Benutzerhandbuch

DC-Bürstenlosmotorregler

RS Stock No.: 206417

1. Produktbezeichnung

Bürstenlosmotorregler RS 206417 sind elektronische Geräte, die für den Betrieb und die Steuerung von bürstenlosen synchronen 3-Phasen-Gleichstrommotoren mit Hall-Encoder entwickelt wurden.

2. Funktionen und Möglichkeiten

Die Regler sind für die Steuerung von Drehzahl, Beschleunigung, Verzögerung und Drehrichtung des Motors ausgelegt. Die Geräte ermöglichen auch die Positionierung basierend auf Signalen von eingebauten Hall-Sensoren. Positionshaltung ist ebenfalls möglich.

Der bürstenlose Motor wird entweder durch externe Signale oder durch Befehle gesteuert, die über RS-485 mit dem Modbus-Protokoll übertragen werden. Der Regler kann auch nach einem Algorithmus arbeiten, der zuvor vom Benutzer im Blockspeicher aufgezeichnet wurde.

Steuerung über RS-485 Modbus

Der Regler kann über die physische RS-485-Kommunikationsleitung unter Verwendung des industriellen Modbus-Protokolls ferngesteuert werden:

- Die Einstellung erfolgt durch Schreiben oder Lesen der entsprechenden Parameter in/aus den Reglerregistern.
- Unterstützte Protokolle sind RTU und ASCII, Geschwindigkeiten sind 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600, 115200, 128000 Baud.
- Bewegungen zu einer voreingestellten Zielposition oder zu einem von vier vordefinierten Referenzpunkten werden ausgeführt.
- Der Regler kann autonom unter der Steuerung eines Benutzerprogramms (bis zu 1024 Befehle) arbeiten, das im nichtflüchtigen Speicher vorgespeichert ist. Bedingte und unbedingte Sprünge (relativ und absolut), Unterprogrammaufrufe, Schleifen, Timer werden unterstützt, logische und mathematische Operationen mit Daten werden bereitgestellt.
- Programmierbare Eingänge IN1 und IN2 können als Signale START/STOP, REVERS oder für andere Zwecke nach Ermessen des Benutzers verwendet werden.
- Der Regler kann eine Positionierung im Bereich von -2147483647 bis +2147483648 Hall-Sensor-Schaltungen durchführen.
- Der Regler verfügt über RT-Kontakte an der Vorderseite für den Anschluss eines Abschlusswiderstands.

Steuerung über externe Signale

Zur Steuerung des Motors durch externe Signale stehen folgende Möglichkeiten zur Verfügung:

- Klemmen zum Anschluss eines Potentiometers oder eines externen Signals 0-5VDC für die analoge Drehzahlregelung.
- Kontakte zum Anschluss der externen Signale In1 und In2, deren Zweck und Verarbeitung vom Benutzer festgelegt wird.Die Eingänge können auch als START/STOP (Bewegungsstart/Stopp) und DIR (Richtung) Signale verwendet werden.
- HARD STOP-Kontakt zum Anschluss eines Alarmsignals, der eine kontrollierte Motorbremsung im Falle einer Notstromunterbrechung gewährleistet. Nach dem Stoppen schaltet der Motor in den stromlosen Modus, ein versehentliches Starten wird verhindert.

3. Technische Eigenschaften

Modell	RS 206417
Versorgungsspannung, VDC	24 - 48
Schutz der Stromversorgung, VDC	20 - 51
Bemessungs-Motorstrom, A	<20
Max. Motorstrom, A	<30
Eingangswiderstand des SPEED-Eingangs, kOhm	20
Eingangsspannungsbereich des SPEED-Eingangs, VDC	0..5
Abmessungen (nicht mehr), mm	116x100x23

Kommunikationsschnittstelle	RS-485, Modbus – ASCII oder RTU

Die Gesamt- und Anschlussabmessungen des Reglers sind in Abb. 1 dargestellt.

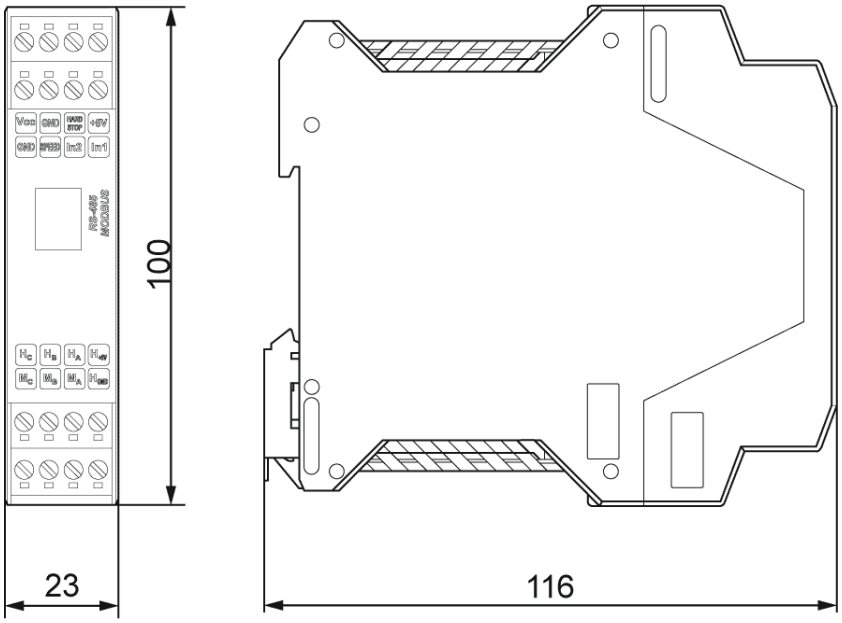

Abb.1. Die Gesamt- und Anschlussabmessungen des Reglers RS 206417

Das Anschlussschema ist in Abb. 2 dargestellt.

Umgebungsbedingungen:

Umgebungstemperatur: 0...+50°C

Luftfeuchtigkeit: 90% RH oder weniger bei +25°C

Kondensation und Gefrieren: keine

4. Aufbau und Bedienelemente

RS 206417 ist als Leiterplatte mit elektronischen Elementen aufgebaut, die mit einem Gehäuse mit DIN-Schienenhalterung abgedeckt ist. Auf der Oberseite des Gehäuses befinden sich grafische Symbole für die Bedienelemente und die Pinbelegung. Neben elektronischen Komponenten befinden sich auf der Platine Anzeige- und Bedienelemente, Anschlussklemmen und Steckverbinder:

- Schraubklemmen zum Anschluss der Stromversorgung, der bürstenlosen Motorwicklungen, der Encoderleitungen und des Steuerkreises;
- - Klemmen IN1 und IN2 zum Anschluss von Steuereingangssignalen;
- - Klemmen zum Anschluss eines externen Potentiometers zur Regelung der Motordrehzahl;
- - Klemme zum Anschluss von Not-Aus-Signalkontakten;
- - LED-Anzeige für den Gerätebetrieb;
- - RJ11 (6P6C) Anschluss zum Anschluss von RS-485 Leitungen;
- - Kontakte zum Anschluss des internen Abschlusswiderstands RT;
- - eingebaute Bremschaltung zur Absorption der vom Motor erzeugten Energie (beim Auslaufen, erzwungenen Drehung).

Der Eingang „SPEED“ ist für die Drehzahlregelung durch ein externes analoges Signal vorgesehen. Ein externer Signaleingang "HARD STOP" ist für den Notstopp des Motors vorgesehen. Die externen Eingänge IN1 und IN2 können verwendet werden, um den Motor über externe Signale zu starten und zu stoppen sowie die Drehrichtung des Motors zu steuern.

Alle Parameter des Motorbetriebs und der Bewegungssteuerung können per Software und durch Befehle, die über RS-485 über das Modbus-Protokoll übertragen werden, ausgeführt werden.

Die Anordnung und der Zweck der Klemmen sind in Abb. 2 dargestellt.

1. Ausgang +5 VDC für externes Potentiometer
2. Not-Halt-Signal „HARD STOP“
3. Stromversorgung GND
4. Stromversorgung 24 – 48 VDC
5. Signal „IN1“ (reiner Kontakt)
6. Signal „IN2“ (reiner Kontakt)
7. Analogeingangssignal – zum Anschluss eines externen Drehzahlregelpotentiometers
8. Signalmasse
9. Ausgang für die Versorgung der Hallsensoren
10. Hallsensor – Phase A
11. Hallsensor – Phase B
12. Hallsensor – Phase C
13. GND der Hallsensoren
14. Motorphase A
15. Motorphase B
16. Motorphase C

RS-485 Modbus - RJ12 (6P6C) Anschluss für den Anschluss von RS-485 Datenleitungen

RT - Anschlüsse für den Anschluss des Abschlusswiderstands

Abb. 2. Layout und Zweck der Anschlüsse und Bedienelemente

5. Montage und Anschluss

Bitte lesen Sie diese Anleitung vor dem Anschluss und der Montage sorgfältig durch.

Bitte verkabeln Sie nur bei ausgeschalteter Stromversorgung. Versuchen Sie nicht, die Verkabelung bei eingeschalteter Stromversorgung zu ändern.

Bitte sorgen Sie für einen zuverlässigen Kontakt in den Anschlussklemmen. Beachten Sie bei der Verdrahtung die Polarität und die Kabelbelegung. Verpolung und Überspannung beschädigen den Controller.

Befolgen Sie beim Anschluss die folgenden Anweisungen:

1. Schließen Sie einen Motor gemäß Abb. 2 an den Controller an. Die Motorphasen müssen an die Klemmen 14 - 16 angeschlossen werden. Die Signale der Hallsensoren müssen an die Klemmen 10 - 12 angeschlossen werden. GND der Hallsensoren muss an Klemme 13 angeschlossen werden, die Versorgung der Hallsensorsignale muss an Klemme 9 angeschlossen werden.

2. Schließen Sie externe Steuerelemente gemäß den Anschlussplänen in Abb. 3 an:

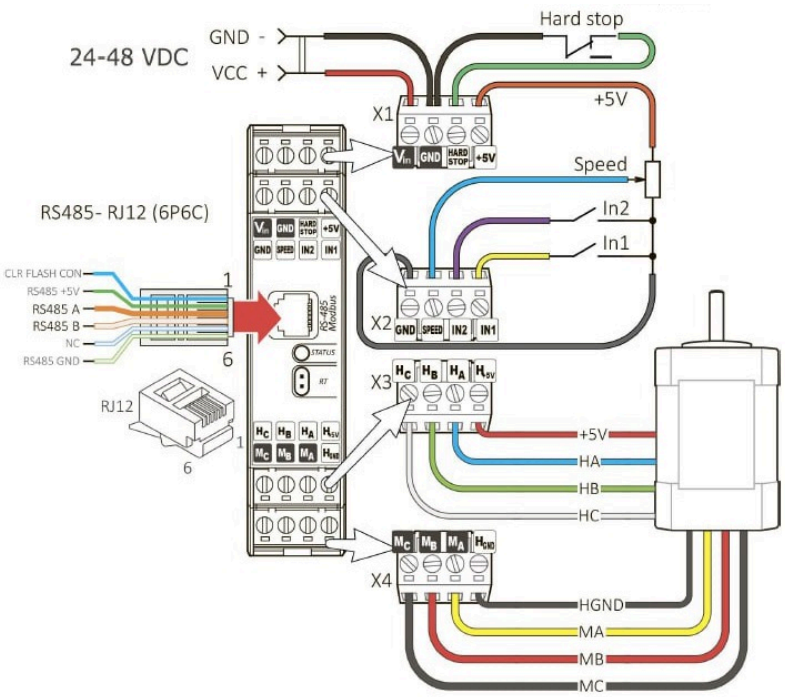

Abb. 3. Anschluss von Stromversorgung und externen Bedienelementen

- Art der externen Signale «IN1», «IN2», «HARD STOP» - sauberer Kontakt;

- Gesamtwiderstand des externen Potentiometers zur Drehzahlregelung - ca. 4..5KOhm.

3. Schließen Sie die Leitungen der RS-485-Schnittstelle gemäß Abb. 4 an den RJ11-Stecker an.

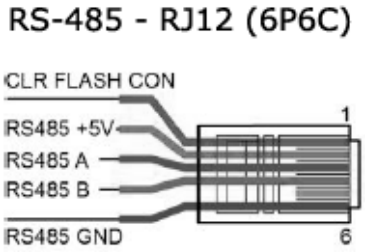

Abb. 4. Pinbelegung des RJ12-Steckers - RS-485 Modbus

4. Setzen Sie bei Bedarf eine Brücke auf die RT-Pins, um den internen Abschlusswiderstand anzuschließen.

5. Schließen Sie das Gerät unter Beachtung der Polarität an das Netzteil an. Das Netzteil muss mit einem Sicherheitszuschlag gewählt werden (um Spannungseinbrüche zu vermeiden). Die Dicke der Anschlussleitungen muss dem Stromverbrauch des Motors entsprechen. Schließen Sie "+" des Netzteils an Eingang 4 an, schließen Sie "-" des Netzteils an Eingang 3 an. Die Demontage erfolgt in umgekehrter Reihenfolge.

6. Betrieb

1. Stellen Sie sicher, dass die Stromversorgung ausgeschaltet ist. Bitte verkabeln Sie nur bei ausgeschalteter Stromversorgung.

2. Schließen Sie den Motor und das Netzteil gemäß Abschnitt 5 an den Controller an.

3. Wählen Sie die Steuerungsmethode: Steuerung durch Befehle über Modbus oder externe Signale (Register MODE_DEVICE - siehe Abschnitt 6.5, Abb. 6 und Abb. 7).

4. Um die Drehzahl über eine externe Signalquelle einzustellen, schließen Sie einen externen Widerstand an die Klemmen 1 "+5 V", 7 "SPEED" und 8 "GND" an. Der minimale Widerstand entspricht der maximalen Drehzahl; mit zunehmendem Widerstand nimmt die Drehzahl ab. Um die Drehzahl über ein externes analoges Signal steuern zu können, stellen Sie das Register USE_EXTERN_SPEED = 1 ein (siehe Beschreibung der Register unten).
6. Schließen Sie die Steuersignale "IN1", "IN2" und das Not-Aus-Signal "HARD STOP" gemäß Abschnitt 5 an. Das Signal "HARD STOP" dient zum Notstopp des Motors. Der Betrieb ist bei geschlossenem Kontakt zulässig.
7. Schließen Sie die Leitungen der RS-485-Schnittstelle gemäß Abschnitt 5 an.
8. Schalten Sie die Stromversorgung ein. Das Gerät ist bereit für die Konfiguration über das Modbus-Protokoll.
9. Stellen Sie die notwendigen Betriebsparameter über Befehle via Modbus-Protokoll ein - Motorstrombegrenzung, Drehrichtung, Einstellung der Funktion der externen Eingänge IN1 und IN2, Steuerungsmethode.
10. Um den Antrieb über das Modbus-Protokoll zu steuern, senden Sie Befehle über die RS-485-Schnittstelle.Die Registertabelle des Controllers, deren Zweck und mögliche Motorsteuerbefehle sind unten aufgeführt.
11. Um den Antrieb mit externen Signalen zum Starten und Stoppen des Motors und zur Steuerung der Drehrichtung zu steuern, verwenden Sie die Signale IN1 und IN2.Die Drehzahlregelung erfolgt durch ein externes analoges Signal, das an den SPEED-Eingang angelegt wird, oder durch Befehle über das Modbus-Protokoll.

MODBUS-Steuerung

Für die Datenübertragung über die RS-485-Schnittstelle wird das Standard-Modbus-Kommunikationsprotokoll (ASCII oder RTU) verwendet.

Die Steuereinheit hat folgende Werkseinstellungen:

- ID = 1
- Rate: 115200 Baud
- Paritätsprüfung: gerade
- Datenbits: 8
- Stoppbit: 1
- MODBUS RTU

6.1. Eingabesteuerregister

Adresse	Typ	Name	Größe	Beschreibung
1000h	Discrete Input	IN1_bit	1-Bit	Status des Eingangssignals IN1
1001h	Discrete Input	IN2_bit	1-Bit	Status des Eingangssignals IN2
1002h	Discrete Input	IN_HARD_STOP_bit	1-Bit	Status des Eingangssignals HARD_STOP
5007h	Holding Register	MODE_EXT_IN	16-Bit	Konfiguration der externen Eingänge IN, IN2.
5013h	Holding Register	PRESSED_INPUTS_EXTERN	16-Bit	Minimale positive Impulsdauer an den digitalen Eingängen IN1, IN2 (ms)
5014h	Holding Register	WAITED_INPUTS_EXTERN	16-Bit	Mindestzeit für den negativen Teil des Impulses an den digitalen Eingängen IN1, IN2 (ms)

Eingangsstatusregister 1000h..1002h sind schreibgeschützt.

1000h -**IN1_bit**– repräsentiert den Zustand des physikalischen Eingangs IN1.

1001h –**IN2_bit**– repräsentiert den Zustand des physikalischen Eingangs IN2.

1002h -**IN_HARD_STOP_bit**– repräsentiert den Zustand des physikalischen Eingangs HARD_STOP.

Mögliche Registerwerte 1000h..1002h:

- 1 – Eingang kurzgeschlossen mit Ausgang GND
- 0 – Eingang offen mit Ausgang GND

5007h -**MODE_EXT_IN**– der Registerwert bestimmt den Zweck und die Methode der Signalverarbeitung von IN1 und IN2 (siehe Beschreibung in Abschnitt 6.5).

5013h –**PRESSED_INPUTS_EXTERN** und 5014h –**WAITED_INPUTS_EXTERN**– die minimale Zeit (eingestellt in ms) der positiven und negativen Teile des Impulses an den digitalen Eingängen IN1, IN2 - Register werden verwendet, um Kontaktprellen zu unterdrücken (siehe Abb. 5).

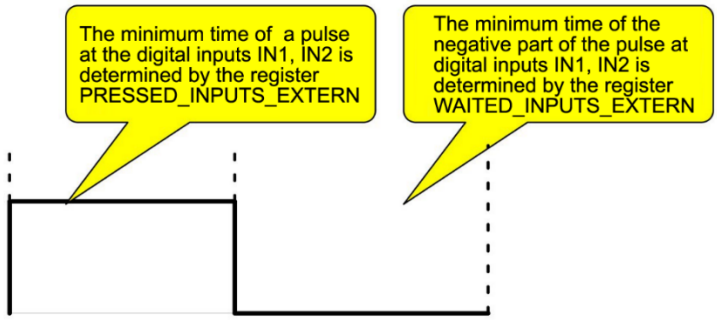

Abb. 5. Kontaktprellunterdrückung

6.2. Motorsteuerungsregister

Adresse	Typ	Name	Größe	Beschreibung
2000h	Coils	START_bit	1-Bit	Motorrotation mit eingestellter Beschleunigung starten
2001h	Coils	STOP_bit	1-Bit	Motorrotation mit eingestellter Verzögerung stoppen
2002h	Coils	HARD_STOP_bit	1-Bit	Abrupter Notstopp der Motorrotation
2003h	Coils	CLR_POSITION_bit	1-Bit	Zurücksetzen des aktuellen Positionsregisters POSITION_VALUE auf Null.
2004h	Coils	INV_DIR_bit	1-Bit	Umkehren der Motordrehrichtung
2005h	Coils	INV_CNT_bit	1-Bit	Zählrichtung der Position umkehren
2006h	Coils	ACC_ON_bit	1-Bit	Aktivierung der durch das ACC-Register festgelegten Beschleunigung
2007h	Coils	DEC_ON_bit	1-Bit	Aktivierung der durch das DEC-Register festgelegten Verzögerung
2008h	Coils	FQ_REGULATOR_ON_bit	1-Bit	Aktivierung der Vier-Quadranten-Drehzahlregelung
2009h	Coils	AUTO_HOLD_MODE_bit	1-Bit	Aktivierung des automatischen Haltemodus

Diese Steuerregister sind sowohl lesbar als auch schreibbar. Beim Schreiben in das entsprechende Register aktiviert der Wert 1 die jeweilige Funktion. Die Register 2000h...2003h werden nach Anwendung der Funktion auf 0 zurückgesetzt.

6.3. Statusregister

Adresse	Typ	Name	Größe	Beschreibung
3000h	Input Register	STATUS	16-Bit	Der aktuelle Zustand der Motorsteuerung.
3001h	Input Register	CURRENT_VALID	16-Bit	Der aktuelle Wert des vom Motor verbrauchten Stroms.
3002h	Input Register	SPEED_VALID	16-Bit	Der Momentanwert der Drehzahl.
3003h	Input Register	CURRENT_POSITION	32-Bit	Aktuelle Position. Wertebereich von -2147483647 bis +2147483648
3005h	Input Register	TEMPERATURE_ MCU	16-Bit	CPU-Temperatur
3006h	Input Register	TEMPERATURE_ MOSFET	16-Bit	Temperatur des Leistungskreises
3007h	Input Register	TEMPERATURE_ BRAKE	16-Bit	Bremsstromkreis-Temperatur
3008h	Input Register	TASK_COUNTER	16-Bit	Gibt einen Zufallswert zurück (um die Funktion des Übertragungskanals zu überprüfen)
3009h	Input Register	STATUS_USER_PROGRAM	16-Bit	Der aktuelle Status des Anwenderprogramms
300Ah	Input Register	SPEED_INPUT_VALUE	16-Bit	ADC-Messwerte des SPEED-Eingangs

Diese Registergruppe ist schreibgeschützt und stellt den aktuellen Zustand des Controllers dar.

3000h –**STATUS**- aktueller Zustand der Motorsteuerung, mögliche Werte:

0 Motorstopp
1 Vorwärtslauf
2 Rückwärtslauf

3001h -**CURRENT_VALID**- der aktuelle Wert des vom Motor verbrauchten Stroms. Stellt den Wert des vom Motor aus der Stromversorgung verbrauchten Stroms dar, der Wert in mA.

3002 -**SPEED_VALID**- der momentane Wert der Drehzahl. Die Maßeinheit sind Umdrehungen pro Minute.

3003h -**CURRENT_POSITION**– die aktuelle Position. Die Positionierung erfolgt im Wertebereich von -2147483647 bis + 2147483648 der Gesamtzahl der Schaltungen des Hall-Sensors, wobei sowohl Vorder- als auch Hinterflanken berücksichtigt werden.

3005h -**TEMPERATURE_MCU**— Temperatur des Zentralprozessors. Die Temperatur wird als TEMPERATURE_MCU/10 Grad/C berechnet.

3006h -**MOSFET-TEMPERATUR**- Leistungskreistemperatur - die Temperatur im Installationsbereich der MOSFET-Schalter. Die Temperatur wird als TEMPERATURE_MOSFET/10 Grad/°C berechnet.

3007h - **TEMPERATURE_BRAKE** - Bremskreistemperatur – die Temperatur im Bereich des Bremswiderstands. Die Temperatur wird als TEMPERATURE_MOSFET/10 Grad/°C berechnet.

3008h -**TASK_COUNTER**- liefert eine Zufallszahl im Bereich von 0x0000 bis 0xFFFF.

3009h -**STATUS_USER_PROGRAM** - Der aktuelle Status des Benutzerprogramms, mögliche Werte:

1 – Benutzerprogramm wird durch einen Befehl über Modbus gestoppt
2 – Benutzerprogramm wird durch einen Befehl über Modbus gestartet (dieser Zustand ist nur von kurzer Dauer, bevor Zustand 4 gesetzt wird)
3 – Benutzerprogramm wird gestartet, nachdem die Versorgungsspannung eingeschaltet wurde (dieser Zustand ist nur von kurzer Dauer, bevor Zustand 4 gesetzt wird)
4 – Benutzerprogramm wird ausgeführt
5 – Benutzerprogramm wurde mit dem Befehl END beendet
6 – Benutzerprogramm wurde mit dem Befehl ENDF beendet
7 – Benutzerprogramm wurde aufgrund eines Fehlers gestoppt

300Ah - SPEED_INPUT_VALUE - Messwerte des Analog-Digital-Wandlers des SPEED-Eingangs (Messung in relativen Einheiten).

6.4. RS-485 Modbus Datenübertragungs-Einstellungsregister

Adresse	Typ	Name	Größe	Beschreibung
5000h	Holding Register	SLAVE_ADDRESS_MODBUS	16-Bit	ID des Controllers (Geräteadresse).Gültige Werte: 0.. .247. (0 - Broadcast-Adresse, keine Antwortnachricht)
5001h	Holding Register	TYPE_MODBUS	16-Bit	Einstellungen der Datenübertragung
5002h	Holding Register	BITRATE_MODBUS	16-Bit	Einstellen der Baudrate
5003h	Holding Register	TIMEOUT_BROADCAST_MODBUS	16-Bit	Zusätzliche Verzögerung zwischen dem empfangenen Paket und der Antwortnachricht.

5001h -**TYPE_MODBUS**– Einstellungen der Datenübertragung, gültige Werte:

- 1 – ASCII, 7 Datenbits, gerade, 1 Stoppbit,
- 2 – ASCII, 7 Datenbits, ungerade, 1 Stoppbit
- 3 – RTU, 8 Datenbits, gerade, 1 Stoppbit
- 4 – RTU, 8 Datenbits, ungerade, 1 Stoppbit
- 5 – RTU, 8 Datenbits, keine, 2 Stoppbits

5002h -**BITRATE_MODBUS**– Modbus-Baudrate, mögliche Werte:

- 0 – 600
- 1 – 1200,
- 2 – 2400,
- 3 – 4800,
- 4 – 9600,
- 5 – 14400,
- 6 – 19200,
- 7 – 38400,
- 8 – 57600,
- 9 – 115200,
- 10 – 128000

5003h -**TIMEOUT_BROADCAST_MODBUS**- wird verwendet, wenn das Steuergerät eine lange Zeit benötigt, um vom Sendemodus in den Empfangsmodus zu wechseln.

Nach dem Ändern der Kommunikationseinstellungen müssen die neuen Werte mithilfe des Registers FLAG_SAVE_INI gespeichert (siehe Abschnitt 6.5) und anschließend der Controller neu gestartet werden. Nach dem Neustart wird die RS-485-Verbindung mit den neuen Einstellungen hergestellt.

6.5. Register zur Einstellung des Antriebsbetriebs

Adresse	Typ	Name	Größe	Beschreibung
5004h	Holding Register	MODE_DEVICE	16-Bit	Betriebsart des Controllers
5005h	Holding Register	MODE_USER_PROGRAM	16-Bit	Steuerbefehl zum Starten eines Anwenderprogramms
5006h	Holding Register	MODE_ROTATION	16-Bit	Drehmodus.
5007h	Holding Register	MODE_EXT_IN	16-Bit	Konfiguration der externen Eingänge IN, IN2.
5008h	Holding Register	POSITION_N	16-Bit	Die Positionsnummer, zu der gefahren werden soll. Wertebereich von 1 bis 4
5009h	Holding Register	REF_CURRENT	16-Bit	Begrenzung des maximalen Betriebsstroms in der Motorwicklung. Bereich von 1000 mA bis 20000 mA
500Ah	Holding Register	HOLD_CURRENT	16-Bit	Haltestrom im Stopp-Modus. Bereich von 0 bis 1000 mA

500Bh	Holding Register	SPEED	16-Bit	Drehzahl einstellen. Bereich von 35 bis 14000 U/min.
500Ch	Holding Register	ACC	16-Bit	Beschleunigung einstellen. Bereich von 10 bis 1000.
500Dh	Holding Register	DEC	16-Bit	Bremsverzögerung einstellen. Bereich von 10 bis 1000.
500Eh	Holding Register	DIRECTION	16-Bit	Drehrichtung. 1 – vorwärts 2 – rückwärts
500Fh	Holding Register	N_POLE	16-Bit	Anzahl der Motorpole. Bereich von 1 bis 12
5010h	Holding Register	USE_EXTERN_SPEED	16-Bit	Drehzahlregelungsmethode: 1 – Befehle über Modbus 2 – externes Signal am SPEED-Eingang
5011h	Holding Register	RESERVED	16-Bit	reserviert
5012h	Holding Register	OFFSET_COMPENSATION	16-Bit	Motorbremskorrektur
5013h	Holding Register	PRESSED_INPUTS_EXTERN	16-Bit	Minimale positive Impulsdauer an den digitalen Eingängen IN1, IN2 (ms)
5014h	Holding Register	WAITED_INPUTS_EXTERN	16-Bit	Mindestzeit für den negativen Teil des Impulses an den digitalen Eingängen IN1, IN2 (ms)
5015h	Holding Register	OFFSET	32-Bit	Der Versatz zum Bewegen. Wertebereich von -2147483647 bis +2147483648
5017h	Holding Register	OFFSET_CONST	32-Bit	Inkrement – der Versatz, um den eine Bewegung erfolgen muss. Der Wert wird vom Controller während des Betriebs geändert.
5019h	Holding Register	TARGET_POSITION	32-Bit	Die zu bewegende Position. Wertebereich von -2147483647 bis +2147483648
501Bh	Holding Register	TARGET_POSITION1	32-Bit	Vom Benutzer voreingestellte Position Nr. 1 Wertebereich von -2147483647 bis +2147483648
501Dh	Holding Register	TARGET_POSITION2	32-Bit	Benutzerdefinierte Position Nr. 2 Wertebereich von -2147483647 bis +2147483648
501Fh	Holding Register	TARGET_POSITION3	32-Bit	Vom Benutzer voreingestellte Position Nr. 3 Wertebereich von -2147483647 bis +2147483648
5021h	Holding Register	TARGET_POSITION4	32-Bit	Benutzerdefinierte Position Nr. 4

				Wertebereich von -2147483647 bis +2147483648
5023h	Holding Register	ERROR	16-Bit	Fehlerregister
5024h	Holding Register	FLAG_SAVE_INI	16-Bit	Register zur Speicherung von Benutzereinstellungen.(Register 5000h..501Fh). Wert: 0x37FA
5025h	Holding Register	FLAG_SAVE_USER_PROGRAM	16-Bit	Register zum Schreiben oder Lesen eines Anwenderprogramms in den oder aus dem nichtflüchtigen Speicher. Zu schreibender Wert: 0x8426 Auszulesender Wert: 0x9346
5026h	Holding Register	FLAG_RESTART	16-Bit	Register zum Neustart. Wert: 0x95AF

5004h -**MODE_DEVICE**-Betriebsmodus des Controllers, mögliche Werte:

- 1 – Steuerung über Modbus
- 2 – Steuerung durch externe Signale

5005h –**MODE_USER_PROGRAM**– Steuerbefehl zum Starten eines Benutzerprogramms, mögliche Werte:

- 1 – Benutzerprogramm stoppen
- 2 – Benutzerprogramm starten
- 3 – Benutzerprogramm beim Einschalten starten (Vorspeicherung der Einstellungen erforderlich - siehe Register FLAG_SAVE_INI).

5006h –**MODE_ROTATION**– Drehmodus, mögliche Werte:

- 1 – kontinuierliche Drehung
- 2 – Versatz um den durch das OFFSET-Register festgelegten Betrag
- 3 – Bewegung zu einer gegebenen Position. Die Positionsnummer wird durch das Register POSITION_N angegeben (von 1 bis 4), die Koordinaten der Positionen werden durch die Register POSITION1, POSITION2, POSITION3, POSITION4 entsprechend angegeben.

5007h –**MODE_EXT_IN**– Konfiguration der Betriebsart der externen Eingänge IN1, IN2 im Steuermodus externer Signale, mögliche Werte:

- 1 – Eingang IN1 wird als Start-/Stoppsignal des Antriebs verwendet, er wird an der fallenden Flanke des Impulses verarbeitet; Eingang IN2 wird als Umkehrsignal verwendet, er wird an der fallenden Flanke des Impulses verarbeitet.
- 2 - Eingang IN1 wird als Start-/Stoppsignal des Antriebs verwendet, er wird an der fallenden Flanke des Impulses verarbeitet; Eingang IN2 wird als Drehrichtungs-Referenzsignal verwendet, verarbeitet entsprechend dem Signalpegel.
- 3 - Eingang IN1 wird als Start-/Stoppsignal des Antriebs verwendet, er wird entsprechend dem Signalpegel verarbeitet: , Signal vorhanden - Motorrotation aktivieren, kein Signal - Stopp; Eingang IN2 wird als Umkehrsignal verwendet, er wird an der fallenden Flanke des Impulses verarbeitet.
- 4 – Eingang IN1 wird als Start-/Stoppsignal des Antriebs verwendet und entsprechend dem Signalpegel verarbeitet: Signal vorhanden - Motorrotation aktivieren, kein Signal - Stopp; Eingang IN2 wird als Richtungsreferenzsignal verwendet und entsprechend dem Signalpegel verarbeitet.
- 5 – Eingang IN1 wird als Signal zum Starten und Stoppen des Antriebs in Vorwärtsrichtung verwendet, IN2 – als Signal zum Starten und Stoppen des Antriebs in Rückwärtsrichtung; beide Signale werden nach Pegel verarbeitet.

5008h –**POSITION_N**– Auswahl der Positionsnummer für die Bewegung (von 1 bis 4) - wird im Modus der Bewegung zu einer gegebenen Position (MODE_ROTATION = 3) in Verbindung mit den Registern POSITION1, POSITION2, POSITION3, POSITION4 verwendet.

5009h –**REF_CURRENT**– Einstellung des maximalen Betriebsstroms in der Motorwicklung – von 1000 mA bis 20000 mA

500Ah –**HOLD_CURRENT**– Haltestrom im Stopp-Modus (von 0 bis 1000 mA)

500Bh –**SPEED**– Soll-Drehzahl für die Modbus-Steuerung (von 35 bis 14000 U/min).

500Ch –**ACC**– vorgegebene Beschleunigung (von 10 bis 1000)

500Dh –**DEC**– vorgegebene Verzögerung (von 10 bis 1000)

Die Beschleunigungs- und Verzögerungswerte in den Registern ACC und DEC sind konventionelle lineare Werte, die die Geschwindigkeit der Beschleunigung/Verzögerung bestimmen. Ein Wert von 10 entspricht 100 U/s2, ein Wert von 1000 entspricht 5000 U/s2.

500Eh –**DIRECTION**– Drehrichtung, mögliche Werte:

- 1 – Vorwärtsdrehung
- 2 – Rückwärtsdrehung.

500Fh –**N_POLE**– Anzahl der Motorpole (von 1 bis 12).

5010h –**USE_EXTERN_SPEED**– Einstellung des Drehzahl-Einstellmodus, mögliche Werte:

- 1 – Befehle über Modbus
- 2 – externes analoges Signal am Eingang SPEED

5012h -**OFFSET_COMPENSATION**– Experimentell berechnete Motorbremskorrektur für die aktuellen Einstellungen ACC, DEC, SPEED. Wenn der Stopp jenseits des berechneten Punktes erfolgt, ist der Kompensationswert gleich dem Positionierungsfehler mit einem negativen Vorzeichen, wenn der Motor vor dem Stopppunkt stoppt, ist der Kompensationswert positiv. Beispielsweise beträgt der Fehler beim Erreichen der angegebenen Position 15 Inkremente, was bedeutet, dass OFFSET_COMPENSATION = -15 ist, und diese Korrektur gilt nur für die aktuellen Einstellungen von Beschleunigung, Verzögerung und Geschwindigkeit.

5013h und 5014h –**PRESSED_INPUTS_EXTERN** und **WAITED_INPUTS_EXTERN**– Mindestdauer eines positiven und eines negativen Teils eines Impulses an den digitalen Eingängen IN1, IN2 – dient zur Unterdrückung von Kontaktprellen (siehe Abb. 5).

5015h –**OFFSET**– der zu verschiebende Offset wird verwendet, wenn MODE_ROTATION = 2 ist, gültige Werte von -2147483647 bis + 2147483648.Vor Beginn der Bewegung muss der erforderliche Offset-Wert im OFFSET-Register eingestellt werden, der vom Controller als Zähler der verbleibenden Bewegung verarbeitet wird. Während der Ausführung der angegebenen Bewegung nimmt der OFFSET-Wert ab.

5017h –**OFFSET_CONST**-Inkrement - der Offset, zu dem verschoben werden muss, der Wert wird vom Controller während des Betriebs geändert.

5019h –**TARGET_POSITION**– Position, zu der verschoben werden soll, zulässige Werte von -2147483647 bis + 2147483648.Im Modus der Bewegung zu einer gegebenen Position wird die Koordinate von POSITION1-POSITION4 vor der Bewegung in dieses Register kopiert, während die Register POSITION1-POSITION4 ihre Werte nicht ändern.

501Bh - 5021h –**TARGET_POSITION1..4**– Vom Benutzer voreingestellte Position №1..4, wird verwendet, wenn MODE_ROTATION = 3 ist, die Positionsnummer wird durch das Register POSITION_N bestimmt, zulässige Werte von -2147483647 bis + 2147483648.

5023h – **ERROR** – Fehler, die während des Controllerbetriebs auftreten - jedes Bit des Registers signalisiert einen bestimmten Fehler:

- Bit 0 - außerhalb des Versorgungsspannungsbereichs;
- Bit 1 - Kurzschluss der Motorwicklungen;
- Bit 2 - Überhitzung des Bremskreises;
- Bit 3 - Überhitzung des Leistungskreises;
- Bit 4 - Fehler beim Anschließen der Hallsensoren;
- Bit 5 - Not-Aus;
- Bit 6 - Überhitzung der MCU;
- Bit 7 - Test-Steuerprogramm;
- Bit 8 - Ausführungsfehler des Benutzerprogramms;
- Bit 9 - Fehler beim Lesen oder Schreiben der Einstellungen;
- Bit 10 - Fehler beim Betrieb der Ausgangstransistorschalter;
- Bit 12 - Warnung vor der Unmöglichkeit, den Haltepunkt zu berechnen;
- Bit 13 - Warnung vor dem Versuch, einen Wert außerhalb des zulässigen Bereichs in das Register zu schreiben;
- Bit 14 – RS-485 Übertragungs-Paritätsfehler

5024h –**FLAG_SAVE_INI**– Register zum Speichern der Benutzereinstellungen - beim Schreiben des Wertes 0x37FA in dieses Register werden die durch die Register 5000h definierten Einstellungen.501Fh im nichtflüchtigen Speicher gespeichert.

5025h –**FLAG_SAVE_USER_PROGRAM**– das Schreiben des Wertes 0x8426 in dieses Register startet den Vorgang zum Speichern des Benutzerprogramms aus dem temporären Puffer in den nichtflüchtigen Speicher des Controllers. Das Schreiben des Wertes 0x9346 startet den Vorgang zum Lesen des Benutzerprogramms aus dem nichtflüchtigen Speicher des Controllers in einen temporären Puffer (siehe Abschnitt 6.6.).

5026h –**FLAG_RESTART**– das Schreiben des Wertes 0x95AF in dieses Register führt zum Neustart des Controllers.

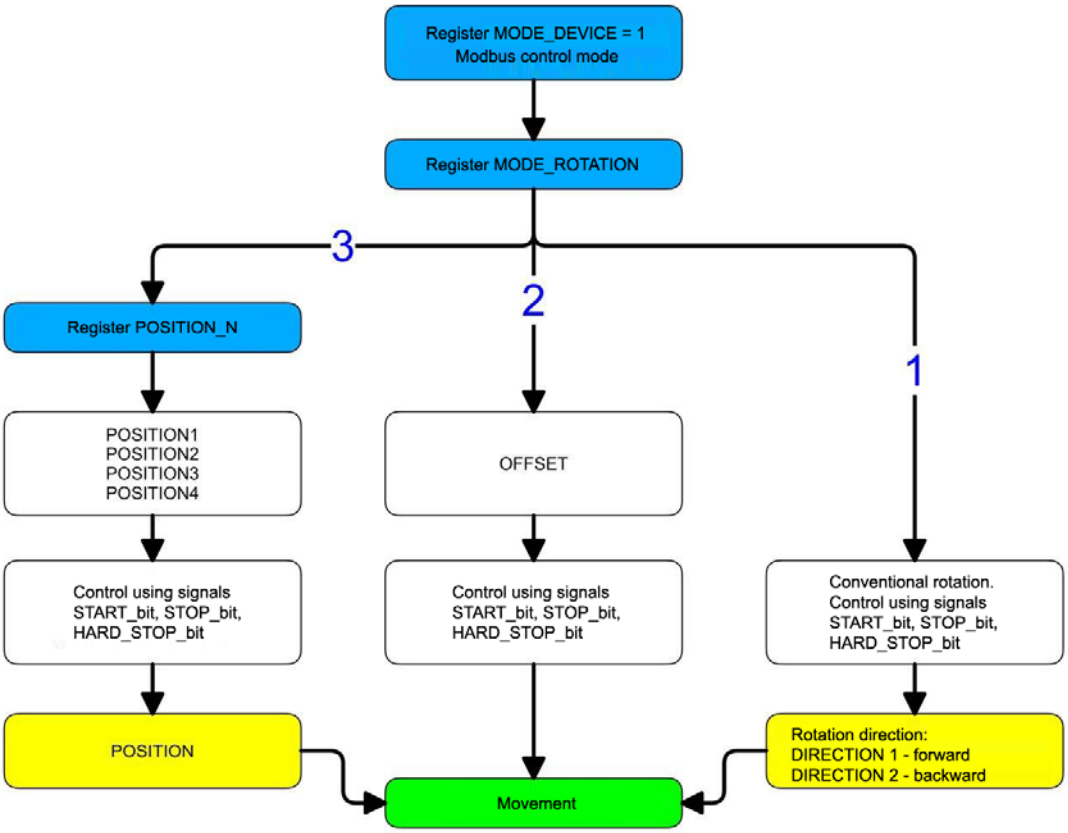

Abb.6. Flussdiagramm zur Auswahl des Betriebsmodus bei der Steuerung des Antriebs über Modbus

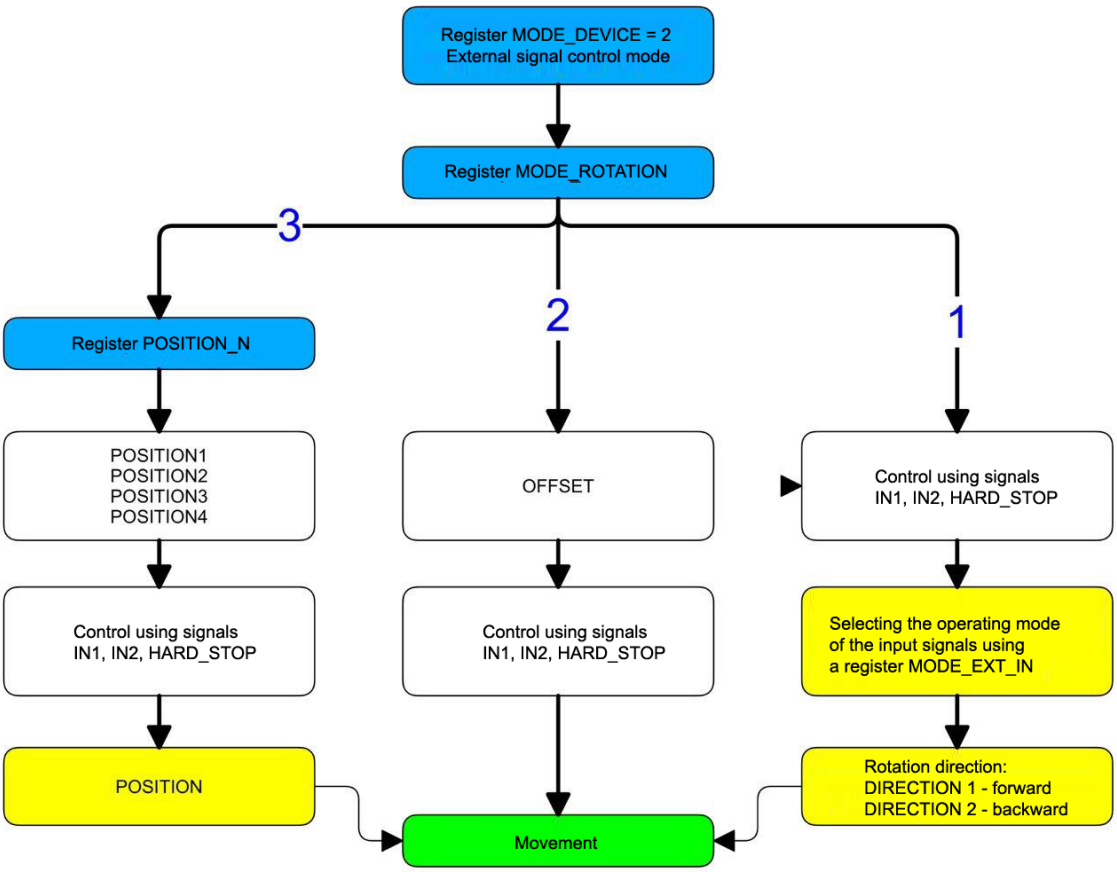

Abb.7. Flussdiagramm zur Auswahl des Betriebsmodus bei der Steuerung des Antriebs über externe Signale

6.6. Lesen und Schreiben eines Anwenderprogramms

Ein temporärer Puffer von 1024 Befehlen wird verwendet, um ein Benutzerprogramm zu lesen und zu schreiben. Das spezielle Register FLAG_SAVE_INI wird verwendet, um das Programm aus dem temporären Puffer im nichtflüchtigen Speicher zu speichern und das Programm aus dem Controllerspeicher in den temporären Puffer zu lesen.

Adresse	Typ	Name	Größe	Beschreibung
5025h	Holding Register	FLAG_SAVE_ USER_PROGRAM	16-Bit	Register zum Schreiben oder Lesen eines Anwenderprogramms in den nichtflüchtigen Speicher des Controllers oder aus diesem. Schreibwert: 0x8426 Lesewert: 0x9346
6000h	Holding Register	WRITE_CMD	16-Bit	Adressregister. Wenn der Adresswert in dieses Register geschrieben wird, wird der Befehl vom CMD_W-Feld an die angegebene Adresse des temporären Puffers des Anwenderprogramms übertragen.
6001h	Holding Register	CMD_W	32-Bit	Anweisungen des Benutzerprogramms
6003h	Holding Register	READ_CMD	16-Bit	Adressregister. Wenn der Adresswert in dieses Register geschrieben wird, wird der Befehl von der angegebenen Adresse des temporären Puffers des Anwenderprogramms in das Feld CMD_R übertragen.
6004h	Holding Register	CMD_R	32-Bit	Anweisungen des Benutzerprogramms

Erstellen und Schreiben eines Benutzerprogramms in den Controllerspeicher

Jeder Benutzerprogrammbefehl besteht aus zwei Speicherworten (32 Bit) - Befehl (16 Bit) und Befehlsdaten (16 Bit).Beim Erstellen eines Benutzerprogramms werden die Befehle zunächst in einen temporären Puffer im Controller geschrieben. Um in den temporären Puffer zu schreiben, muss ein Befehl in das CMD_W-Register geschrieben und dann die Adresse dieses Befehls im internen Puffer in das WRITE_CMD-Register geschrieben werden. Wenn die Adresse in das WRITE_CMD-Register geschrieben wird, wird der Befehl vom CMD_W-Register in den internen Puffer übertragen. Nach dem Erstellen eines Benutzerprogramms in einem temporären Puffer muss der Wert 0x8426 in das Register FLAG_SAVE_USER_PROGRAM geschrieben werden - das Programm wird vom temporären Puffer in den FLASH-Speicher des Controllers übertragen.

Lesen eines Benutzerprogramms aus dem Controllerspeicher

Um ein Benutzerprogramm aus dem FLASH-Speicher zu lesen, muss es in den temporären Puffer des Controllers übertragen werden. Schreiben Sie dazu den Wert 0x9346 in das Register FLAG_SAVE_USER_PROGRAM - das Programm wird vom FLASH-Speicher des Controllers in den temporären Puffer übertragen. Um dann einen Befehl aus dem temporären Puffer zu lesen, muss die Befehlsadresse in das READ_CMD-Register geschrieben werden - wenn die Adresse in dieses Register geschrieben wird, wird der Befehl vom temporären Puffer in das CMD_R-Register kopiert.

Ein Diagramm der Verfahren zum Lesen und Schreiben eines Benutzerprogramms ist in Abb. 8 dargestellt.

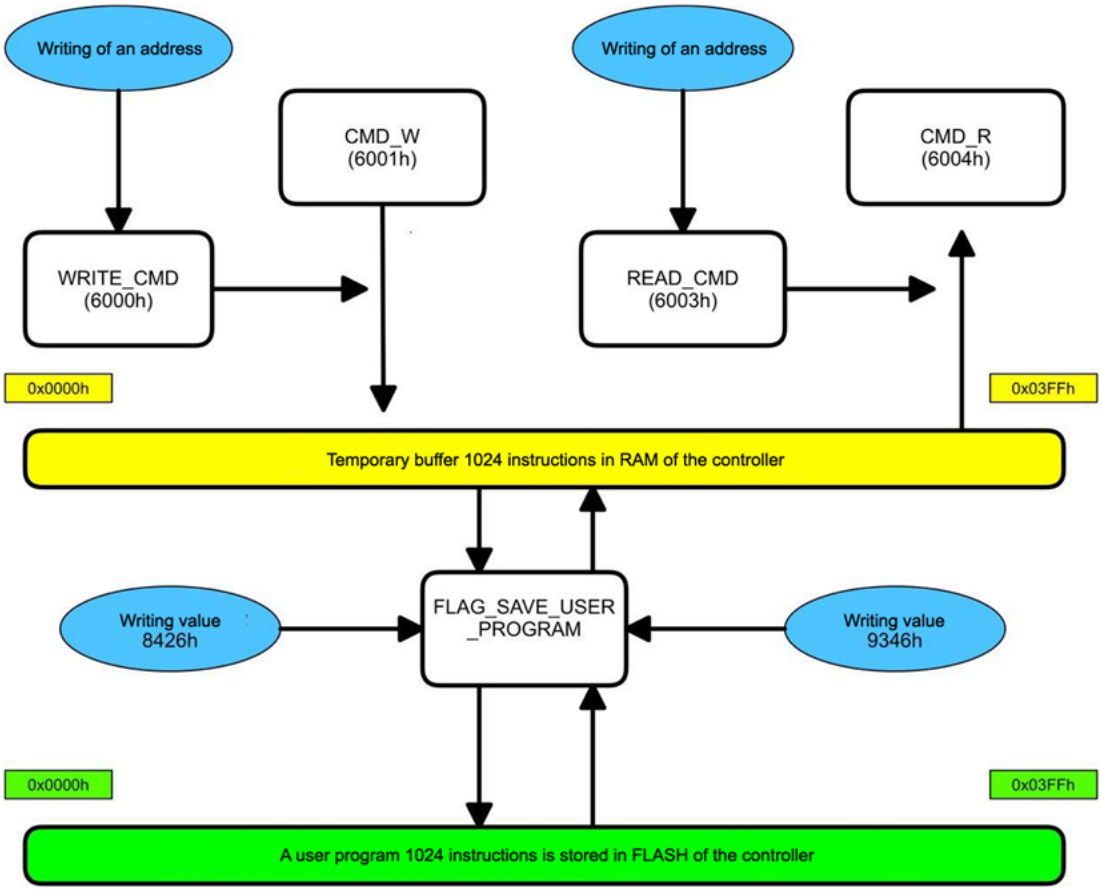

Abb.8. Flussdiagramm der Verfahren zum Lesen und Schreiben eines Benutzerprogramms

6.7. Systemregister

Adresse	Typ	Name	Größe	Beschreibung
7000h	Holding Register	AX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm
7001h	Holding Register	BX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm
7002h	Holding Register	CX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm
7003h	Holding Register	DX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm
7004h	Holding Register	EX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm
7005h	Holding Register	FX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm
7006h	Holding Register	PC_REG	16-Bit	Registerzeiger auf den aktuell ausgeführten Benutzerbefehl
7007h	Holding Register	GX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm
7008h	Holding Register	HX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm
7009h	Holding Register	IX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm
700Ah	Holding Register	JX_REG	16-Bit	Datenspeicherregister für Anwenderprogramm

Systemregister AX_REG..FX_REG (Wertebereich 0..65535) dienen zur temporären Speicherung von Daten während der Ausführung eines Benutzerprogramms. PC_REG ist ein Zeiger auf den aktuell ausgeführten Benutzerprogrammbefehl. Eine detailliertere Beschreibung finden Sie in Abschnitt 6.9.

6.8. Identifikationsregister

Adresse	Typ	Name	Größe	Beschreibung
8001h	Input Register	HW_MAJOR	16-Bit	Treibertyp
8002h	Input Register	HW_MINOR	16-Bit	Hardwareversion
8003h	Input Register	FW_MAJOR	16-Bit	Softwarekennung
8004h	Input Register	FW_MINOR	16-Bit	Softwareversion

Die Register sind notwendig, um den Funktionszweck der Steuereinheit, ihre Eigenschaften und die Softwareversion über das Netzwerk zu bestimmen.

Für RS 206417 sind die Werte:

- HW_MAJOR 1000
- HW_MINOR x
- FW_MAJOR x
- FW_MINOR x

6.9. Anweisungen des Benutzerprogramms

Die Struktur der Benutzerprogrammanweisungen ist im Diagramm in Abb. 9 dargestellt.

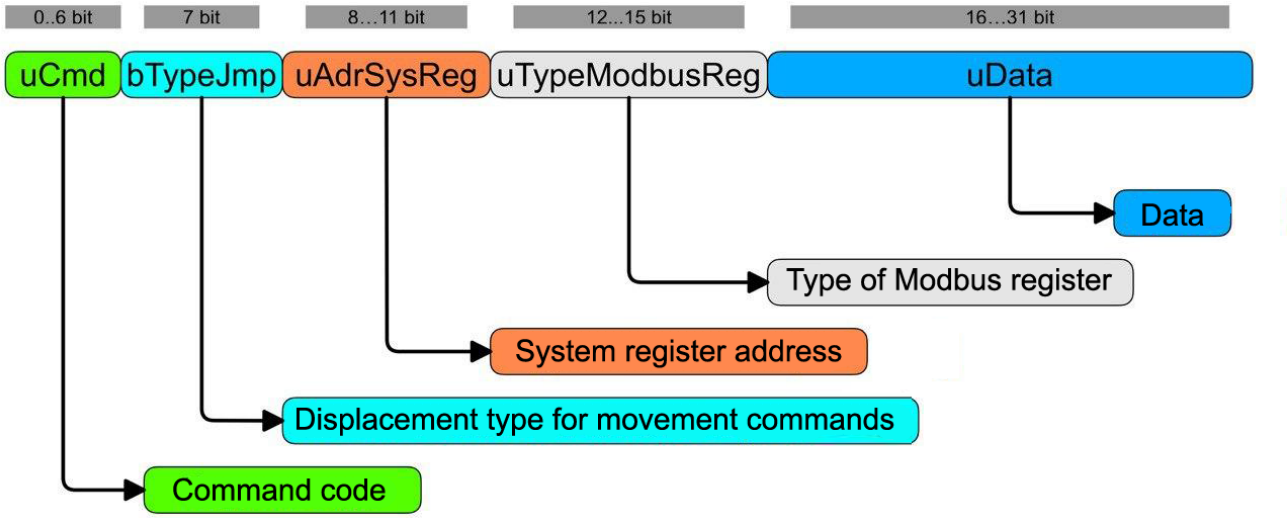

Abb. 9. Aufbau eines Anwenderprogrammbefehls

Die Benutzerprogrammanweisung ist 32 Bit groß und enthält die folgenden Felder:

uCmd – 7 Bit – Befehlscode

bTypeJmp – 1 Bit – Verschiebungstyp für Bewegungsbefehle:

- 0 – Absolutwert
- 1 – Relativwert

uAdrSysReg – 4 Bit - Adresse des Systemregisters AX_REG ... FX_REG mit den Nummern 0 ... 5

uTypeModbusReg – 4 Bit – Typ eines Modbus-Registers:

- 0 – Diskrete Eingänge
- 1 – Coils
- 2 – Eingänge
- 3 – Haltungsregister

uData – 16 Bit - Daten.

64-Bit-Anweisungen bestehen aus zwei 32-Bit-Befehlszeilen.Die erste Befehlszeile dieser Anweisung ist ein Befehl ähnlich einer 32-Bit-Anweisung.Die zweite Befehlszeile enthält 16- oder 32-Bit **uCMD_DATA Daten**.

Anweisungen mit einem D-Präfix, wie z. B. DMOV, arbeiten mit 32-Bit-Daten, die sich in zwei aufeinanderfolgenden 16-Bit-Registern befinden. Im Adressfeld solcher Anweisungen wird das untere Register angegeben. Beim Lesen eines Wertes liest der Befehl automatisch zwei aufeinanderfolgende 16-Bit-Register ab der angegebenen Adresse und bildet einen einzelnen 32-Bit-Wert.Beim Schreiben eines Wertes teilt die Anweisung eine 32-Bit-Zahl in zwei 16-Bit-Teile auf und schreibt diese in zwei aufeinanderfolgende Register ab der angegebenen Adresse.

Die folgende Tabelle listet Befehle mit Optionen zum Füllen von Feldern auf.Wenn das Feld nicht angegeben ist, wird es in diesem Befehl nicht verwendet.

uCmd		-	-	-
0x00	CMD_STOP_PROGRAM	-	-	-
0x0D	CMD_FULL_STOP_PROGRAM			
uCmd		**bTypeJmp**	**uData**	-
0x05	CMD_JMP	0 – Absolutwert 1 – Relativwert	Adresse verschieben 0..1024 oder Offset ±1024	-
0x06	CMD_JMP_AX_PARI_BX			
0x07	CMD_JMP_AX_NOPARI_BX			
0x08	CMD_JMP_AX_MORE_BX			
0x09	CMD_JMP_AX_LESS_BX			
0x63	CMD_DJMP_GX_PARI_IX			
0x64	CMD_DJMP_GX_NOPARI_IX			
0x65	CMD_DJMP_GX_MORE_IX			
0x66	CMD_DJMP_GX_LESS_IX			

uCmd		uData	-	-
0x04	CMD_DELAY	Verzögerungszeit 0..65535 ms	-	-
0x0A	CMD_CALL	Unterprogrammaufruf adresse 0..1024		
0x0B	CMD_RETURN	-		
0x0C	CMD_FOR	Zykluslänge und Anzahl der Zyklen		
uCmd		**uAdrSysReg**	-	-
0x15	CMD_NOT_SYSREG	Systemregisteradresse (0..9)	-	-
0x17	CMD_DNOT_SYSREG			
uCmd		**uTypeModbusReg**	**uData**	-
0x16	CMD_NOT_MODBUS	1 – Spulen 3 – Haltungsregister	Modbus-Registeradresse 0..65535	-
0x18	CMD_DNOT_MODBUS			
uCmd		**uAdrSysReg**	**uData**	-
0x01	CMD_MOV_SYSREG_CONST	Systemregisteradresse (0..9)	Konstante 0..65535	-
0x19	CMD_ADD_SYSREG_CONST			
0x1A	CMD_SUB_SYSREG_CONST			
0x1B	CMD_DIV_SYSREG_CONST			
0x1C	CMD_MUL_SYSREG_CONST			
0x1D	CMD_AND_SYSREG_CONST			
0x1E	CMD_OR_SYSREG_CONST			
0x1F	CMD_XOR_SYSREG_CONST			
uCmd		**uAdrSysReg**	**uTypeModbusReg**	**uData**
0x03	CMD_MOV_SYSREG_MODBUS	Systemregisteradresse (0..9)	0 – Diskrete Eingänge 1 – Spulen 2 – Eingänge 3 – Haltungsregister	Modbus-Registeradresse 0..65535
0x21	CMD_ADD_SYSREG_MODBUS			
0x22	CMD_SUB_SYSREG_MODBUS			
0x23	CMD_DIV_SYSREG_MODBUS			
0x24	CMD_MUL_SYSREG_MODBUS			
0x25	CMD_AND_SYSREG_MODBUS			
0x26	CMD_OR_SYSREG_MODBUS			
0x27	CMD_XOR_SYSREG_MODBUS			
uCmd		**uTypeModbusReg**	**uData**	**uCMD_DATEN**
0x0E	CMD_MOV_MODBUS_CONST	1 – Spulen 3 – Haltungsregister	Modbus-Registeradresse 0..65535	Konstante 0..65535 (separater Befehl Leitung
0x28	CMD_ADD_MODBUS_CONST			
0x29	CMD_SUB_MODBUS_CONST			
0x2A	CMD_DIV_MODBUS_CONST			
0x2B	CMD_MUL_MODBUS_CONST			
0x2C	CMD_AND_MODBUS_CONST			
0x2D	CMD_OR_MODBUS_CONST			
0x2E	CMD_XOR_MODBUS_CONST			
uCmd		**uAdrSysReg**	**uTypeModbusReg**	**uData**
0x02	CMD_MOV_MODBUS_SYSREG	Systemregisteradresse (0..9)	0 – Diskrete Eingänge 1 – Spulen 2 – Eingänge 3 – Haltungsregister	Modbus-Registeradresse 0..65535
0x30	CMD_ADD_MODBUS_SYSREG			
0x31	CMD_SUB_MODBUS_SYSREG			
0x32	CMD_DIV_MODBUS_SYSREG			
0x33	CMD_MUL_MODBUS_SYSREG			
0x34	CMD_AND_MODBUS_SYSREG			
0x35	CMD_OR_MODBUS_SYSREG			
0x36	CMD_XOR_MODBUS_SYSREG			
uCmd		**uAdrSysReg**	**uData**	-
0x0F	CMD_MOV_SYSREG_SYSREG			-

0x37	CMD_ADD_SYSREG_SYSREG	Systemregisteradresse (0..9)	Systemregisteradresse (0..9)	
0x38	CMD_SUB_SYSREG_SYSREG			
0x39	CMD_DIV_SYSREG_SYSREG			
0x3A	CMD_MUL_SYSREG_SYSREG			
0x3B	CMD_AND_SYSREG_SYSREG			
0x3C	CMD_OR_SYSREG_SYSREG			
0x3D	CMD_XOR_SYSREG_SYSREG			
uCmd		**uAdrSysReg**	**uCMD_DATEN**	-
0x10	CMD_DMOV_SYSREG_CONST	Systemregisteradresse (0..9)	Konstante 0...4294967295 (separater Befehl Leitung	-
0x3E	CMD_DADD_SYSREG_CONST			
0x3F	CMD_DSUB_SYSREG_CONST			
0x40	CMD_DDIV_SYSREG_CONST			
0x41	CMD_DMUL_SYSREG_CONST			
0x42	CMD_DAND_SYSREG_CONST			
0x43	CMD_DOR_SYSREG_CONST			
0x44	CMD_DXOR_SYSREG_CONST			
uCmd		**uAdrSysReg**	**uTypeModbusReg**	**uData**
0x11	CMD_DMOV_SYSREG_MODBUS	Systemregisteradresse (0..9)	0 – Diskrete Eingänge 1 – Spulen 2 – Eingänge 3 – Haltungsregister	Modbus-Registeradresse 0..65535
0x46	CMD_DADD_SYSREG_MODBUS			
0x47	CMD_DSUB_SYSREG_MODBUS			
0x48	CMD_DDIV_SYSREG_MODBUS			
0x49	CMD_DMUL_SYSREG_MODBUS			
0x4A	CMD_DAND_SYSREG_MODBUS			
0x4B	CMD_DOR_SYSREG_MODBUS			
0x4C	CMD_DXOR_SYSREG_MODBUS			
uCmd		**uTypeModbusReg**	**uData**	**uCMD_DATEN**
0x12	CMD_DMOV_MODBUS_CONST	1 – Spulen 3 – Haltungsregister	Modbus-Registeradresse 0..65535	Konstante 0...4294967295 (separater Befehl Leitung
0x4D	CMD_DADD_MODBUS_CONST			
0x4E	CMD_DSUB_MODBUS_CONST			
0x4F	CMD_DDIV_MODBUS_CONST			
0x50	CMD_DMUL_MODBUS_CONST			
0x51	CMD_DAND_MODBUS_CONST			
0x52	CMD_DOR_MODBUS_CONST			
0x53	CMD_DXOR_MODBUS_CONST			
uCmd		**uAdrSysReg**	**uTypeModbusReg**	**uData**
0x13	CMD_DMOV_MODBUS_SYSREG	Systemregisteradresse (0..9)	0 – Diskrete Eingänge 1 – Spulen 2 – Eingänge 3 – Haltungsregister	Modbus-Registeradresse 0..65535
0x55	CMD_DADD_MODBUS_SYSREG			
0x56	CMD_DSUB_MODBUS_SYSREG			
0x57	CMD_DDIV_MODBUS_SYSREG			
0x58	CMD_DMUL_MODBUS_SYSREG			
0x59	CMD_DAND_MODBUS_SYSREG			
0x5A	CMD_DOR_MODBUS_SYSREG			
0x5B	CMD_DXOR_MODBUS_SYSREG			
uCmd		**uAdrSysReg**	**uData**	-
0x14	CMD_DMOV_SYSREG_SYSREG	Systemregisteradresse (0..9)	Systemregisteradresse (0..9)	-
0x5C	CMD_DADD_SYSREG_SYSREG			
0x5D	CMD_DSUB_SYSREG_SYSREG			
0x5E	CMD_DDIV_SYSREG_SYSREG			
0x5F	CMD_DMUL_SYSREG_SYSREG			
0x60	CMD_DAND_SYSREG_SYSREG			
0x61	CMD_DOR_SYSREG_SYSREG			
0x62	CMD_DXOR_SYSREG_SYSREG			

uCmd		uAdrSysReg	bTypeJmp	uData
0x20	CMD_SH_SYSREG_CONST	Systemregisteradresse (0..9)	0 – Linksverschiebung 1 – Rechtsverschiebung	Verschiebungswert
0x45	CMD_DSH_SYSREG_CONST			
uCmd		**uTypeModbusReg**	**uData**	**bTypeJmp**
0x2F	CMD_SH_MODBUS_CONST	1 – Spulen 3 – Haltungsregister	Modbus-Registeradresse 0..65535	0 – Linksverschiebung 1 – Rechtsverschiebung
0x54	CMD_DSH_MODBUS_CONST			

CMD_STOP_PROGRAM – (Befehlscode 0x00) – Stoppen der Ausführung eines Benutzerprogramms, ohne den Ausführungsmodus des Benutzerprogramms zu verlassen. Am Ende der Programmausführung bleiben alle Register und der Zustand des Motors so, wie sie vor der Ausführung des Befehls waren (der Motor dreht sich weiter, wenn er sich vor der Ausführung des Befehls drehte).Vor dem nächsten Start des Programms müssen Sie den Befehl CMD_FULL_STOP_PROGRAM senden.

CMD_FULL_STOP_PROGRAM – (Befehlscode 0x0D) - Stoppen der Ausführung des Benutzerprogramms und Verlassen des Programmbetriebsmodus.Am Ende der Programmausführung kehren alle Register und der Zustand des Motors zu ihren ursprünglichen Werten zurück, der Motor stoppt.

CMD_MOV_SYSREG_CONST – (Befehlscode 0x01) – Schreiben in das Systemregister mit der Adresse uAdrSysReg, Werte aus dem Datenfeld uData

CMD_MOV_MODBUS_SYSREG – (Befehlscode 0x02) – Schreiben des Inhalts aus dem Systemregister uAdrSysReg in den ModBUS-Registerbereich, der durch das Feld TypeModbusReg und seine Adresse im Feld uData definiert ist.

CMD_MOV_MODBUS_SYSREG – (Befehlscode 0x03) – Lesen des Inhalts aus dem ModBUS-Registerbereich, der durch das Feld TypeModbusReg und seine Adresse im Feld uData bestimmt wird, in eines der Systemregister uAdrSysReg

CMD_MOV_MODBUS_CONST – (Befehlscode 0x0E) – Schreiben der in der folgenden Befehlszeile uCMD_DATA enthaltenen Konstante in den ModBUS-Registerbereich, der durch das Feld TypeModbusReg und seine Adresse im Feld uData definiert ist.

CMD_MOV_SYSREG_SYSREG – (Befehl Code 0x0F) – Schreiben des Inhalts aus dem Systemregister uAdrSysReg in ein anderes Systemregister mit der Adresse uData.

CMD_DELAY – (Befehl Code 0x04) – Pause, ms.

CMD_JMP – (Befehl Code 0x05) – Sprung zur im uData-Feld angegebenen Adresse.

CMD_JMP_AX_PARI_BX – (Befehl Code 0x06) – Sprung zur im uData-Feld angegebenen Adresse, wenn der Wert im Systemregister AX_REG gleich dem Wert in BX_REG ist.

CMD_JMP_AX_NOPARI_BX – (Befehl Code 0x07) – Sprung zur im uData-Feld angegebenen Adresse, wenn der Wert im Systemregister AX_REG nicht gleich dem Wert in BX_REG ist.

CMD_JMP_AX_MORE_BX – (Befehl Code 0x08) – Sprung zur im uData-Feld angegebenen Adresse, wenn der Wert im Systemregister AX_REG größer als der Wert in BX_REG ist.

CMD_JMP_AX_LESS_BX – (Befehl Code 0x09) – Sprung zur im uData-Feld angegebenen Adresse, wenn der Wert im Systemregister AX_REG kleiner als der Wert in BX_REG ist.

CMD_CALL – (Befehl Code 0x0A) – Aufruf einer Unterroutine, beginnend an der im uData-Feld angegebenen Adresse.

CMD_RETURN – (Befehl Code 0x0B) – Rückkehr aus der Unterroutine.

CMD_FOR – (Befehl Code 0x0C) – Zyklische Ausführung einer Befehlsfolge. Das High-Byte des uData-Felds enthält die Anzahl der Befehle, die sich nach dem CMD_FOR-Befehl befinden und in einem Zyklus wiederholt werden. Das Low-Byte des uData-Felds enthält die Anzahl der Wiederholungen. Zum Beispiel: uData = 0x1705 - 0x17 = 23 Befehle, die in einer Schleife ausgeführt werden, 0x05 = 5 - die Anzahl der Wiederholungen.

CMD_DJMP_GX_PARI_IX – (Befehl Code 0x63) – Sprung zur im uData-Feld angegebenen Adresse, wenn der Inhalt eines Paares von Systemregistern GX_REG und HX_REG gleich dem Inhalt von IX_REG und JX_REG ist.

CMD_DJMP_GX_NOPARI_IX – (Befehl Code 0x64) – Sprung zur im uData-Feld angegebenen Adresse, wenn der Inhalt eines Paares von Systemregistern GX_REG und HX_REG nicht gleich dem Inhalt von IX_REG und JX_REG ist.

CMD_DJMP_GX_MORE_IX – (Befehl Code 0x65) – Sprung zur im uData-Feld angegebenen Adresse, wenn der Inhalt eines Paares von Systemregistern GX_REG und HX_REG größer als der Inhalt von IX_REG und JX_REG ist.

CMD_DJMP_GX_LESS_IX – (Befehl Code 0x66) – Sprung zur im uData-Feld angegebenen Adresse, wenn der Inhalt eines Paares von Systemregistern GX_REG und HX_REG kleiner als der Inhalt von IX_REG und JX_REG ist.

Mathematische Operationen sind zwischen Systemregistern, Modbus-Registern und Konstanten möglich. Die Operanden von Anweisungen, die mit 32-Bit-Daten arbeiten, befinden sich in zwei aufeinanderfolgenden 16-Bit-Registern.Beim Zugriff auf ein Register gibt die Anweisung die Adresse des unteren Registers an. Die Adresse des oberen Registers wird durch Inkrementieren der Adresse des unteren Registers um eins erhalten.

Die folgende Tabelle zeigt mathematische Anweisungen, Anweisungscodes für Operationen mit 16-Bit- und 32-Bit-Daten und die Position der Operanden:

Befehl			Operand 1	Operand 2	Ergebnis
Name	Code				
	16-bit data	32-bit data	S1	S2	D
Addition (S1 + S2 = D)					
CMD_ADD_SYSREG_CONST	0x19	0x3E	SYS_REG_1	CONST_1	SYS_REG_1
CMD_ADD_SYSREG_MODBUS	0x21	0x46	SYS_REG_1	Modbus_REG	SYS_REG_1
CMD_ADD_MODBUS_CONST	0x28	0x4D	Modbus_REG	CONST_2	Modbus_REG
CMD_ADD_MODBUS_SYSREG	0x30	0x55	Modbus_REG	SYS_REG_1	Modbus_REG
CMD_ADD_SYSREG_SYSREG	0x37	0x5C	SYS_REG_1	SYS_REG_2	SYS_REG_1
Subtraction (S1 - S2 = D)					
CMD_SUB_SYSREG_CONST	0x1A	0x3F	SYS_REG_1	CONST_1	SYS_REG_1
CMD_SUB_SYSREG_MODBUS	0x22	0x47	SYS_REG_1	Modbus_REG	SYS_REG_1
CMD_SUB_MODBUS_CONST	0x29	0x4E	Modbus_REG	CONST_2	Modbus_REG
CMD_SUB_MODBUS_SYSREG	0x31	0x56	Modbus_REG	SYS_REG_1	Modbus_REG
CMD_SUB_SYSREG_SYSREG	0x38	0x5D	SYS_REG_1	SYS_REG_2	SYS_REG_1
Division, remainder discarded (S1 / S2 = D)					
CMD_DIV_SYSREG_CONST	0x1B	0x40	SYS_REG_1	CONST_1	SYS_REG_1
CMD_DIV_SYSREG_MODBUS	0x23	0x48	SYS_REG_1	Modbus_REG	SYS_REG_1
CMD_DIV_MODBUS_CONST	0x2A	0x4F	Modbus_REG	CONST_2	Modbus_REG
CMD_DIV_MODBUS_SYSREG	0x32	0x57	Modbus_REG	SYS_REG_1	Modbus_REG
CMD_DIV_SYSREG_SYSREG	0x39	0x5E	SYS_REG_1	SYS_REG_2	SYS_REG_1
Multiplication (S1 * S2 = D)					
CMD_MUL_SYSREG_CONST	0x1C	0x41	SYS_REG_1	CONST_1	SYS_REG_1
CMD_MUL_SYSREG_MODBUS	0x24	0x49	SYS_REG_1	Modbus_REG	SYS_REG_1
CMD_MUL_MODBUS_CONST	0x2B	0x50	Modbus_REG	CONST_2	Modbus_REG

CMD_MUL_MODBUS_SYSREG	0x33	0x58	Modbus_REG	SYS_REG_1	Modbus_REG
CMD_MUL_SYSREG_SYSREG	0x3A	0x5F	SYS_REG_1	SYS_REG_2	SYS_REG_1
Bitwise logical AND (S1 & S2 = D)					
CMD_AND_SYSREG_CONST	0x1D	0x42	SYS_REG_1	CONST_1	SYS_REG_1
CMD_AND_SYSREG_MODBUS	0x25	0x4A	SYS_REG_1	Modbus_REG	SYS_REG_1
CMD_AND_MODBUS_CONST	0x2C	0x51	Modbus_REG	CONST_2	Modbus_REG
CMD_AND_MODBUS_SYSREG	0x34	0x59	Modbus_REG	SYS_REG_1	Modbus_REG
CMD_AND_SYSREG_SYSREG	0x3B	0x60	SYS_REG_1	SYS_REG_2	SYS_REG_1
Bitwise logical OR (S1 | S2 = D)					
CMD_OR_SYSREG_CONST	0x1E	0x43	SYS_REG_1	CONST_1	SYS_REG_1
CMD_OR_SYSREG_MODBUS	0x26	0x4B	SYS_REG_1	Modbus_REG	SYS_REG_1
CMD_OR_MODBUS_CONST	0x2D	0x52	Modbus_REG	CONST_2	Modbus_REG
CMD_OR_MODBUS_SYSREG	0x35	0x5A	Modbus_REG	SYS_REG_1	Modbus_REG
CMD_OR_SYSREG_SYSREG	0x3C	0x61	SYS_REG_1	SYS_REG_2	SYS_REG_1
Bitwise logical XOR (S1 ^ S2 = D)					
CMD_XOR_SYSREG_CONST	0x1F	0x44	SYS_REG_1	CONST_1	SYS_REG_1
CMD_XOR_SYSREG_MODBUS	0x27	0x4C	SYS_REG_1	Modbus_REG	SYS_REG_1
CMD_XOR_MODBUS_CONST	0x2E	0x53	Modbus_REG	CONST_2	Modbus_REG
CMD_XOR_MODBUS_SYSREG	0x36	0x5B	Modbus_REG	SYS_REG_1	Modbus_REG
CMD_XOR_SYSREG_SYSREG	0x3D	0x62	SYS_REG_1	SYS_REG_2	SYS_REG_1
Data shift (S1 >> S2 = D or S1 << S2 = D – Verschieberichtung wird im Feld bTypeJmp angegeben: 0 –links, 1 - rechts)					
CMD_SH_SYSREG_CONST	0x20	0x45	SYS_REG_1	CONST_1	SYS_REG_1
CMD_SH_MODBUS_CONST	0x2F	0x54	Modbus_REG	CONST_2	Modbus_REG

CONST_1 - Konstante im uData-Datenfeld

CONST_2 - Eine Konstante in der folgenden uCMD_DATA-Befehlszeile

SYS_REG_1 ist der Wert, der im Systemregister (oder zwei aufeinanderfolgenden Systemregistern bei 64-Bit-Befehlen) unter uAdrSysReg enthalten ist.Adressen der Systemregister 0..9.

SYS_REG_2 – der Wert ist in dem Systemregister (oder zwei aufeinanderfolgenden Systemregistern für 32-Bit-Daten) enthalten, dessen Adresse in der folgenden uCMD_DATA-Befehlszeile angegeben ist.

Modbus_REG – der Wert, der im Modbus-Register enthalten ist: Registertyp im Feld TypeModbusReg, Registeradresse im Feld uData.

Wichtig: Wenn die Ausführung eines Benutzerprogramms durch den Befehl CMD_STOP_PROGRAM gestoppt wird, bleiben der Zustand des Motors und aller Register so, wie sie während des Programms eingestellt waren. Aus diesem Grund wird der Motor, wenn er zum Zeitpunkt der Ausführung des Befehls CMD_STOP_PROGRAM lief, die letzte Aufgabe weiterhin ausführen, d. h. die Bewegung wird nach Beendigung des Programms fortgesetzt. Nach Beendigung des Programms kann die Antriebsdrehung durch Schreiben eines Registers über Modbus (Coils 2001h oder Coils 2002h) gestoppt werden. Um eine unkontrollierte Drehung zu verhindern, kann vor dem Programmende-Befehl ein Motorstopp-Befehl gesetzt werden.

7. Zurücksetzen auf Werkseinstellungen

Falls erforderlich, können die Controller-Parameter auf die Werkseinstellungen zurückgesetzt werden. Schließen Sie dazu vor dem Einschalten der Stromversorgung des Geräts den ersten Kontakt des RJ11-Steckers - CLR_FLASH_CON (RS-485-Stecker, siehe Abb. 4) mit der Masse GND der Stromversorgung. Die rote und grüne LED am Gerät leuchten

abwechselnd. Halten Sie CLR_FLASH_CON und GND 5 s lang geschlossen. Danach werden die Controller-Parameter auf die Werkseinstellungen zurückgesetzt.

8. Lieferumfang in kompletten Sets

Bürstenloser Motorregler RS 206417 1 Stück

9. Herstellerinformationen

RS Components verfolgt die Linie der kontinuierlichen Entwicklung und behält sich das Recht vor, Änderungen und Verbesserungen am Design und der Software des Produkts ohne vorherige Ankündigung vorzunehmen.

Die in diesem Handbuch enthaltenen Informationen können jederzeit und ohne vorherige Ankündigung geändert werden.

10. Garantie

Jegliche Reparaturen oder Modifikationen werden vom Hersteller oder einem autorisierten Unternehmen durchgeführt.

Der Hersteller garantiert den fehlerfreien Betrieb des Controllers für 12 Monate ab Verkaufsdatum, wenn die Betriebsbedingungen erfüllt sind.

Die Adresse des Herstellervertriebs:

RS Components Ltd, Birchington Rd, Corby, NN17 9RS, United Kingdom, rs-online.com
RS Components GmbH, Mainzer Landstrasse 180, 60327 Frankfurt/Main, Germany, rs-online.com