb im Fehlerzustand ist.

Die empfangenen H' 02 Bytes zeigen dem Master, daß die Achse 2 den Fehler erkannt hat. Er weiß also: Entweder ist die elektrische Verbindung zwischen Achse 3 und Achse 2 unterbrochen oder Achse 3 ist nicht betriebsbereit.

10. ND21-spezifische Befehle

10.1. Read-Befehle

ReadByte: Liest ein Byte aus dem Speicherbereich des ND21:

H' 2F00-H' 2FBF ROM
H' FD80-H' FF7F RAM
H' FF80-H' FFFF On-chip register field

Read Byte

Befehl	H' C0	AddrL	AddrH	+CheckSum	
Antwort	H' C0	AddrL	Data	-Checksum	

ReadWord: Liest ein Wort (2 Byte) aus dem ND21-Speicher:

H' 2F00-H' 2FDF ROM
H' FD80-H' FF7F RAM

Read Word

Befehl	H' C1	AddrL	AddrH	'?'	+CheckSum	
Antwort	H' C1	AddrL	DataL	DataH	-Checksum	

('?' = H' 3F)

ReadLong: Liest 4 Bytes aus dem ND21-Speicher:

H' FD80-H' FF7F: RAM

Read Long

Befehl	H' C7	AddrL	AddrH	'1' '2'	+ Checksum	
Antwort	H' C7	AddrL	Data0	Data1	Data3	

Checksum ('1' = H' 31, '2' = H' 32, '?' = H' 3F)

10.2. Write-Befehle

WriteByte: Schreibt ein Byte in den Speicher des ND21:

H' FD80-H' FF7F: RAM

Write Byte

Befehl	H' 82	Data	AddrL	AddrH	+ CheckSum
Antwort	H' 82	Data	AddrL	AddrH	- Checksum

WriteWord: schreibt ein Wort (2 Byte) in den ND21-Speicher.

H' FD80-H' FF7F: RAM

Write Word

Befehl	H' 63	DataL	DataH	AddrL	AddrH	+CheckSum
Antwort	H' 63	DataL	DataH	AddrL	AddrH	-Checksum

10.3. Bit-Manipulation Logische Befehle

And: Führt ein logisches AND bitweise in den ND21-Speicher durch.
H' FF00-H' FF7F: RAM (AddrH = H' FF)
RAM := RAM & Data

AND

Befehl	H' A4	Data	AddrL	+Checksum
Antwort	H' A4	Data	AddrL	-Checksum

Or: führt ein logisches OR bitweise in den ND21-Speicher durch.
H' FF00-H' FF7F:RAM (AddrH = H' FF)
RAM := RAM | Data

OR

Befehl	H' A5	Data	AddrL	+Checksum
Antwort	H' A5	Data	AddrL	-Checksum

10.4. Input / Output-Befehle

WriteIO: Ändert die freiprogrammierbaren digitalen Ausgänge.

Write IO

Befehl	H' c8	Code	+Checksum
Antwort	H' c8	Code	-Checksum

Code:	H' 00	Clear	GPO1
	H' 01	Set	GPO1
	H' 02	Clear	GPO2
	H' 03	Set	GPO2

10.5. Reset H8

Löst einen Reset des ND21 Betriebsprogrammes aus. Alle nicht im EEPROM gesicherten Daten gehen verloren.

Reset

Befehl	H' DD	'!'	+Checksum
Antwort	H' DD	'!'	-Check-sum

('!' = H' 21)

11. Der Novobus-Treiber

11.1. Allgemeines

Für IBM-kompatible PCs steht eine kostenlose Laufzeit-Bibliothek (NOVOBUS.LIB) zur Verfügung.

Der Treiber kann Antriebe in einem Ring oder in zwei Ringen bedienen. Die Datenübertragung erfolgt durch die seriellen Schnittstellen der PCs.

Der Bustreiber arbeitet interruptgesteuert im Hintergrund. Die Aufträge sind als übliche Funktionsaufrufe zu programmieren. Um unnötige Wartezeiten zu vermeiden, werden die Aufträge in einem Buffer (FIFO) zwischengespeichert.

Der Treiber enthält Fehlerüberwachung- und Fehlerkorrektur-Routinen. Bei Störungen in der Datenübertragung versucht der Treiber den Fehler automatisch zu korrigieren Nach mehreren erfolglosen Korrekturversuchen, bzw. bei nicht korrigierbaren Fehlern, wird die Kommunikation gestoppt, und der Bustreiber gibt eine Fehlermeldung aus.

Der Bustreiber kann auch mit Echtzeitsystemen zusammenarbeiten. Die verschiedenen Tasks, die mit dem Bus arbeiten wollen, werden durch Semaphoren synchronisiert.

11.2. Hardware- und Software-Voraussetzungen

NOVOBUS.LIB ist eine Laufzeit-Bibliothek für IBM-kompatible MS-DOS Rechner.

Die Bibliothek wurde für die Programmiersprache Turbo C++ entwickelt (Speichermodell: small oder medium). Die Objektdatei kann man auch in Pascal einsetzten.

Die Novobus-Funktionen rufen keine Funktion von der C-Laufzeitbibliothek auf. Von den MS-DOS Funktionen werden nur die Funktionen H' 25 (Set-Interrupt-Vektor) und H' 35 (Get-Interrupt-Vektor) aufgerufen.

Von der BIOS-Interrupts wird nur INT10 genutzt (BIOS-Bildschirmausgabe für die Fehlermeldungen). Der Novobus-Treiber benutzt keinen arithmetischen Coprozessor.

Der Novobus-Treiber greift direkt auf den Interrupt-Controller (PIC, 8259) und auf die serielle Schnittstelle (UART, 8250/16450/16550) zu.

Der Novobus-Treiber arbeitet interruptgesteuert. Bei 38400 bit/s Übertragungsgeschwindigkeit erfolgt alle 286 µs ein Interrupt. Das Programm darf die Interrupts nicht lange sperren, sonst verliert der Rechner Daten (UART-Überlauffehler).

Falls es nicht zu vermeiden ist, muß man entweder den Bus schließen und nach der kritischen Operation neu starten, oder mit dem Baustein 16550 arbeiten (UART mit FIFO).

Der Bustreiber kann auch mit Echtzeitsystemen (wie z.B. RTKernel) zusammenarbeiten, die verschiedenen Tasks, die mit dem Bus arbeiten wollen, werden durch Semaphoren synchronisiert. Die einzigen Funktionen, die der Novobus-Treiber braucht :

- Semaphore setzen,
- warten auf Semaphore.

11.3. Einbinden der Bibliothek

Die Bibliothek NOVOBUS.LIB enthält die Module "NOVOBUS", "NBERROR" und "NBSEMAPHORE".

Die Novobus-Funktionen sind in dem Modul "NOVOBUS" zu finden. Die Prototypen enthält die Header-Datei novobus.h. Die Speicheradressen von ND21 sind in der Datei nd21.h zu finden.

Der Modul "NBERROR" enthält die Fehlerroutine NB_error( ) , die im Fehlerfall eine Fehlermeldung auf den Bildschirm schreibt. Falls eine eigene Fehlerroutine definiert wurde, muß man sie vor der Novobus-Bibliothek einbinden **(siehe Kapitel 11.5. Fehlerroutinen).**

Der Modul "NBSEMAPHORE" enthält zwei leere Funktionsdefinitionen für den Linker. Falls man mit Echtzeitsystem arbeiten möchte, muß man die Funktionen NB_Signal( ) und NB_Wait( ) selbst definieren und vor der Novobus-Bibliothek einbinden **(siehe Kapitel 11.7. Novobus im Echtzeitsystem).**

11.4. Den Bus öffnen und schließen

Der Novobus-Treiber kann mit einem Ring oder mit zwei Ringen arbeiten. Mit der Funktion InitNovobus1( ) kann man den Bus mit einem Ring öffnen, die Funktion InitNovobus2( ) öffnet den Bus mit zwei Ringen.

Mit der Funktion Close Novobus( ) kann man den geöffneten Bus wieder schließen.

Mit der Funktion NB_Status( ) kann man jederzeit abfragen, ob der Bus aktiv ist.

11.5. Fehlerroutinen

Bei einem Übertragungsfehler versucht der Novobus-Treiber den Fehler automatisch zu korrigieren. Bei jeder Fehlerkorrektur erhöht er den Wert des Zählers. Die Zahl der automatisch korrigierten Fehler kann mit der Funktion NB_BusError( ) abgefragt werden. Diese Funktion sollte regelmäßig aufgerufen werden, um die Übertragungsqualität zu kontrollieren. Wenn regelmäßig Übertragungsfehler vorkommen, muß man die Fehlerquelle suchen und sie beseitigen.

Falls die automatische Korrektur nicht mehr möglich ist, stoppt der Treiber den Bus, und gibt eine Fehlermeldung aus.

Bei einem solchen, nicht korrigierbarem Fehler, ruft der Novobus-Treiber die Funktion NB_error( ) auf. Die Bibliothek *NOVOBUS.LIB* enthält ein Modul "NBERROR". Falls die Funktion NB_error( ) im Anwenderprogramm nicht definiert wurde, greift der Treiber auf die eigene Fehlerroutine, die eine dem Fehlercode entsprechende Meldung auf dem Bildschirm bringt.

Falls eine andere Fehlerbehandlung erwünscht ist, kann eine eigene Fehlerbehandlungs-Routine geschrieben werden:

extern void far cdecl NB_error (unsigned char *Ring,* int *Errorcode*);

Parameter: *Ring:*1: Ring1, 2: Ring2, 0: unbestimmt.
Errorcode: Fehlercode.

Der Aufruf kann auch aus einer Interruptroutine erfolgen, deshalb ist größte Vorsicht geboten. (Möglichst schnelle Behandlung, sparsam mit dem Stack umgehen, nur reetrante Routine aufrufen, kein direkter oder indirekter DOS-Aufruf, kein Disketten- oder Festplatte-I/O, den aritmetischen Coprozessor/Emulator nicht benutzen, keinen blockierenden Taskwechsel erzwingen, usw. In einem Multitask-System sollte die Funktion nur eine Semaphore setzen, und damit eine Task mit hoher Priorität für die Fehlerbehandlung aktivieren).

11.6. Daten lesen und schreiben

Der Mikrokontroller des ND21 hat einen Arbeitspeicher (RAM), ein Programmspeicher (ROM), und einen elektrisch programmierbaren und löschbaren, nichtflüchtigen Speicher (EEPROM). Die Adressen sind in der Include-Datei "nd21.h" zu finden.

Funktionen für Lesen aus dem RAM oder ROM.NB_ReadByte( )

- NB_ReadWord( )
- NB_ReadLong()

Funktionen für Schreiben in den RAM:

- NB_WriteByte( )
- NB_WriteWord( )

Funktion für Lesen aus dem EEPROM:

- NB_ReadEEPROM( )

Funktionen für Schreiben in den EEPROM:

- NB_WriteEEPROM( )
- NB_RAMtoEEPROM( )

Bei einem Lesevorgang wird vom PC ein Telegramm geschickt, der Antwort von ND21 überden gesuchten Wert enthält. Die aufgerufene Funktion muß also auf die Antwort warten, die Wartezeit ist abhängig von der Teilnehmerzahl im Ring.

Falls ein Programm (oder eine Task) mehrere Daten nacheinander lesen möchte, muß bei jedem Funktionsaufruf gewartet werden. Für die bessere Organisation stehen die nächsten Funktionen zur Verfügung :

Funktionen für zeitoptimierte Kommunikation:

- NB_StartReadByte( )
- NB_StartReadWord( )
- NB_Test( )
- NB_Result( )

Die Funktionen NB_StartReadByte( ) und NB_StartReadWord() starten das Telegramm für den Lesevorgang. Als Ergebnis liefern sie eine Auftragsnummer zurück. Der Auftrag wird in einem internen Buffer gespeichert.

Mit der Funktion NB_Test( ) kann man jederzeit abfragen, ob das Ergebnis eines Auftrages schon zur Verfügung steht oder nicht.

Mit der Funktion NB_Result( ) kann man das Ergebnis holen. Man kann diese Funktion jederzeit aufrufen, also auch dann, wenn das Ergebnis noch nicht zur Verfügung steht. Falls dies der Fall ist, wartet die Funktion auf das Ergebnis. Der Aufruf von NB_Test( ) ist optional. Man darf NB_Result( ) für jeden Auftrag nur einmal aufrufen.

Die Funktion liefert das Ergebnis des Auftrags, gleichzeitig löscht sie ihn aus dem internen Buffer. Ein zweiter Aufruf findet die Eintragung nicht mehr, er verursacht Fehlermeldung ("Falsche Auftragsnummer").

Um den Buffer nicht überflüssig zu belasten, sollte jedes Ergebnis mit NB_Result( ) rechtzeitig abgeholt werden.

11.7. Novobus im Echtzeitsystem

Die Novobus-Funktionen senden verschiedene serielle Telegramme, die unterschiedlich lang sein können. Bei 38400 bit/s Übertragungsgeschwindigkeit dauert es 286 µs, ein Byte zu senden.

Um Wartezeiten zu vermeiden, werden die Daten in einem internen Buffer (FIFO) zwischengespeichert.

Wenn der Buffer voll ist, muß die Task, die eine Novobus-Funktion aufruft, warten. Bei einem Lesevorgang muß auch auf das Ergebnis gewartet werden.

In einem Echtzeitsystem werden die verschiedenen Tasks, die mit dem Bus arbeiten wollen, durch Semaphoren synchronisiert. Die meisten Funktionen haben einen optionalen Parameter *"Semaphore"*.

Wenn keine Semaphore angegeben wurde, wartet die Novobus-Funktion mit periodischer Abfrage (Polling). Das kostet aber Rechenzeit, in einem Echtzeitsystem sind die Tasks mit niedrigerer Priorität blockiert. Deshalb ist es vorzuziehen, die Synchronisation mit Semaphoren zu lösen.

Alle Lesefunktionen arbeiten intern mit der Funktion NB_Result(). Sie prüft zuerst, ob das Ergebnis des Lesevorganges zur Verfügung steht. Falls es nicht der Fall ist, wartet sie mit Hilfe der Semaphore in suspendiertem Zustand, also ohne Rechenzeit zu verschwenden.

Da mehrere Tasks gleichzeitig Novobus-Funktionen aufrufen können, müssen die Tasks, die mit Novobus arbeiten, eine eigene Semaphore zur Verfügung stellen. Es reicht aber eine Semaphore je Task.

Damit der Novobus-Treiber mit Semaphoren umgehen kann, braucht er zwei Funktionen, die vom Anwender zu definieren sind:

void NB_Signal(void* *Semaphore*);
Funktion: Setzt eine Semaphore.

void NB_Wait (void* *Semaphore*);
Funktion: Wartet auf eine Semaphore.

Beispiel: Mit RTKernel:

```
void near* semaphore = (void near*) RTKCreateSema(Binary,0);

extern void far cdecl  NB_Signal( void near* semaphore )

{

RTKSignal( (Semaphore) semaphore );

}

extern void far cdecl  NB_Wait( void near* semaphore )

{

RTKWait( (Semaphore) semaphore ) ;

}
```

11.8. Die Antriebe starten und stoppen

ND21 hat folgende mögliche Zustände :

Reset Nach dem Einschalten ist der Antrieb in *Reset*-Zustand. Der Antrieb initialisiert die Hardware und die Software, und führt einen umfangreichen Selbsttest durch. Mit der Funktion NB_ResetH8( ) kann man den Antrieb auch durch den Bus zurücksetzen.

Sperre Im *Reglersperre*-Zustand ist der Regler und die Endstufe gesperrt. Es fließt kein Strom, der Motor bringt keinen Drehmoment. Der Antrieb kann durch ein Hardware-Signal *(Hardware-Sperre)* oder durch den Bus *(Software-Sperre)* gesperrt werden. Im Novobus-Treiber steht dafür die Funktion NB_Sperre( ) zur Verfügung. Die *Software-Sperre* kann man nur durch den Bus, die *Hardware-Sperre* nur durch ein Hardware-Signal aufheben. Für die Freigabe müssen sowohl die *Hardware-* als auch die *Software-Sperre* aufgehoben werden.

Stop Im *Stop*-Zustand ist der Regler freigegeben, aber der Antrieb ist gestoppt. Unabhängig vom vorgegebenen Drehzahlsollwert bringt ND21 den Motor zum Stehen. Falls sich der Motor gerade dreht, bremst ihn ND21 mit der vorprogrammierten Bremsrampe. Der Antrieb kann durch ein Hardware-Signal *(Hardware-Stop)* oder durch den Bus *(Software-Stop)* gestoppt werden.

Im Novobus-Treiber steht dafür die Funktion NB_Stop( ) zur Verfügung. Den *Software-Stop* kann man nur durch den Bus, den *Hardware-Stop* nur durch ein Hardware-Signal aufheben. Für den Betriebszustand müssen sowohl der *Hardware-* als auch der *Software-Stop* aufgehoben werden.

Go Im Betriebzustand *(Go)* dreht sich der Antrieb nach dem Drehzahlsollwert entsprechend. Dazu müssen die Hardware-Signale "Freigabe" und "Start" aktiv sein. Die Funktion NB_Go( ) hebt die *Software-Sperre* und den *Software-Stop* auf, und startet den Antrieb.

Error Bei bestimmten Fehlern geht der Antrieb in Fehlerzustand *(Error)*. Er meldet durch die 7-Segment-Anzeige einen 3-stelligen Fehlercode. Er kann auch durch den Bus mit der Funktion NB_ReadWort( ) ausgelesen werden. Im Fehlerzustand ist der Regler gesperrt. Mit der Funktion NB_DeleteError( ) kann man den Fehler quittieren.

11.9. Die Positioniersteuerung bedienen

Der ND21 enthält eine integrierte Einachsen-Positioniersteuerung (PS). Die Positioniersteuerung kann sowohl vom Leitrechner (PC) durch den Novobus, als auch von einer externen Steuerung (SPS) durch Hardware-Signale gesteuert werden.

Der Servoumrichter ND21 arbeitet mit Resolverrückmeldung. Der Resolver liefert den mechanischen Rotorwinkel. Diesen Wert kann man elektronisch verschieben. Der Lageregler von ND21 arbeitet mit 4-Byte-Auflösung. Die unteren 2 Bytes zeigen den Winkelunterschied zum Nullpunkt, die oberen 2 Bytes die Entfernung in Umdrehungen. Falls der Motor nicht mit einem

Multiturn-Resolversystem aufgerüstet ist, weiß der Antrieb nach dem Einschalten nicht, wo sich der Nullpunkt befindet.

Initialisierung

Für die Initialisation der Positioniersteuerung stehen drei Funktionen zur Verfügung. Die Funktion NB_InitPS() initialisiert sie ohne den Lageistwert zu ändern. Die Funktion NB_IstwertSetzen( ) initialisiert die Positioniersteuerung und setzt den aktuellen Lageistwert auf den angegebenen Wert. Die Funktion NB_Referenzfahrt( ) initialisiert die Positioniersteuerung mit einer Referenzfahrt.

Positionierung

Für einen Positioniervorgang muß man zuerst die neue Zielposition mit der Funktion NB_WritePS( ) dem Antrieb mitteilen. Danach kann der Antrieb mit der Funktion NB_StartPS( ) gestartet werden.

12. Die Library Funktionen

12.1. Bus öffnen und schließen, Status abfragen

unsigned int InitNovobus1	(unsigned int	*COMAddress*,
	unsigned int	*COMIRQ*,
	unsigned char	*RingSize*,
	NBSEMAPHORE	*Semaphore*);

Funktion: Initialisiert und startet den Novobus mit einem Ring

Parameter:	*COMAddress:*	Basisadresse des Kommunikationsbausteins (üblicherweise: COM1 = 0x3f8, COM2 = 0x2f8)
	COMIRQ:	Interrupt-Request Belegung (Üblicherweise: COM1 = 4, COM2=3)
	RingSize:	Größe des Ringes, Zahl der Antriebe im Ring (1...250)
	Semaphore:	Für Task-Synchronisation in Multitasksystemen (optional)

Ergebnis: Ungleich 0 bedeutet Fehler bei der Initialisierung

unsigned int InitNovobus2	(unsigned int	*COMAddress1*,
	unsigned int	*COMIRQ1*,
	unsigned char	*RingSize1*,
	unsigned int	*COMAddress2*
	unsigned int	*COMIRQ2*
	unsigned char	*RingSize2*
	NBSEMAPHORE	*Semaphore*);

Funktion:	Initialisiert und startet den Novobus mit zwei Ringen
Parameter:	wie bei InitNovobus()
Ergebnis:	wie bei InitNovobus()

void CloseNovobus (NBSEMAPHORE *Semaphore)*

Funktion:	Schließt den Bus
Parameter:	*Semaphore,* für Task-Synchronisation in Multitasksystemen (optional).

unsigned int NB_Status (void)

Funktion:	Liefert Informationen über den Status des Buses
Ergebnis:	0 = inaktiv 1 = ein Ring ist aktiv 2 = zwei Ringe sind aktiv

unsigned int NB_BusError (void);

Funktion:	Liefert Informationen über Übetragungsfehler
Ergebnis:	Zahl der automatisch korrigierten Fehler

12.2. Daten schreiben

void NB_Write Byte	(unsigned int	*Axis*,
	unsigned int	*Address*,
	unsigned char	*Data*,
	NBSEMAPHORE	*Semaphore*);

Funktion: Schreibt ein Byte in den Arbeitsspeicher des ND21.

Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0xfd80-0xff7f: RAM
	Data:	Neuer Inhalt der Speicherzelle
	Semaphore:	Optional

void NB_WriteWord	(unsigned int	*Axis,*
	unsigned int	*Address*,
	unsigned int	*Data*,
	NBSEMAPHORE	*Semaphore)*

Funktion: Schreibt ein Wort in den Arbeitspeicher des ND21.

Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0xfd80-0xff7f: RAM
	Data:	Neuer Inhalt der Speicherzelle
	Semaphore:	Optional

12.3. Daten lesen

unsigned char NB_ReadByte (unsigned int *Axis*,
unsigned int *Address*,
NBSEMAPHORE *Semaphore*);

Funktion:	Liest ein Byte aus dem Arbeitspeicher des ND21.	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0x2f00-0x2f4f: ROM-Info 0xfd80-0xff7f: RAM 0xff80-0xffff: On-chip Registerfeld
	Semaphore:	Optional
Ergebnis:	Inhalt der Speicherzelle	

unsigned int NB_ReadWord (unsigned int *Axis,*
unsigned int *Address*,
NBSEMAPHORE *Semaphore)*

Funktion:	Liest ein Wort aus dem Arbeitspeicher des ND21.	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0x2f00-0x2f4f: ROM-Info 0xfd80-0xff7f: RAM
	Semaphore:	Optional
Ergebnis:	Inhalt der Speicherzelle.	

unsigned int NB_ReadLong	(unsigned int	*Axis*,
	unsigned int	*Address*,
	NBSEMAPHORE	*Semaphore*);

Funktion:	Liest vier Bytes aus dem Arbeitsspeicher des ND21.	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0x2f00-0x2f4f: ROM-Info 0xfd80-0xff7f: RAM
	Semaphore:	Optional
Ergebnis:	Inhalt der Speicherzelle.	

12.4. Funktionen für zeitoptimierte Kommunikation

NBORDER NB_StartReadByte	(unsigned int	*Axis*,
	unsigned int	*Address*,
	NBSEMAPHORE	*Semaphore*);

Funktion:	Startet einen Lesevorgang.	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0x2f00-0x2f4f: ROM-Info 0xfd80-0xff7f: RAM 0xff80-0xffff: On-chip register field
	Semaphore:	Optional
Ergebnis:	Auftragsnummer	

NBORDER NB_StartReadWord (unsigned int *Axis*,
unsigned int *Address*,
NBSEMAPHORE *Semaphore*);

Funktion:	Startet einen Lesevorgang.	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0x2f00-0x2f4f: ROM-Info 0xfd80-0xff7f: RAM
	Semaphore:	Optional
Ergebnis:	Auftragsnummer	

unsigned int NB_Test (NBORDER Auftrag)

Funktion:	Prüft, ob ein Auftrag erledigt wurde.	
Parameter:	*Auftrag:*	Auftragsnummer
Ergebnis:	0: Im Gange 1: Ergebnis steht zur Verfügung	

unsigned int NB_Result (NBORDER *Auftrag*,
NBSEMAPHORE *Semaphore*);

Funktion:	Liefert die Ergebnisse des Auftrags und löscht ihn.	
Parameter:	*Auftrag:*	Auftragsnummer
	Semaphore	Optional
Ergebnis:	Dem Auftrag entsprechend.	

12.5. Reading and Writing EEPROM

unsigned char NB_ReadEEPROM (unsigned int *Axis*,
unsigned char *Address*,
NBSEMAPHORE *Semaphore*);

Funktion:	Liest ein Byte aus dem nichflüchtigen Speicher	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0x00-0x7f: EEPROM
	Semaphore:	Optional
Ergebnis:	Inhalt der Speicherzelle	

void NB_WriteEEPROM (unsigned int *Axis*,
unsigned char *Address*,
NBSEMAPHORE *Semaphore*);

Funktion:	Liest ein Byte aus dem nichflüchtigen Speicher	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0x00-0x7f: EEPROM
	Semaphore:	Optional
Ergebnis:	Inhalt der Speicherzelle	

void NB_ RAMtoEEPROM (unsigned int *Axis*,
NBSEMAPHORE *Semaphore*);

Funktion:	Kopiert die Reglerparameter aus dem RAM in EEPROM	
RAM-Bereich	0xff60-0xff7f (32 Byte)	
EEPROM-range	0x20 - 0x3f (32 Byte)	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Address:	Adresse der Speicherzelle 0x00-0x7f: EEPROM
	Semaphore:	Optional

12.6. Start / Stop ND21

void NB_Go (unsigned int *Axis*,
NBSEMAPHORE *Semaphore*);

Funktion:	Startet ND21 (inaktiviert Software-Sperre und Software-Stop)	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Semaphore:	Optional

void NB_Stop

Funktion:	Stopt ND21 (Software-Stop).	
Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Semaphore:	Optional

void NB_Sperre (unsigned int *Axis*,
NBSEMAPHORE *Semaphore*);

Funktion: Sperrt den Regler und die Endstufe (Software-Sperre)

Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Semaphore:	Optional

12.7. Reset ND21, Fehler löschen

void NB_DeleteError (unsigned int *Axis*,
NBSEMAPHORE *Semaphore*);

Funktion: Quittiert die Fehlermeldung des ND21

Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Semaphore:	Optional

void NB_ResetH8 (unsigned int *Axis*,
NBSEMAPHORE *Semaphore*);

Funktion: Setzt den Mikrocontroller (H8) des ND21 und damit den Antrieb selbst zruück (Software-Reset)

Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Semaphore:	Optional

12.8. Digitalausgänge setzen

void NB_IO (unsigned int *Axis*,
unsigned int *Code,*
NBSEMAPHORE *Semaphore*);

Funktion: Setzt GP01 oder GP02 auf 1 oder 0

Parameter:	*Axis:*	Adresse des Antriebes Loop 1: 0x0000-0x00f9 Loop 2: 0x0100-0x01f9
	Code	Set GPO1 H' 00 Set GPO2 H' 02 Reset GPO2 H' 03
	Semaphore:	Optional

13. Positioniersteuerung

13.1. Die ND21 - Positioniersteuerung

ACHTUNG! **Die Positioniersteuerung mit SPS-Ankopplung ist nur auf Anfrage verfügbar.**

Der ND21 enthält ein Software-Modul für Einachsen-Positioniersteuerung (PS) als Option. Die Positioniersteuerung wurde ohne Zusatzhardware realisiert.

Die Positioniersteuerung von ND21 arbeitet mit trapezförmiger Geschwindigkeitskurve. Beschleunigung (Rampe) und Fahrgeschwindigkeit sind programmierbar. (Der Fahrweg ist nicht begrenzt, aber die Fahrt eines Positioniervorganges darf bei PS1.0 nicht länger als 26 Sekunden dauern).

Die Positioniersteuerung kann sowohl von einem Leitrechner (z.B. PC) durch den NOVOBUS, als auch von einer externen Steuerung (SPS) durch Hardware-Signale gesteuert werden.

Falls ein Leitrechner im System vorhanden ist, kann er durch sein serielles Interface (RS232 oder RS485) die neuen Zielpositionen übertragen **(siehe Kapitel 11 Der NOVOBUS-Treiber)**. Der Positioniervorgang selbst kann sowohl vom Leitrechner als auch von der SPS gestartet werden.

Der ND21 kann auch ohne Leitrechner positionieren. Dazu enthält er eine interne Ablaufsteuerung (AS). Bis zu 40 Positionen können in ND21 gespeichert werden. Die Programmierung kann, bei der Inbetriebnahme, mit einem PC/Laptop erfolgen. Die SPS kann die Ablaufsteuerung auch on-line programmieren. Sie kann die neue Zielposition entweder bitseriell übertragen, oder durch Teach-In-Verfahren festlegen.

13.1.1. Die Referenzfahrt

Der Servoumrichter ND21 arbeitet mit Resolverrückmeldung. Der Resolver liefert den mechanischen Rotorwinkel. Falls der Motor nicht mit einem Multiturn-Resolversystem ausgerüstet ist, kennt der ND21 nach dem Einschalten nur den Motorwinkel, nicht aber die volle Istposition. Bei einer Referenzfahrt wird ein Referenzschalter gesucht. Da die Lage des Referenzschalters bekannt ist, weiß ND21 nach der Referenzfahrt die absolute Position des Motors.

Ein PC kann die Referenzfahrt durch die NOVOBUS-Funktion *NB_Referenzfahrt( )* durchführen lassen. Falls der Rechner die absolute Istposition auf eine andere Weise feststellen kann (z.B. Blockfahren), kann er die Lageinformation direkt übertragen und die Positioniersteuerung ohne Referenzfahrt initialisieren. Dazu steht die Funktion *NB_IstwertSetzen( )* zur Verfügung.

Die SPS kann auch ohne einen Leitrechner die Referenzfahrt starten. Dazu muß er mit den vier (fünf, falls GPIN8 verwendet wird) Mode-Schaltern die Funktion *"Referenzfahrt"* auswählen und sie mit dem "Start"-Signal starten.

In der Referenzposition muß der Referenzschalter High-Pegel liefern (GPIN4).

Die einzelnen Schritte:

1. Fahren mit der vorprogrammierten Richtung und Drehzahl (*RefV1*) solange GPIN4=Low ist.

2. Wenn GPIN4=High, bremsen mit der vorprogrammierten Rampe (*nRampe-*).

3. Langsameres Fahren mit der vorprogrammierten Geschwindigkeit in die entgegengesetzte Richtung (*RefV2*) solange, GPIN4=High.

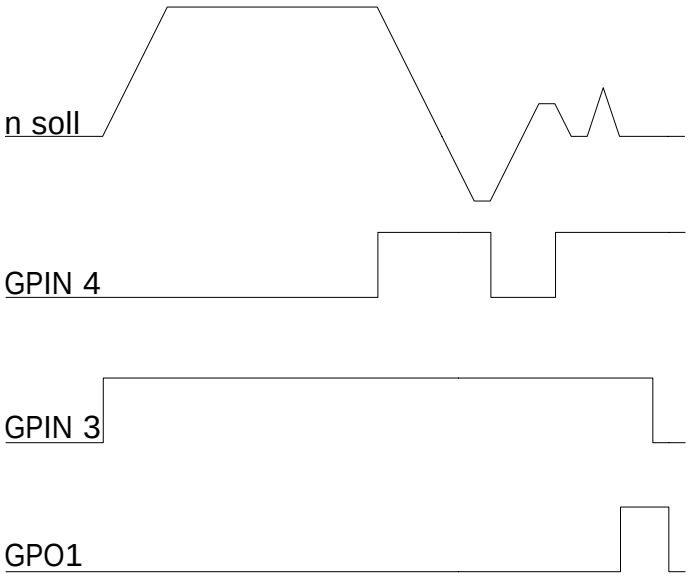
4. Wenn erneut GPIN4=Low, bremsen.

5. Langsames Fahren in die erste Richtung bis wieder GPIN4=High.

6. Positionieren auf die nächste Position, bei der der Rotorwinkel 180° ist (Lageistwert = H'8000). Diese Position läßt sich mit dem Parameter *RefLage* (H' FD9E) verschieben. Um Reproduzierbarkeit der Referenzfahrt zu gewährleisten, muß der Referenzschalter möglichst dicht an dieser Position schalten.

7. Lageistwert setzen. Dem Referenzpunkt entspricht der vorprogrammierte 4 Byte Lageistwert (*RefUmdr0, RefLage0*).

8. Falls die Ablaufsteuerung aktiv ist meldet ND21 mit GPO1=High, daß die Referenzfahrt abgeschlossen ist.



13.1.2. Der Positioniervorgang

Ein Positioniervorgang beginnt damit, daß ND21 eine neue Zielposition bekommt. Diese kann entweder über NOVOBUS oder bitseriell von der SPS, übertragen, oder aus dem ND21 EEPROM ausgelesen werden.

Sobald die neue Zielposition eingetragen ist, kann die ND21 Positionsrechnung gestartet werden. Die Rechnungen können bei PS1.0 nur im Stillstand und bei freigegebenem Reglerzustand durchgeführt werden.

Wenn ND21 mit der internen Rechnungen fertig ist, kann der Positioniervorgang gestartet werden.

Sobald die Positionierung abgeschlossen ist, kann dies mit Digitalsignalen der SPS gemeldet und über NOVOBUS abgefragt werden.

Der Digitalausgang GPO2 kann für eine "In Position"-Meldung programmiert werden.

Die Fahrtgeschwindigkeit und die Rampen sind programmierbar.

In dem Zustand Reglersperre hat der Antrieb kein Haltemoment, der Motor kann mit fremder Kraft bewegt werden. Da damit die Anfangsposition nicht mehr stimmt, löscht die Reglersperre die Rechnungsergebnisse der Positioniersteuerung.

Zum Positionieren muß also die Rechnung neu gestartet werden. Falls die Rechnung im Zustand Reglersperre gestartet wird, wartet der Antrieb solange, bis der Regler wieder freigegeben wird, erst dann führt er die Rechnungen durch.

Der gestartete Positioniervorgang kann mit Stop oder mit Reglersperre gestoppt werden. Bei Stop bremst der Antrieb mit der vorprogrammierter Rampe, bei Reglersperre wird elektrisch nicht gebremst. Sowohl der Leitrechner als auch die SPS kann Stop oder Reglersperre auslösen.

Die SPS macht es durch die ND21-Eingänge:

GPIN3=Low (*Stop*) und

GPIN5=Low (*Reglersperre*).

13.1.3. Konfigurationen

Es sind verschiedene Steuerungskonzepte mit ND21 denkbar:

1.: ND21 und SPS

2.: ND21 und Leitrechner (PC)

3.: ND21 und Leitrechner (PC) und SPS

13.1.3.1. ND21 und SPS

In diesem Fall ist eine SPS vorhanden, die den Maschinenablauf steuert. Dazu gehört auch die Kontrolle über die ND21 Positioniersteuerung. Die SPS kann über Digitalsignale auf einen Satz von 23 Funktionen zurückgreifen, mit denen sie:

- Positionierungen starten,
- Zielpositionen auswählen,
- Tippbetrieb fahren,
- Referenzfahrt starten,
- Zielpositionen übertragen,
- Teach In durchführen,
- Fehler auslesen und quittieren,
- Istposition zurücklesen,

- Positioniergeschwindigkeit programmieren,
- Tippgeschwindigkeit programmieren,
- Rampe programmieren,
- Stromgrenzen programmieren,
- auf Analogsollwert umschalten

kann.

13.1.3.2. ND21 und Leitrechner

In diesem Fall müssen bei ND21 nur wenige Digitalsignale (Start, Freigabe, Endschalter, Referenzschalter) angeschlossen sein.

Die ND21 Positioniersteuerung wird über NOVOBUS vom Leitrechner gesteuert. Dazu stehen spezielle Befehle im NOVOBUS Treiber zur Verfügung.

13.1.3.3. ND21 und Leitrechner und SPS

Diese Konfiguration findet man bei Maschinen, die einer schnellen Echtzeitsteuerung mit SPS bedürfen, aber gleichzeitig einen hohen Aufwand an Datenverarbeitung, z. B. für Visualisierung, haben.

In diesem Fall kann der Leitrechner die neue Zielposition ermitteln, über NOVOBUS an ND21 übertragen und die Positionierrechnung starten.

Die SPS startet die Positionierung. Über den Befehl *"WaitPS*" kann eine Synchronisierung zwischen SPS und Leitrechner erfolgen.

Die SPS wählt diese Funktion aus und startet sie mit dem "Start"-Signal (GPIN3). Der ND21 meldet mit dem digitalen Ausgangsignal GPO1 = 1 zurück, wenn er die neue Zielposition vom Rechner erhalten und sie aufgearbeitet hat. Danach kann die SPS mit der Funktion *"StartPS"* den Positioniervorgang starten.

13.1.4. Programmierung von ND21

Bevor mit der Positioniersteuerung gearbeitet werden kann, müssen natürlich der Stromregler, Drehzahlregler und der Lageregler richtig eingestellt sein. Die Rampen, die Strom- und Drehzahlgrenzen müssen bestimmt und eingegeben werden **(siehe Handbuch Inbetriebnahme und Parameter-einstellungen von ND21).**

Damit man mit der Ablaufsteuerung arbeiten kann, muß sie eingeschaltet werden (Bit 3 in *HwVersion*=1). Die Positioniersteuerung kann nur dann richtig arbeiten, wenn die überlagerte Lageregelung aktiv ist (Bit 6 in *SwVersion*=1), der Sollwert als Digital-Sollwert programmiert ist (Bit1 und Bit 0 in *SwVersion*=1). Den Ausgang GPO1 sollte man als Digitalausgang programmieren (Bit 6 in *HwVersion*=1).

Der Ausgang GPO2 kann beliebig programmiert werden:

HwVersion Bit 6	Bit 5	GPO2
0	0	Durch den Bus programmierbar
0	1	Strombegrenzung (Motor blockiert)
1	0	In-Position (PEH)
1	1	Motor steht

In Position: Mit dem Parameter *Window* (H' FFB7) wird das In Positionsfenster eingestellt. Die "In Position"-Meldung wird auf GPO1 geschaltet, wenn die Istposition von der Zielposition um weniger als (4 x *Window*) abweicht.

Betriebsart Tippen:

Die Geschwindigkeit ist mit *Vtippen* (2 Byte, Speicheradresse H' FD94) festgelegt. Nach dem Einschalten initialisiert ND21 den Wert aus dem EEPROM.

EEPROM-Adresse: *EE_Vtippen* (H' 50, H' 51).

Falls Bit 0 in *SwVersion2* gleich 0 ist, wird die Stromgrenze (maximales Drehmoment) mit dem Analogeingang vorgegeben.

Betriebsart Referenzfahrt

Die Refenzfahrt-Geschwindigkeiten sind programmierbar:

1. Geschwindigkeit:

RefV1 (2 Byte, H' FD8C)/*EE_RefV1* (H' 48, H' 49)

2. Geschwindigkeit:

RefV2 (2 Byte, H' FD8E)/*EE_RefV2* (H' 4A, H' 4B)

Die Geschwindigkeiten müssen mit Vorzeichen programmiert werden (2's Komplement), und in die entgegengesetzte Richtung zeigen (unterschiedliches Vorzeichen).

Nach der Referenzfahrt wird der Istwert gesetzt:

RefUmdr0 (2 Byte, H' FD90)/*EE_RefUmdr0* (H' 4C, H' 4D)

RefLage0 (2 Byte, H' FD92)/*EE_RefLage0* (H' 4E, H' 4F)

Die Motor-Nullposition kann mit dem Wert

RefLage (2 Byte, H' FD9E)/*EE_RefLage* (H' 54, H' 55)

verschoben werden.

Betriebsart Positionierung

Die Positioniergeschwindigkeit ist programmierbar:

ps_v0 (2 Byte, H' FF4C)

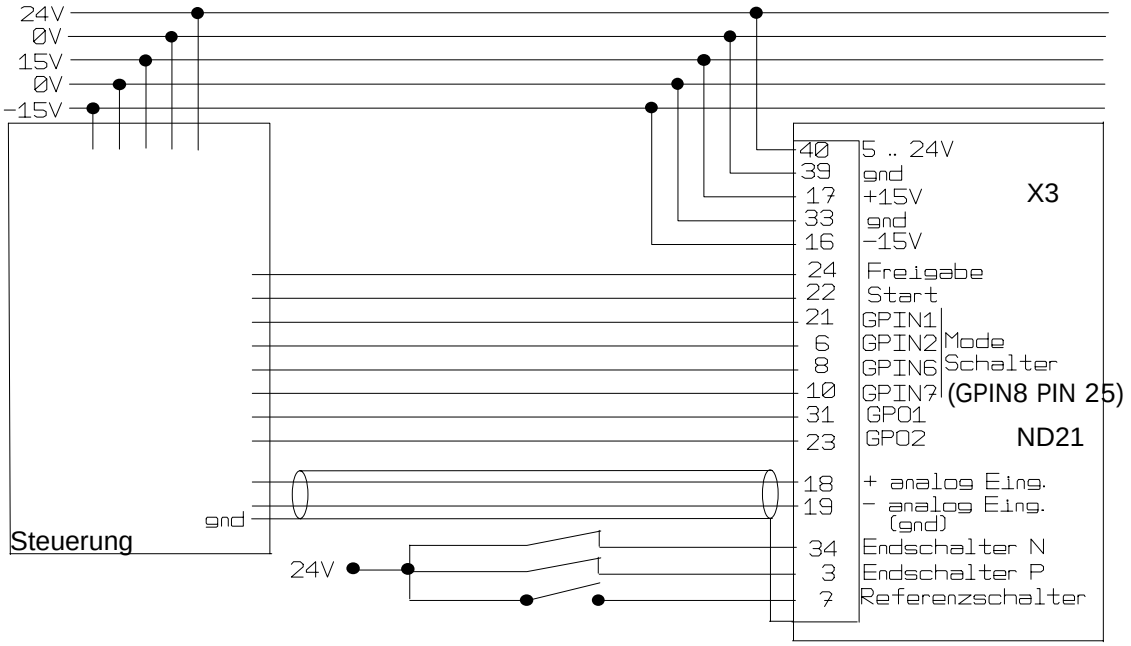
Diese Geschwindigkeit wird mit der maximale Geschwindigkeit initialisiert.

nMax (2 Byte, H' FF68)

13.1.5. Steuerung durch die SPS (Ablaufsteuerung (AS 1.0))

ACHTUNG! **Diese Funktion ist nur auf Anfrage verfügbar.**

13.1.5.1. Anschlußdiagramm



Ansteuerung ND21 Ablaufsteuerung

Hinweis: Bei GPO1 und GPO2 handelt es sich um Open Kollektor Ausgänge mit 6.8 kOHM internem Pull-up Widerstand. Eventuell sind in der Steuerung geeignete weitere Pull-up Widerstände vorzusehen.

Hinweis: Alle Verbindungen sollten möglichst kurz sein, das Kabel zum Analog Eingang sollte abgeschirmt sein.

Hinweis: Die 24V und die +- 15V Versorgung müssen gut geerdet sein.

Hinweis: Der Analogeingang und die +/- 15 V Versorgung werden nur dann benötigt, wenn beim Tippen eine Drehmomentbegrenzung mit einem Analogsignal erfolgen soll.

13.1.6. Funktionsweise

In ND21 ist eine Ablaufsteuerung integriert. Sie ermöglicht es auf dem ND21 bis zu 40 Positionen zu speichern. Diese 40 Positionen können mit der ND21 Positioniersteuerung angefahren werden. Die Positionen können in einem Teach-In Verfahren ermittelt werden oder bitseriell von der SPS übertragen werden. Es gibt Funktionen für: Handbetrieb (Tippen), Referenzfahrt, Positionsauswahl, Starten der Positionierung, usw.

Eine Funktion wird immer so ausgewählt, daß an den Signalen GPIN (8),7,6,2,1 der Funktionscode und anschießend am Start-Eingang (GPIN3) eine steigende Flanke, angelegt wird. Mit GPO1=Low meldet die Ablaufsteuerung, daß sie bereit ist, eine neue Funktion durchzuführen. GPO1 wird High, wenn die Funktion durchgeführt wurde.

Der Start-Eingang (GPIN3) muß solange High sein bis GPO1 mit High den Abschluß der Funktion anzeigt. Vorzeitiges Low des Start-Einganges (GPIN3) bedeutet den Abbruch der Funktion.

Beispiel Referenzfahrt:

1. Anlegen des Funktionscodes für Referenzfahrt:

   GPIN1 = High

   GPIN2 = High

   GPIN6 = Low

   GPIN7 = Low

   (GPIN8 = Low, wenn verwendet)

2. Steigende Flanke Low - High am Start-Eingang GPIN3.

3. ND21 führt die Referenzfahrt durch. Die SPS hält den Start-Eingang GPIN3 solange High bis GPO1 High wird.

4. ND21 meldet mit GPO1 = High, daß die Referenzfahrt abgeschlossen ist.

5. Die SPS schaltet den Start-Eingang (GPIN3) auf Low, damit wird GPO1 auch wieder Low und zeigt damit der SPS an, daß ND21 bereit ist weitere Funktionen auszuführen.

13.1.7. Signale und Funktionen

13.1.7.1. Eingänge

GPIN8 (X3, 25): zusätzlicher Mode-Schalter

GPIN7 (X3.10): Mode-Schalter

GPIN6 (X3.8): Mode-Schalter

GPIN2 (X3.6): Mode-Schalter

GPIN1 (X3.21): Mode-Schalter

GPIN3 (X3.22): Start-Signal

GPIN4 (X3.7): Referenzschalter

GPIN5 (X3.24): Regler-Freigabe
Low: Regelsperre (Endstufe gesperrt)
High: Freigabe

13.1.7.2. Ausgänge

GPO1 (X3.31): Auftrag durchgeführt (Quittung)

GPO2 (X3.23): programmierbar:

- Strombegrenzung (Motor blockiert)
- In Position
- Motor steht

13.1.7.3. Mode-Schalter

Mit den Mode-Schaltern wird die gewünschte Funktion ausgewählt.

(GPIN8= Low, wenn verwendet).

GPIN7	GPIN6	GPIN2	GPIN1	Funktion:
Low	*Low*	*Low*	*Low*	StartPS
Low	*Low*	*Low*	*High*	Tippen-
Low	*Low*	*High*	*Low*	Tippen+
Low	*Low*	*High*	*High*	Referenzfahrt
Low	*High*	*Low*	*Low*	RdEEWrPS
Low	*High*	*Low*	*High*	ClrPZ
Low	*High*	*High*	*Low*	IncPZ
Low	*High*	*High*	*High*	DecPZ
High	*Low*	*Low*	*Low*	Teach In
High	*Low*	*Low*	*High*	WrEE
High	*Low*	*High*	*Low*	Wr0
High	*Low*	*High*	*High*	Wr1
High	*High*	*Low*	*Low*	WaitPS
High	*High*	*Low*	*High*	RdError
High	*High*	*High*	*Low*	RdPos
High	*High*	*High*	*High*	WrPS

13.1.7.4. SPS-Funktionen

ClrPZ Setzt den Positionszeiger auf Null (PZ = 0)

IncPZ Erhöht den PZ um eins (PZ++)

DecPZ Erniedrigt den PZ um eins (PZ--)

Wr0, Wr1 Mit *Wr0* bzw. *Wr1* kann eine neue Zielposition (32 Bit) bitweise gesendet werdet.(Erste Sendung: MSB, letzte: LSB). Während der Übertragung darf keine andere SPS-Funktion durchgeführt werden. Die Zielposition kann entweder in die Positioniersteuerung geladen werden (*WrPS*) oder im EEPROM gespeichert werden (*WrEE*).

WrPS Mit dieser Funktion kann man die bitweise übertragene Position in die Positioniersteuerung (PS) laden.

WrEE Mit dieser Funktion kann man die bitweise vorgegebene Position im EEPROM speichern. Danach wird PZ automatisch inkrementiert.

RdEEWrPS Liest die aktuelle Zielposition aus dem Speicher (EEPROM),ladet sie in die Positioniersteuerung (PS), startet die Rechnung, und erhöht den Positionszeiger (PZ) um eins (auto-increment).

Anmerkung: Wenn der Regler gesperrt ist, wartet der Antrieb auf die Freigabe, und erst danach startet er die Rechnung.

StartPS Startet den Positioniervorgang. Die Funktion wird nur dann durchgeführt, wenn vorher die neue Zielposition mit *WrPS* oder *RdEEWrPS* in die Positioniersteuerung (PS) geladen wurde.

Anmerkung: Mit GPIN3=Low (Stop) oder GPIN5=Low (Sperre) kann der Positioniervorgang gestoppt werden.

Tippen+, Tippen- Handbetrieb startet den Motor mit einer vorprogrammierten Geschwindigkeit (Vtippen). Mit GPO1=High meldet ND21, daß er gestartet wurde. Mit GPIN3=Low (Stop) kann man den Motor stoppen.

Referenzfahrt Mit der vorprogrammierten Richtung und Geschwindigkeit sucht ND21 den Referenzpunkt. Der Referenzschalter (Schließer) muß an den Eingang GPIN4 angeschlossen sein.

Anmerkung: Mit GPIN3=Low (Stop) oder GPIN5=Low (Sperre) kann die Referenzfahrt vorzeitig gestoppt werden.

Teach in Die Ist-Position des Antriebs wird im EEPROM auf die aktuelle Speicherzelle (nach PZ) gespeichert, danach wird der Positionszeiger (PZ) inkrementiert.

RdPos Mit dieser Funktion kann die Position des Antriebes (Istwert) bitweise ausgelesen werden. (32 bit, LSB zuerst).

Wird dies nicht benötigt, so kann über RdPos auch auf Analogsollwerte umgeschaltet werden. Dazu muß Bit 4 in HwVersion2 auf 0 gesetzt sein.

RdError Mit dieser Funktion kann der 12-Bit Errorcode ausgelesen werden. Die übertragenen Bits erscheinen an GPO1. Im fehlerfreien Betriebszustand enthält das Wort den Wert Null. Mit dem 13ten Lesebefehl kann eine Fehlermeldung quittiert werden.

WaitPS Diese Funktion dient der Synchronisation von Leitrechner (falls vorhanden) und Steuerung (SPS). Falls der Leitrechner durch den Bus (NOVOBUS) die neue Zielposition sendet, der Positioniervorgang aber von der SPS gestartet wird, soll die SPS ein *WaitPS* ausgeben. Mit GPO1=High meldet ND21, daß er die neue Zielposition erhalten und die Rechnung beendet hat.

13.1.7.5. Zusätzlicher Mode-Schalter GPIN8

Der bereits bestehende Befehlsumfang der Ablaufsteuerung wurde um 8 zusätzliche Funktionen erweitert. Zur Ansteuerung wird der GPIN8 (X3 Pin 25) verwendet. Um GPIN8 als zusätzlichen Mode-Schalter nutzen zu können, müssen das Bit 1 im Parameter SwVersion2 und das Bit 7 in HwVersion auf 0 gesetzt werden. Erst danach stehen die Funktionen zur Verfügung.

Funktionsweise Es werden immer 16 Bit Parameter an ND 21 geschickt. Die einzelnen Bits werden mit den Funktionen Wr0 Param und Wr1 Param bitseriell an das ND 21 in den Parameterpuffer geschrieben. Das höchstwertige Bit muß dabei zuerst geschickt werden.

Mit den Funktionen Wr nMax, Wr Vtippen, Wr nRampe und Wr i kann dann der Parameterpuffer auf die gewünschte Speicherzelle kopiert werden.

GPIN8	GPIN7	GPIN6	GPIN2	GPIN1	Funktion
High	*Low*	*High*	*High*	*High*	*Wr0 Param*
High	*Low*	*Low*	*High*	*High*	*Wr1 Param*
High	*High*	*High*	*High*	*Low*	*Wr nMax*
High	*High*	*Low*	*High*	*Low*	*Wr Vtippen*
High	*Low*	*High*	*High*	*Low*	*Wr nRampe*
High	*Low*	*Low*	*High*	*Low*	*Wr i*

High	*High*	*High*	*High*	*High*	*InitPS*
High	*High*	*Low*	*High*	*High*	*Leitungstest*

Wr0 Param Ein 0 Bit wird in den Parameterpuffer eingetragen.

Wr1 Param Ein 1 Bit wird in den Parameterpuffer eingetragen.

Wr nMax Kopiert den Parameterpuffer auf nMax. nMax ist die Geschwindigkeit, mit der die Positionierung erfolgt.

Wr Vtippen Kopiert den Parameterpuffer auf Vtippen. Vtippen ist die Geschwindigkeit, mit der im Tipp-Betrieb verfahren wird.

Wr nRampe Kopiert den Parameterpuffer auf nRampe- (Bremsrampe) und nRampe+ (Beschleunigungsrampe).

Wr i Kopiert die ersten 8 Bit des Parameterpuffers auf imax und die zweiten 8 Bit auf i0. Dabei ist imax der Grenzstrom und i0 der Motornennstrom.

Init PS Initialisiert die Positioniersteuerung nach dem Einschalten.

Leistungstest Erzeugt die Fehlermeldung 0x602. Diese Fehlermeldung kann dann mit der Funktion Rderror bitseriell zürückgelesen werden.

13.1.8. Aktivieren der Ablaufsteuerung

Bit 3 im Byte *HwVersion* (H' FF60) auf 1 setzen.

13.1.9. Positionsdatenverwaltung

Im nichtflüchtigen Speicher von ND21 (EEPROM) kann man bis zu 40 Positionen ablegen. Die 4 Byte Positionen werden ab EEPROM-Adresse H' 60 gespeichert:

PZ	1. Byte	2.Byte	3. Byte	4. Byte
0	H' 60	H' 61	H' 62	H' 63
1	H' 64	H' 65	H' 66	H' 67
2	H' 68	H' 69	H' 6A	H' 6B
3	H' 6C	H' 6D	H' 6E	H' 6F
4	H' 70	H' 71	H' 72	H' 73

5	H' 74	H' 75	H' 76	H' 77
6	H' 78	H' 79	H' 7A	H' 7B
7	H' 7C	H' 7D	H' 7E	H' 7F
8	H' 80	H' 81	H' 82	H' 83
9	H' 84	H' 85	H' 86	H' 87
10	H' 88	H' 89	H' 8A	H' 8B
11	H' 8C	H' 8D	H' 8E	H' 8F
12	H' 90	H' 91	H' 92	H' 93
13	H' 94	H' 95	H' 96	H' 97
14	H' 98	H' 99	H' 9A	H' 9B
15	H' 9C	H' 9D	H' 9E	H' 9F
16	H' A0	H' A1	H' A2	H' A3
17	H' A4	H' A5	H' A6	H' A7
18	H' A8	H' A9	H' AA	H' AB
19	H' AC	H' AD	H' AE	H' AF
20	H' B0	H' B1	H' B2	H' B3
21	H' B4	H' B5	H' B6	H' B7
22	H' B8	H' B9	H' BA	H' BB
23	H' BC	H' BD	H' BE	H' BF
24	H' C0	H' C1	H' C2	H' C3
25	H' C4	H' C5	H' C6	H' C7
26	H' C8	H' C9	H' CA	H' CB
27	H' CC	H' CD	H' CE	H' CF
28	H' D0	H' D1	H' D2	H' D3
29	H' D4	H' D5	H' D6	H' D7
30	H' D8	H' D9	H' DA	H' DB
31	H' DC	H' DD	H' DE	H' DF

32	H' E0	H' E1	H' E2	H' E3
33	H' E4	H' E5	H' E6	H' E7
34	H' E8	H' E9	H' EA	H' EB
35	H' EC	H' ED	H' EE	H' EF
36	H' D0	H' D1	H' D2	H' D3
37	H' D4	H' D5	H' D6	H' D7
38	H' D8	H' D9	H' DA	H' DB
39	H' DC	H' DD	H' DE	H' DF

Die einzelnen Positionen können bei der Inbetriebnahme mit einem PC (Laptop) programmiert werden. Die SPS kann die Positionen bitweise oder durch Teach-In-Verfahren programmieren.

Jede Position ist in 4 Bytes (32 Bits = 1 Mode Bit + 31 Bit Position) angegeben:

Bit31 (MSB)	Bit28	1. Byte
Bit27	Bit 16	2. Byte
Bit15	Bit 8	3. Byte
Bit7	Bit 0 (LSB)	4. Byte

Bit 31 bestimmt ob es sich um relative oder absolute Positionen handelt. Wenn Bit 31=0, sind die Positionen absolut gemeint (Entfernung vom Nullpunkt). Wenn Bit 31=1, sind die Positionen relativ angegeben (Fahrweg). Beim Teach-In-Verfahren werden immer absolute Positionen gespeichert.

Bit 30...Bit 16 enthalten die ganzen Umdrehungen

(-16384...+16383), Bit 15...Bit0 enthalten den Winkel in einer Umdrehung. ND21 arbeitet mit 16 Bit Winkelauflösung, es entspricht 65536 Impulse/Umdrehung.

Bit 30...Bit 0 enthalten die Position in Zweierkomplement-Darstellung. Das Zweierkomplement muß aus den 31 Bit insgesamt gebildet werden. Eine byteweise oder wortweise Zweierkomplementbildung führt zu falschen Ergebnissen.

Beispiele

1. Zielposition: 10,25 Umdrehungen vom Nullpunkt in positiver Richtung.

absolut:	Bit31 = 0
10 ganze Umdrehungen:	dezimal 10 = H' 000A
Bits 30 - 16:	B' 000 0000 0000 1010
0,25 Umdrehung (90°):	0.25 · 65536
	= dezimal 16384
	= H' 4000
Bits 15 - 0:	B' 0100 0000 0000 0000
In hexadezimaler Form:	1. Byte = H' 00
	2. Byte = H' 0A,
	3. Byte = H' 40
	4. Byte = H' 00
zusammengefaßt:	H' 000A 4000

2. Zielposition: -10,25 Umdrehungen vom Nullpunkt.

absolut:	Bit31 = 0
31 Bit-Zweierkomplement von	H' 000A 4000 = H' 7FF5 C000
1. Byte = H' 7F	
2. Byte = H' F5,	
3. Byte = H'C0,	
4. Byte = H' 00	

3. Fahrweg: 50°

relative Angabe:	Bit 31 = 1,
Keine ganzen Umdrehungen (0)	Bits 30 - 16 = H' 0000,
50 Grad entspricht:	$65536 \cdot \frac{50}{360}$
= dezimal 9102	Bits 15 - 0 = H' 238E.
1. Byte = H' 80	
2. Byte = H' 00.	
3. Byte = H' 23	
4. Byte = H' 8E,	
zusammengefaßt:	H' 8000 238E

4. Fahrweg: -50°

relative Angabe:	Bit31 = 1
Zweierkomplement vom	
H' 0000 238E in 31 Bit	H' 7FFF DC72,
Ergebnis = H' FFFF DC72.	

5. Weitere Beispiele:

Ziel = 6,5 Umdr. vom Nullpunkt:	H' 0006 8000
Ziel = -6,5 Umdr. vom Nullpunkt:	H' 7FF9 8000
Ziel = 100 Umdr. vom Nullpunkt:	H' 0064 0000
Ziel = -100 Umdr. vom Nullpunkt:	H' 7F9C 0000
Fahrweg = 6,5 Umdrehungen:	H' 8006 8000
Fahrweg = -6,5 Umdrehungen:	H' FFF9 8000
Fahrweg = 100 Umdrehungen:	H' 8064 0000
Fahrweg = -100 Umdrehungen:	H' FF9C 0000

13.1.9.1. Positionszeiger (PZ)

Der Positionszeiger (PZ) zeigt auf die aktuelle Zielposition (0 <= PZ < 40).

Die SPS kann den Zeiger mit den Funktionen *"ClrPZ"* (zurücksetzen), *"IncPZ"* (inkrementieren) und *"DecPZ"* (dekrementieren) modifizieren.

Der Befehl *"RdEEWrPS*" inkrementiert den Positionszeiger automatisch.

13.1.9.2. Positionsdaten mit der SPS programmieren

Dazu muß zuerst der Positionszeiger auf die gewünschte Speicherstelle im EEPROM gestellt werden.

Die Zielposition kann bitseriell mit den Befehlen "*Wr0*" und "*Wr1*" übertragen werden. Angefangen mit Bit 31 überträgt "*Wr0*" ein 0 Bit und "*Wr1*" ein 1 Bit. Nach 32 Befehlen ist die gesamte Position übertragen. Sie kann mit dem Befehl "*WrEE*" ins EEPROM gespeichert werden. Es ist auch möglich mit "*WrPS*" diese Position direkt der Positionsrechnung zu zu führen. Die Übertragung der Zielposition darf nicht durch andere Befehle unterbrochen werden.

13.1.9.3. Positionsdaten mit "*Teach In*" programmieren

Dazu muß zuerst der Positionszeiger auf die gewünschte Speicherstelle im EEPROM gestellt werden.

Mit "*Tippen+*" und "*Tippen-*" den Antrieb in die gewünschte Position bringen. Mit "*Teach In*" diese Position in die Stelle im EEPROM übertragen, auf die der Positionszeiger zeigt.

13.1.9.4. Positionen mit dem PC programmieren

Im Servicemenü, Untermenü EEPROM mit der Funktion Write Byte die Zielposition direkt eintragen, wie oben berechnet.

13.1.9.5. Istposition auslesen

Mit dem Befehl "*RdPos*" kann bitseriell die 32 Bit Istposition ausgelesen werden. Jeder Befehl "*RdPos*" überträgt ein Bit auf GPO1. Bit 0 wird zuerst übertragen. Nach 32 Befehlen ist die Übertragung komplett. Die Übertragung des Istwertes darf nicht durch andere Befehle unterbrochen werden.

Wird dies nicht benötigt, so kann über RdPos auch auf Analogsollwerte umgeschaltet werden. Dazu muß Bit 4 in HwVersion2 auf 0 gesetzt sein.

13.1.10. Referenzfahrt

Die Referenzfahrt wird mit dem Befehl "*Referenzfahrt*" gestartet. ND21 meldet mit GPO1 = 1 zurück, wenn die Referenzfahrt abgeschlossen ist.

13.1.11. Positionieren

Den Positionszeiger auf die gewünschte Position stellen. Mit *RdEEWrPS* die Position aus dem EEPROM auslesen und die Positionierrechnung durchführen. Mit *StartPS* die Positionierung starten.

Der Antrieb meldet mit GPO1=*High*, wenn der Positioniervorgang abgeschlossen wurde.

13.1.12. Tippen

Mit *Tippen+* oder *Tippen-* kann der Motor mit der vorprogrammierten Tippgeschwindigkeit bewegt werden. Die Bewegung hält solange an, wie das Startsignal (GPIn 3) anliegt.

13.1.13. Restweg fertigfahren

Nachdem mit Stop oder Sperre der Antrieb stillgesetzt wurde, kann die Positionierung durch folgenden Vorgehensweise fortgesetzt werden:

- Antrieb freigeben.

- mit WrPS Positioniersteuerung neu laden.

- mit StartPS Positionierung starten.

13.2. Steuerung über NOVOBUS

13.2.1. Initialisierung der Positioniersteuerung

Es gibt 4 Arten die Positioniersteuerung zu initialisieren:

- Referenzfahrt
- Istwertsetzen
- InitPS
- Automatische Initialisierung nach dem Einschalten. Dazu muß Bit 1 in HwVersion2 auf 0 gesetzt sein.

13.2.1.1. Referenzfahrt

Dazu gibt es im NOVOBUS Treiber eine Spezialfunktion:

void NB_Referenzfahrt (unsigned int *Axis*,
NBSEMAPHORE *Semaphore*);

Funktion:	Initialisiert die Positioniersteuerung (PS) mit Referenzfahrt	
Parameter:	*Axis:*	Adresse des Antriebes Ring 1: 0x0000-0x00f9 Ring 2: 0x0100-0x01f9
	Code	Set GPO1 H' 00 Set GPO2 H' 02 Reset GPO2 H' 03
	Semaphore:	Optional

Bei NB_Referenzfahrt handelt es sich um eine Funktion, die mehrere Aktionen auf NOVOBUS durchführt:

1. ps_v0 (H' FF4C) wird ausgelesen und zwischengespeichert
2. Status (H' FF00) wird ausgelesen, und geprüft:

   ob Reset noch aktiv ist,

   und kein Fehler vorliegt.

3. Falls kein Reset mehr aktiv ist, und kein Fehler vorliegt, wird ps_status (H' FF43) auf 0 gesetzt.

4. Die oberen 5 Bits von Flags2 (H' FF57) werden auf Null gesetzt.

5. nsoll (H' FF08) wird auf Null gesetzt.

6. Bit 7 in Flags2 (H' FF57) wird auf 1 gesetzt um die Referenzfahrt zu starten.

7. Flags2 (H' FF57) Bits 5, 6 und 7 werden kontrolliert. Solange nicht alle 3 Bits Null sind ist die Referenzfahrt aktiv. Gleichzeitig wird in Flags (H' FF56) Bits 5 und 7 kontrolliert, daß die Referenzfahrt nicht mit Stop oder Sperre unterbrochen wurde.

8. Sobald der Referenzfahrtablauf beendet wurde, wird ps_status (H' FF43) kontrolliert, bis der Antrieb mit ps_status = H' 01 "In Position" zurückmeldet. Auch dabei wird Stop (Bit5) und Sperre (Bit 7) in Flags kontrolliert.

9. ps_v0 wird wieder zurückgeschrieben.

13.2.1.2. Istwert setzen

Dazu gibt es im NOVOBUS-Treiber eine Spezialfunktion.

void NB_IstwertSetzen	(unsigned int	*Axis*,
	unsigned long	*Position,*
	NBSEMAPHORE	*Semaphore*);

Funktion: Initialisiert die Positioniersteuerung (PS) ohne Referenzfahrt. Und weist der momentanen Position einen neuen Wert zu.

Parameter:	*Axis:*	Adresse des Antriebes Ring 1: 0x0000-0x00f9 Ring 2: 0x0100-0x01f9
	Position	Aktuelle Rotorposition entsprechend dem neuen Lageistwert.
	Semaphore:	Optional

Bei NB_IstwertSetzen handelt es sich um eine Funktion, die mehrere Aktionen auf NOVOBUS durchführt:

1. Istposition (H' FF16 4 Byte) und dLage (H' FF74) werden eingelesen:
   ....Lage = NB_ReadLong(Antrieb, 0xFF16)
   ....dlage = NB_ReadWord(Antrieb, 0xFF74)

2. Von der Istposition die Rotorlage-Verschiebung abziehen:
   Lage = Lage - dlage.

3. Neue Rotorlage-Verschiebung berechnen:
   dlage = Istwert - Lage

4. Abfragen ob der Antrieb gesperrt ist. Wenn nein sperren.

5. Neue dlage in ND21 (H' FF74) übertragen.

6. Istwert-Umdrehungen in in umdrist (H' FF16) eintragen.

7. Falls der Antrieb bei 4. nicht gesperrt war jetzt wieder freigeben.

8. Die neue Istposition (H' FF16 4Byte) zurücklesen Lage = NB_ReadLong(Antrieb, =xFF16) und mit Istwert vergleichen.delta = Istwert - Lage.
   Falls die Abweichung kleiner 256 Inkremente ist, weiter mit 9., sonst ab 2. wiederholen.

9. Bit3 in Flags2 (H' FF57) auf 0 setzen.

10. In ps_status (H' FF43) H' 01 eintragen.

13.2.1.3. Init PS

Dazu gibt es im NOVOBUS-Treiber eine Spezialfunktion

void NB_InitPS (unsigned int *Axis*,
NBSEMAPHORE *Semaphore*);

Funktion:	Initialisiert die Positioniersteuerung (PS) ohne Referenzfahrt, und ohne den Lageistwert zu ändern.	
Parameter:	*Axis:*	Adresse des Antriebes Ring 1: 0x0000-0x00f9 Ring 2: 0x0100-0x01f9
	Semaphore:	Optional

NB_InitPS macht folgendes:

1. Bit 3 in Flags2 (H' FF57) wird auf Null gesetzt.

2. Bit 6 in ps_status (H' FF43) wird auf Null gesetzt.

3. ps_status = H' 01 ? Wenn ja, dann fertig. Wenn nein 4.

4. Falls ps_umdrehung = H' FF dann ps_status auf H' 01 setzen, fertig. Falls ps_umdrehung <> H' FF dann 5.

5. Warten bis ps_status Bit5 oder Bit1 1 geworden sind, oder ps_status = 0 geworden ist.

6. ps_status auf H' 01 setzen.

13.2.2. Neues Ziel übertragen und Positionierrechnung starten

void NB_WritePS	(unsigned int	*Axis*,
	unsigned long	*Position,*
	NBSEMAPHORE	*Semaphore*);

Funktion: Sendet die neue Zielposition der Positioniersteuerung und startet die Rechnung.

Parameter:	*Axis:*	Adresse des Antriebes RIng 1: 0x0000-0x00f9 Ring 2: 0x0100-0x01f9
	Position	Neue Zielposition
	Semaphore:	Optional

NB_WritePS macht folgendes:

1. Flags2 (H' FF57) Bit 3 kontrollieren und warten bis es Null ist.

2. Ziel eintragen: obere 2 Bytes in ps_positionH (H' FF44)
   untere 2 Bytes in ps_positionL (H' FF46)

3. Flags2 Bit3 auf 1 setzen.

13.2.3. Positionierung starten

void NB_StartPS (unsigned int *Axis*,
NBSEMAPHORE *Semaphore*);

Funktion: Startet die Positioniersteuerung (Software-Start).

Parameter:	*Axis:*	Adresse des Antriebes Ring 1: 0x0000-0x00f9 Ring 2: 0x0100-0x01f9
	Semaphore:	Optional

NB_StartPS macht folgendes:

1. Warten bis Bit 5 von ps_status (H' FF43) 1 wird

2. Bit 4 von ps_status auf 1 setzen.

13.2.4. Ende der Positionierung

Die Positionierung ist dann abgeschlossen, wenn ps_status den Wert H' 01 angenommen hat. Dieses kann mit der Funktion NB_ReadByte ausgelesen werden.

Falls GPO2 für "In Position" programmiert wurde, kann die "In Position"-Meldung in H' FFB7 mit Bit 5 = 1 ausgelesen werden.

13.3. Interner Ablauf der Positioniersteuerung

13.3.1. Status

Die Positioniersteuerung wird durch die Speicherzelle ps_status (Adresse H' FF43) kontrolliert.
Die Bits haben folgende Bedeutung:

Bit7:	Richtung der Positionierung
Bit6:	1: startet die PS-Rechnung
Bit5:	1: meldet "PS-Rechnung fertig"
Bit4:	1: startet die Positionierung
Bit3:	1: Phase "Beschleunigen und Konstantfahren"
Bit2:	1: Phase "Bremsen"
Bit1:	1: Phase "Zielbremsen"
Bit0:	1: meldet "Positionierung abgeschlossen"

13.3.2. Ablauf einer Positionierung

13.3.2.1. Zielposition übertragen

Die absolute Zielposition mit 32 Bit (16 Bit Umdrehungen und 16 Bit Rotorlage) wird in ps_positionH (H' FF44) und ps_positionL (H' FF46) eingetragen.

13.3.2.2. Rechnung starten

Dazu wird Bit 3 von Flags2 (H' FF57) auf 1 gesetzt.

ND21 berechnet die Positionierrichtung und trägt sie in Bit 7 von ps_status ein. Zusätzlich berechnet ND21 die Anzahl der erforderlichen Motorumdrehungen und Impulse und trägt sie in ps_umdrehung (H' FF48) und ps_impulse (H' FF4A) ein.

Dann wird in ps_status Bit 6 auf 1 gesetzt. Dadurch wird die eigentliche Positionierrechnung gestartet.

Die Positionierrechnung ermittelt die Werte ps_v0 (H' FF4C), ps_k0 (H' FF4E), ps_v1 (H' FF51), ps_k1 (H' FF52) und ps_delta (H' FF53). Diese Werte bestimmen, wie lange mit welchen Geschwindigkeiten gefahren wird um im Ziel stehen zu bleiben.

Sobald diese Rechnung abgeschlossen ist, wird Bit 5 in ps_status auf 1 gesetzt.

13.3.2.3. Positionierung starten

Nachdem mit Bit 5 in ps_status das Ende der Rechnung angezeigt wurde, kann mit ps_status Bit 4 = 1 die Positionierung gestartet werden. Den Ablauf der Positionierung kann man mit ps_status verfolgen.

Während dem Beschleunigen und Konstantfahren ist Bit 3 gesetzt.

Während dem Bremsen ist Bit 2 gesetzt.

Während dem Zielbremsen ist Bit 1 gesetzt

Sobald die Positionierung abgeschlossen ist wird Bit 0 gesetzt.

Gleichzeitig wird Bit 3 Flags2 zurückgesetzt, um anzuzeigen, daß eine neue Berechnung stattfinden kann.

13.3.2.4. Relativ positionieren

Zum Relativpositionieren genügt es, die Anzahl der Umdrehungen in ps_umdrehungen, die Anzahl der Motorimpulse in ps_impulse, und die Richtung in Bit 7 von ps_status einzutragen. Bit 6 von ps_status auf 1 setzen um die PS-Rechnung zu starten. Abwarten bis Bit 5 von ps_status 1 wird. Dann mit ps_status Bit 4 = 1 die Positionierung starten.

13.4. Fehlerquellen

Problem	Ursache
PS-Rechnung wird nicht durchgeführt.	Antrieb gesperrt.
Fehler 600 PS-Überlauf.	Positionierzeit > 26 sec Positionierweg verkürzen oder Geschwindigkeit erhöhen.
"In Position"-Meldung erscheint nicht an GPO2.	Programmierung HwVersion 2 prüfen, Einstellung Window prüfen.
Referenzfahrt endet gelegentlich eine Motorumdrehung weiter.	Referenzschalter neu justieren.
Fehler 601.	Positioniersteuerung nicht freigegeben.

13.5. Positioniersteuerung mit Sinus^2-Kurve

13.5.1. Allgemeines

Die Positionierung mit einer trapezförmigen Geschwindigkeitsrampe hat hohe Beschleunigungssprünge zur Folge. Diese wirken sich bei schwingungsfähigen Systemen störend auf den Bewegungsablauf aus. Gleichzeitig wird die Mechanik überdurchschnittlich stark belastet, was wiederum die Lebensdauer einer Anlage verringern kann. Um dies zu vermeiden, gibt es beim ND 21 die Möglichkeit, die Geschwindigkeit mit einer Sinus^2-Funktion (S-Rampe) ansteigen zu lassen.

13.5.2. Voraussetzungen

Da die Vorberechnungen für die S-Rampe nicht im ND21, sondern in einem PC stattfinden, muß dieser im System eingebunden sein. Er kann dann natürlich auch alle anderen Aufgaben übernehmen, wie Anzeigen von Antriebsinformationen, Übertragung von Steuerbefehlen, usw.

Zusätzlich zu einem PC müssen noch die folgenden Voraussetzungen erfüllt sein:

- ND 21 Hauptprozessor H8 ab Version V3.1.
- ND 21 Coprozessor Novochip 2204.
- Novobus-Treiber NOVOBUS.LIB ab V3.1.
- ND 21- Parameter: Bit 2 in SwVersion=0.

13.5.3. Funktionsbeschreibung

Die Programmierung der S-Rampen-Parameter wie Rampensteigung, Fahrgeschwindigkeit, Fahrweg und Fahrrichtung erfolgt dabei über einen IBM kompatiblen PC, der über eine RS232 oder RS485 Schnittstelle mit dem ND 21 verbunden ist. Die Übertragungsrate beträgt 38400 Baud.

Der Novobus-Treiber (NOVOBUS.LIB) steht jedem Anwender als Software-Bibliothek kostenlos zur Verfügung. Er enthält ab Version 3.1 auch drei spezielle Befehle zur Programmierung der Positioniersteuerung mit S-Rampe:

SK_WritePS Mit diesem Befehl werden dem NOVOBUS-Treiber die neuen Positions- und Geschwindigkeitsdaten übergeben. Der Treiber berechnet daraus dann mehrere Stützpunkte für die S-Rampe und sendet sie an den adressierten Antrieb. Zwischen den Stützpunkten wird mit konstanter Beschleunigung verfahren.

Die maximale Dauer des Positioniervorgangs ist auf 120 Minuten begrenzt, d.h. nach dieser Zeit muß der programmierte Weg zurückgelegt worden sein. Bei der Wegprogrammierung ist darauf zu achten, daß keine Positionen, sondern Wege vorgeben werden müssen (inkremental).

Eine Absolutpositionierung, wie beim trapezförmigen Geschwindigkeitsverlauf, ist mit der S-Rampe z. Z. nicht integriert. Die Fahrrichtung wird erst im Befehl SK_StartPS festgelegt.

SK_StartPS Mit diesem Befehl wird der Positioniervorgang im ND 21 gestartet. Zusätzlich wird hier die Fahrrichtung mit angegeben. Dies hat den Vorteil, daß bei gleichen Weglängen die Stützpunkte nicht mehr neu berechnet werden müssen. Es reicht dann aus, den Befehl SK_StartPS erneut abzusetzen.

BYTE SK_StatusPS Diese Funktion liefert den Status der Positionierung mit S-Rampe zurück. Es sind dabei mehrere Zustände möglich.

Ready Die Übertragung der Weg- und Geschwindigkeitsinformationen vom PC ins ND 21 ist abgeschlossen. Die Positionierung kann nun über den Befehl SK_StartPS aktiviert werden.

Walt Dieser Zustand kommt dann vor, wenn vor dem Starten des Positioniervorgangs ein Hard- oder Software-Stop anliegt. ND 21 wartet dann auf einen Hard- oder Software-Start.

Stopped Wird der Antrieb während einer Positionierung gestoppt oder gesperrt, so geht der Status in den Zustand ‘stopped’ über. Die folgenden Wege werden dann von dieser Position aus verfahren.

Doing Die Positionierung ist gerade aktiv.

Done Nach dem Erreichen der Zielposition wird der Status ‘done’ zurückgeliefert.

13.5.4. Syntax der Befehle zur Positionierung mit S-Rampe

Im Folgenden sind die Befehle erläutert, die zur Positionierung mit einer Sinus^2-Kurve zusätzlich benötigt werden und die in der Treiber-Version 3.1 enthalten sind.

unsigned char SK_WritePS	(unsigned int	*Achse*,
	unsigned long	*Weg*
	unsigned int	Upm,
	unsigned char	*Rampe*,
	NBSEMAPHORE	*Semaphore*);

Erklärung:	Die Funktion sendet den neuen Fahrweg (inkremental) für die Positionierung mit Sinus^2-Kurve. Die benötigten Vorberechnungen finden im PC statt.	
Parameter:	*Address:*	Novobus-Adresse.
	Weg:	Fahrweg in Impulsen (0x10000 entspricht 1 Umdrehung).
	Upm:	Größe des Ringes, Zahl der Antriebe im Ring (1...250)
	Rampe:	Beschleunigungszeit in ms.
	Semaphore:	Optional
Ergebnis:	= 0: Befehl erfolgreich ausgeführt <> 0: Fehler bei Befehlsausführung.	

unsigned char SK_StartPS	(unsigned int	*Achse*,
	char	*Richtung*,
	NBSEMAPHORE	*Semaphore*);

Erklärung: Die Funktion startet den Positioniervorgang mit Sinus^2-Kurve. Als Fahrweg gilt der letzte mit SK_WritePS übertragen Weg.

Parameter:	*Address:*	Novobus-Adresse.
	Richtung:	'+' oder '-'
	Semaphore:	Optional

Ergebnis: = 0: Befehl erfolgreich ausgeführt
<> 0: Fehler bei Befehlsausführung.

unsigned char SK_StatusPS	(unsigned int	*Achse*,
	NBSEMAPHORE	*Semaphore*);

Erklärung: Die Funktion liefert den Status der Positionierung mit Sinus^2-Kurve zurück.

Parameter:	*Address:*	Novobus-Adresse.
	Semaphore:	Optional

Ergebnis: = 0: ready (Informationsübertragung PC-ND 21 abgeschlossen)
= 1: wait (Wartet auf Hard- oder Software-Start)
= 2: doing (Positionierung läuft)
= 3: done (Positionierung abgeschlossen)
= 0xFF: Stop/Sperre (Antrieb gestoppt)

14. Einstell- und Ausgabemöglichkeiten des Novodrive

In diesem Kapitel erfahren Sie, welche Einstellmöglichkeiten der Novodrive bietet. Ferner werden hier die Parameter, die Sie auslesen können, erklärt.

Parameter, die den Regelkreis betreffen, nur dann einstellen, wenn Sie die Bedeutung des jeweiligen Parameters und seine Auswirkungen verstanden haben.

Sie haben zwei Möglichkeiten, die Einstellung des Novodrives zu ändern:

- über die Inbetriebnahme-Software
- über den NOVOBUS

14.1. Voraussetzungen

Mit der Inbetriebnahme-Software können Sie viele Parameter komfortabel über Menüs einstellen. Sollten diese Möglichkeiten für Ihre Anwendungen nicht ausreichen, dann finden Sie die notwendigen Informationen in diesem Kapitel.

Alle Parameter können auch über den NOVOBUS geändert oder ausgelesen werden. Die NOVOBUS-Treibersoftware stellt die dafür erforderlichen Write- und Read-Befehle zur Verfügung.

Mit der Inbetriebnahme-Software können Sie am Novodrive sämtliche Einstellungen vornehmen und die einzelnen Parameter auslesen.

Um die Parameter mit der Inbetriebnahme-Software einstellen oder auslesen zu können, benötigen Sie einen IBM kompatiblen PC und ein Buskabel.

14.2. Speicher des Novodrive

Ihr Novodrive hat 5 verschiedene Speicher.

- RAM-Speicher mit 512 Byte, Adressen: FD80h - FF7Fh.
- Speicher im ASIC.
- EEPROM mit 256 Bytes, Adressen: 00h - FFh.

- Speicher des Mikrocontrollers, Adressen: FF90h - FFA7h
- FFB0h - FFFFh.
- Eprom-Speicher, Adressen: 0000h - 3FFFh.

Siehe Kapitel 14.2.1. RAM Speicher.

RAM-Speicher Eine Änderung im RAM-Speicher wirkt sich unmittelbar auf die Regelung aus. Beim Aus- und Wiedereinschalten der Versorgungsspannung des ND21 gehen Änderungen, die im RAM-Speicher vorgenommen wurden, verloren, wenn sie nicht zuvor im EEPROM-Speicher des ND21 gesichert wurden.

Der RAM-Speicher des ND21 kann im Servicemenü mit der Funktion RAM-Monitor angesehen werden. Veränderungen werden mit der Funktion Write RAM durchgeführt **(siehe Handbuch Inbetriebnahme und Parametereinstellungen von ND21, Kapitel 1.4.11. Servicemenü).**

ASIC Bei den ASIC-Parametern handelt es sich um die Einstellungen des PI-Stromreglers. Änderungen dieser Parameter wirken sich unmittelbar auf die Regelung aus. Die ND21 Inbetriebnahmesoftware sichert sie immer automatisch im EEPROM.

Änderungen im Servicemenü mit der Funktion ASIC **(siehe Handbuch Inbetriebnahme und Parametereinstellungen von ND21, Kapitel 1.4.11. Servicemenü).**

EEPROM Speicher ND21 ist mit einem seriellen EEPROM mit 256 Bytes ausgerüstet.

Das EEPROM ist nichtflüchtig, d. h. es verliert seine Daten beim Ausschalten der Versorgungsspannung nicht.

Änderungen im Servicemenü mit dem Menü EEPROM **(siehe Handbuch Inbetriebnahme und Parametereinstellungen von ND21, Kapitel 1.4.11. Servicemenü).**

H8-Registerfeld Hier befinden sich die Adressen zur Steuerung der µController Peripherie.

EPROM Speicher Im EPROM Speicher des ND21 befindet sich die ND21 Firmware, unveränderliche Parameter und Tabellen.

14.2.1. RAM Speicher

Adresse	Bezeichnung	Bytes	Bedeutung
FD80	PoleJump	2	Sprungadresse für Motorpolzahl
FD82	errorcode	2	Fehlercode
FD84	i2	4	
FD88	EEPROMbuffer	2	intern EEPROM Prog.
FD8A	EEPROM control	1	intern EEPROM Prog.
FD8B	reserviert	1	
FD8C	RefV1	2	1. Referenzgeschwindigkeit
FD8E	RefV2	2	2. Referenzgeschwindigkeit
FD90	RefUmdr0	2	Auf diesen Wert wird der Umdrehungszähler nach der Referenzfahrt gesetzt
FD92	RefLage0	2	Auf diesen Wert wird die Rotorlage nach der Referenzfahrt gesetzt
FD94	Vtippen	2	Geschwindigkeit in Tippen
FD96	iOffset	2	intern Strommessung
FD98	dRESOLVER	2	Resolverjustage
FD9A	PZ	1	Positionszeiger in der Ablaufsteuerung
FD9B	BZ	1	Bitzeiger Ablaufsteuerung
FD9C	AS_Command	1	Befehl in Ablaufsteuerung
FD9D	reserviert	1	
FD9E	RefLage	2	Nullpunktverschiebung bei Referenzfahrt
FDA0			
- FE9F	scope_puffer	256	intern Speicheroszilloskop
FEA0			

- FEFF	stack	96	intern Programmstack
FF00	Status	1	Statusbyte
FF01	mmax	1	Maximales Drehmoment
FF02	msoll	2	Drehmoment Sollwert
FF04	iasoll	1	Stromsollwert Phase A
FF05	ibsoll	1	Stromsollwert Phase B
FF06	iaist	1	Stromistwert Phase A
FF07	ibist	1	Stromistwert Phase B
FF08	nsoll	2	Drehzahlsollwert
FF0A	nsollrampe	2	Drehzahlsollwert nach dem Rampengenerator
FF0C	nist	2	Drehzahlistwert
FF0E	nIntegrator	2	Drehzahlregler Integrator
FF10	emk	1	Motor EMK
FF11	lagesollL	1	
FF12	umdrsoll	2	Umdrehungen Sollwert
FF14	lagesoll	2	Rotorlage Sollwert
FF16	umdrist	2	Umdrehungen Istwert
FF18	lageist	2	Rotorlage Istwert
FF1A	lage0	2	Rotorlage Istwert alt
FF1C	nsollAnalog	2	reserviert
FF1E	icra	2	Input Capture Register A
FF20	icra0	2	Input Captuer Register A alt
FF22	VCOoffset	2	Offset Analogeingang
FF24	adcsr	1	A/D-Control-Status-Register
FF25	reserviert	1	
FF26	TempKK	1	Kühlkörpertemperatur

FF27	TempMot	1	Motortemperatur
FF28	nsollDigital	2	reserviert
FF2A	Novobus	1	Kommunikationsbytes
FF2B	Command	1	
FF2C	ParamBuffer	4	
FF30	ChecksumIn	1	
FF31	ChecksumOut	1	
FF32	DataIn	1	Datenkanal Eingangsadr.
FF33	DataInBuffer	1	Datenkanal Eingangsbytes
FF34	DataOut	1	Datenkanal Ausgabeadresse
FF35	DataOutBuffer	1	Datenkanal Ausgangsbytes
FF36	BusAddr	1	
FF37	reserviert	1	
FF38	scope_status	1	Oszilloskopstatus
FF39	scope_timer0	1	Zeitbasis
FF3A	scope_counter	2	Triggerverzögerung
FF3C	scope_delay	1	Verzögerungszähler
FF3D	scope_pointer	1	intern
FF3E	scope_trigger	1	Adresse Triggerquelle
FF3F	scope_level	1	Triggerschwelle
FF40	scope_signal1	1	Adresse Signal 1
FF41	scope_signal2	1	Adresse Signal 2
FF42	reserviert	1	
FF43	ps_status	1	Status Positioniersteuerung
FF44	ps_positionH	2	Ziel Umdrehungen
FF46	ps_positionL	2	Ziel Rotorlage
FF48	ps_umdrehung	2	Umdrehungszähler Pos.

FF4A	ps_impuls	2	Impulszähler Pos.
FF4C	ps_v0	2	erste Geschwindigkeit
FF4E	ps_k0	2	Dauer erste Geschwindigkeit
FF50	ps_v1	2	zweite Geschwindigkeit
FF52	ps_k1	2	Dauer 2. Geschwindigkeit
FF53	ps_delta	1	Restweg
FF54	reserviert	1	
FF56	Flags	1	ND21 Statusbyte
FF57	Flags2	1	ND21 Statusbyte
FF58	Cnt2	1	interner Zähler
FF59	reserviert	1	
FF5A	uP	2	µP Auslastung
FF5C	RealTimeClock	2	Zeit-Taktgeber
FF5E	CntBlink	1	Zykluszähler Anzeige
FF5F	CntRevers	1	Zykluszähler Reversieren
FF60	HwVersion	1	ND21 Konfigurationsbyte
FF61	HwVersion2	1	ND21 Konfigurationsbyte
FF62	SwVersion	1	ND21 Konfigurationsbyte
FF63	SwVersion2	1	ND21 Konfigurationsbyte
FF64	MaxTempKK	1	Kühlkörper-Temperaturgrenze
FF65	MaxTempMot	1	Motor-Temperaturgrenze
FF66	imax	1	Spitzenstrom
FF67	i0	1	Motornennstrom
FF68	nMax	2	Maximaldrehzahl
FF6A	reserviert	2	
FF6C	nRampe+	2	Beschleunigungsrampe
FF6E	nRampe-	2	Bremsrampe

FF70	nKp	1	P-Anteil Drehzahlregler
FF71	nKi	1	I-Anteil Drehzahlregler
FF72	LKp	1	P-Anteil Lageregler
FF73	LKd	1	D-Anteil Lageregler
FF74	dLage	2	Nullpunktsverschiebung
FF76	Schleppfehler	2	nur bei Lageregelung
FF78	VCOlin	1	Skalierung Analog- und Frequenzeingang
FF79	nFilter	1	Tachofilter
FF7A	emk0	1	EMK Vorsteuerung
FF7B	Window	1	Fenster für In Position, oder Drehzahl 0
FF7C	AnOut	1	Auswahlbyte Analogausg.
FF7D	AnOutOffset	1	Offsetabgleich Analogausg.
FF7E	ReversTakt	1	Zyklus für Testbetrieb Reversieren
FF7F	VCOlinL	1	Feinskalierung Frequenzeingang

14.2.2. EEPROM Speicher

00	EE_LA	reserviert
01; 02; 03	EE_Seriennummer	Seriennummer
04		reserviert
05; 06	EE_BetriebStd	Betriebsstunden
07	reserviert	
08; 09	EE_SperreStd	Sperrestunden
0A; 0B	EE_InitDatum	Datum EEPROM Init.
0C	EE_Version	Konfigurationsbyte
0D; 0E; 0F	reserviert	00

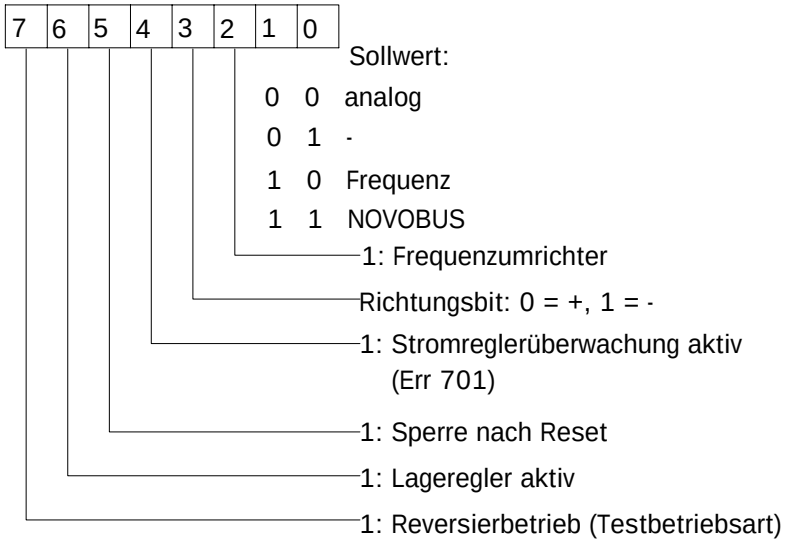
10 - 1E	reserviert für	Fehlerhistory
1F	EE_parity	EEPROM Kontrollbyte
20 - 3F	EE_RAM	RAM-Spiegel entspricht RAM FF60 - FF7F, (Parameterbereich)
40	EE_a_param3	ASIC Einstellung
41	EE_a_mpr1	ASIC Einstellung
42	EE_a_mpr2	ASIC Einstellung
43	EE_a_mpr98	ASIC Einstellung
44; 45	EE_impulszahl	Enkoderemulation
46	EE_pole	Motorpolzahl
47	EE_selftest	Konfigurationsbyte
48; 49	EE_RefV1	Referenzgeschwindigkeit
4A; 4B	EE_RefV2	Referenzgeschwindigkeit
4C; 4D	EE_RefUmdr0	Referenzpunkt-Koordinate
4E; 4F	EE_RefLage0	Referenzpunkt-Koordinate
50; 51	EE_Vtippen	Tippgeschwindigkeit
52; 53	EE_dRESOLVER	Resolverjustage
54; 55	EE_RefLage	Nullpunktsverschiebung
56	EE_nd2204	ASIC Einstellung
H'57	reserviert	
58; 59	EE_ProgDatum	Datum Parameteränderung
5A; 5B	EE_ProgName	
5C - 5F	EE_Kunde	Frei für Kunden (4 Byte)
60 - FF	EE_Ablauf	Ablaufsteuerung (160 Byte)

14.3. Konfiguration

14.3.1. Das Byte SwVersion

RAM-Parameter Adresse: H' FF62
EPROM: H' 22

Das Byte kann gelesen und geschrieben werden.



Erläuterungen:

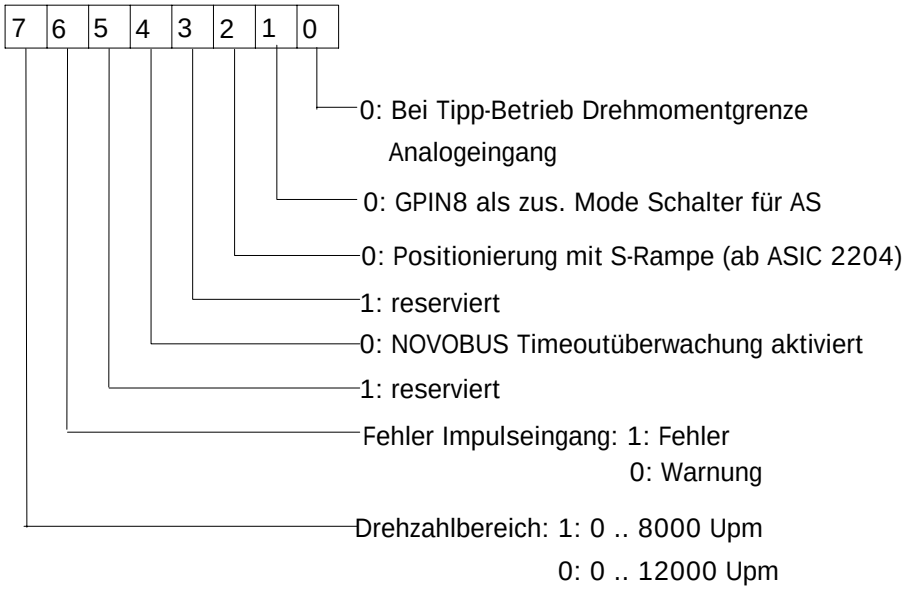
Bit 0 und 1	Bestimmen, woher der Drehzahlregler des ND21 seinen Sollwert bekommt.
Bit 2	1 ist Frequenzumrichterbetrieb 0 ist Servoumrichter
Bit 3	Damit kann der Drehsinn des Motors geändert werden. D.h., wenn der Motor bei positiver Richtungsvorgabe in die als negativ bezeichnete Drehrichtung läuft, kann durch Ändern des Bit 3 dieses korrigiert werden.
Bit 4	Mit 1 kann eine zusätzliche Überwachungsfunktion des Stromreglers aktiviert werden.

Bit 5	Mit 1 wird das ND21 nach dem Einschalten erst aktiv, wenn die Freigabe über NOVOBUS erfolgt. Mit 0 wird nach dem Einschalten des ND21 der Regler aktiv, wenn am Regler-Sperre-Eingang eine Spannung angelegt wird.
Bit 6	Mit 1 wird der Lagerregler aktiviert. Dies ist auch dann möglich, wenn mit Drehzahlsollwerten gearbeitet wird. In diesem Fall errechnet ND21 sich aus den Drehzahlsollwerten die Lagesollwerte. Dadurch wird eine höhere Steifigkeit des Motors erreicht.
Bit 7	Mit 1 wird in einen internen Testbetrieb umgeschaltet, damit das ND21 den Motor mit der eingestellten Drehzahl reversiert. Diese Betriebsart ist besonders zur Optimierung der Reglerparameter geeignet.

14.3.2. Das Byte SwVersion2

RAM-Parameter Adresse: H' FF63
EEPROM: H' 23

Das Byte kann gelesen und geschrieben werden.



Erklärungen:

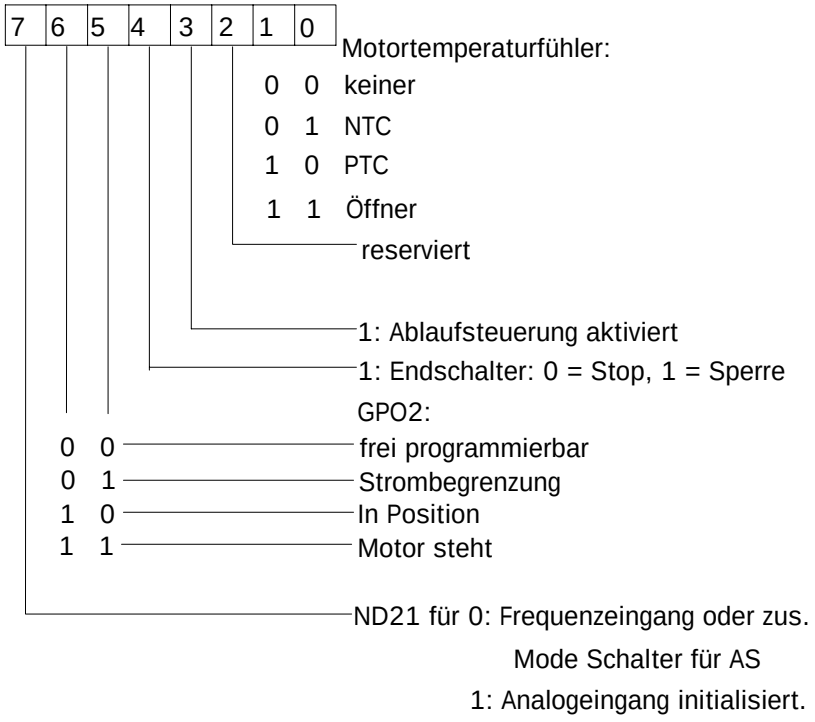
Bit0	Mit 0 kann hier programmiert werden. Das Maximaldrehmoment vom Analogeingang kann eingelesen werden.
Bit 1-5	Reserviert - auf 1 zu setzen.
Bit 6	Falls hier eine 1 programmiert ist, führt ein Fehler am Impulseingang zum Abschalten der Endstufe mit Fehlermeldung. Bei 0 erfolgt lediglich eine Warnung ohne Abschalten der Endstufe.
Bit 7	Einstellung des ND 21-Drehzahlbereichs.

14.3.3. Das Byte HwVersion

RAM-Parameter Adresse: H' FF60
EEPROM: H' 20

Das Byte kann gelesen und geschrieben werden. Dieses Byte wurde bei der Weiterentwicklung von ND21 geändert.

Neue Version: (ab H8 Version 1. 7. 93)

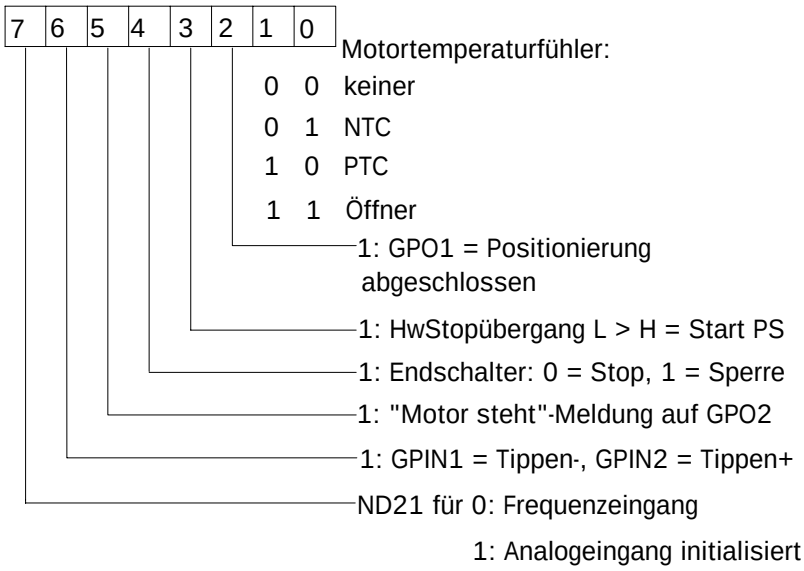


Erläuterungen:

Bit 0 und 1:	Zeigen dem ND 21 an, welcher Temperaturfühler sich im Motor befindet.
Bit 2:	1: reserviert
Bit 3:	Mit 1 wird die eingebaute Ablaufsteuerung aktiviert.
Bit 4:	Bestimmt die Reaktion auf einen angefahrenen Endschalter. Bei 0 stoppt der Antrieb, mit 1 wird der Regler gesperrt.
Bit 5 und 6:	Diese beiden Bits bestimmen, welche Meldung auf den GPO2-Ausgang geschaltet wird.
Bit 7:	Bestimmt die Initialisierung des 16 Bit Zählers im ND 21. Mit 1 wird ND 21 für Analogeingang initialisiert, mit 0 für Frequenzeingang.

Nachdem HwVersion geändert wurde, muß im Servicemenü mit [R] ein Reset des Antriebs ausgelöst werden, um alle Änderungen aktiv werden zu lassen (vorher im EEPROM sichern).

Alte Version: Bis H8 Version 30.06.93.



Erläuterungen:

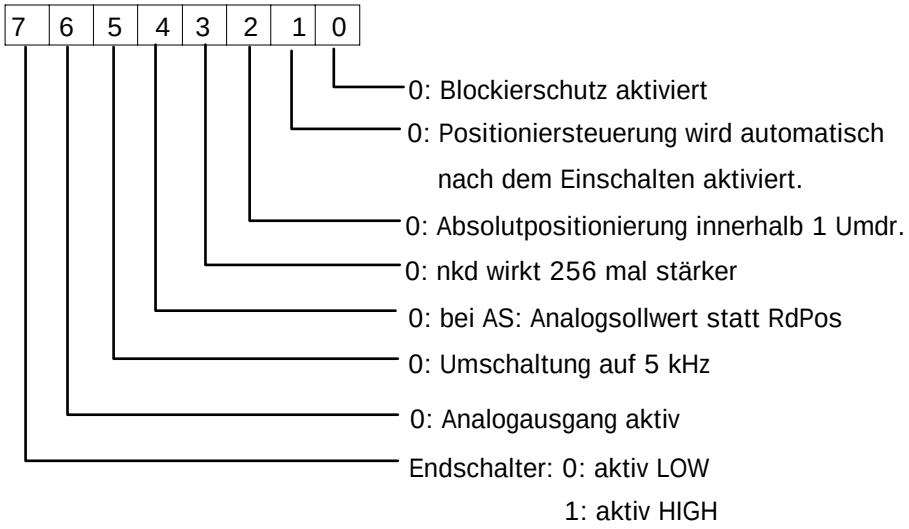
Bit 0 und 1:	Zeigen dem ND 21 an, welcher Temperaturfühler sich im Motor befindet.
Bit 2:	Mit 1 wird der Ausgang GPO1 für die Meldung "Position erreicht" in der Positioniersteuerung verwendet. Ist Bit 2 = 0, kann dieser Ausgang für andere Funktionen verwendet werden.
Bit 3:	Ist dieses Bit gesetzt, wird der Positioniervorgang durch eine steigende Flanke am Start-Stop-Eingang (GPIN3) ausgelöst.
Bit 4:	Bestimmt die Reaktion auf einen angefahrenen Endschalter. Bei 0 stoppt der Antrieb, mit 1 wird der Regler gesperrt.
Bit 5:	Ist dieses Bit gesetzt, wird vom ND 21 bei stehendem Motor eine Stillstandsmeldung auf den Ausgang GPO2 geschaltet.
Bit 6:	Ist dieses Bit gesetzt, werden die Eingänge GPIN1 und GPIN2 für die Tippen-Funktion der Positioniersteuerung verwendet.
Bit 7:	Bestimmt die Initialisierung des 16-Bit-Zählers im ND 21. Mit 1 wird ND 21 für Analogeingang initialisiert, mit 0 für Frequenzeingang.

Nachdem die HwVersion geändert wurde, muß nun im Servicemenü mit [R] ein Reset des Antriebs ausgelöst werden. Damit können alle Änderungen aktiv werden (vorher im EEPROM sichern).

14.3.4. Das Byte HwVersion2

RAM-Parameter Adresse: H' FF61
EEPROM: H' 21

Das Byte kann gelesen und geschrieben werden.



Erläuterungen:

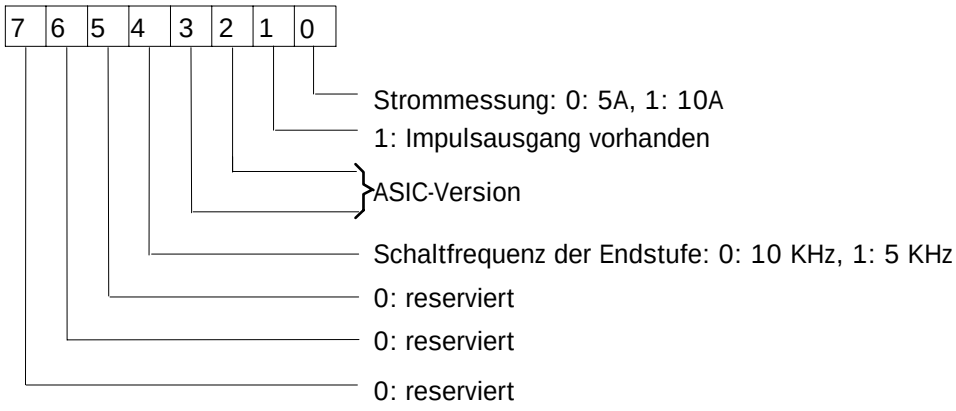
Bit 0 - 4:	Reserviert für zukünftige Erweiterungen, alle auf 1 setzen.
Bit 5:	Mit 0 kann die Schaltfrequenz der Endstufe auf 5 KHz reduziert werden.
Bit 6:	Wenn nicht gesetzt, wird der Analogausgang aktiviert. In diesem Fall kann GPO1 nicht gleichzeitig als Digitalausgang verwendet werden.
Bit 7:	Damit wird der Aktivpegel der Endschalter festgelegt.

Hinweis: Die Funktionen der Bits 0 bis 4 stehen erst seit der H8-Version 3.3 vom 30.12.94 zur Verfügung.

14.3.5. Das Byte EEVersion

EEPROM-Parameter EEPROM: H' 0C

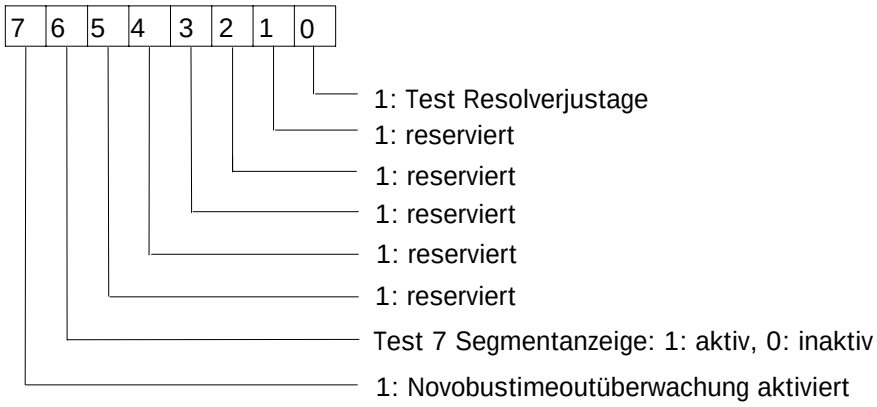
Dieses Byte ist unveränderlich. Es kann nur gelesen, aber nicht geschrieben werden.



14.3.6. Das Byte EESelftest

EEPROM-Parameter EEPROM: H' 47

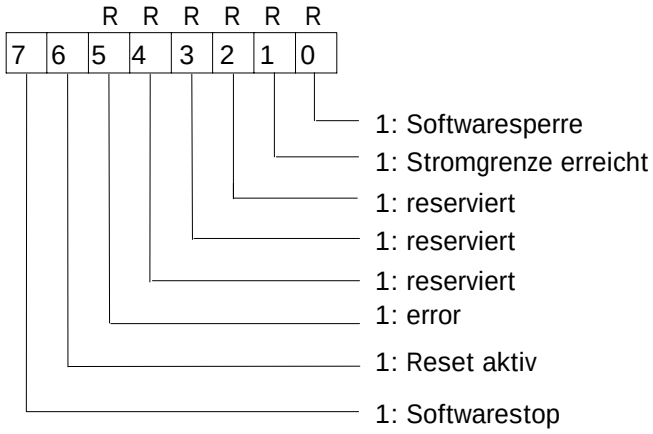
Das Byte kann gelesen und geschrieben werden.



14.4. ND21 Status

14.4.1. Das Byte Status

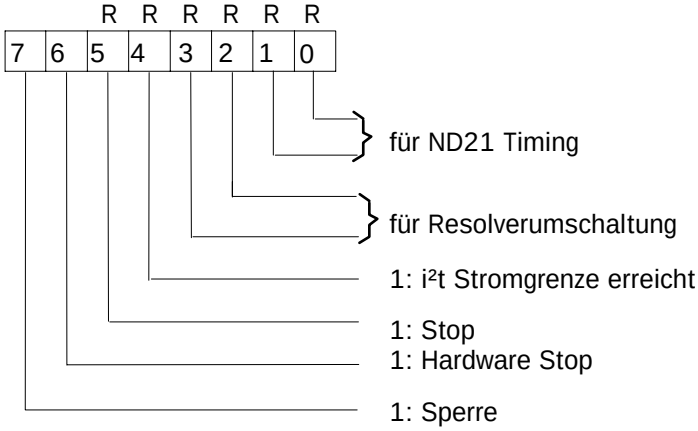
RAM Zelle Adresse H' FF00



Erläuterung: R: diese Bits dürfen nur gelesen werden.

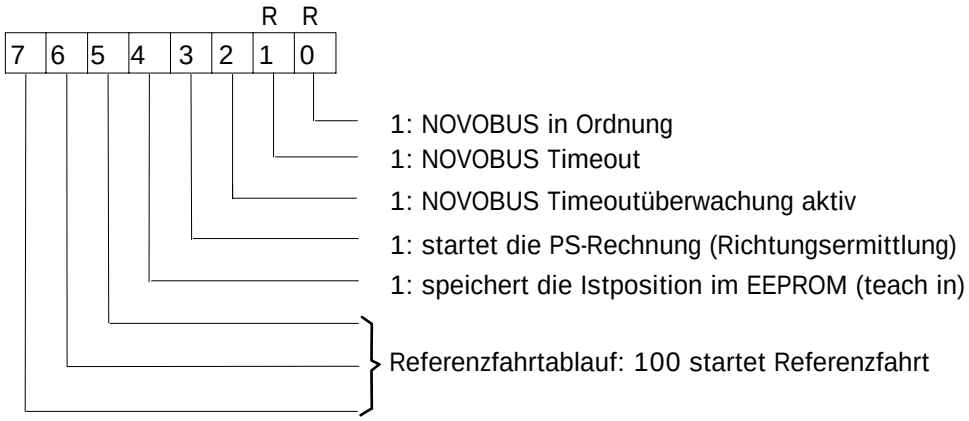
14.4.2. Das Byte Flags

RAM Zelle Adresse H' FF56



Das Byte Flags darf nur gelesen, aber nicht geschrieben werden.

14.4.3. Das Byte Flags2



Erläuterung: R: diese Bits dürfen nur gelesen werden.

14.5. Istwerte, Sollwerte, Grenzwerte

14.5.1. Ströme

14.5.1.1. Soll- und Istwerte

Diese Werte dürfen nur gelesen werden.

Motorstrom-Istwerte:	iaist:	Adresse H' FF06
	ibist:	Adresse H' FF07
Motorstrom-Sollwerte:	iasoll:	Adresse H' FF04
	ibsoll:	Adresse H' FF05

Darstellung: Byte, Zweierkomplement

Skalierung	**5605**	**5610**	**5615**	**5620**
A_{eff} pro Bit	0,122	0,244	0,355	0,488

14.5.1.2. Grenzwerte

Diese Werte werden vom Anwender festgelegt.

Spitzenstrom:	imax:	Adresse H' FF66 EEPROM H' 26
Motornennstrom:	i0:	Adresse H' FF67 EEPROM H' 27

Darstellung: Byte

Skalierung	**5605**	**5610**	**5615**	**5620**
Aeff pro Bit	0,086	0,172	0,251	0,345
Maximum	H' 74	H' 74	H' 52	H' 74

14.5.2. Drehmomente

Diese Werte dürfen nur gelesen werden.

Drehmoment-Sollwert:

msoll: Adresse H' FF02

Darstellung: Wort, Zweierkomplement

Drehmomentgrenze:

mmax: Adresse H' FF01

Darstellung: Byte, Zweierkomplement

Skalierung:	**5605**	**5610**	**5615**	**5620**
mmax Aeff pro Bit	0,086	0,172	0,251	0,345
msoll mAeff pro Bit	0,336	0,671	0,980	1,347

14.5.3. Drehzahlen

14.5.3.1. Soll- und Istwerte

Die Werte nist und nsollrampe dürfen nur gelesen werden.

Drehzahlistwert:	nist	Adresse: H' FF0C
Drehzahlsollwert:	nsoll	Adresse: H' FF08
Drehzahlsollwert nach Rampe:	nsollrampe	Adresse: H' FF0A

Darstellung: Wort, Zweierkomplement

Skalierung:

0,264909525 Upm pro Bit für SwVersion2 Bit7 = 1

0,52981905 Upm pro Bit für SwVersion2 Bit7 = 0

14.5.3.2. Grenzwerte

Dieser Wert wird vom Anwender programmiert.

Maximaldrehzahl:

nMax: Adresse: H' FF68
EEPROM: H' 28

Darstellung: Wort

Skalierung: wie Soll- und Istwerte

14.5.4. Rampen

Die Rampen werden vom Anwender programmiert.

Beschleunigungsrampe:	nRampe+	Adresse: H' FF6C EEPROM: H' 2C
Bremsrampe:	nRampe-	Adresse: H' FF6E EEPROM: H' 2E

Darstellung: Wort

Skalierung:

64,18332625 rad/s² pro Bit für SwVersion2 Bit 7 = 1

128,3666525 rad/s² pro Bit für SwVersion2 Bit 7 = 0

14.5.5. Wege

Sollwerte können vom Anwender programmiert werden.

Rotorlage-Sollwert	lagesoll	Adresse: H' FF14
Umdrehungs-Sollwert	umdrsoll	Adresse: H' FF12
Rotorlage-Istwert	lageist	Adresse: H' FF18
Umdrehungs-Istwert	umrdist	Adresse: H' FF16

Darstellung: umdrsoll und lagesoll, sowie umdrist und lageist bilden jeweils zusammen ein 32 Bit Doppelwort. Zweierkomplement.

14.5.6. Schleppfehler

Beim Betrieb mit Lageregler kann mit Schleppfehler-Überwachung gearbeitet werden. Bei Schleppfehler wird die Endstufe gesperrt und Fehlermeldung H' 700 wird generiert.

Schleppfehler Adresse: H' FF76
EEPROM: H' 36, 37

Darstellung: Wort

Skalierung: Deaktiviert falls H' 8000 eingetragen. 1 Bit bedeutet 0.024543692 rad.

14.5.7. Temperaturen

14.5.7.1. Kühlkörpertemperatur

Diese Werte dürfen nur gelesen werden.

Kühlkörpertemperatur	tempKK	Adresse: H' FF26
Übertemperaturschwelle	MaxTempKK	Adresse: H' FF64

EEPROM: H' 24

Darstellung: Byte

Skalierung: Es sei: $temp1 = \frac{tempKK}{256 - tempKK} \cdot 0{,}27$

damit: Kühlkörpertemperatur (°C]

$$T = \frac{1}{\frac{\ln(temp1)}{3600} + \frac{1}{298}} - 273$$

14.5.7.2. Motortemperatur

Der Motortemperatur-Istwert darf nur gelesen werden. Der Motortemperatur-Grenzwert kann gelesen und geschrieben werden.

Motortemperatur:	tempMot	Adresse: H' FF27
Temperaturschwelle:	MaxTempMot	Adresse: H' FF65 EEPROM: H' 25

Darstellung: Byte

Skalierung: Der Widerstandswert des Motortemperatur-Fühlers berechnet sich folgendermaßen:

$$R = \frac{tempMot \cdot 1227067 - 7180800}{-1 \cdot tempMot \cdot 1298{,}5 + 326400}$$

Die Motortemperatur kann aus der Kennlinie des im Motor eingebauten Sensors errechnet werden.

14.6. Reglerparameter

Alle Reglerparameter können vom Anwender verändert werden.

14.6.1. Stromregler

ND21 arbeitet mit zwei PI-Stromreglern

P-Anteil ikp:	a_mpr1	Adresse: H' FF80 EEPROM: H' 41

a_mpr98 Adresse: H' FF82
EEPROM: H' 43

Darstellung: 10 Bit ohne Vorzeichen. Untere 8 Bit in a_mpr1, obere 2 bits: Bit 7 und 6 von a_mpr98.

I-Anteil iki: a_mpr2 EEPROM: H' 42

a_mpr98 EEPROM: H' 43

Darstellung: 10 Bit ohne Vorzeichen. Untere 8 Bit in a_mpr2, obere 2 bits: Bit 5 und 4 von _mpr98.

Änderungen im EEPROM bei a_mpr1, a_mpr2 und a_mpr98 werden erst nach einem Reset aktiv. Die Stromregler-Parameter dürfen nur im EEPROM geändert werden (außer mit der ND21 Inbetriebnahmesoftware). Sie werden dann nach einem Reset aktiv. Falls die Notwendigkeit besteht während des Betriebes diese Parameter anzupassen, bitte erkundigen Sie sich bei Novotron.

14.6.2. EMK-Kompensation

Die EMK-Kompensation dient der Spannungsvorsteuerung zur Kompensation der Gegen-EMK des Motors.

EMK-Kompensation: emk0 Adresse: H' FF7A
EEPROM: H' 3A

Darstellung: Byte, kein Vorzeichen

Skalierung: emk0 = 0.38636 · (Spannungsgradient des Motors in V/1000Upm)

14.6.3. Tachofilter

Mit dem Tachofilter ist es möglich den Drehzahlistwert zu filtern. Beim Tachofilter handelt es sich um einen Filter 1. Ordnung.

Tachofilter: nFilter Adresse: H' FF79
EEPROM: H' FF39

Darstellung: Byte, kein Vorzeichen, Bereich: H' 00 - H' 7F.

Skalierung: Zeitkonstante

$$\text{Tachofilter} = \frac{432\mu s}{1 - \frac{\text{nFilter}}{128}}$$

14.6.4. Drehzahlregler

ND21 arbeitet mit einem PI-Drehzahlregler.

P-Anteil	nKp	Adresse: H' FF70 EEPROM: H' 30
I-Anteil	nKi	Adresse: H' FF71 EEPROM: H' 31

Darstellung: Byte, kein Vorzeichen, Bereich: H' 00 - H' 7F.

14.6.5. Lageregler

ND21 arbeitet mit einem PD-Lageregler.

P-Anteil	LKp	Adresse: H' FF72 EEPROM: H' 32
D-Anteil	LKd	Adresse: H' FF73 EEPROM: H' 33

Darstellung: Byte, kein Vorzeichen, Bereich: H' 00 - H' 7F.

14.6.6. Resolverjustage

Die Resolver-Einbaulage kann elektronisch verschoben werden, um eine korrekte Kommutierung zu erreichen.

Resolvereinstellung: dRESOLVER Adresse: H' FD98
EEPROM: H' 52,53

Darstellung: Wort, Höherwertiges Byte in H' 52

Skalierung: 1 Bit bedeutet: 1 Umdrehung / 65536

14.6.7. Motorpolzahl

Es ist möglich in ND21 eine Motorpolzahl von 2 bis 12 zu programmieren.

Motorpolzahl: EE_pole EEPROM: H' 46

Darstellung: Byte, kein Vorzeichen, Wertebereich:
H' 02, H' 04, H' 06, H' 08, H' 0A, H' 0C

14.7. Signal Ein- und Ausgänge

14.7.1. Digitalsignale

Tabelle der digital Eingänge ND21:

Digitaleingang	Speicherzelle	Bit	Port	X3 Pin	0V =
GPIN1	H' FFB3	4	2	21	"1"
GPIN2	H' FFB3	5	2	6	"1"
GPIN3	H' FFB3	6	2	22	"1"
GPIN4	H' FFB3	7	2	7	"1"
GPIN5	H' FFBE	3	7	24	"1"
GPIN6	H' FFBE	7	7	8	"1"
GPIN7	H' FFBB	0	6	10	"1"
GPIN8	H' FFBB	2	6	25	"1"

Endschalter: H'FFB3 0,1,2,3 2

Für die Endschalter gilt folgende Kodierung:

(0 = 0V, 1 = 5 - 24V) X3 Pin

Endschalter N	0	1	0	1	3
Endschalter P	0	0	1	1	34

H'FFB3Bit 0	0	0	1	0
H'FFB3 Bit 1	0	1	1	1
H'FFB3 Bit 2	1	0	0	1
H'FFB3 Bit 3	1	1	1	1

Anmerkung: Falls ein interner Fehler vorliegt wird der Endschalter-Code durch einen Fehlercode überschrieben.

Adressen Digitalausgänge:

Digitalausgang	Speicherzelle	Bit	Port	X3 Pin	0V =
GPO1	H'FFB7	7	4	31	"0"
GPO2	H'FFB7	5	4	23	"0"

Die Signalzustände können in den betreffenden Speicherzellen ausgelesen werden. Um die GPO's direkt zu schreiben stehen spezielle NOVOBUS-Befehle zur Verfügung.

14.7.1.1. Enkoderemulation

Impulszahl: EE_impulszahl
EEPROM: H' 44; 45

Darstellung: Wort, kein Vorzeichen, Bereich H' 0001 bis H' 0400. Höherwertiges Byte in H' 44. Die neue Impulszahl wird erst nach einem Reset aktiv.

14.7.2. Analogeingang

ND21 wird mit HwVersion (H' FF60) Bit 7 = 1 auf Analogeingang programmiert. SwVersion (H' FF62) Bits 1,0 = 00 wählt den Analogeingang für Sollwertvorgabe aus. Dazu ist es erforderlich HwVersion auf dem EEPROM zu sichern, und dann einen Reset auszulösen.

Sollwert-Skalierung VCOlin Adresse: H' FF78
EEPROM: H' 38

Skalierung: ca. VCOlin = (Drehzahl[Upm] bei 10V)/60

Offset: VCOoffset

Adresse: H' FF22

Der Offset wird automatisch einmal pro Sekunde kalibriert.

14.7.3. Frequenzeingang

ND21 wird mit HwVersion (H' FF60) Bit 7 = 0 auf Frequenzeingang programmiert. SwVersion (H' FF62)

Bits 1,0 = 10 wählt den Frequenzeingang für Sollwertvorgabe aus. Dazu ist es erforderlich HwVersion auf dem EEPROM zu sichern, und dann einen Reset auszulösen. Für Frequenzvorgabe müssen die Rampen deaktiviert sein:

nRampe- (H' FF6E) = nRampe+ (H' FF6C) = H' 7FFF.

Der Lageregler muß aktiviert sein:

SwVersion (H' FF62) Bit 6 = 1. Sollwert-Skalierung:

VCOlin Adresse: H' FF78
EEPROM: H' 38

VCOlinL Adresse: H' FF7F
EEPROM: H' 3F

Skalierung:

VCOlin · 256 + VCOlinL = 33554432/(Impulse pro Umdrehung)

14.7.4. Analogausgang

ND21 hat einen 8 Bit Analogausgang. Der Analogausgang kann nur dann verwendet werden, wenn GPO1 nicht verwendet wird. Aktivieren des Analogausganges mit:

HwVersion2 (H' FF61) Bit 6 = 0.

Signalauswahl:

AnOut Adresse: H' FF7C
EEPROM: H' 3C

Es ist eine Signaladresse im H' FFXX Bereich einzutragen.

Beispiel: Für nist (H' FF0C) den Wert H' 0C.

Offset:

AnOutOffset Adresse: H' FF7D
EEPROM: H' 3D

14.7.5. GPO2

Für GPO2 gibt es 4 Programmiermöglichkeiten:

HwVersion (H' FF60)	Bit 6	Bit5
frei programmierbar	0	0
Strombegrenzung erreicht	0	1
In Position	1	0
Motor steht	1	1

14.7.5.1. Frei programmierbar

GPO2 kann mit NOVOBUS auf Null oder 1 gesetzt werden.

14.7.5.2. Strombegrenzung erreicht

GPO2 wird auf 1 gesetzt wenn der Motorstrom durch die Strombegrenzung begrenzt wird.

14.7.5.3. In Position

(siehe Kapitel 13.1.4. Programmierung von ND21)

14.7.5.4. Motor steht

Mit dem Parameter Window (H' FF7B) kann eine Minimaldrehzahl programmiert werden, bei deren Unterschreitung GPO1 auf 1 gesetzt wird. Window wird dabei mit dem niederwertigeren Byte von nist verglichen.

14.8. Antriebsinfos

14.8.1. Seriennummer

Die Seriennummer kann aus dem EEPROM aus den Speicherzellen H' 01, 02, 03 ausgelesen werden.

Darstellung: BCD, Höchstwertiges Byte H' 01

14.8.2. Betriebsstunden

Die Betriebsstunden können aus dem EEPROM ausgelesen werden.

Aktivzeit: EEPROM: H' 05, H' 06

Darstellung: Wort, höherwertiges Byte in H' 06

Gesperrtzeit: EEPROM: H' 08, H' 09

Darstellung: Wort, höherwertiges Byte in H' 09

14.8.3. Inbetriebnahme-Datum

Das Inbetriebnahme-Datum kann aus dem EEPROM ausgelesen werden.

EE_InitDatum
EEPROM: H' 0A, H' 0B

Darstellung: Wort, höherwertiges Byte H' 0A

Bits 0 - 4: Tag

Bits 5 - 8: Monat

Bits 9 - 15: Jahr

Das Kalenderjahr ergibt sich, indem man zum Kalenderjahr 1993 hinzuzählt.

14.8.4. Datum der letzten Parameteränderung

Dieses Datum kann aus dem EEPROM ausgelesen werden. Falls mit der Inbetriebnahmesoftware mit Save! oder ProgParam Parameter im EEPROM gespeichert wurden wird dieses Datum aktualisiert. Ein Anwenderprogramm, das Parameter im EEPROM ändert, sollte ebenfalls dieses Datum immer aktualisieren.

EE_ProgDatum
EEPROM: H' 58, H' 59

Darstellung: Wort, höherwertiges Byte H' 58

Bedeutung der Bits wie EE_InitDatum

14.9. Steuerung von ND21 über NOVOBUS

Die Steuerung über NOVOBUS erfolgt, abgesehen von wenigen Spezialbefehlen wie z. B. Reset, nach dem Prinzip, Speicherzellen im RAM des µControllers H8 zu lesen und zu schreiben. Dadurch werden die Steuerbytes beeinflußt und die Statusbytes können zurückgelesen werden.

14.10. Betriebszustände

Sperren:	Bit 0 Status (H' FF00) auf 1 setzen
Freigabe:	Bit 0 Status (H' FF00) auf 0 setzen
Stop:	Bit 7 Status (H' FF00) auf 1 setzen
Stop aufheben:	Bit 7 Status (H' FF00) auf 0 setzten
Reset auslösen:	NOVOBUS-Befehl ResetH8 ausführen

Der jeweilige Betriebszustand kann in den Speicherzellen: Status (H' FF00), Flags (H' FF56) und Flags2 (H' FF57) ausgelesen werden.

14.10.1. Fehlerzustand

Fehlerzustand abfragen:	Bit 5 Status (H' FF00) lesen
Fehlercode:	errorcode Adresse: H' FD82
Darstellung:	12 Bit, höherwertiges Byte H' FD82
Fehler quittieren:	H' AF in RAM Zelle errorcode (H' FD82) schreiben

14.10.2. NOVOBUS Timeout

Die NOVOBUS-Timeout-Überwachung wird mit Flags2 (H' FF57) Bit 2 = 1 aktiviert. Initialisiert wird dieses Bit aus dem EEPROM mit Bit 7 aus EE_selftest (H' 47).

Die Timeout-Überwachung spricht an, wenn nach kürzestens 10 ms und längstens 20 ms keine Daten auf NOVOBUS empfangen wurden. Dabei spielt es keine Rolle, ob die Daten für diesen Antrieb bestimmt waren, oder nicht.

14.10.3. Soll- Istwert-Austausch

Es gibt 2 Möglichkeiten Sollwerte über NOVOBUS vorzugeben und Istwerte zurückzulesen:

1. Im Parameterkanal mit den entsprechenden NOVOBUS-Befehlen die Sollwerte in die zugehörigen Speicherzellen eintragen. Die Istwerte aus den entsprechenden Speicherstellen zurücklesen **(siehe Kapitel 11 NOVOBUS-Treiber).**

2. Im Prozessdaten-Kanal schneller Soll- Istwert-Austausch.

In jedem Telegramm mit Prozessdaten-Kanal werden 2 Byte empfangen und 2 Byte weitergeschickt.

Empfangsadresse	DataIn	Adresse: H' FF32
Sendeadresse	DataOut	Adresse: H' FF34

Die ankommenden Daten werden in Speicherzellen im RAM-Bereich H' FFXX eingetragen. Die Adresse dieser Speicherzellen werden in DataIn eingetragen.

Beispiel: Die ankommenden Daten sollen Drehzahl-Sollwerte bedeuten: nsoll hat die Adresse H' FF08, also H' 08 in DataIn eintragen.

Die Sendedaten werden aus Speicherzellen im RAM-Bereich H' FFXX entnommen. Die Adresse dieser Speicherstellen werden in DataOut eingetragen.

Beispiel: Die Sendedaten sollen Lage-Istwerte bedeuten: lageist hat die Adresse H' FF18, also H' 18 in DataOut eintragen.

14.10.4. Positioniersteuerung

(siehe Kapitel 13. Die ND21-Positioniersteuerung)

14.10.5. EEPROM-Steuerung

14.10.5.1. EEPROM-Befehle

Die EEPROM-Befehle werden in die Speicherzelle:

EEPROMcontrol Adresse: H' FD8A im RAM-Speicher von ND21 eingetragen.

Befehl	**Code**
Read Byte	H' 81
Write Byte	H' 82
ProgParam	H' 84

14.10.5.2. Byte lesen

Die EEPROM-Adresse der Bytes im EEPROMbuffer (H' FD88) eintragen. H' 81 in EEPROMcontrol (H' FD8A) eintragen. Warten, bis Bit 5 in EEPROMcontrol den Wert 1 annimmt. Das Ergebnisbyte aus dem EEPROMbuffer auslesen. H' 10 in EEPROMcontrol eintragen.

14.10.5.3. Byte schreiben

Die EEPROM-Adresse des Bytes im EEPROMbuffer (H' FD88) eintragen. Das Datenbyte in Speicherzelle mit Adresse H' FD89 eintragen. H' 82 in EEPROMcontrol eintragen. Der Schreibvorgang ist abgeschlossen, sobald Bit 4 in EEPROMcontrol den Wert 1 annimmt.

14.10.5.4. ProgParam

Damit ist es möglich mit einem Befehl die Reglerparameter (Speicherbereich H' FF60 bis H' FF7F) ins EEPROM zu kopieren.

Dazu H' 84 in EEPROMcontrol eintragen. Der Kopiervorgang ist abgeschlossen, sobald Bit 4 in EEPROMcontrol den Wert 1 annimmt.

14.10.6. Oszilloskop

14.10.6.1. Signalauswahl

Es können Signale ausgewählt werden die sich im RAM-Speicher im Bereich H' FFxx befinden. Die Signalauswahl erfolgt, indem das niederwertige Byte der RAM-Adresse des gewünschten Signals als:

Signal1 in scope_signal1 H' FF40,

Signal2 in scope_signal2 H' FF41 oder

Triggersignal in scope_trigger H'FF3E,

eingetragen wird.

Beispiel: H' 0C in die RAM Zelle H' FF40 bewirkt, daß für Signal1 der Drehzahlistwert (RAM Zelle H' FF0C) ausgewählt wurde.

Falls nur ein Signal angesehen werden soll, in scope_signal1 und in scope_signal2 die gleiche Adresse eintragen.

Es werden die Inhalte der ausgewählten Speicherzellen aufgezeichnet. Die Bedeutung und die Skalierung der Aufzeichnung ergibt sich aus den ausgewählten Signalen selbst.

Beispiel: Falls mit H' 06 in RAM Zelle H' FF40, iaist als Signal1 ausgewählt wurde, bedeutet das:
iaist: Motorstrom-Istwert Phase A,
Zweierkomplement-Darstellung, bei ND21 5610:
0,244 A pro Bit.

Falls z. B. ein Wert von H' 4C aufgezeichnet wurde, entspricht das einem Stromwert von +18,5 A. Bei H' C8: -13,7 A.

Die Signalauswahl muß bei gesperrter Aufzeichnung erfolgen.

14.10.6.2. Zeitbasis

Die Abtastrate des Speicheroszilloskop wird in RAM Zelle scope_timer0 H' FF39 eingetragen. Dies bedeutet, daß Immer nach einer Abtastzeit die Speicherung eines Messwertes erfolgt.

$T_{AB} = 108\ \mu s \cdot$ scope_timer0

Es werden insgesamt 256 Messwerte eingelesen.

Beispiel: scope_timer = 0: Alle 108 µs wird ein Messwert im Scope Puffer gespeichert. Der volle Scope Puffer hat dann eine zeitliche Ausdehnung von 256 x 108 µs = 27,648 ms.

scope_timer = H' C8: Alle 200 x 108 µs = 21,6 ms wird ein Messwert im Scope Puffer gespeichert. Der volle Scope Puffer hat dann eine zeitliche Ausdehnung von 256 x 21,6 ms = 5,5296 s. Die Zeitbasis muß bei gesperrter Aufzeichnung eingestellt werden.

14.10.6.3. Triggerschwelle

Die Triggerschwelle wird mit invertiertem Bit7 in die RAM Zelle scope_level H' FF3F eingetragen.

Beispiel: Bei iaist soll bei -7 A getriggert werden.
7A bei ND21 5610: 7 / 0.244 = 29 Bit = H' 1D.
-7 A entspricht: -H' 1D = H' E3.
Bit 7 invertieren: Aus H' E3 wird H' 63
H' 63 in RAM Zelle scope_level eintragen.
Die Triggerschwelle kann immer verändert werden.

14.10.6.4. Triggerverzögerung

Zur Triggerverzögerung müssen die RAM Zellen scope_delay H' FF3C und scope_counter H' FF3A programmiert werden. Beide RAM Zellen müssen für jede Aufzeichnung neu programmiert werden. Die beiden Werte werden wie folgt berechnet:

Gegeben sei die gewünschte Verzögerungszeit T_{VZ}:

scope_counter = H' 0100 + T_{VZ} / T_{AB} für T_{VZ} >= 0

scope_delay = H' 00 für T_{VZ} < 0

scope_delay = -T_{VZ} / T_{AB}

Beispiel: Abtastzeit 108 µs, Verzögerungszeit -5ms. D. h. es sollen 46 Abtastpunkte vor, und 210 Abtastintervalle nach dem Triggerzeitpunkt aufgezeichnet werden.
scope_delay = 46 = H' 2E,
scope_counter = H' 0100 - H' 2E = H' 00D2

14.10.6.5. Scope Status

Das Byte scope_status H' FF38 steuert die Aufzeichnung.

Bit 7	Bezeichnet den Zustand des Oszilloskopes:
Bit 7 = 0:	Aufzeichnung läuft.
Bit 7 = 1:	Aufzeichnung gestoppt.
Bit 6	bezeichnet den Triggerstatus: 0: kein Trigger erfolgt 1: Triggerung erfolgt
Bit 5:	Triggerstatus, mit 1 initialisieren.
Bit 4:	Triggerflanke
Bit 3 .. 0	reserviert = 0

Bit 7 wird am Ende einer Aufzeichnung automatisch auf 0 gesetzt. Eine Aufzeichnung kann auch gestoppt werden, indem Bit 7 auf 0 gesetzt wird.

14.10.6.6. Aufzeichnungsablauf

Bei gestoppter Aufzeichnung werden scope_delay und scope_counter geladen. Anschließend wird scope_status mit H' 20 bei negativer und mit H' 30 bei positiver Flanke beschrieben, und damit die Aufzeichnung gestartet. Zuerst läuft die Delayzeit ab. Dann wird die Triggerbedingung geprüft.

Bit 5 geht auf Null, sobald das Triggersignal bei positiver Triggerflanke, die Triggerschwelle unterschritten, bzw. bei negativer Triggerflanke die Triggerschwelle überschritten hat. Bit 5 geht wieder auf 1, sobald die Triggerschwelle erneut passiert wurde. Damit wird das Triggersignal ausgelöst, Bit 6 wird auf 1 gesetzt. Sobald getriggert wurde läuft die in scope_counter programmierte Zeit ab.

Danach wird die Aufzeichnung automatisch gestoppt und Bit 7 in scope_status auf 1 gesetzt. Die aufgezeichneten Daten stehen jetzt im Scope Puffer zum Auslesen zur Verfügung. Die geraden Adressen im Scope Puffer sind mit Daten aus Signal1 belegt, die ungeraden Adressen mit Signal2.

Der Datenbereich sieht dann folgendermaßen aus:

Anfang: H' FDA0 + scope_pointer + 1
bis: H' FE9F
von: H' FDA0
Ende: H' FDA0 + scope_pointer.

14.10.6.7. Auto Trigger

Eine Aufzeichnung ohne Trigger kann erfolgen, indem Bit 7 im scope_status für die Dauer einer Aufzeichnung auf 0 und anschließend wieder auf 1 gesetzt wird.

14.11. Fehlercodes

14.11.1. H8 Port Fehler

P10 H'0010

P11 H'0011

P12 H'0012

P13 H'0013

P14 H'0014

P15 H'0015

P16 H'0016

P17 H'0017

P20 H'0020

P21 H'0021

P22 H'0022

P23 H'0023

P24 H'0024

P25 H'0025

P26 H'0026

P27 H'0027

P30 H'0030

P31 H'0031

P32 H'0032

P33 H'0033

P34 H'0034

P35 H'0035

P36 H'0036

P37 H'0037

P40 H'0040

P41 H'0041

P42 H'0042

P43 H'0043

P45 H'0045

P46 H'0046

P47 H'0047

P50 H'0050

P51 H'0051

P52 H'0052

P60 H'0060

P61 H'0061

P62 H'0062

P63 H'0063

P64 H'0064

P65 H'0065

P66 H'0066

P67 H'0067

P70 H'0070

P71 H'0071

P72 H'0072

P73 H'0073

P74 H'0074

P75 H'0075

P76 H'0076

P77 H'0077

P80 H'0080

P81 H'0081

P82 H'0082

P83 H'0083

P84 H'0084

P85 H'0085

P86 H'0086

P90 H'0090

P91 H'0091

P92 H'0092

P93 H'0093

P94 H'0094

P95 H'0095

P96 H'0096

P97 H'0097

14.11.2. EEPROM-Fehler

EEPROMrdwr	H'0100
EEPROMvoid	H'0101
EEPROMparity	H'0102
EPROMwr	H'0103
EEPROMbus	H'0104

14.11.3. A/D-Fehler

TempMot	H'0110
TempKK	H'0111
Strom A	H'0112
Strom B	H'0113
BitSplitting	H'0114
BitSplittingA	H'0115
BitSplittingB	H'0116

14.11.4. Fehler Analogeingang

AnInput0V	H'0120
AnInput10V	H'0121
AnInput12V	H'0122

14.11.5. Resolverfehler

ResolverIC	H'0130
RslvUeberw	H'0131
RAMSt	H'0132
RslvJustage	H'0133
ResolvKabel	H'0134
Rslv	H'0135

14.11.6. Watchdogfehler

WdogHw	H'0140
IntWdog	H'0141

14.11.7. ASIC-Fehler

ASICfr	H'0150
ASICread0	H'0151
ASICrdwr	H'0152
ASICtype	H'0153
ASICram	H'0154

14.11.8. Modulatorfehler

PhaseABC	H'0160
PhaseA	H'0161
PhaseB	H'0162
PhaseC	H'0163

14.11.9. Ablauffehler

PC	H'0201
IT	H'0202
reset	H'0203

14.11.10. Fehler H8 Register

SP	H'0210
SYSCR	H'0211
IER	H'0212
ISCR	H'0213
TMR0	H'0214
TMR1	H'0215
FRC	H'0216
CCR	H'0217
RAM	H'0218
R0	H'0220
R1	H'0221
R2	H'0222
R3	H'0223

14.11.11. Fehler Motorströme

imax	H'0250
ia	H'0251
ib	H'0252
ic	H'0253

14.11.12. PAL Fehler

PAL	H'0300
PALfrgabe	H'0301
5V	H'0302
15V	H'0303
ZkUeberSp	H'0304
ZkUnterSp	H'0305
Wdog	H'0306
Endstufe	H'0307
UeberStrom	H'0308
Resolver	H'0309
EndSchaltP	H'0310
EndSchaltN	H'0311
EndSchalter	'0314

14.11.13. Temperaturfehler

OverTempKK	H'0400
OverTempMot	H'0401
OverTempKK_	H'0410

14.11.14. Fehler serielle Schnittstelle

SCIoverrun	H'0501
SCIparity	H'0502
SCIframing	H'0503
NBsync	H'0504
NBcommand	H'0505
NBparam	H'0506
NBchecksum	H'0507
NBtimeout	H'0508
Bextern	H'0509
NBdata	H'0510

14.11.15. Fehler Positioniersteuerung

PS	H'0600
DisPS	H'0601

14.11.16. Fehler Regelung

Schleppfehler	'0700
Strom0	H'0701

14.11.17. Fehler Sollwerte

ImpulsEingang	H'0800

Anhang

A1. Zahlendarstellungen

A1.1. Zahlensysteme

Ein und die selbe Zahl kann in verschiedenen Zahlensystemen dargestellt werden.

A1.1.1. Das Dezimalsystem

Das Dezimalsystem, auch Zehnersystem genannt, ist das Zahlensystem, in dem der Mensch gewohnt ist zu rechnen. Das Zehnersystem kennt Ziffern von 0 - 9. Um größere Zahlen darzustellen, werden die Ziffern aneinander gereiht. Die Ziffer ganz rechts ist die niederwertigste Ziffer. Sie gibt die Einer einer Zahl an. Die nächste Ziffer bezeichnet die Zehner, die nächste die Hunderter, dann die Tausender usw.. Mit jeder Stelle kommt eine Zehnerpotenz hinzu. Mit einer dreistelligen Dezimalzahl lassen sich Zahlen von 000 - 999 darstellen.

A1.1.2. Das Binärsystem

Das Binärsystem wird auch Zweiersystem genannt. In diesen Zahlensystem rechnen alle Digitalrechner. Das Binärsystem kennt nur die Ziffern 0 und 1. Um größere Zahler darzustellen, werden die Ziffern aneinander gereiht. Die Ziffer ganz rechts ist die niederwertigste Ziffer. Sie gibt die Einer an. Die nächste Ziffer bezeichnet die Zweier, die nächste die Vierer, dann die Achter, die Sechzehner usw.. Mit jeder Stelle kommt eine Zweierpotenz hinzu. Mit einer dreistelligen Binärzahl lassen sich Zahlen von 000 bis 111 darstellen. Zum Beispiel bedeutet 101: 1 Einer + 0 Zweier + 1 Vierer = 5 im Dezimalsystem.

A1.1.3. Das Hexadezimalsystem

Das Hexadezimalsystem wird auch Sechzehner-System genannt. Große Zahlen im Binärsystem sind recht lang und umständlich aufzuschreiben. Deshalb bedient man sich des Hexadezimal-systems. 4 Stellen (oder Bits) im Binärsystem entsprechen einer Stelle im Hexadezimalsystem. Das Hexadezimalsystem kennt die Ziffern 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Die Ziffern A bis F haben die Wertigkeiten 10 bis 15. Um größere Zahler darzustellen, werden die Ziffern aneinander gereiht. Die Ziffer ganz rechts ist die niedrigste Ziffer, die die Einer angibt.

Die nächste Ziffer bezeichnet die Sechzehner, die nächste die Zweihunderd-Sechsundfünfziger, dann die 4096'er, die 65536' usw.. Mit jeder Stelle kommt eine Sechzehnerpotenz hinzu. Mit einer dreistelligen Hexadezimalzahl lassen sich Zahlen von 000 bis FFF darstellen. Zum Beispiel bedeutet 3E1: 1 Einer + 14 16'er + 3 256'er = 993 im Dezimalsystem.

A1.2. Zahlen im Digitalrechner

A1.2.1. Bit

Kleinste Einheit. Ein Bit kann die Werte 0 und 1 annehmen

A1.2.2. Byte

8 Bit. Achtstellige Binärzahl. Die acht Bits werden von 0 bis 7 durchnummeriert. Das niederwertigste Bit ist Bit Nummer 0.

Ein Byte kann auch als zweistellige Hexadezimalzahl aufgeschrieben werden. Mit einem Byte kann man Zahlen von 0000 0000 bis 1111 1111 (entspricht H' 00 bis H' FF im Hexadezimalsystem oder 0 - 255 im Dezimalsystem) darstellen.

A1.2.3. Wort

16 Bit, 2 Byte. Ein Wort enthält die Bits 0 bis 15. Ein Wort ist eine sechzehnstellige Binärzahl, das entspricht einer vierstelligen Hexadezimalzahl. Mit einem Wort können Zahlen von 0000 0000 0000 0000 bis 1111 1111 1111 1111 (entspricht H' 0000 bis H' FFFF oder dezimal 0 bis 65535) darstellen.

A1.2.4. Doppelwort

32 Bit, 4 Byte. Ein Doppelwort enthält die Bits 0 bis 31. Ein Doppelwort ist eine 32-stellige Binärzahl, das entspricht einer achtstelligen Hexadezimalzahl. Mit einem Doppelwort können Zahlen von 0000 0000 0000 0000 0000 0000 0000 0000 bis 1111 1111 1111 1111 1111 1111 1111 1111 (entspricht H' 00000000 bis H' FFFFFFFF oder dezimal 0 bis 4294967295) darstellen.

A1.3. Negative Zahlen

Mit Byte, Wort und Doppelwort sind bisher nur positive Zahlen beschrieben. Um negative Zahlen darzustellen definiert man:

Für: Byte 0 - 127 positiv

128 - 255 negativ, Zahlenwerte -128 - -1

Wort: 0 - 32767 positiv

32768 - 65536 negativ, Zahlenwerte -32768 - -1

Doppelwort: 0 - 2147483647 positiv

2147483648 - 4294967295 negativ,

Zahlenwerte -2147483647 - -1

Das bedeutet bei Byte zum Beispiel: Wenn man zu 0 1 addiert, so erhält man 0000 0001 also 1. Wenn man von 0 1 sutrahiert, so erhält man 1111 1111 (H' FF oder 255 dezimal) das bedeutet nach obiger Definition -1.

Man negiert eine Zahl, indem man das sogenannte Zweierkomplement bildet. Ein Komplement einer Binärzahl erhält man, indem man aus jeder Null eine Eins macht, und aus jeder Eins eine Null. Das Zweierkomplement erreicht man, wenn man nach der Komplementbildung noch 1 addiert.

Im obigen Beispiel:

1 = 0000 0001

Komplement: 1111 1110 **+ 1:** 1111 1111 = -1

Dezimal 25= x 128 + 0 x 64 + 0 x 32 + 1 x 16 + 1 x 8 + x 4 0 x 2 1 x 1 = 0001 1001

Komplement: 1110 0110 **+ 1:** 1110 0111 = - 25

Hinweis: Die Zweierkomplementbildung muß immer auf alle Bits angewendet werden. Das heißt, bei Zahlen die in einem Byte dargestellt werden, müssen 8 Bit komplementiert werden, bei Zahlen die mit einem Wort dargestellt werden, 16 Bit und bei Doppelwort 32 bit.