Handbuch

zum Messautomat

Ausgabe April 2004

Dieses Handbuch ist Umfang der

Technikerarbeit 2003 / 2004

Automatisierung von Akustikmessungen

für BOSE®

von Lorenz Kulf
Jürgen Roos

1 Wichtige Hinweise

1.1 Zweck des Handbuches

Die Informationen dieses Handbuches ermöglichen es Ihnen :

- den Messautomat zur Messung vorzubereiten und aufzubauen.
- den Messautomat über das integrierte Touchpanel zu bedienen.
- Funktionsabläufe des Messsystems kennenzulernen.
- technische Daten des Messautomaten und seiner Komponenten nachzuschlagen.

1.2 Dokumentation zur Programmierung

Um Änderungen im Ablauf des Messverfahrens prozesssicher durchführen zu können, ist diese Dokumentation nicht ausreichend. Auch Änderungen der Bedienoberflächen des integrierten Touchpanels sollten ausschließlich durch eingewiesenes Fachpersonal mit Hilfe der herstellerspezifischen Dokumentation durchgeführt werden.

Weitere Hinweise zu herstellerbezogenen Dokumentationen erhalten Sie in den betreffenden Kapiteln.

1.3 CD-ROM

Diese Dokumentation ist auch auf beiliegender CD-ROM enthalten.

Weitere technische Daten zu Komponenten, die dieser Messautomat beinhaltet und nicht in der Handbuchausgabe enthalten sind, finden Sie ebenso auf der CD-ROM.

1.4 Wegweiser

Um Ihnen einen schnellen Zugriff auf spezielle Informationen zu erleichtern, enthält das Handbuch folgende Zugriffshilfen :

- Am Anfang des Handbuchs steht Ihnen ein Gesamtinhaltsverzeichnis zur Verfügung.
- Ein Abbildungsverzeichnis ist im Gesamtinhaltsverzeichnis integriert.

1.5 Weitere Unterstützung

Bei Fragen zur Nutzung der im Handbuch beschriebenen Produkte, die Sie hier nicht beantwortet finden, wenden Sie sich bitte jeweils an den entsprechenden Komponentenhersteller.

Bei Fragen, die das Komplettsystem betreffen, wenden Sie sich bitte an Ihren BOSE®-Ansprechpartner.

1.6 Ständig aktuelle Informationen

Ständig aktuelle Informationen zu diesem Messsystem erhalten Sie über Ihren BOSE®- Ansprechpartner.

Inhalt

2 Sinn und Zweck

2.1 Warum werden Akustikmessungen durchgeführt ?

Um das akustisch hochwertige Ergebnis eines Bose Sound Systems zu erhalten, sind viele Entwicklungsschritte notwendig. Das Entwickeln eines Sound Systems für ein Kraftfahrzeug stellt die Bose Automotive Systems Division immer wieder vor neue Herausforderungen.

Das Klangbild in einem Fahrzeug wird von vielen Faktoren beeinflusst. An erster Stelle sind natürlich die Lautsprecher und deren Einbauort zu nennen. Jeder Fahrzeuginnenraum „klingt“ anders, das Interieur beeinflusst den Klang maßgeblich: Leder- oder Stoffausstattung, Form der Kopfstützen, Art der Seitenverkleidungen, Armaturen und die Neigungswinkel der Scheiben sind Faktoren, welche die Akustik im Fahrzeug beeinflussen.

Um das gewünschte Klangbild eines Sound Systems in einem Kraftfahrzeug zu erhalten, muß das Sound System an den Fahrzeugtyp angepasst werden. Hierzu sind zuerst die akustischen Vorrausetzungen im Fahrzeuginneren festzustellen. Dies erreicht man durch Messen der Frequenzgänge aller Lautsprecher im Fahrzeuginnenraum.

Die aus diesen Messungen gewonnenen Daten bilden die Grundlage für die Anpassung, das „tunen“, des Sound Systems.

Um möglichst allen Fahrzeuginsassen besten Klang zu bieten, müssen die Akustikmessungen nicht nur auf den vorderen Sitzplätzen, sondern auch auf dem linken und rechten Sitzplatz im Fond durchgeführt werden.

2.1.1 Was ist „tunen“ ?

Zum „tunen“ eines Sound Systems gibt es verschiedene Hilfsmittel, die je nach Anwendungsfall eingesetzt werden können.

- Equalizerfilter
- Phasengangkorrekturen
- Laufzeitkorrekturen

Durch das Messen der Frequenzgänge und das Auswerten der gewonnenen Daten kann man feststellen, daß ein Lautsprecher einen nichtlinearen Frequenzgang hat. Diese Nichtlinearität gilt es zu korrigieren.

Durch das Einsetzen von Equalizerfiltern wird der Frequenzgang linearisiert.

Mit der Verwendung von Hoch-, Tief- und Bandpässen wird erreicht, daß ein Lautsprecher nur in diesem Frequenzbereich eingesetzt wird, für den er auch konzipiert wurde.

Mit „Allpässen“ werden die Phasengänge korrigiert.

Als weitere Möglichkeit bleibt, Zeitverzögerungen, sogenannte „delays“, einzusetzen. Durch dieses Hilfsmittel wird die Wiedergabe entsprechender Lautsprecher zeitverzögert, bzw. Laufzeitunterschiede werden korrigiert. Dadurch kann ein räumliches Klangbild hervorgerufen werden.

2.2 Warum Automatisierung des Messsystems ?

Die Messungen werden durchgeführt mit Hilfe einer geschlossenen „Kiste“, die durch ihre Größe in etwa einen menschlichen Oberkörper darstellt und auf deren oberen Ende zwei Mikrofone installiert sind.

Um verschiedene Sitzhaltungen und Körpergrößen der Passagiere zu berücksichtigen, werden die Messungen nicht nur auf jedem Sitzplatz, sondern auch an fünf verschiedenen Positionen auf dem jeweiligen Sitzplatz durchgeführt.

Ziel der Messungen ist es, die Frequenzgänge aller im Kraftfahrzeug eingebauten Lautsprecher zu erfassen – an jeder Position, an der sich der Kopf eines Fahrzeuginsassen während der Fahrt befinden könnte.

Die verschiedenen Positionen erreichen die Mikrofone durch nach Vorne kippen und durch zur Seite neigen der „Kiste“. Zusätzlich kann die Höhe der Mikrofone durch Verstellen eines versenkbaren Halses variiert werden.

Diese Positionsänderungen geschahen seither durch manuellen Eingriff des Bedieners der Messeinrichtung. Dieser legte nach eigenem Ermessen die Kipp- und Neigungswinkel durch z.B. Unterlegen von Schaumstoffkeilen fest.

Je nach Bediener und Wiederholungen der Messungen variierten somit die Messpositionen im Fahrzeug, Vergleichsmessungen nach Änderungen am Tuning des Sound Systems konnten nur unzureichend durchgeführt werden.

Zu jeder Positionsänderung mußte der Bediener den Messaufbau manuell bewegen, kontrollieren und der Messsoftware bestätigen, daß die Messung fortgesetzt werden konnte.

Zusätzlich wurden an weiteren Geräten wie Notebook oder Psion, verschiedene Eingaben, wie z. B. Umschaltung der zu messenden Frequenz auf andere Lautsprecher des Systems, verlangt.

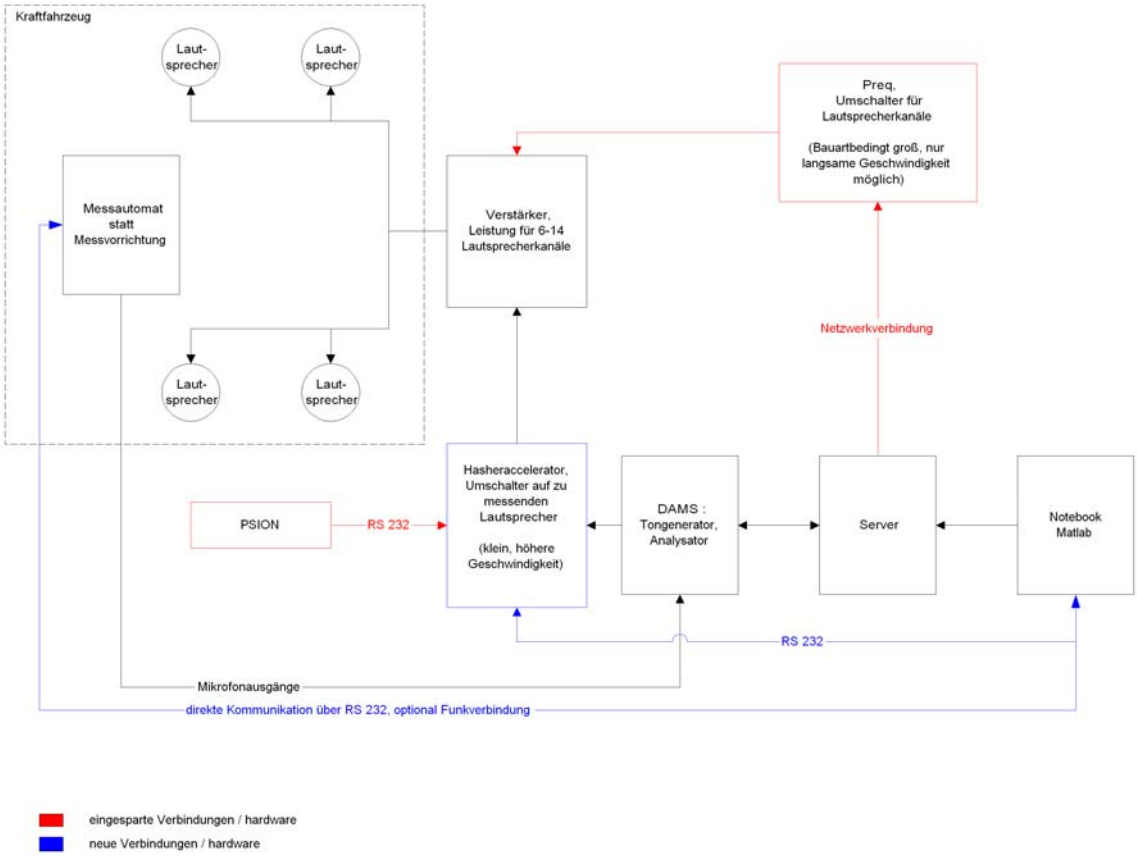
Für die Messungen bestanden somit keine nachvollziehbaren Standards, dadurch ergab sich Handlungsbedarf. Durch Automatisierung des Messvorganges wurde dieser optimiert :

- Vergleichsmessungen nach Änderungen am Sound System können zuverlässig durchgeführt werden.
- Durch Automatisierung wird eine selbstständige Positionierung des Messsystems im Fahrzeug möglich. Der Bediener kann sich auf die Überwachung des Systems konzentrieren.
- Durch selbstständige Kommunikation zwischen Messsoftware und Messsystem, wird Fehlbedienung durch den Benutzer ausgeschlossen.
- Durch Automatisierung werden Funktionen von externer Hardware implementiert, d. h., Geräte zur Eingabe von Daten, wie z. B. einem Psion, können eingespart werden.
- Der Messaufbau wird bedienerfreundlicher, während der Messung ist kein Bedienereingriff mehr notwendig.

Ziel der Automatisierung ist es, die Gewinnung der Rohdaten zu standardisieren, Peripherie zu rationalisieren und Arbeitsaufwand zu minimalisieren.

3 Der Messaufbau

3.1 Messaufbau Blockschaltbild



(Abb. 3 – 1)

Durch die Automatisierung des Messvorganges und Änderungen in der Art der Kommunikation der einzelnen Komponenten des Messaufbaus untereinander ist es möglich, Hardwarekomponenten einzusparen. Zum Teil wird auch Hardware gegen Software ersetzt.

Wie aus der Übersicht im Blockschaltbild zu erkennen ist, werden Geräte wie der Handheld Psion, der seither zum umschalten auf den zu messenden Lautsprecher benötigt wurde, eingespart.

Ebenso kann auf den bisherigen Preq, der Umschalter der Lautsprecherkanäle, in Zukunft verzichtet werden. Diese Aufgabe übernimmt der Hasheraccelerator, der bauartbedingt kleiner und leichter ist, als sein Vorgänger. Außerdem lassen sich mit dem Hasheraccelerator höhere Umschaltgeschwindigkeiten der Lautsprecherkanäle erzielen.

Die Messvorrichtung wird zum Messautomat ausgebaut und kommuniziert direkt über eine RS 232 - Schnittstelle mit der Messsoftware, programmiert in Matlab, auf einem Notebook.

Der Ausbau der Messvorrichtung zum Messautomat wird in Kapitel 4 verdeutlicht.

4 Übersicht Aufbau Messautomat

4.1 Allgemein

Der Messautomat ist in Modulbauweise aufgebaut. Der Aufbau läßt sich weiter gliedern. Eine Übersicht über den Aufbau können Sie nachfolgend entnehmen, genauere Beschreibungen und Detailinformationen sind in Kapitel 5 ersichtlich.

4.2 Teil 1 : Mechanik

4.2.1 Gehäuse

- Rahmen aus Aluminiumprofilen
- Füllung aus Mehrschichtplatten

4.1.2 Antriebseinheiten

- Doppelte Linearführungen
- Kugelrollspindel mit entsprechender Lagerung
- Kupplung zur Kraftübertragung zwischen Antriebsmotor und Getriebe
- Antriebsschlitten mit Gleitlager
- Getriebe
- DC – Antriebsmotor
- Inkrementalgeber zur Lageregelung
- Seitliche Schiene als Endschalterhalterung

4.3 Teil 2 : Elektrik

4.3.1 Hauptplatine

- Netzteil zur Bereitstellung verschiedener Spannungen
- Steuerungsteil
- Leistungsteil
- Kommunikation über serielle Schnittstellen
- Anschluss der Peripherie

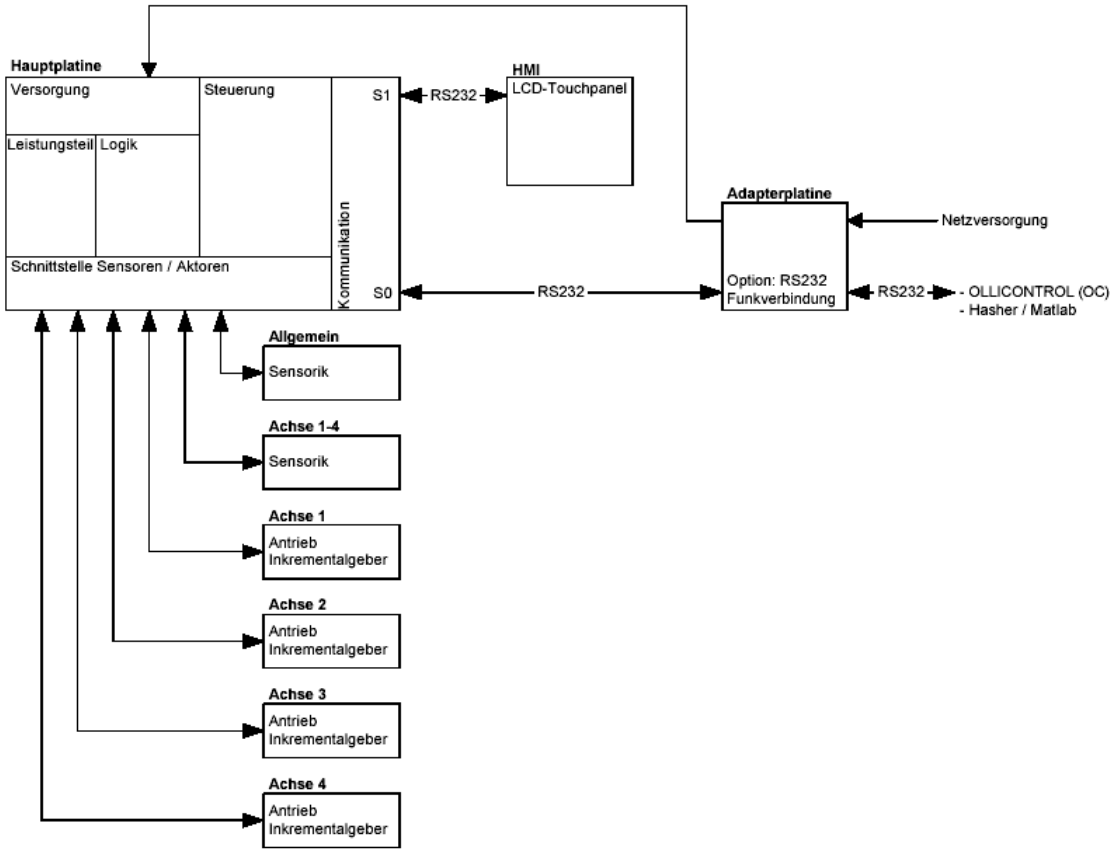
4.3.2 HMI – Human Machine Interface

- Zum Bedienen des Systems und zur Bedienerführung
- Touchpanel mit beleuchtetem LCD – Display

4.3.3 Externe Anschlüsse der Frontplatte

- Spannungsversorgung, Netzeinspeisung 240 V~
- Mikrofonausgänge
- Serielle Schnittstelle RS 232

4.4 Blockschaltbild : neue Komponenten des Messautomaten



(Abb. 4-1)

5 Kurzfunktionsbeschreibung

5.1 Kurzfunktionsbeschreibung Ablauf Messautomat

Der Messautomat dient zum akustischen Ausmessen eines Fahrzeuginnenraumes. An fünf Messpositionen auf jedem der vier Standartsitzplätze im Kraftfahrtzeug werden Messungen vorgenommen. Der Messautomat soll sich selbstständig zwischen den fünf Messpositionen eines Sitzplatzes bewegen.

Verschiedene Betriebsarten stehen hierzu zur Verfügung :

- Betriebsart Automatik
- Betriebsart Teilautomatik
- Betriebsart Hand

Der Messautomat verfügt über ein beleuchtetes LCD-Touchpanel als HMI (Human Machine Interface). Über die HMI werden Eingaben durch den Benutzer angefordert. Die HMI ist die dominante Schnittstelle zur Bedienung des Messautomaten. Die umfangreichsten Eingaben können über diese Schnittstelle getätigt werden.

Die HMI dient selbstverständlich auch zur Bedienerführung. Meldungen werden hier ausgegeben, genauso wie Störungen dargestellt.

Als weiteres Werkzeug zur Bedienung des Messautomaten steht die Applikation OLLICONTROL (OC) zur Verfügung. Genauso wie die HMI kommuniziert OC über eine serielle Schnittstelle mit der Steuerung.

Mit OC kann im Servicefall das Matlab-Hasherprogramm simuliert werden. Der Messautomat kann also autark von einem Messaufbau wie im Blockschaltbild in Kapitel 3.1 (Abb. 3-1) dargestellt, betrieben werden.

OC wird über die Steuerung in Abhängigkeit der HMI zur Bedienung des Automaten freigeschalten bzw. gesperrt.

Befehle, die über das Hasherprogramm aus Matlab, OC oder die HMI an die Steuerung gesendet werden, werden dort weiterverarbeitet. In der Steuerung werden Reaktionen auf die Befehle ausgelöst.

Entsprechend den Befehlen werden logische Ausgänge der Steuerung gesetzt bzw. rückgesetzt oder Befehle über die seriellen Schnittstellen an entsprechende Adressen weitergeleitet.

Nach dem Verbinden des Automaten mit der Netzeinspeisung, führt dieser jedes Mal einen Selbsttest aus.

Durch automatische Freigabe der Bewegungen nach einem erfolgreichen Selbsttest wird eine Referenzpunktfahrt ausgelöst. Bei dieser Referenzpunktfahrt werden die Antriebseinheiten auf ihre mechanischen Bezugspunkte gefahren und mit der Steuerung abgeglichen.

Nach der Referenzpunktfahrt ist die Betriebsart Automatik vorgewählt. Ohne zusätzlichen Bedienungsaufwand kann nun mit der Messung durch das Hasherprogramm begonnen werden.

Durch Fahrbefehle werden Positionsänderungen des Messautomaten ausgelöst. Entsprechend dieser Befehle werden die Treiberstufen angesteuert. Die Treiberstufen stellen die Leistung für die Antriebsmotoren zur Verfügung, welche nun in Verbindung mit den Antriebseinheiten eine Positionsänderung durchführen.

Die Achsen verfahren nicht interpolierend, da kein NC-technisches Bahnfahren zur Positionierung des Messautomaten notwendig ist.

Entsprechende Kommunikation zwischen Mechanik und Steuerung sorgt für Prozesssicherheit.

Detailinformationen über die Antriebseinheiten bietet Ihnen Kapitel 6.1.2.

6 Detail Aufbau Messautomat

6.1 Teil 1 : Mechanik

6.1.1 Gehäuse

Da der Messautomat nicht nur für den stationären, sondern auch für den mobilen Einsatz gedacht ist, ist eine außerordentlich stabile und gegen Umwelteinflüsse robuste Bauweise nicht nur für das Gehäuse, sondern für alle Komponenten des Messautomaten, erforderlich.

Das Gehäuse ist eine Rahmenkonstruktion aus Aluminiumprofilen, welche an den Knotenpunkten mit Würfelprofilen verschraubt sind. Diese Rahmenkonstruktion bringt ein Höchstmaß an Stabilität mit sich.

(Abb. 6-1)

Als Füllung wurden Mehrschichtplatten verwendet, die eine hohe Verwindungssteifigkeit des Körpers garantieren. Diese Mehrschichtplatten wurden mit der Oberfräse bearbeitet, um sie in die Rahmenprofile einlassen zu können.

Die eigentlichen Seitenverkleidungen sind auf die eingelassenen Mehrschichtplatten geschraubt. Dadurch wurde im Inneren des Messautomaten deutlich an Raum gewonnen.

Tragegriffe sind im Gehäuse integriert.

Der Deckel, der die Mikrofone vor Beschädigungen beim Transport des Messautomaten schützt, ist verdrehsicher mit Schnellspannverschlüssen montiert. Dieser beinhaltet auch ein separates Fach, um Anschlussleitungen für die Netzversorgung und eine Datenleitung sicher aufbewahren zu können.

6.1.2 Antriebseinheiten

Um den Messautomat in seinem Arbeitsumfeld bewegen zu können, sind vier Linearachsen notwendig. Prinzipiell sind alle vier Achsen Z-Achsen, also hängende Achsen, da sie sich im Koordinatensystem des Messautomaten senkrecht bewegen und somit eine Auf- / Abbewegung auslösen.

Anstatt die Achsen somit mit Z1 – Z4 zu bezeichnen, wurden sie einfach durchnummeriert, Achse 1 bis Achse 4.

Eine Antriebseinheit besteht aus zwei parallel laufenden Präzisionsstahlwellen mit je 8 mm Durchmesser. Diese Wellen werden in gefrästen massiven Aluminiumklemmblöcken fixiert.

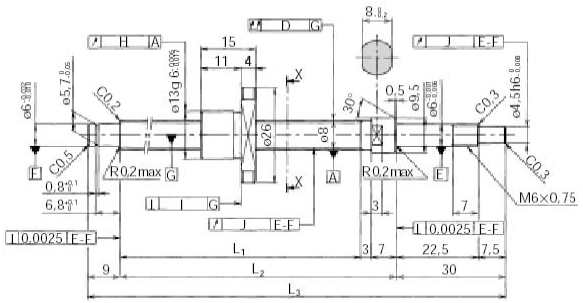
Der Schlitten wird durch eingepresste Linearkugellager auf den Präzisionsstahlwellen geführt.

Den Vortrieb übernimmt eine Kugelrollspindel, deren Spindelmutter in den Schlitten eingelassen ist.

Die Kugelrollspindel wird motorseitig von einer Festlagereinheit in Flanschausführung, auf der anderen Seite von einer Loslagereinheit, ebenso in Flanschausführung, gelagert.

Präzisions-Kugelgewindetrieb BNK 0801-3

Durchmesser: 8 mm; Steigung: 1 mm



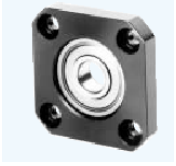
(Abb. 6-2)

Achse 1 bis Achse 3 bestehen aus identischen Kugelrollspindeln mit 145 mm Länge. Die Antriebseinheit der Achse 4 erfordert einen größeren Arbeitsbereich und wird deshalb mit einer Kugelrollspindel der Länge von 175 mm ausgerüstet.

Baugröße	max. Hub	Länge Gewindespindel L_1	L_2	L_3
BNK0801-3G0+115LC3Y	40	66	76	115
BNK0801-3G0+115LC5Y				
BNK0801-3G2+115LC7Y				
BNK0801-3G0+145LC3Y	70	96	106	145
BNK0801-3G0+145LC5Y				
BNK0801-3G2+145LC7Y				
BNK0801-3G0+175LC3Y	100	126	136	175
BNK0801-3G0+175LC5Y				
BNK0801-3G2+175LC7Y				
BNK0801-3G0+225LC3Y	150	176	186	225
BNK0801-3G0+225LC5Y				
BNK0801-3G2+225LC7Y				

(Abb. 6-3)

Die Präzisionsstahlwellen stehen über die Festlagereinheit hinaus und dienen somit als Befestigung für die Motorhalterung. Die Motorhalterung wird im Gegensatz zu der Los- und Festlagereinheit nicht mit der Seitenwand verschraubt, um eventuelle Vibrationen im Antriebsstrang nicht direkt auf das Gehäuse zu übertragen.



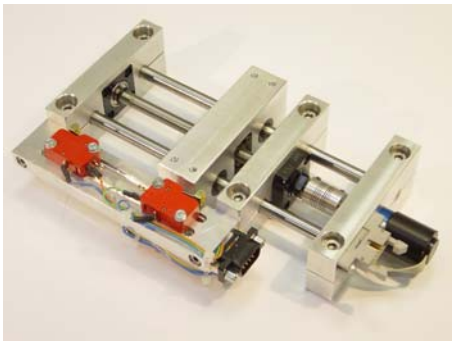
(Abb. 6-4,5)

Das Drehmoment des DC – Antriebsmotor mit Edelmetallkommutierung wird über ein hochwertiges Planetengetriebe in Stahlausführung mit Hilfe einer drehsteifen Federstegkupplung aus Aluminium an die Kugelrollspindel übertragen.

Mit der Festlagereinheit auf der einen Seite und der Loslagereinheit auf der anderen Seite ist eine mehrfach gefräste Aluminiumschiene verschraubt. Über die gesamte Länge dieser Schiene sind zwei Langlöcher eingearbeitet, die eine verstellbare Befestigungsmöglichkeit für

zwei Grenztaster als Endlagenüberwachung und einem Grenztaster als Referenzpunktschalter bietet.

Nach Möglichkeit wurden sämtliche Schrauben in Innensechskant-Ausführung gewählt, um hohe Anzugsmomente zu erreichen und um den Werkzeugeinsatz im Servicefall so gering wie möglich zu halten.



(Abb. 6-6)

Die Antriebseinheiten sind mit den jeweiligen Seitenverkleidungen verschraubt und können somit nach dem Lösen der elektrischen Verbindungen, die zu diesem Zweck alle steckbar sind, mit der Seitenverkleidung demontiert werden. So bildet jede Achse in Verbindung mit der entsprechenden Seitenwand ein Modul, das zu Servicezwecken ausgebaut, neu justiert oder auch nur überprüft werden kann.

Weitere Detailinformationen können Sie den Skizzen (Kapitel 6.1.5.1) entnehmen.

6.1.3 Dimensionierung der mechanischen Komponenten

6.1.3.1 Dimensionierung des Gehäuses

Die äußerlichen baulichen Abmaße änderten sich zu der Messvorrichtung nur geringfügig, um keine Nachteile in der Mobilität des Systems zu erlangen.

Die Rahmenprofile wurden im Querschnitt 30 x 30 mm gewählt, um den höchsten Gewichts- / Stabilitätsfaktor zu erreichen.

Die Mehrschichtplatten haben eine Stärke von 10 mm, dies ist mehr als ausreichend, um den Rahmen zu stabilisieren und eine außerordentliche Grundlage, um die Antriebseinheiten präzise justieren zu können.

6.1.3.2 Dimensionierung der Antriebseinheit

Die Kugelrollspindeln wurden mit einer Steigung von 1 mm/Umdrehung ausgewählt. Die geplante Schlittenverfahrgeschwindigkeit beträgt 5 mm/s. Die abgangsseitige Drehzahl des Getriebes muß somit 300 Umdrehungen/min betragen.

Bei einem Motor mit einer Leerlaufdrehzahl von 7800 Umdr./min ergibt dies eine erforderliche Getriebeuntersetzung von 26:1. Ausgewählt wurde eine Getriebeübersetzung von 23:1, da ein Getriebe mit der berechneten Untersetzung nicht im Produktkatalog der Fa. Faulhaber zu finden ist.

Auf einen anderen Hersteller wird aus Rücksicht auf das Prinzip der Modulbauweise des Messautomaten verzichtet.

Die erforderliche Kraft ergibt sich aus diversen Gewichten und Anstellwinkeln, in denen Massen bewegt werden, mit 98 Newton. Daraus läßt sich unter Berücksichtigung von Reibungsverlusten und Wirkungsgraden ein an der Spindel erforderliches Drehmoment von 21,1 mNm berechnen.

Durch die Getriebeuntersetzung von 23:1 und einem Wirkungsgrad des Getriebes von 0,75 berechnet sich das erforderliche Drehmoment des Antriebsmotor auf 1,22 mNm.

Hier wurde eine vierfache Sicherheit eingerechnet und ein DC-Motor mit Edelmetallkommutierung mit einem Nenndrehmoment von 5 mNm gewählt.

Das kugelgelagerte Planetengetriebe ist mit 0,5 N belastbar. Der Gehäusewerkstoff des Planetengetriebes ist Stahl, der Zahnräderwerkstoff ist Metall.

Eine Federstegkupplung aus Aluminium bildet die Verbindung zwischen Planetengetriebe und Kugelrollspindel.

Die Berechnung der Komponenten erfolgte auf Dimensionierungsmethoden der Fa. Oriental Motor, Fa. SKF und Fa. THK, welche auf die Verhältnisse am Messautomaten angepaßt wurden.

Umdrehungen / Motor	:	7800
abgangsseitige Umdrehungszahl des Getriebes	:	N_G
Getriebeuntersetzung	:	i
Gesamtbelastung der Kugelrollspindel	:	F
erforderliches Drehmoment an der Spindel	:	T_L
erforderliches Drehmoment am Motor	:	T_M
externe Kräfte	:	F_A
Masse des Schlittens und der zu tragenden Last	:	m_1
Koeffizient Reibungsverluste an der Spindel	:	μ
Wirkungsgrad des Getriebes	:	ηG

Berechnung der abgangsseitigen Umdrehungszahl des Getriebes :

$$NG := \frac{V \cdot 60}{PB}$$

$$NG := 300 \frac{U}{min}$$

Berechnung der Getriebeuntersetzung :

$$i := \frac{7800 \frac{U}{min}}{NG}$$

$i = 26$, *gewählt wurde die Getriebeuntersetzung 23:1*

Berechnung der Gesamtbelastung der Kugelrollspindel :

$$F := FA + m1 \cdot g \cdot (\sin(\alpha) + \mu \cdot \cos(\alpha))$$

$F = 98\ N$

Berechnung des erforderlichen Drehmomentes an der Spindel :

$$TL := \frac{98N \cdot 1mm}{2\pi \cdot 0.95} \blacksquare + \frac{0.3 \cdot 98N \cdot 1mm}{2\pi}$$

TL = 21.1 mNm

Berechnung des erforderlichen Drehmomentes am Motor :

$$TM := \frac{21.1mNm}{23 \cdot \eta G}$$

TM = 1.22 mNm

Um bei der Antriebseinheit ohne zusätzliche mechanische Bremse oder Klemmung auszukommen, wird auf Verfahrgeschwindigkeit verzichtet und deshalb ein Getriebe mit hoher Untersetzung gewählt.

Die Spindelsteigung wird ebenso extrem flach gehalten, um der Notwendigkeit eines drehmomentstärkeren Antriebmotors zu entgehen.

Das Getriebe in Verbindung mit der flachen Spindelsteigung sorgt für genügend Widerstand, um genaues Positionieren zu ermöglichen und ein Absacken der Einheit im Stillstand zu verhindern.

6.1.4 Sonstige mechanische Komponenten

6.1.4.1 Mikrofonhalterung

Die Achse 4 bildet die Einheit um die Mikrofone in ihrer Höhe zu verstellen. Diese Achse hat auch den längsten Arbeitsbereich. Auf den Schlitten wird nicht wie bei den Achsen 1 bis 3 ein stabiler, aus Vollmaterial gefräster Aluminiumwinkel justiert, sondern eine Präzisionsstahlwelle.

Die Welle wird aus Gründen der Stabilität und um Verwindungen im Lagerbereich des Schlittens zu vermeiden, durch ein separates Gleitlager geführt.

Die Antriebseinheit 4 wird durch diese Maßnahme nicht den Hebelkräften ausgesetzt, die eventuell durch unsachgemäße Handhabung des Messautomaten entstehen könnten.

Die ebenso gefräste Mikrofonhalterung wird stirnseitig mit der Präzisionsstahlwelle verschraubt.

Weitere Informationen können Sie Kapitel 6.1.5.2 entnehmen.

6.1.5 Technische Zeichnungen

6.1.5.1 Technische Zeichnungen Antriebseinheit

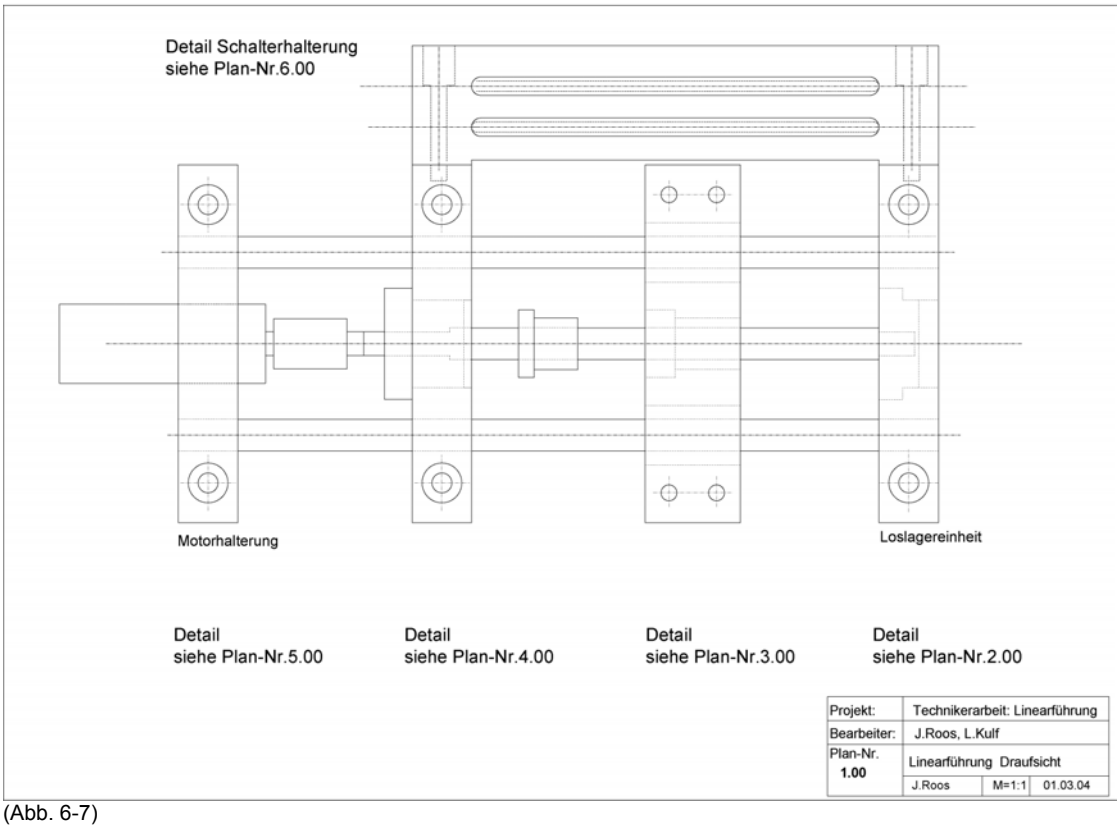
Im Folgenden die technischen Zeichnungen zu den Antriebseinheiten. Die Zeichnungen sind hier nicht maßstabsgetreu dargestellt und dienen nur der Übersicht.

Die Zeichnungen werden auf der beiliegenden Dokumentations – CDR im CAD – Dateiformat *.mcd, *.dxf und *.dwg für beispielsweise VectorWorks zur Verfügung gestellt.

Selbstverständlich können die Grafikdateien auch von anderen CAD - Zeichenprogrammen importiert werden.

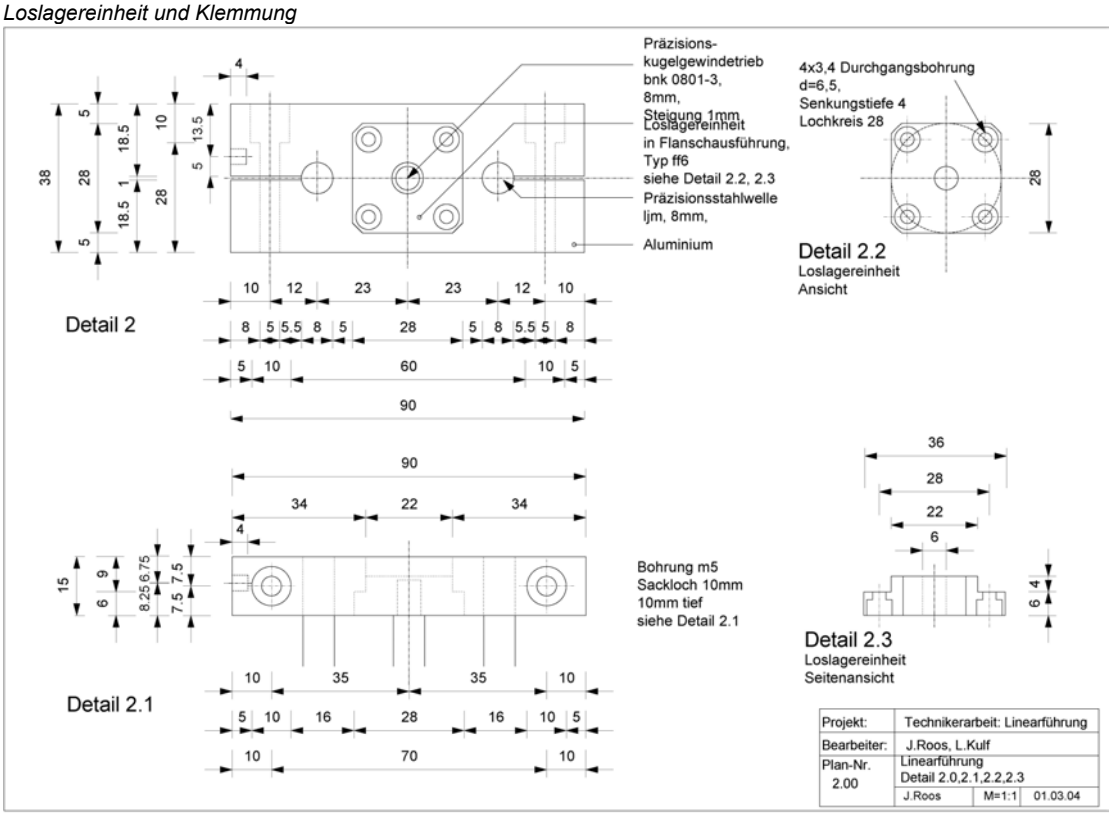
Um die Grafikdateien hier darzustellen zu können, wurden sie mit Adobe Illustrator bzw. Corel Draw bearbeitet.

Übersicht Antriebseinheit



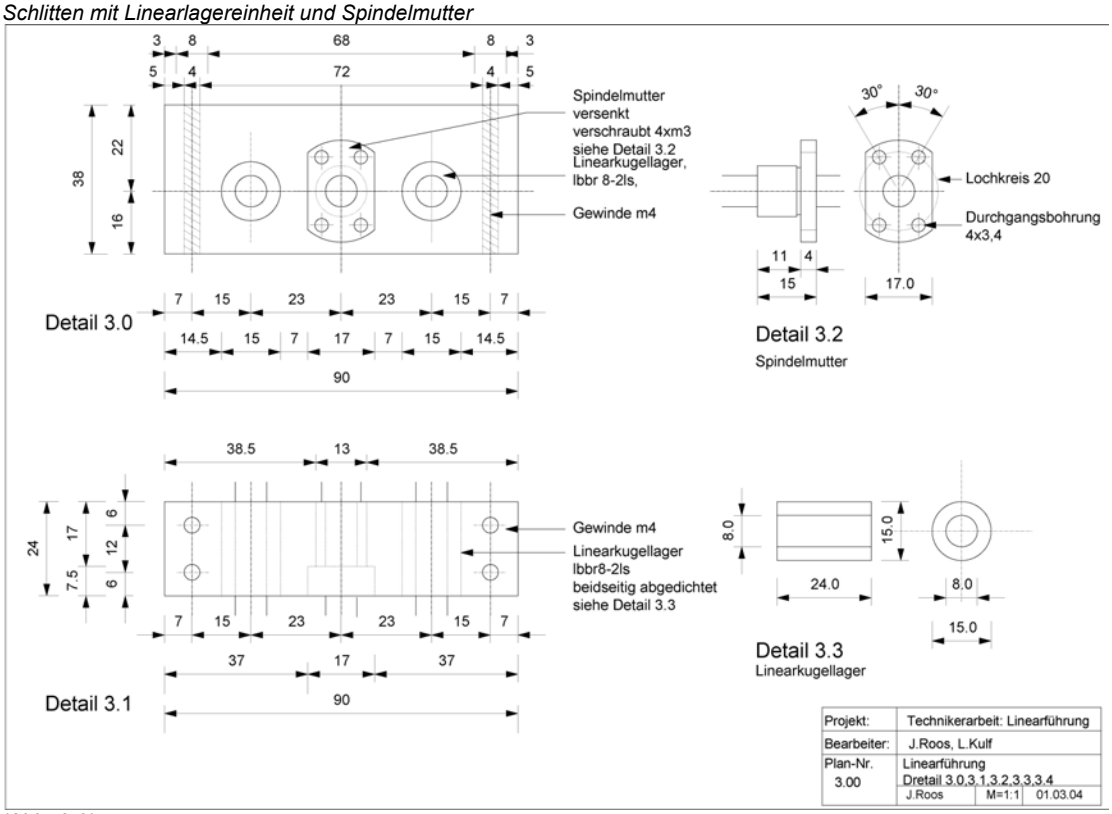
(Abb. 6-7)

Loslagereinheit und Klemmung



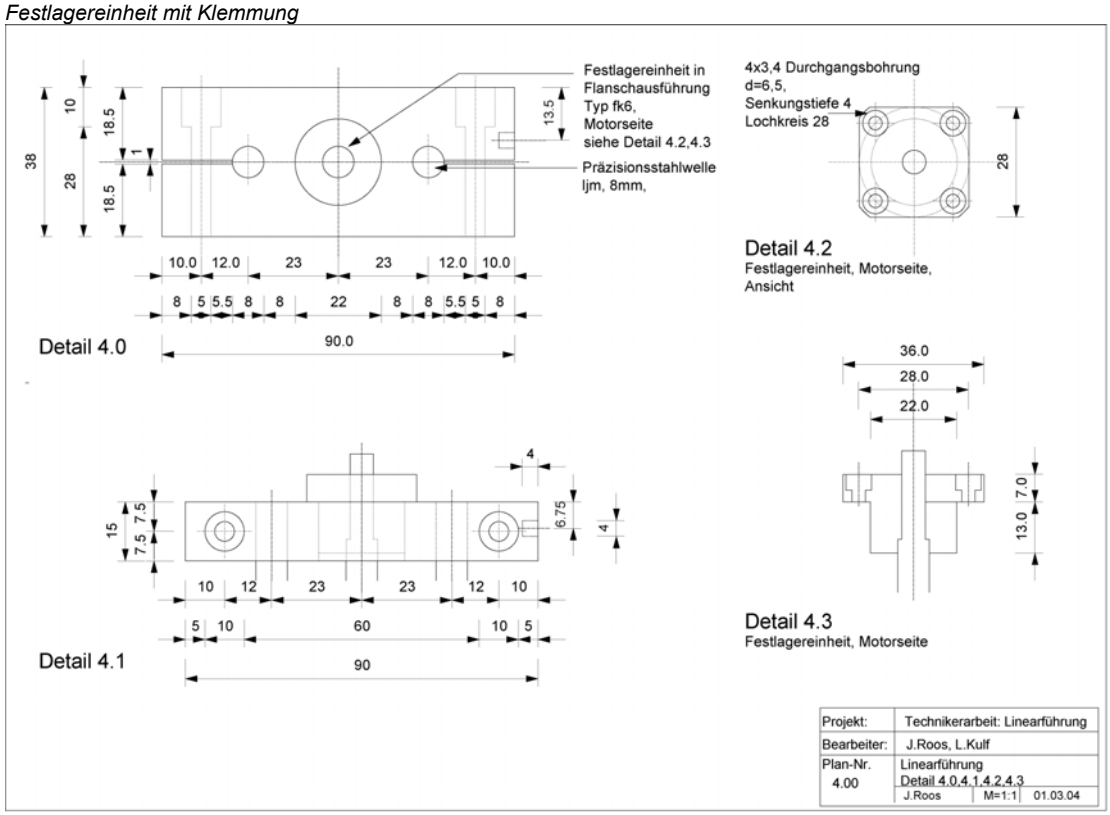
(Abb. 6-8)

Schlitten mit Linearlagereinheit und Spindelmutter



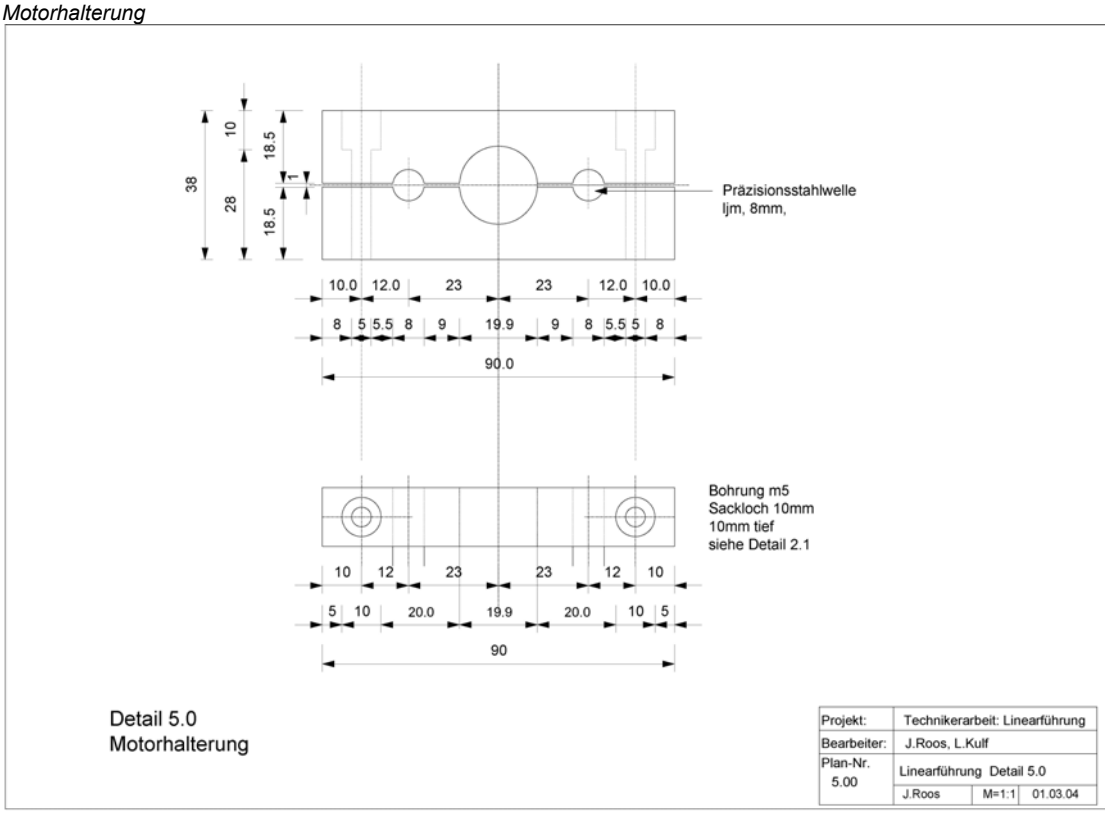
(Abb. 6-9)

Festlagereinheit mit Klemmung



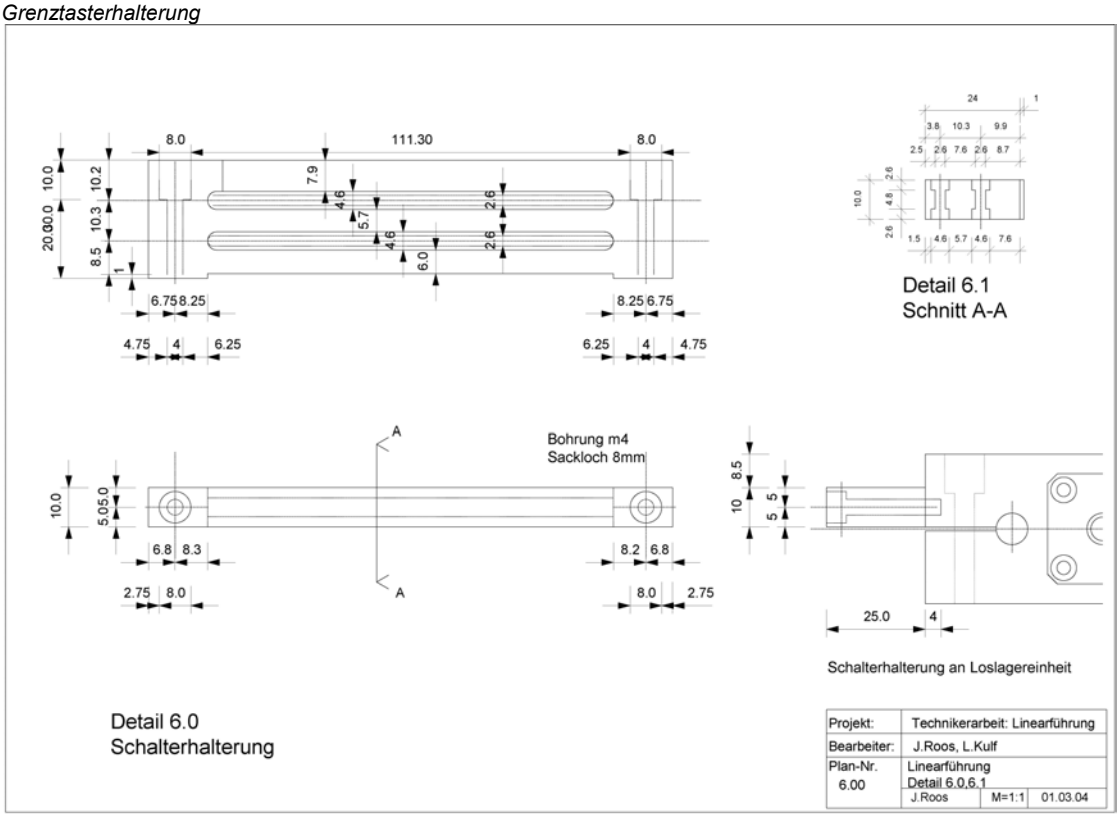
(Abb. 6-10)

Motorhalterung



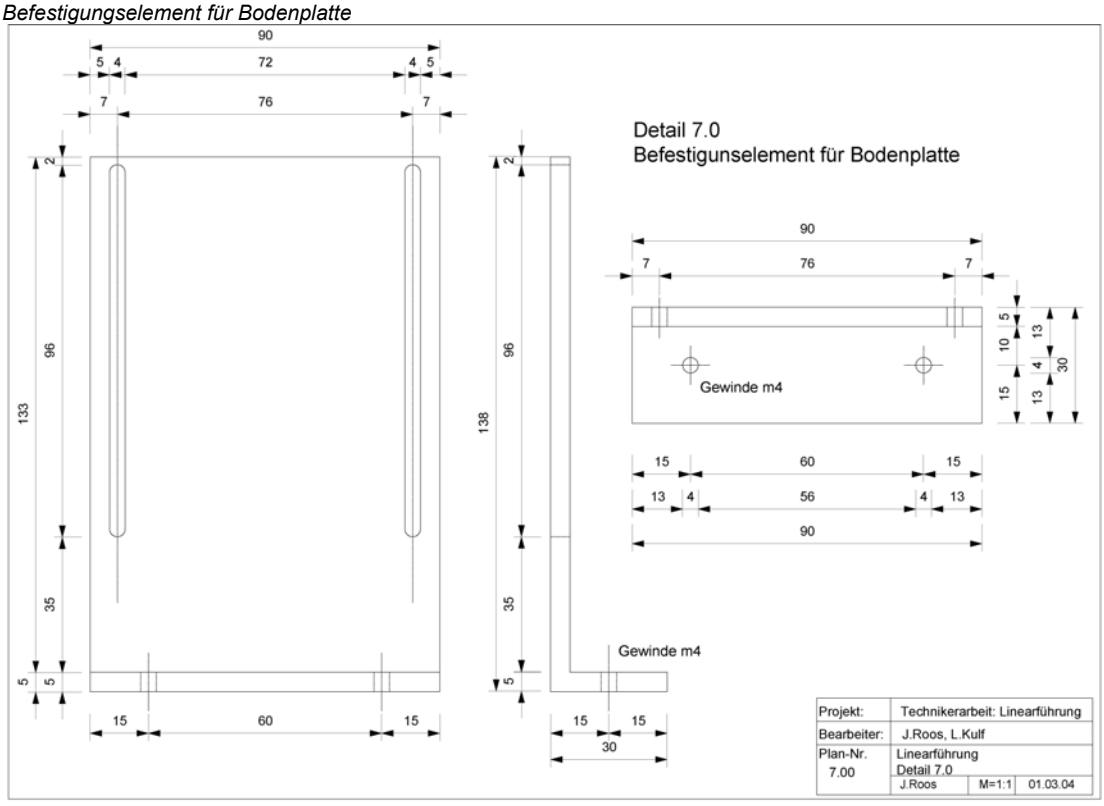
(Abb. 6-11)

Grenztasterhalterung



(Abb. 6-12)

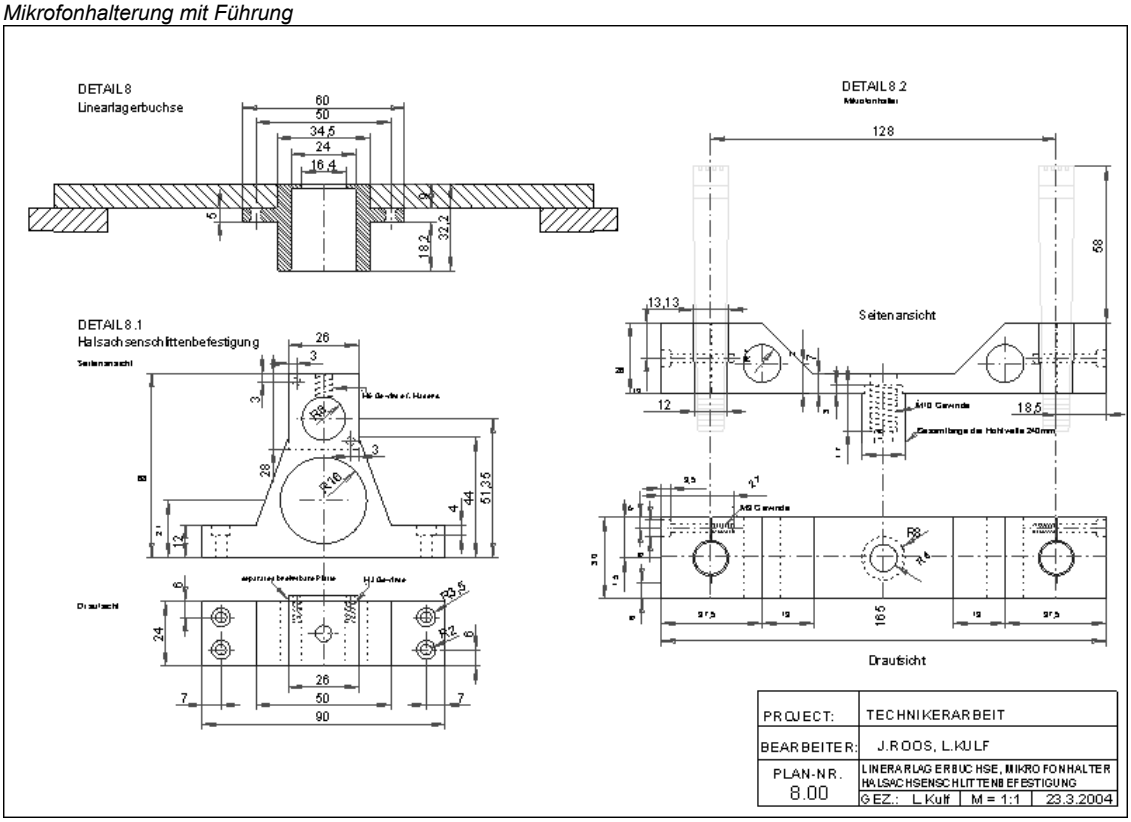
Befestigungselement für Bodenplatte



(Abb. 6-13)

6.1.5.2 Technische Zeichnung Mikrofonhalterung

Mikrofonhalterung mit Führung



(Abb. 6-14)

6.2 Teil 2 : Elektrik

6.2.1 Spannungsversorgung

6.2.1.1 Spannungsversorgung Messautomat

(Abb. 6-15)

Die Spannungsversorgung erfolgt über die Adapterplatte im Gehäuse des Messautomaten (Abb. 6-15). Die Netzeinspeisung von wahlweise 240 V~ / 50 Hz oder 110 V~ / 60 Hz ist steckbar ausgeführt.

Bei Betrieb an 110 V~ Netzen muß an internen Netzteilen ebenso eine Anpassung durch Auswahl der entsprechenden Versorgungsspannung vorgenommen werden.

Die Anschlussleitungen sind in vertauschsicherer Ausführung gestaltet. Die Netzanschlussbuchse ist an der Adapterplatte mit der Gravur 110-230V~ gekennzeichnet.

6.2.1.2 Spannungsversorgung Hauptplatine

Die Spannungsversorgung der Hauptplatine erfolgt über ein Netzteil, das über die Adapterplatte in der Front des Gehäuses an die Netzversorgung angeschlossen ist. Der Anschluß an die Hauptplatine erfolgt mittels eines verpolungssicheren Steckers. Die Versorgungsspannung der Hauptplatine beträgt 12 – 15 V=.

Die Anschlussbezeichnungen können dem Stromlaufplan entnommen werden.

6.2.1.3 Spannungsversorgung des Netzteils der Mikrofone

Die Spannungsversorgung des Netzteils der Mikrofone erfolgt ebenso über die Adapterplatte in der Front des Gehäuses. Die Verteilung der Spannungsversorgungen und deren Absicherungen erfolgen intern.

Detailinformationen können dem Stromlaufplan entnommen werden.

6.2.1.4 Spannungsversorgung der HMI

Die Spannungsversorgung der HMI erfolgt über das gleiche Netzteil, wie die Spannungsversorgung der Hauptplatine. Das Netzteil ist über die Adapterplatte in der Front des Gehäuses mit der Netzeinspeisung verbunden. Die Verteilung der Spannungsversorgungen und deren Absicherungen erfolgen intern.

Der Anschluß an die HMI erfolgt über die Schraubklemmen J2. Die Versorgungsspannung der HMI an J2 darf zwischen 9 und 35 V= betragen.

Eine Verpolung der Versorgungsspannung der HMI an den Schraubklemmen J2, kann zu sofortiger Zerstörung der HMI führen.

Detailinformationen können dem Stromlaufplan entnommen werden.

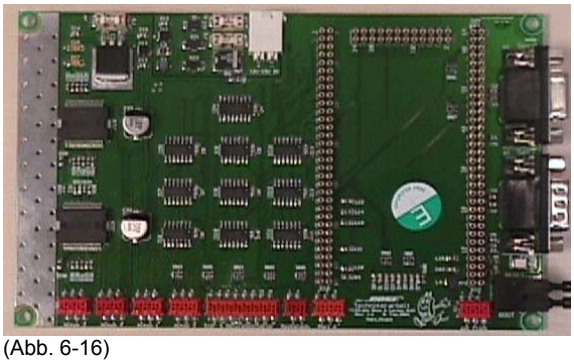
6.2.2 Hauptplatine

Die Hauptplatine wird aus zwei Teilen zusammengeführt :

- der Basisplatine
- der Prozessorplatine

Die Prozessorplatine wird dabei über die Steckleisten X1A, X1B und X1C auf der Basisplatine fixiert. Auf genaues Einsetzen der Prozessorplatine muß dringend geachtet werden. Das Einsetzen der Platine darf nicht unter Spannung erfolgen.

6.2.2.1 Teil 1 : Basisplatine



(Abb. 6-16)

Die Basisplatine (Abb. 6-16) ist in mehrere Bereiche unterteilt:

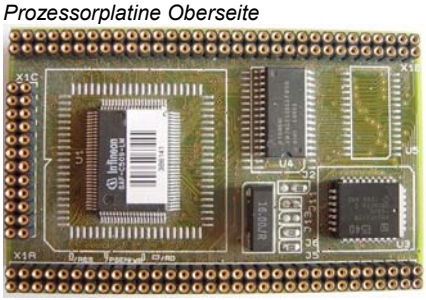
- Spannungsversorgung
- Logikteil
- Leistungsteil
- Anschlussmöglichkeiten zur Peripherie
- serielle Schnittstellen
- Debugbereich
- reservierter Bereich für die Prozessorplatine

Sämtliche Komponenten im Layout der Platine wurden in SMD-Ausführung gehalten, dadurch wurde ermöglicht, eine kompakte Einheit, die Steuerung, Logik- und Leistungsteil vereint, zu generieren.

Detailinformationen zur Basisplatine erhalten Sie in Kapitel 7.

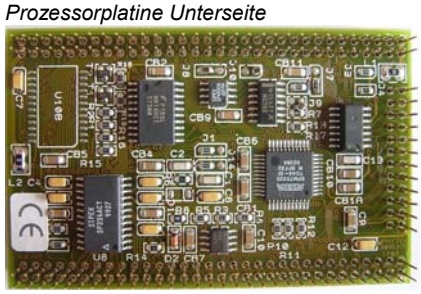
6.2.2.2 Teil 2 : Prozessorplatine

Prozessorplatine Oberseite



(Abb. 6-17)

Prozessorplatine Unterseite



(Abb. 6-18)

Die Prozessorplatine ist ein miniModul-509 der Firma Phytec. Zu beziehen ist dieses Modul über die Phytec Messtechnik GmbH.

Das Modul ist ESD-empfindlich und darf nur an ESD geschützten Arbeitsplätzen von geschultem Fachpersonal gehandhabt werden.

Das miniModul-509 repräsentiert einen funktionellen, universellen Single Board Computer (SBC) im Scheckkartenformat. Die Größe der Prozessorplatine beschränkt sich auf 85 x 55 mm.

Der SBC basiert auf einem Mikrocontroller C509 von Infineon. Der C509 Controller kann mit maximal 16 Mhz getaktet werden und erreicht damit die Rechenleistung eines mit 32 Mhz getakteten 8032. Daraus ergibt sich eine Zykluszeit von 375 ns.

Die controllerspezifischen Eigenschaften entnehmen Sie bitte der Dokumentations-CDR.

Alle Verbindungen zur Basisplatine werden über Stift-/Buchsenleisten durch einfaches Aufstecken der Prozessorplatine auf die Basisplatine hergestellt. Dieser Vorgang wurde bereits in Kapitel 6.2.2 beschrieben.

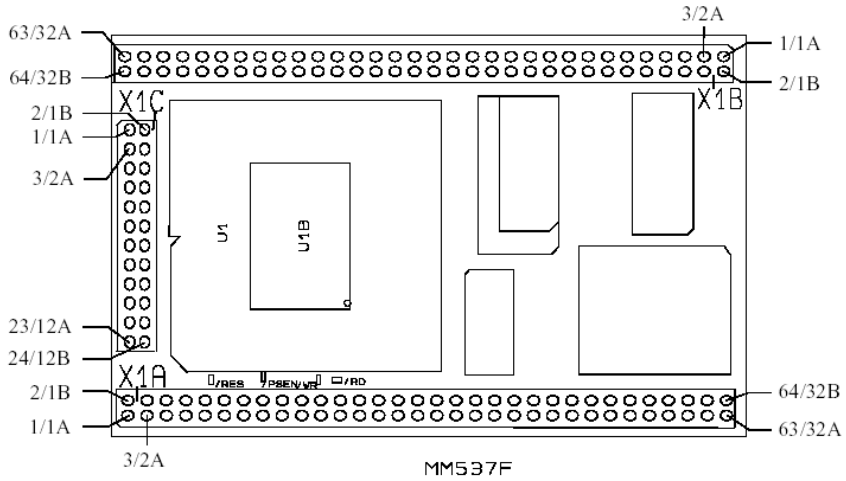
Mit der Prozessorplatine wird die eigentliche Steuerung des Messautomaten verwirklicht.

Über die im Standartmaß von 2.54 mm ausgeführten Stift-/Buchsenleisten werden über die bereits in der Basisplatine anstehenden und aufbereiteten Signale, Informationen eingelesen.

Diese Informationen werden auf der Prozessorplatine ausgewertet, weiterverarbeitet und entsprechende Signale über die Stift-/Buchsenleiste an die Basisplatine zurückgegeben.

Die Lage der Pins an den Stiftleisten kann der Abbildung 6-19 entnommen werden.

Lage der Pins an den Stiftleisten



(Abb. 6-19)

Diese Ausgangssignale werden über die Schnittstellen oder über die Anschlüsse X4 bis X10 der Basisplatine, die in Kapitel 7 näher beschrieben werden, an die Peripherie ausgegeben.

Nachfolgend zusammengefasst, die wichtigsten Eigenschaften und Features der Prozessorplatine miniModul-509 :

- SBC im Format 85 x 55 mm durch SMD-Technik
- Störsicher durch Multilayerplatine
- Infineon Controller C509 im QFP100-Gehäuse, befehlskompatibel zur INTEL 8051-Prozessor Familie.
- Versorgungsspannung 5V über Netzteil auf der Basisplatine
- Ports, Daten- und Adressleitungen über Stiftleisten abgreifbar
- Die Speicher-Standartkonfiguration des miniModuls bietet 32 kByte externen SRAM und 128 kByte externen EEPROM als Speicher für Anwenderquellcode.
- Drei freie chip-select Signale sind verfügbar, um externe Ein-/Ausgänge anzubinden bzw. nachzurüsten.
- Wie bereits aus vorangestellten Kapiteln ersichtlich, verfügt das miniModul-509 über zwei RS232–Schnittstellen, die zur Kommunikation mit der HMI bzw. zur Kommunikation mit dem Hasherprogramm, wahlweise OLLICONTROL (OC) genutzt werden.
- Das Modul ist ausgelegt für den Industrie-Standarttemperaturbereich von 0 bis 70° C.

Weitere Informationen über die Prozessorplatine miniModul-509 können Sie der Phytec-Dokumentation oder der beigelegten Dokumentations-CDR entnehmen.

In Kapitel 8 erhalten Sie spezifische Detailinformationen der Prozessorplatine in Verbindung mit dem Messautomat.

6.2.3 Spindelantrieb

Der Spindelantrieb besteht aus mehreren Komponenten:

- DC-Motor
- Inkrementalgeber
- Getriebe
- Kupplung

In den folgenden Kapiteln erhalten Sie weitere Informationen zu den einzelnen Komponenten des Antriebes der Spindel.

6.2.3.1 DC-Motor

Der Antriebsmotor des Spindeltriebes ist ein bürstenloser DC-Motor mit Edelmetallkommutierung der Fa. Faulhaber des Typs 2224 – 0012 SR.

Die wichtigsten Daten können der folgenden Tabelle entnommen werden.

Serie 2224 ... SR

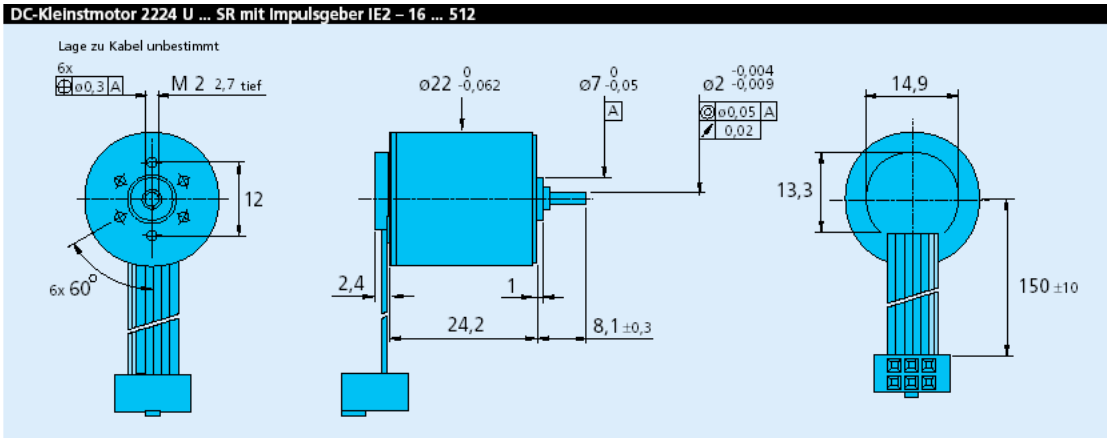
		2224 U	003 SR	006 SR	012 SR	018 SR	024 SR	036 SR	
1	Nennspannung	U_N	3	6	12	18	24	36	Volt
2	Anschlusswiderstand	R	0,56	1,94	8,71	17,50	36,30	91,40	Ω
3	Abgabeleistung	$P_{2\ max.}$	3,92	4,55	4,05	4,54	3,88	3,46	W
4	Wirkungsgrad	$\eta_{max.}$	80	82	82	82	81	80	%
5	Leerlaufdrehzahl	n_o	8 100	8 200	7 800	8 100	7 800	7 800	rpm
6	Leerlaufstrom (bei Wellen ø 2,0 mm)	I_o	0,066	0,029	0,014	0,010	0,007	0,005	A
7	Anhaltemoment	M_H	18,5	21,2	19,8	21,4	19,0	16,9	mNm
8	Reibungsdrehmoment	M_R	0,23	0,2	0,2	0,21	0,2	0,22	mNm
9	Drehzahlkonstante	k_n	2 730	1 380	657	454	328	219	rpm/V
10	Generator-Spannungskonstante	k_E	0,366	0,725	1,520	2,200	3,040	4,560	mV/rpm
11	Drehmomentkonstante	k_M	3,49	6,92	14,50	21,00	29,10	43,50	mNm/A
12	Stromkonstante	k_I	0,286	0,144	0,069	0,048	0,034	0,023	A/mNm
13	Steigung der n-M-Kennlinie	$\Delta n/\Delta M$	438	387	394	379	411	462	rpm/mNm
14	Anschlussinduktivität	L	11	45	200	450	800	1 800	µH
15	Mechanische Anlaufzeitkonstante	τ_m	11	11	11	11	11	11	ms
16	Rotorträgheitsmoment	J	2,4	2,7	2,7	2,8	2,6	2,3	gcm²
17	Winkelbeschleunigung	$\alpha_{max.}$	77	78	74	77	74	74	$\cdot 10^3$ rad/s²
18	Wärmewiderstände	R_{th1} / R_{th2}	5 / 20						K/W
19	Thermische Zeitkonstante	τ_{w1} / τ_{w2}	6,8 / 440						s
20	Betriebstemperaturbereich:								
	– Motor		– 30 ... + 85 (Sonderausführung – 55 ... + 125)						°C
	– Rotor, max. zulässig		+125						°C
21	Wellenlagerung		Sinterlager		Kugellager		Kugellager, vorgespannt		
22	Wellenbelastung, max. zulässig:		(standard)		(Sonderausführung)		(Sonderausführung)		
	– für Wellendurchmesser		2,0		2,0		2,0		mm
	– radial bei 3000 rpm (3 mm vom Lager)		1,5		8		8		N
	– axial bei 3000 rpm		0,2		0,8		0,8		N
	– axial im Stillstand		20		10		10		N
23	Wellenspiel:								
	– radial	≤	0,03		0,015		0,015		mm
	– axial	≤	0,2		0,2		0		mm
24	Gehäusematerial		Stahl, schwarz beschichtet						
25	Gewicht		46						g
26	Drehrichtung		rechtsdrehend auf Abtriebswelle gesehen						
Empfohlene Werte									
27	Drehzahl bis	$n_{e\ max.}$	8 000	8 000	8 000	8 000	8 000	8 000	rpm
28	Dauerdrehmoment bis	$M_{e\ max.}$	5	5	5	5	5	5	mNm
29	Thermisch zulässiger Dauerstrom	$I_{e\ max.}$	2,200	1,200	0,570	0,400	0,280	0,180	A

(Abb. 6-20)

Weitere Informationen zur Dimensionierung des Antriebmotors entnehmen Sie dem Kapitel 6.1.3.2.

CAD-Zeichnungen und weitere Skizzen zu der kompletten Antriebseinheit finden Sie im Kapitel 6.1.5.1.

Auf den technischen Zeichnungen in Kapitel 6.1.5.1 ist der Antriebsmotor nicht in allen Ansichten dargestellt, deshalb können Sie Details zur Zeichnung diesem Kapitel in Abbildung 6-21 entnehmen.



(Abb. 6-21)

6.2.3.2 Inkrementalgeber

Der Inkrementalgeber ist ein magnetischer Impulsgeber des Typs IE2 – 16 der Fa. Faulhaber. Der Inkrementalgeber liefert 16 Impulse pro Umdrehung und besitzt 2 Ausgänge.

Der Geber wäre somit für eine Erkennung der Drehrichtung geeignet, dies ist aber für diese Anwendung nicht notwendig.

Der Geber ist kombinierbar mit dem DC-Motor des Typs 2224 – 012 SR der Fa. Faulhaber und wird auch in Kombination mit diesem geliefert.

In Kombination mit dem Antriebsmotor trägt der Impulsgeber gerade 1,4 mm am hinteren Lagerschild des Motors auf und ist somit wegen seiner kurzen Bauform ideal für diese Anwendung.

Einige technische Daten können Sie der folgenden Tabelle entnehmen.

Serie IE2 – 16

		IE2 – 16	
Impulse pro Umdrehung	N	16	
Ausgangssignal, rechteckig		2	Ausgänge
Betriebsspannung	V_{DD}	4 ... 18	V DC
Nennstromaufnahme, Mittelwert (V_{DD} = 12 V DC)	I_{DD}	typ. 6, max. 12	mA
Ausgangsstrom, max. zulässig	I_{OUT}	15	mA
Pulsbreite [2]	P	180 ± 45	°e
Signal-Phasenverschiebung, Kanal A zu B [2]	Φ	90 ± 45	°e
Signal-Anstiegs-/Abfallzeit, max.(C_{LOAD} = 100 pF)	tr/tf	2,5 / 0,3	µs
Frequenzbereich [1], bis	f	7	kHz
Trägheitsmoment der Impulsscheibe	J	0,11	gcm²
Betriebstemperaturbereich		– 25 ... +85	°C

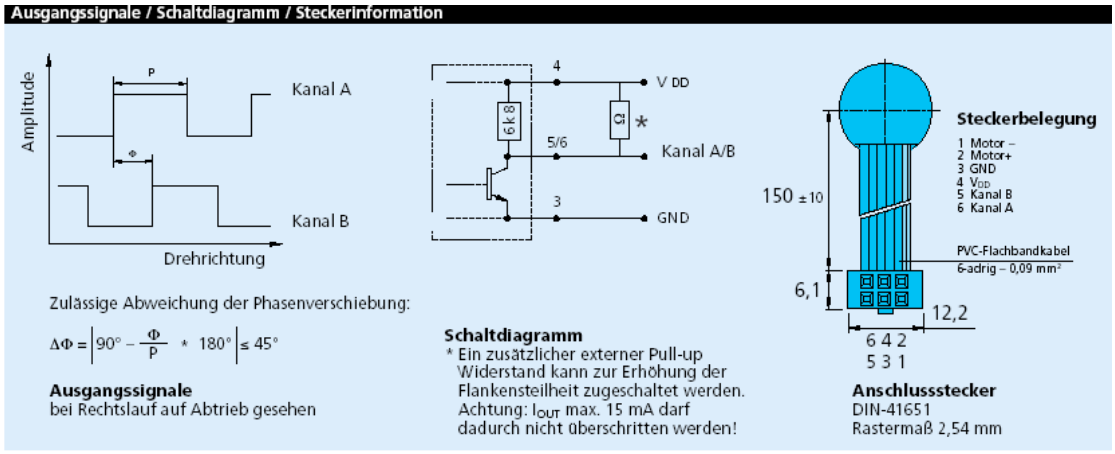
[1] Drehzahl (rpm) = f (Hz) x 60/N
[2] bei 2 kHz geprüft

(Abb. 6-22)

Durch die Verwendung von Hallsensoren und einem mehrteiligen Magnetring ergeben sich zwei um 90° phasenverschobene Kanäle.

Die Versorgungsspannung für den Impulsgeber und den DC-Kleinstmotor sowie die Ausgangssignale werden über ein Flachbandkabel mit einem Stecker angeschlossen.

Der nachfolgenden Abbildung können Informationen zu den Ausgangssignalen, dem Schaltdiagramm und der Steckerbelegung (im Original) entnommen werden.



(Abb. 6-23)

Die Steckerbelegung zeigt den Auslieferungszustand der Fa. Faulhaber. Sie wurde aus Servicegründen der allgemeinen Linie im System angepaßt.

Die Pinbelegung ist dem Stromlaufplan zu entnehmen.

6.2.3.3 Getriebe

Als Getriebe wurde ein Planetengetriebe der Serie 20/1 der Fa. Faulhaber gewählt. Das Übersetzungsverhältnis ist 23:1.

Weitere Informationen zur Dimensionierung des Getriebes erhalten Sie in Kapitel 6.1.3.2. Dort wurden die Berechnungen zu den erforderlichen Drehmomenten dokumentiert.

Aus der folgenden Tabelle können einige technische Daten entnommen werden.

	20/1
Gehäusewerkstoff	Stahl
Zahnräderwerkstoff	Metall
Max. empfohlene Eingangsdrehzahl für:	
– Dauerbetrieb	5000 rpm
Getriebespiel, unbelastet	≤ 1°
Abtriebswellenlager	Kugellager, vorgespannt
Maximal zulässige Wellenbelastung:	
– radial (8,5 mm vom Befestigungsflansch)	≤ 75 N
– axial	≤ 20 N
Maximale Aufpresskraft	≤ 35 N
Lagerspiel (gemessen am Lager):	
– radial	≤ 0,02 mm
– axial	= 0 mm
Betriebstemperaturbereich	– 30 … + 100 °C

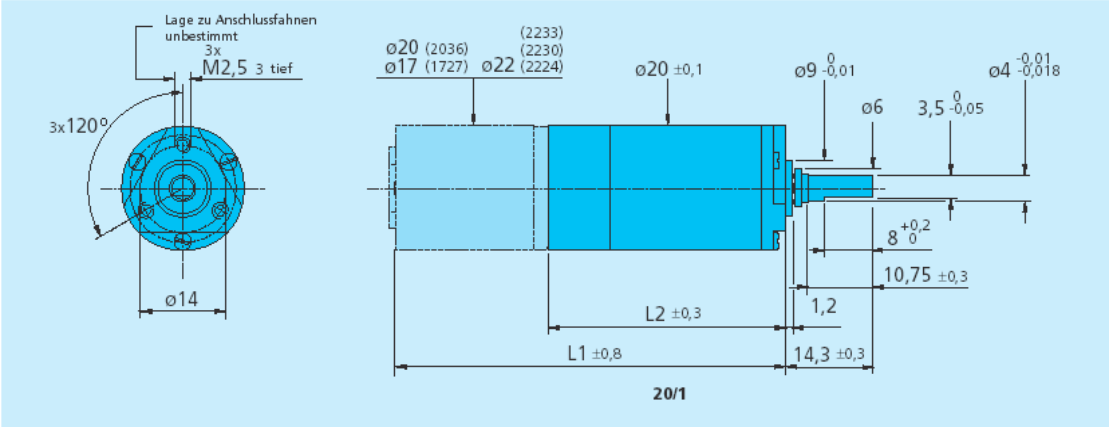
Technische Daten

Untersetzungs-verhältnis (nominal)	Gewicht ohne Motor	Länge ohne Motor L2	Länge mit Motor					Drehmoment Dauer-betrieb	Drehmoment Kurzzeit-betrieb	Drehsinn der Welle	Wirkungs-grad
			1727 U L1	2036 U L1	2224 U L1	2230 U L1	2233 U L1				
	g	mm	mm	mm	mm	mm	mm	M max. mNm	M max. mNm		%
3,71 :1	28	18,35	45,55	54,35	42,55	48,35	50,95	500	700	=	88
9,7 :1	38	23,45	50,65	59,45	47,65	53,45	56,05	500	700	=	80
14 :1	38	23,45	50,65	59,45	47,65	53,45	56,05	500	700	=	80
23 :1	38	23,45	50,65	59,45	47,65	53,45	56,05	500	700	=	80
43 :1	48	28,55	55,75	64,55	52,75	58,55	61,15	500	700	=	70
66 :1	48	28,55	55,75	64,55	52,75	58,55	61,15	500	700	=	70
86 :1	48	28,55	55,75	64,55	52,75	58,55	61,15	500	700	=	70
134 :1	58	33,65	60,85	69,65	57,85	63,65	66,25	500	700	=	60
159 :1	58	33,65	60,85	69,65	57,85	63,65	66,25	500	700	=	60
246 :1	58	33,65	60,85	69,65	57,85	63,65	66,25	500	700	=	60
415 :1	68	38,75	65,95	74,75	62,95	68,75	71,35	500	700	=	55
592 :1	68	38,75	65,95	74,75	62,95	68,75	71,35	500	700	=	55
989 :1	68	38,75	65,95	74,75	62,95	68,75	71,35	500	700	=	55

(Abb. 6-24)

Das Getriebe wurde zusammen mit der Kupplung und den restlichen Komponenten in dieses Kapitel aufgenommen, da diese Komponenten zusammen eine Einheit bilden.

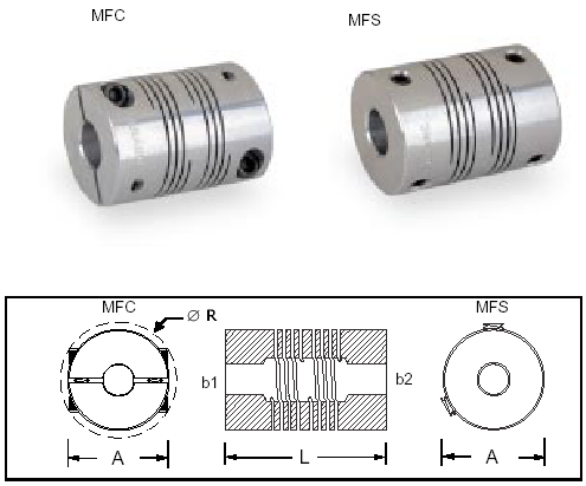
Der Abbildung 6-25 kann die Bemaßung der Motor-/Getriebeeinheit entnommen werden. Zur Ergänzung können die technischen Zeichnungen aus Kapitel 6.1.5.1 hinzugenommen werden.



(Abb. 6-25)

6.2.3.4 Kupplung

Als Kupplung wird eine torsionssteife Flexbeam - Aluminiumkupplung der Fa. Ruland verwendet. Hierbei handelt es sich um eine Federsteg - Aluminiumkupplung.



(Abb. 6-26)

Näher soll hier nicht auf die Kupplung eingegangen werden. Weitere Informationen erhalten Sie auf der Dokumentations-CDR im entsprechenden Datenblatt zu den hier abgebildeten Kupplungen.

6.2.4 Sensorik

Als Sensorik werden an die Steuerung angeschlossene Taster, Schalter, Initiatoren, o. Ä. bezeichnet.

In diesem System werden ausschließlich Referenzpunktschalter und Endlagenschalter verwendet.

Zusätzlich wird mit einem Reedkontakt der Deckel überwacht und mit einem Neigungssensor der Arbeitsbereich des Messautomaten kontrolliert.

6.2.4.1 Referenzpunktschalter

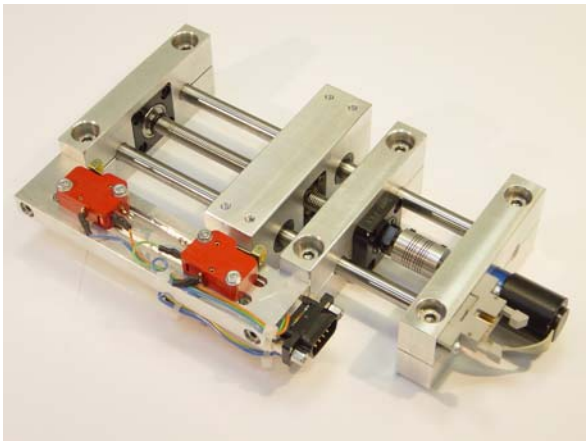
Der Referenzpunktschalter ist ebenso wie die Endlagenschalter auf der Endschalterhalterungsschiene montiert.

Der Referenzpunktschalter ist als Schließerkontakt ausgeführt und an der Unterseite der Schiene für die Endschalter montiert.

Der Referenzpunktschalter ist auf Abbildung 6-27 nicht zu sehen.

Die Sensorik einer Antriebsachse ist als Modulbaustein zu sehen und deshalb auch zu Servicezwecken komplett zu wechseln.

Die Anschlüsse der Schalter wurden steckbar ausgeführt.



(Abb. 6-27)

Genaue Informationen zur Verdrahtung können dem Schaltplan zur Hauptplatine oder dem Stromlaufplan zum Gesamtsystem entnommen werden.

Eine Referenzpunktfahrt auf den Referenzpunktschalter ist notwendig, um eine eindeutige mechanische Position zu einem Bezugspunkt herzustellen. Die Positionierungen der Achsen werden in Bezug auf diesen Referenzpunkt hergestellt.

Die Referenzpunktschalter sind als mechanische Schalter mit Rollenhebel ausgeführt und nicht als Initiator. Somit kann auf eine Versorgungsspannung, wie sie zum Betrieb eines Initiators benötigt wird, verzichtet werden.

6.2.4.2 Endlagenschalter

Die Endlagenschalter sind baulich die gleichen Schalter wie die Referenzpunktschalter der Antriebseinheiten.

Es wird jedoch der Öffnerkontakt des Wechslers verwendet, somit wird für Kabelbruchsicherheit garantiert.

Die Endlagenschalter sind in Abbildung 6-27 dargestellt. Es wird jeweils ein Endlagenschalter für jede Drehrichtung eines Antriebsmotors verwendet, somit wird die Endlage in Drehrichtung (+) und die Endlage in Drehrichtung (-) kabelbruchsicher überwacht.

Da die Endlagenschalter ebenso wie die Referenzpunktschalter auf der Endschalterschiene montiert sind, sind sie steckbar ausgeführt und als Modulbaustein bei Bedarf zu wechseln.

Durch die Langlochfräsung, sind die Endlagenschalter ebenso wie die Referenzpunktschalter einstellbar.

Genauere Informationen zur Verdrahtung können dem Schaltplan zur Hauptplatine oder dem Stromlaufplan zum Gesamtsystem entnommen werden.

6.2.4.3 Deckelschalter

Der Deckelschalter ist als Reedkontakt ausgeführt. Der Schalter ist im oberen Rahmen des Gehäuses eingelassen und wird durch einen Magneten im Deckel bedämpft.

Um den Deckel auch verdreht aufsetzen zu können, sind zur Sicherheit zwei Magnete gegenüberliegend im Rahmen des Deckels verarbeitet. Ein Aufsetzen des Deckels bleibt nicht unerkannt.

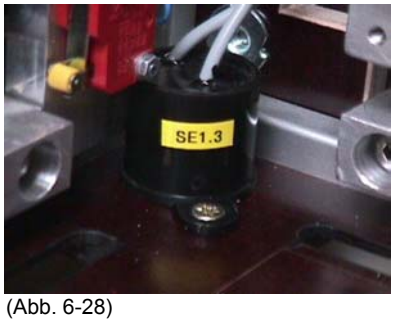
Eine diagnostizierte Störung wird auf der HMI visualisiert. Weitere Informationen zur Bedienerführung werden ausgegeben.

Der Deckelschalter führt die Bezeichnung SE 1.5 im Stromlaufplan und ist entsprechend an Port 1.5 der Steuerung angeschlossen.

6.2.4.4 Neigungssensor

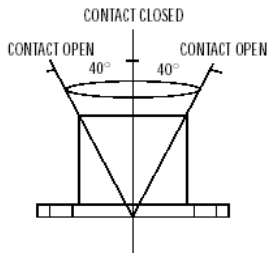
Ein Neigungssensor wird eingesetzt, um zu überwachen, daß der Messautomat sich in seinem Arbeitsbereich befindet.

Der Neigungssensor ist auf dem unteren Boden des Messautomaten installiert. Die Verschaltung und Verdrahtung des Neigungssensors kann dem Stromlaufplan des Systems entnommen werden.



(Abb. 6-28)

Der Neigungssensor führt die Bezeichnung SE 1.3 im Stromlaufplan und ist entsprechend an Port 1.3 der Steuerung angeschlossen.



(Abb. 6-29)

Einige technische Daten können der folgenden Tabelle entnommen werden.

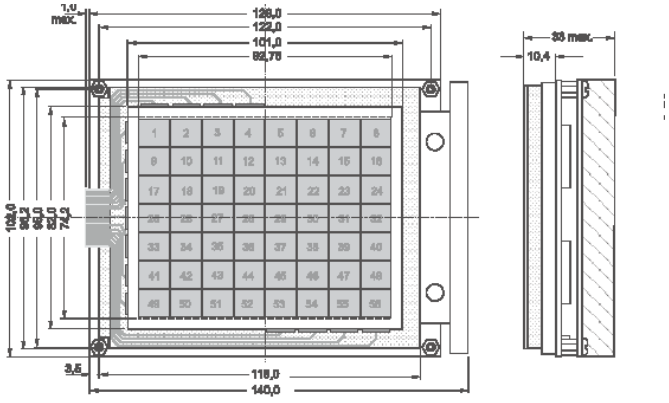
CONTACT FORM / STYLE		Normally Closed
SWITCHING VOLTAGE	Max. Vac RMS	60
SWITCHING CURRENT	Max. A	0.25
SWITCHING CAPACITY	Max. VA	3
OPERATING ANGLE	Deg.° Max.	40° ± 15°
CONTACT RESISTANCE	Max. Ω	6
OPERATING TEMPERATURE	Deg. °C	-20° +70°
STORAGE TEMPERATURE	Deg. °C	-25° +70°
CASE MATERIAL		Polypropylene
CABLE / TERMINATION		2 x round 0.14² PVC covered and insulated

(Abb. 6-30)

Weitere technische Daten zum Neigungssensor erfahren Sie auf der Dokumentations-CDR im entsprechenden Datenblatt.

6.2.5 HMI – Human Machine Interface

Als HMI wird ein 5,1“ LCD-Touchpanel des Typs EA KIT 160-7 der Fa. Electronic Assembly verwendet.



(Abb. 6-31)

Das Touchpanel hat eine Bildschirmauflösung von 160x128 Pixel. Wie in Abbildung 6-31 dargestellt, wird der Bildschirm in 56 Felder aufgeteilt. Diese 56 Felder können als touch-Felder aktiviert werden.

Einige technische Daten:

- Bildschirm mit Hintergrundbeleuchtung
- Bildschirmauflösung 160 x 128 Pixel
- Touchpanelauflösung 8 x 7 Felder
- Versorgungsspannung 12 V=
- Kommunikation über serielle Schnittstelle RS 232

Weitere technische Daten können Sie dem Datenblatt auf der beiliegenden Dokumentations-CDR entnehmen.

Das LCD-Touchpanel dient der Bedienung des Messautomaten. Informationen zur Bedienerführung werden über die HMI visualisiert.

Die Bedieneinheit enthält neben kompletten Grafikroutinen zur Displayausgabe auch verschiedene Schriften.

Die Programmierung erfolgt über hochsprachenähnliche Grafikbefehle. Das Display muß textbasierend programmiert werden. Eine grafisch orientierte Programmierung ist nicht möglich.

Eine Programmierung kann mit jedem ASCII-Editor realisiert werden. Der Quelltext muß compiliert werden und kann dann anschließend über die serielle Schnittstelle RS 232 in ein EEprom des LCD-Touchpanels übertragen werden.

Mit Bildern oder Grafiken wird ebenso verfahren. Die Bilder werden wie der compilierte Quelltext im EEprom des Display gespeichert und mit Hilfe von Makros über die serielle Schnittstelle RS 232 aufgerufen.

Die Bilder können auch je nach Bedarf über die Schnittstelle direkt auf den Bildschirm geladen werden, hiervon wird aber wegen des relativ geringen Datendurchsatzes der Schnittstelle abgeraten.

In diesem System wurden sämtliche Daten im EEprom des Displays abgelegt und nach Bedarf abgerufen.

Ein Upload der Daten vom EEprom des Displays ist nicht möglich.

Die Baudrate der seriellen Schnittstelle RS 232 ist voreingestellt auf 9600 Baud und lässt sich bei Bedarf ändern. Das Datenformat ist fest auf 8, N, 1 eingestellt.

In diesem System ist die Baudrate unverändert, der dem Auslieferungszustand entsprechend, auf 9600 Baud eingestellt.

Baudraten

DIP Schalter			Datenformat 8,N,1
1	2	3	
ON	ON	ON	1200
OFF	ON	ON	2400
ON	OFF	ON	4800
OFF	OFF	ON	9600
ON	ON	OFF	19200
OFF	ON	OFF	38400
ON	OFF	OFF	57600
OFF	OFF	OFF	115200

(Abb. 6-32)

Über DIP – Schalter 6 lässt sich ein versehentliches Überschreiben der Makros, Bilder und Schriften verhindern.

Schreibschutz	
DIP 6	Schreibschutz für EEPROM
ON	Ein keine Makroprog. mögl.
OFF	Aus Makroprog. möglich

(Abb. 6-33)

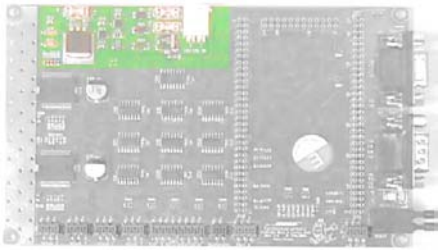
Die Versorgungsspannung wird an den Schraubklemmen J2 angeschlossen.

Versorgungsspannung nicht verpolen !
Eine noch so kurzzeitige Verpolung kann zur sofortigen Zerstörung des LCD-Displays führen.

Weitere Informationen zur HMI erhalten Sie in den Kapiteln zur Bedienung des Messautomaten und zur Programmierung des Displays.

7 Detail Basisplatine

7.1 Spannungsversorgung



(Abb. 7-1)

Der Bereich der Spannungsversorgung befindet sich im oberen linken Teil der Basisplatine. Herzstück der Spannungsversorgung ist ein Festspannungsregler 7805. Dieser ist in seiner Bauform so gewählt, daß er ohne Kühlkörper auskommt. Eine ausreichende Kühlfläche ist auf der Basisplatine so gestaltet, daß Wärme ungehindert abgeführt werden kann.

Der Grundsatz, die Komponenten in ihrer Bauform so zu wählen, daß sie ohne entsprechende Kühlkörper auskommen, wurde im kompletten Projekt verwirklicht.

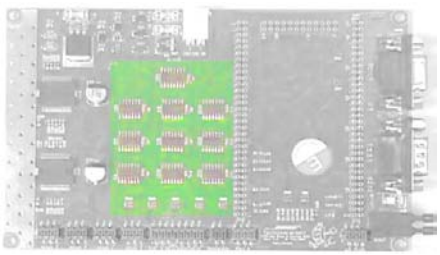
Eine Glättung bzw. Siebung der Spannung wurde durch entsprechende Beschaltung des 7805 erreicht.

Für den Leistungsteil steht eine Spannung von 12 V= bei 6 A zur Verfügung. Für den Logikteil stehen 5 V= zur Verfügung.

Durch eine LED auf der Basisplatine wird die korrekte Spannungsversorgung des Logikteils angezeigt. Diese liegt an mit VCC gekennzeichneten Punkten.

Peripherie wie Grenztaster, wird über die mit VDD gekennzeichneter Spannung versorgt. Eine entsprechende Absicherung der Spannungen ist über die mit F1 bis F3 gekennzeichneten Sicherungen auf der Basisplatine im Bereich der Spannungsversorgung verwirklicht.

7.2 Logikteil



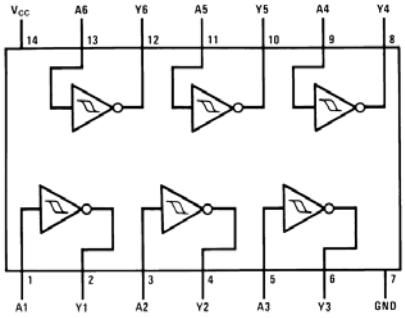
(Abb. 7-2)

Der Logikteil befindet sich im linken zentralen Bereich der Basisplatine. Der Bereich Logik besteht aus 6 SMD-Bausteinen der Reihe 74LS. Grundsätzlich wurde versucht, durch Verwendung von Bausteinen der Reihe 74LS (Low-Power-Schottky) oder 74ALS (Advanced Low-Power-Schottky), die Belastung des Netzteils so gering wie möglich zu halten.

Durch dieses Konzept kann auch die Wärmeentwicklung eingedämmt werden und somit auf externe Kühlung bzw. Lüftung verzichtet werden.

Die Bausteine des Typs 14M enthalten invertierende Schmitt-Trigger. Diese sind erforderlich, um eindeutige Pegel an den Eingängen (DI-Ports) von der Peripherie wie Endlagenschalter oder Referenzpunktschalter zu erhalten.

Blockschaltbild des 74LS14M



(Abb. 7-3)

Die Eingänge der Referenzpunktschalter wurden zweimal invertiert, da diese Schalter als Schließerkontakt ausgeführt sind. Die Endlagenschalter sind dagegen als Öffnerkontakt ausgeführt, um Kabelbruchsicherheit zu garantieren und werden deshalb nur einmal invertiert.

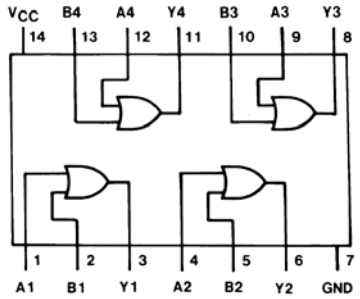
Der Neigungssensor und die Schließüberwachung des Gehäuses am Deckel sind ebenso über invertierende Schmitt-Trigger an die digitalen Eingänge der Steuerung geführt. Auch hier ist es erforderlich, eindeutige Pegel an den DI-Ports zu erhalten.

Der Logikteil besteht aus weiteren drei Bausteinen der Reihe 74ALS. Diese ODER-Bausteine des Typs 32M werden dazu verwendet, um eventuelle Störungen, die durch betätigte Endlagenschalter hervorgerufen werden, an einem separatem DI-Port, der Steuerung zu melden.

Die Steuerung liest dann die DI-Ports aller Endlagenschalter ein und wertet diese dann explizit aus. Durch diesen Vorgang müssen die Endlagenüberwachungen nicht zyklisch eingelesen werden. Es wird somit Zykluszeit bei der Abarbeitung des Programmablaufes eingespart.

Durch diese Maßnahme kann sofort bei anstehender Störung entsprechend reagiert werden und eventuell eingeschaltete Antriebseinheiten mit sofortiger Wirkung stillgesetzt werden um mechanische Schäden an der Antriebseinheit, insbesondere Beschädigungen der Spindelmutter des Gewindetriebes, zu verhindern.

Blockschaltbild des 74ALS32M



(Abb. 7-4)

Durch dieses Prinzip wird eine Ereignissteuerung des Programmablaufes verwirklicht. Diese Interruptsteuerung des Programmablaufes vermischt sich mit der prinzipiellen

Verknüpfungssteuerung des Ablaufes. Diese Mischform zur Steuerung eines Programmablaufes wird im Kapitel *Software* verdeutlicht.

Weitere ODER-Bausteine werden verwendet, um die Überlastausgänge der Leistungsstufen für die Ansteuerung der Antriebsmotoren zu überwachen.

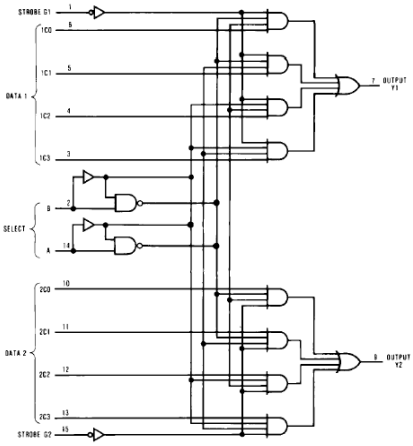
Die Leistungsstufen, über die Sie im Abschnitt *Leistungsteil* mehr Informationen erhalten, verfügen über sogenannte OCD-Ausgänge. Diese OCD-Ausgänge können durch entsprechende Beschaltung des Bausteines dazu genutzt werden, um eine Überlast des angeschlossenen Antriebsmotor durch eine erhöhte Stromaufnahme festzustellen und anzuzeigen.

Die Schaltschwelle dieses Signals ist durch die Beschaltung des Bausteines festzulegen. Diese Beschaltung können Sie dem Schaltplan entnehmen.

Ein weiterer Baustein des Logikteiles ist ein nicht invertierender 4-Kanal Multiplexer mit enable-Eingang, ein SMD-Baustein im SO16-Gehäuse des Typs 74ALS153M. Der Baustein dient der Feedbackumschaltung der einzelnen Inkrementalgeber der Antriebseinheiten von Achse 1 bis Achse 3. Die Impulse des Inkrementalgebers der Achse 4 werden nicht über diesen Baustein geführt.

Durch entsprechende Ansteuerung des Bausteines wird der Geberdatenkanal der im Moment angesteuerten Antriebseinheit auf einen DI-Port geschaltet.

Blockschaltbild des 74ALS153M

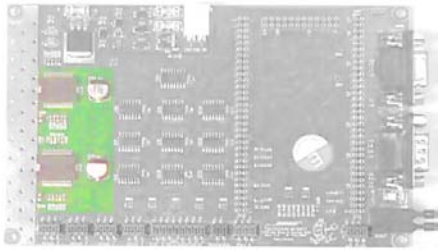


(Abb. 7-5)

Es kann durch diese Art der Ausführung jeweils eine Achse der Achsen 1 bis 3 und die Achse 4 in Betrieb sein, wenn eine Lageüberwachung zur Positionierung der Einheit erforderlich ist.

Im Servicefall gibt es auch Ausnahmen, bei welchen auch mehr als zwei Achsen in Betrieb gehen können. Allerdings ist dann keine absolut prozesssichere Lageüberwachung garantiert.

7.3 Leistungsteil



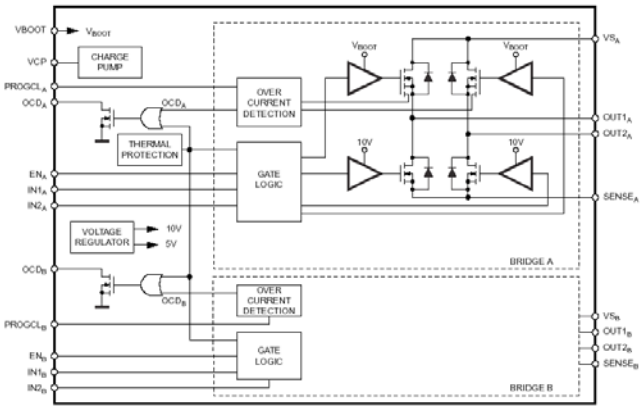
(Abb. 7-6)

Der Leistungsteil befindet sich auf der linken Seite der Basisplatine und besteht hauptsächlich aus zwei Motorsteuer-ICs und deren Beschaltung. Bei den Motorsteuer-ICs handelt es sich um zwei SMD-Bausteine im SO36-Gehäuse des Typs L6206PD des Herstellers ST-Microelectronics.

Von Ausgängen (DO-Ports) der Steuerung angesprochen, haben diese ICs die Aufgabe, die Motoren der Antriebseinheiten ein- und auszuschalten.

Die Motorsteuer-ICs arbeiten nach dem Prinzip einer H-Brücke. Jeder Baustein enthält zwei H-Bücken, es können also zwei Motoren pro IC gesteuert werden.

Blockschaltbild des L6206PD

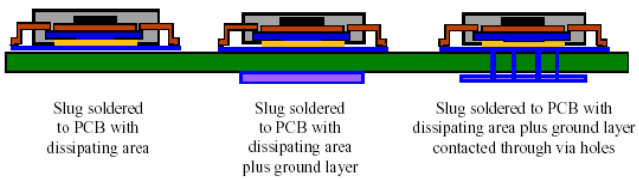


(Abb. 7-7)

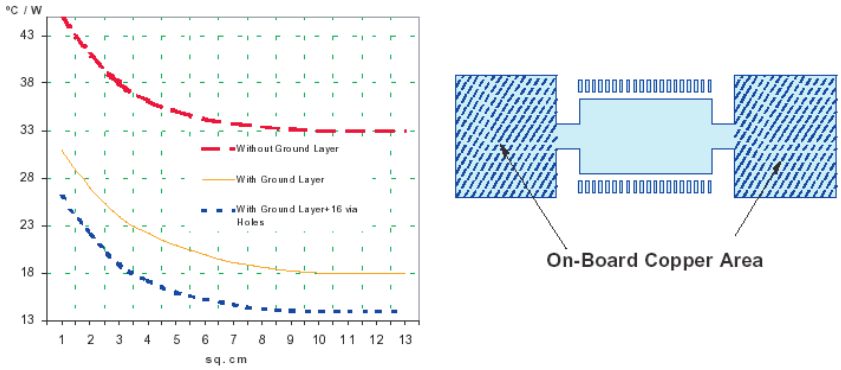
Um die Verlustwärme ableiten zu können, sind diese Bausteine mit einer Metallfläche an ihrer Unterseite versehen. Diese Metallfläche ist mit einer beidseitigen, durchkontaktierten Kühlfläche auf der Basisplatine verbunden.

Durch dieses Prinzip wird eine Beschädigung der Bauteile durch thermische Überlastung ohne Verwendung von Kühlkörpern verhindert.

Folgende Skizzen verdeutlichen dies :



(Abb. 7-8)



(Abb. 7-9)

Neben den Anschlüssen für die Motoren sind in den Bausteinen enable-Eingänge integriert um die ICs freizugeben bzw. zu sperren.

Wie bereits schon im Abschnitt *Logikteil* beschrieben, verfügen die Bausteine über sogenannte OCD-Ausgänge. Diese Ausgänge bieten Diagnosemöglichkeiten :

- Die OCD-Ausgänge können dazu genutzt werden, um eine Überlast des angeschlossenen Antriebsmotor durch eine erhöhte Stromaufnahme festzustellen und anzuzeigen. Die Schaltschwelle dieses Signals ist durch die Beschaltung des Bausteines festzulegen.

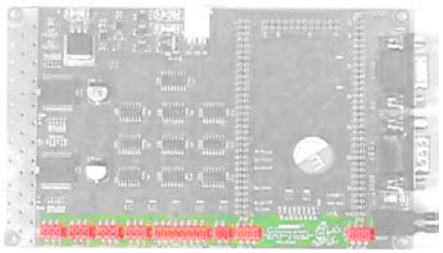
- Durch eine weitere Verschaltung dieser Ausgänge und Weiterführung auf einen DI-Port der Steuerung, kann sofort auf ein Diagnosesignal reagiert werden und entsprechende Vorgänge abgebrochen bzw. gestartet werden.

- Die Signale werden in der Steuerung weiterverarbeitet und über die HMI wird eine Meldung zur Bedienerführung ausgegeben.

Um bei den Motorsteuer-ICs ebenso wie bei allen anderen Komponenten der Basisplatine auf Kühlkörper verzichten zu können, wird die entstehende Wärme auf eine Kühlfläche der Basisplatine abgeführt. Diese Kühlfläche ist ausreichend groß, um Schäden durch thermische Überlastung zu verhindern.

Die Strombegrenzung der Antriebe wurde durch Beschaltung der ICs mit 18k Widerständen auf 1,2 A festgelegt.

7.4 Anschlussmöglichkeiten zur Peripherie



(Abb. 7-10)

Im unteren Bereich der Basisplatine sind die Anschlussmöglichkeiten zur Peripherie. Diese Leiterplattenbuchsen sind in SMD-Ausführung gestaltet. Die Steckplätze X4, X5, X6 und X7

sind für die Achsen 1 bis 4 reserviert. Über diese Steckverbindungen werden die Achsantriebe versorgt. Aufgelegt sind jeweils die Leistung für Drehrichtung plus, Leistung für Drehrichtung minus, Masse, die Spannungsversorgung für die Inkrementalgeber und das Feedback der Inkrementalgeber.

Über den Steckverbinder X8 werden die Signale der Referenzpunktschalter und der Endlagenschalter der Achsen 1 bis 4 aufgelegt. Die Spannungsversorgung wird pro Achse einmal aufgelegt und jeweils an den Antriebseinheiten entsprechend der Schalter verteilt.

Die Buchse X9 dient dem Anschluss des Neigungssensors und des Gehäuseüberwachungsschalters im Deckel des Messautomaten.

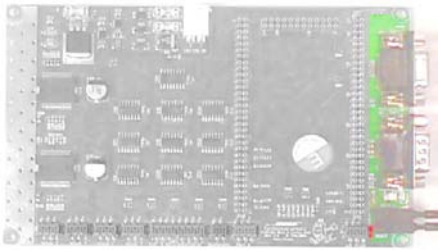
An der Leiterplattenbuchsen X10 kann weitere Peripherie wie Schalter, Taster, etc. angeschlossen werden. Diese Buchse ist bereits intern mit der Steuerung verschaltet und über invertierende Schmitt-Trigger Bausteine geführt um eindeutige Signalpegel von der Peripherie zu erhalten. Diese Eingänge sind auf bisher freie DI-Ports der Steuerung gelegt und dienen der optionalen Erweiterung der Basisplatine.

Über die Leiterplattenbuchse X11 wird die Verbindung zur Adapterplatine an der Frontplatte im Gehäuse des Messautomaten hergestellt. Dort befindet sich ein weiterer Reset-Taster, um die Steuerung rückzusetzen.

Um Softwareupdates der Steuerung benutzerfreundlich durchführen zu können, wurde dort ebenso ein Boot-Taster installiert. Mit diesen Tastern lässt sich die Steuerung in den Boot-Strap-Mode setzen, der erforderlich ist, um ein update zu übertragen.

Eine V24-Schnittstelle ist auf der Adapterplatine enthalten, welche auch in Abbildung 6-9 zu erkennen ist.

7.5 Serielle Schnittstellen



(Abb. 7-11)

Die seriellen Schnittstellen S0 und S1 befinden sich im rechten seitlichen Bereich der Basisplatine. Die RS232-Schnittstelle S0 ist für den Anschluss eines kompatiblen PCs reserviert. Über diese Schnittstelle kann der Messautomat wie in Abbildung 3-1 dargestellt, in den Messaufbau integriert werden.

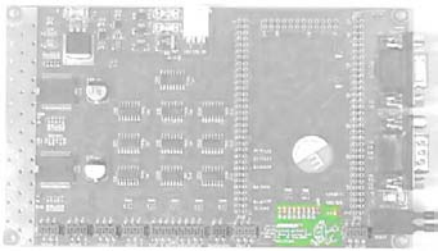
Im Servicefall kann über die Schnittstelle S0 der Messautomat mit der Software OLLICONTROL (OC) gesteuert werden. Diese Schnittstelle ist ebenso auf die Frontplatte im Gehäuse des Messautomaten über die Adapterplatine nach außen geführt.

Wie auch auf der Basisplatine vorhanden, sind über die Adapterplatine zwei Taster nach außen geführt. Die Funktion des Reset- und des Boot-Tasters wurde bereits im Abschnitt *Anschlussmöglichkeiten zur Peripherie* beschrieben.

Die RS232-Schnittstelle S1 ist für den Anschluss der HMI reserviert. Das LCD-Touchpanel kommuniziert über diese Schnittstelle mit der Steuerung.

Um eine Verwechslung der beiden Schnittstellen zu vermeiden, sind sie auf der Basisplatine gekennzeichnet. Zusätzlich wurde darauf geachtet, daß auf der Basisplatine eine Schnittstelle als Buchse, die andere Schnittstelle als Stecker ausgeführt ist.

7.6 Debugbereich

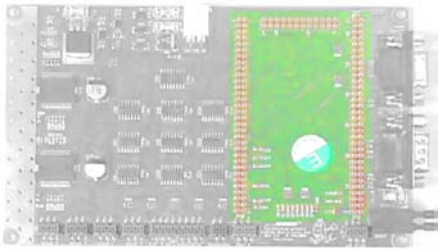


(Abb. 7-12)

Unterhalb der Prozessorplatine befindet sich der Debugbereich. Dort sind zu Simulations- und Testzwecken 8 LEDs angebracht. Diese LEDs sind an nicht für die Automation benötigte Ports des Mikrocontrollers angeschlossen und können für Debugzwecke benutzt werden.

Die Pinbelegung des Mikrocontrollers ist in Kapitel 8 ersichtlich. Dort können Sie auch die Bezeichnungen für die verwendeten Portpins der Debug-LEDs ersehen.

7.7 Reservierter Bereich



(Abb. 7-13)

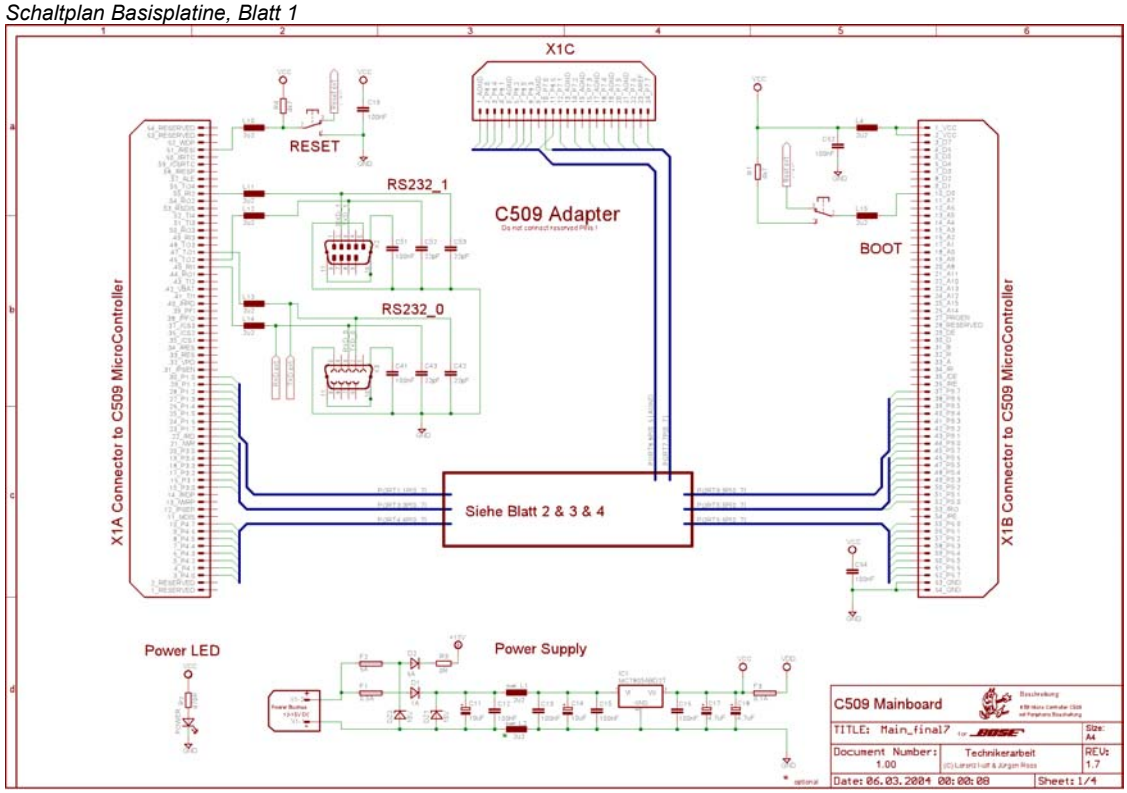
Dieser Bereich ist für die Prozessorplatine reserviert. Aus EMV-technischen Gründen wurde dieser Bereich für andere Bauelemente nicht genutzt und wurde somit ausschließlich für die Prozessorplatine freigehalten.

Wie bei der gesamten Basisplatine wurde auch hier darauf geachtet, die Führung der Leiterbahnen unter EMV-technischen Aspekten zu gestalten.

7.8 Schaltpläne der Basisplatine

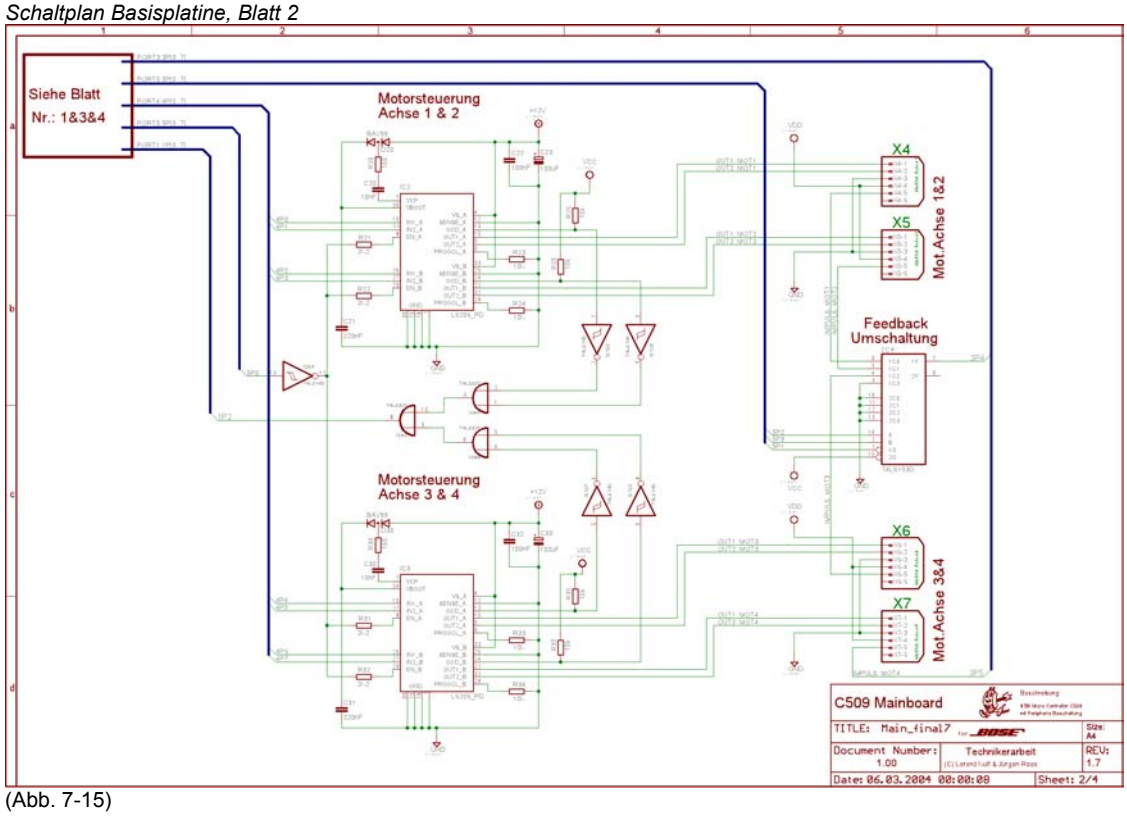
Die Schaltpläne und das gesamte Layout der Basisplatine wurden mit der CAD-Software Eagle, Version 4.11, erstellt.

Schaltplan Basisplatine, Blatt 1



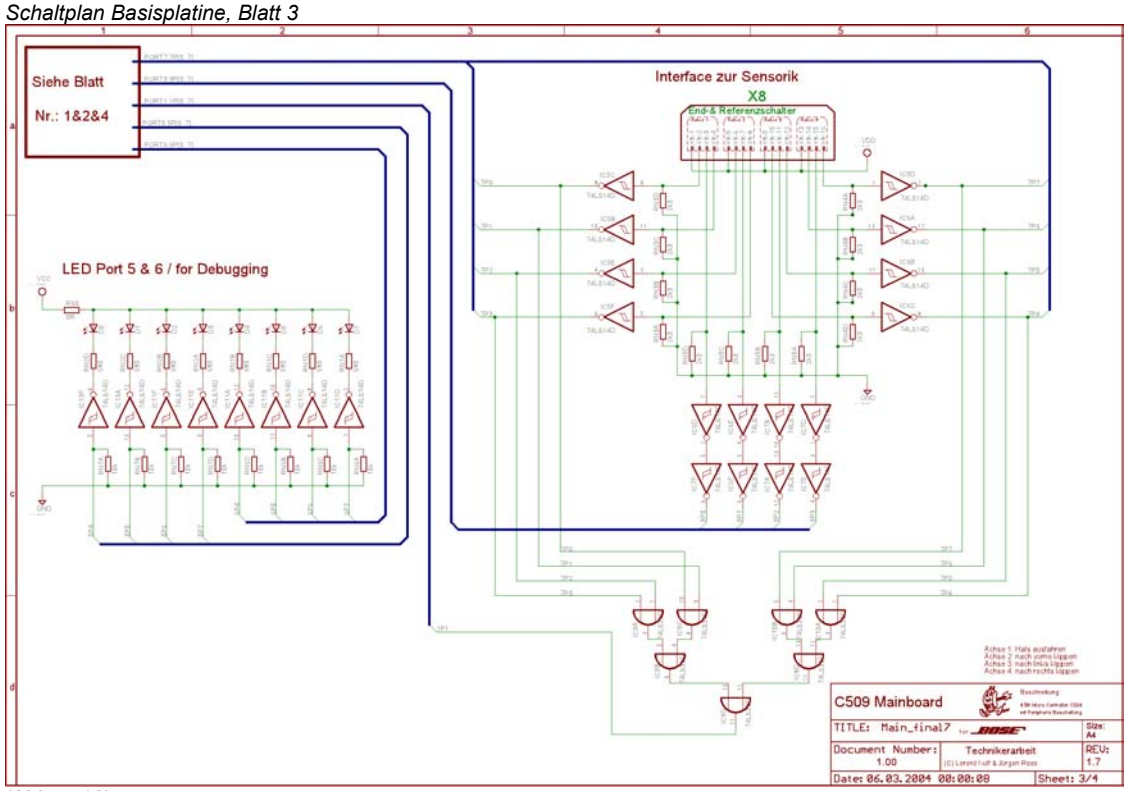
(Abb. 7-14)

Schaltplan Basisplatine, Blatt 2



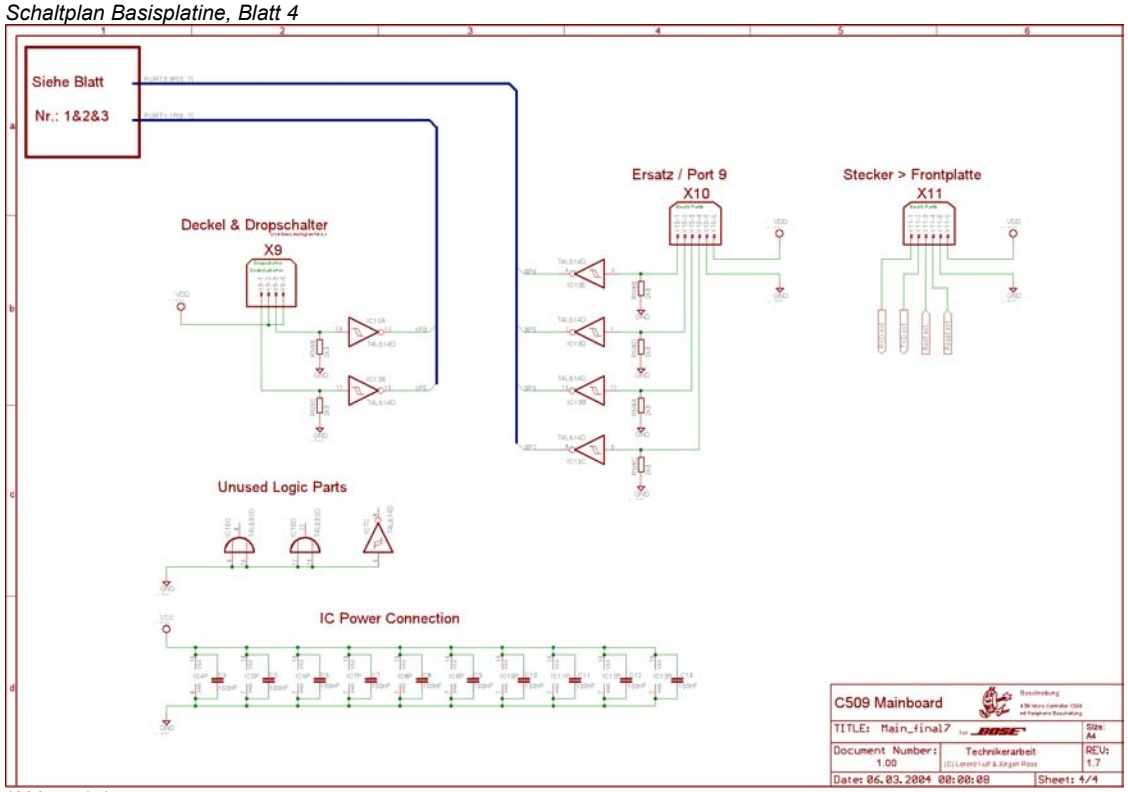
(Abb. 7-15)

Schaltplan Basisplatine, Blatt 3



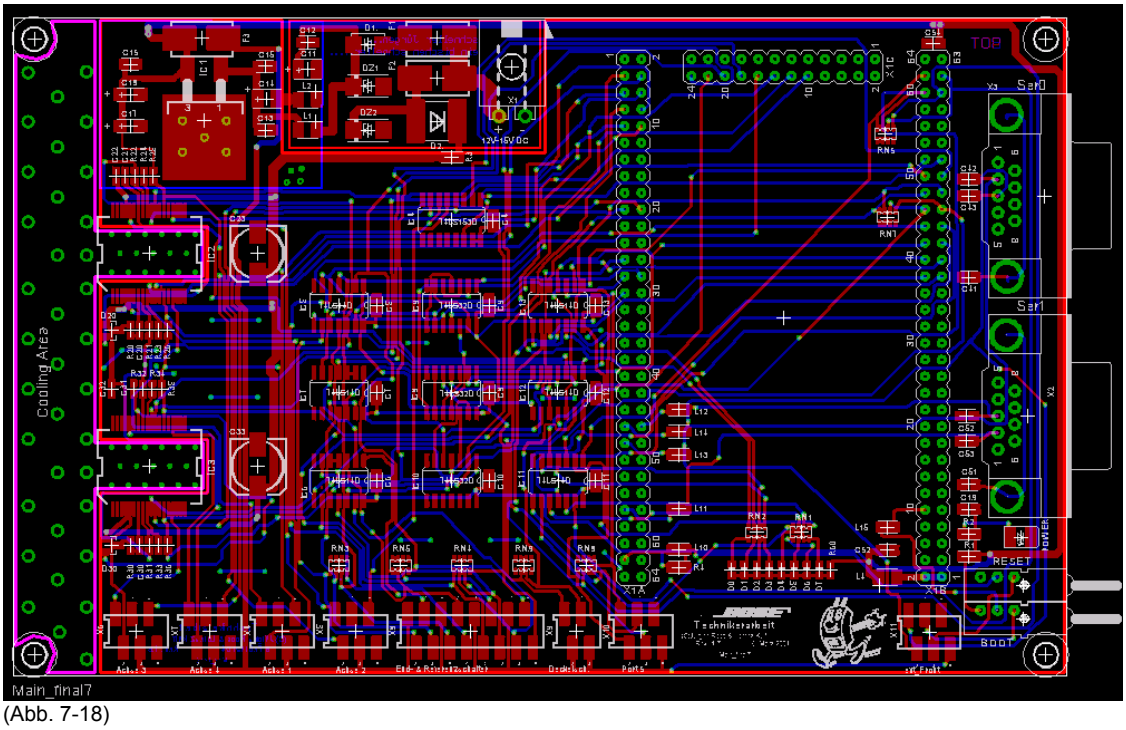
(Abb. 7-16)

Schaltplan Basisplatine, Blatt 4



(Abb. 7-17)

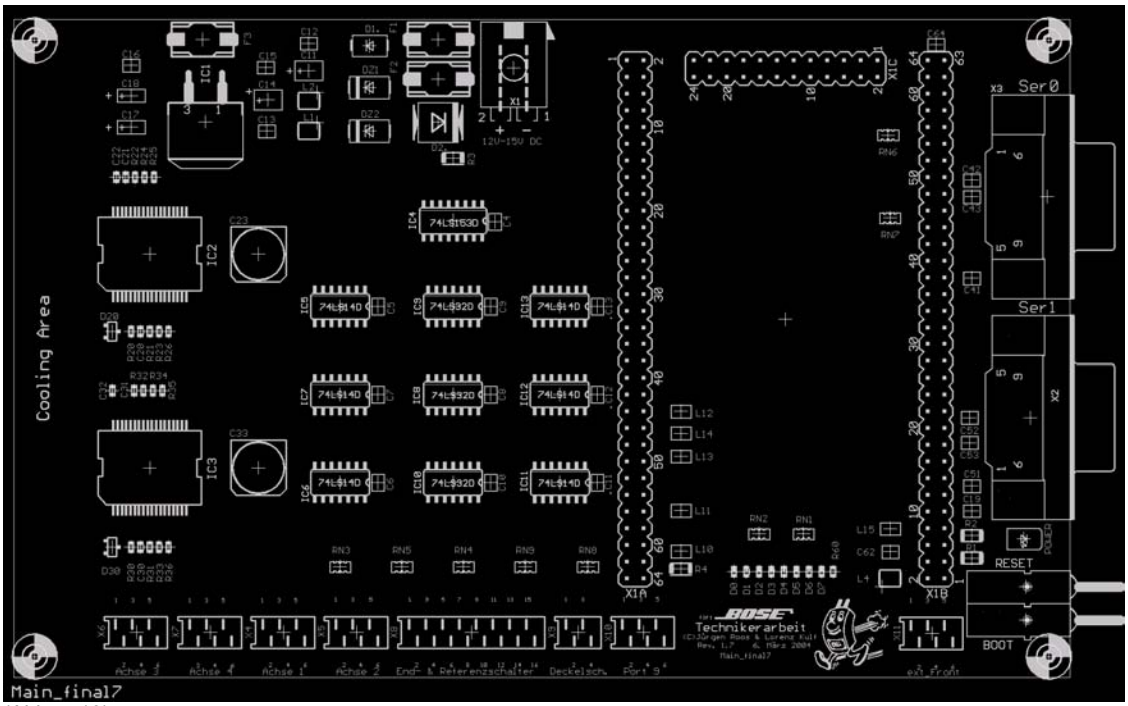
7.9 Layout Basisplatine



(Abb. 7-18)

Eine Gesamtansicht des Platinenlayouts. Es wird darauf verzichtet, jeden Layer der Platine hier abzubilden. Die vollständigen Ansichten der verschiedenen Layer sind auf der Dokumentations-CDR enthalten und können u. A. mit der CAD-Software Eagle, Version 4.11, eingesehen werden.

7.10 Bestückungsdruck Basisplatine



(Abb. 7-19)

8 Details Prozessorplatine

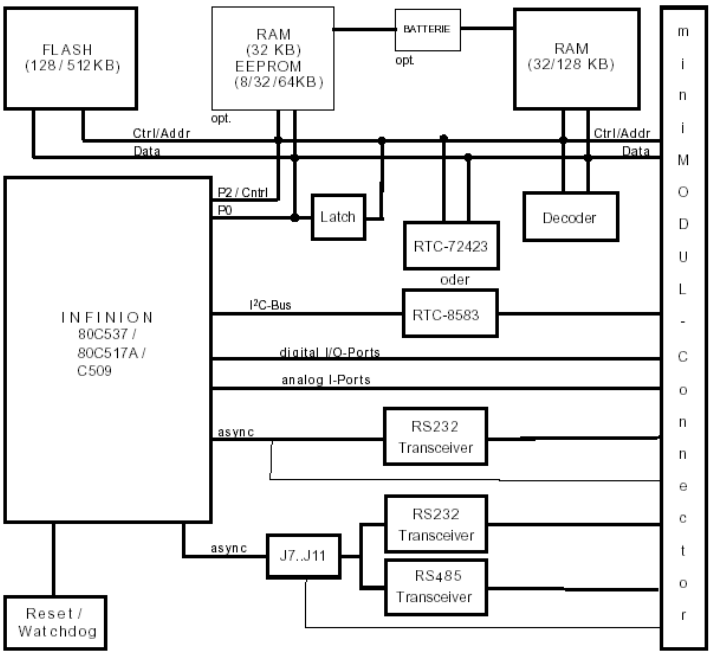
8.1 Allgemein

Details, die Sie in diesem Kapitel vermissen werden, können Sie Kapitel 6.2.2.2 entnehmen. Dort wurden bereits einige Informationen, die auch Details betreffen, behandelt.

In diesem Kapitel erhalten Sie Informationen über spezifische Eigenschaften bzw. Zusammenhänge der Prozessorplatine in Verbindung mit dem Messautomat.

8.2 Blockschaltbild Prozessorplatine

Blockschaltbild des miniModuls-509



(Abb. 8-1)

8.3 Pinbelegung Mikrocontroller

Portpin	bitadr.	Funktion	altern. Funktion	Pinnr. Phytec				Funktion	Beschaltung
P1.0	ja	In/Out	/INT 3 input	X1A	15b	30	30		
P1.1	ja	In/Out	INT 4 input	X1A	15a	29	29	Interrupt Endschalterschalter	Pulldown
P1.2	ja	In/Out	INT 5 input	X1A	14b	28	28	Interrupt OCD Motorsteuer-ICs	Pullup
P1.3	ja	In/Out	INT 6 input	X1A	14a	27	27	Neigungssensor	Pulldown
P1.4	ja	In/Out	/INT 2 input	X1A	13b	26	26		
P1.5	ja	In/Out	In/Out	X1A	13a	25	25	Deckelschalter	Pulldown
P1.6	ja	In/Out	CLKOUT	X1A	12b	24	24		
P1.7	ja	In/Out	Timer2 extern input	X1A	12a	23	23		
P3.0	ja	In/Out	Serial0 RxD 0	X1A	8a	15	15	Serielle Schnittstelle 0 RxD	
P3.1	ja	In/Out	Serial0 TxD 0	X1A	8b	16	16	Serielle Schnittstelle 0 TxD	
P3.2	ja	In/Out	/INT 0 input	X1A	9a	17	17		
P3.3	ja	In/Out	/INT 1 input	X1A	9b	18	18		
P3.4	ja	In/Out	Counter 0 input	X1A	10a	19	19	Inkrementalgeber Achse 1-3	
P3.5	ja	In/Out	Counter 1 input	X1A	10b	20	20	Inkrementalgeber Achse 4	
P3.6	ja	In/Out	/WR	X1A	11a	21	21		
P3.7	ja	In/Out	/RD	X1A	11b	22	22		
P4.0	ja	In/Out	CM0	X1A	2a	3	3	Antrieb Achse 1 Richtung (-)	
P4.1	ja	In/Out	CM1	X1A	2b	4	4	Antrieb Achse 1 Richtung (+)	
P4.2	ja	In/Out	CM2	X1A	3a	5	5	Antrieb Achse 2 Richtung (-)	
P4.3	ja	In/Out	CM3	X1A	3b	6	6	Antrieb Achse 2 Richtung (+)	
P4.4	ja	In/Out	CM4	X1A	4a	7	7	Antrieb Achse 3 Richtung (-)	
P4.5	ja	In/Out	CM5	X1A	4b	8	8	Antrieb Achse 3 Richtung (+)	
P4.6	ja	In/Out	CM6	X1A	5a	9	9	Antrieb Achse 4 Richtung (-)	
P4.7	ja	In/Out	CM7	X1A	5b	10	10	Antrieb Achse 4 Richtung (+)	
P5.0	ja	In/Out	CCM0	X1B	26b	52	116	enable PWR Motoren - enable mit 0	
P5.1	ja	In/Out	CCM1	X1B	26a	51	115	enable Multiplexer - enable mit 0	
P5.2	ja	In/Out	CCM2	X1B	25b	50	114	Multiplexer select Geberdatenkanal	
P5.3	ja	In/Out	CCM3	X1B	25a	49	113	Multiplexer select Geberdatenkanal	
P5.4	ja	In/Out	CCM4	X1B	24b	48	112	DebugLED 0	Pulldown
P5.5	ja	In/Out	CCM5	X1B	24a	47	111	DebugLED 1	Pulldown
P5.6	ja	In/Out	CCM6	X1B	23b	46	110	DebugLED 2	Pulldown
P5.7	ja	In/Out	CCM7	X1B	23a	45	109	DebugLED 3	Pulldown
P6.0	nein	In/Out		X1B	28a	55	119		
P6.1	nein	In/Out	Serial1 RxD 1	X1B	28b	56	120	Serielle Schnittstelle 1 RxD	
P6.2	nein	In/Out	Serial1 TxD 1	X1B	29a	57	121	Serielle Schnittstelle 1 TxD	
P6.3	nein	In/Out		X1B	29b	58	122		
P6.4	nein	In/Out		X1B	30a	59	123	DebugLED 4	Pulldown
P6.5	nein	In/Out		X1B	30b	60	124	DebugLED 5	Pulldown
P6.6	nein	In/Out		X1B	31a	61	125	DebugLED 6	Pulldown
P6.7	nein	In/Out		X1B	31b	62	126	DebugLED 7	Pulldown
P7.0	nein	In	AnIn0	X1C	5b	10	138	Endlage-, Achse 1	Pulldown
P7.1	nein	In	AnIn1	X1C	6b	12	140	Endlage+, Achse 1	Pulldown
P7.2	nein	In	AnIn2	X1C	7b	14	142	Endlage-, Achse 2	Pulldown
P7.3	nein	In	AnIn3	X1C	8b	16	144	Endlage+, Achse 2	Pulldown
P7.4	nein	In	AnIn4	X1C	9b	18	146	Endlage-, Achse 3	Pulldown
P7.5	nein	In	AnIn5	X1C	10b	20	148	Endlage+, Achse 3	Pulldown
P7.6	nein	In	AnIn6	X1C	11b	22	150	Endlage-, Achse 4	Pulldown
P7.7	nein	In	AnIn7	X1C	12b	24	152	Endlage+, Achse 4	Pulldown
P8.0	nein	In	AnIn8	X1C	1b	2	130		
P8.1	nein	In	AnIn9	X1C	2b	4	132		
P8.2	nein	In	AnIn10	X1C	3b	6	134		
P8.3	nein	In	AnIn11	X1C	4b	8	136		
P8.4	nein	In	AnIn12	X1C	2a	3	131		
P8.5	nein	In	AnIn13	X1C	4a	7	135		
P8.6	nein	In	AnIn14	X1C	6a	11	139		
P9.0	nein	In/Out	CC10	X1B	22b	44	108	Referenzpunktschalter Achse 1	Pulldown
P9.1	nein	In/Out	CC11	X1B	22a	43	107	Referenzpunktschalter Achse 2	Pulldown
P9.2	nein	In/Out	CC12	X1B	21b	42	106	Referenzpunktschalter Achse 3	Pulldown
P9.3	nein	In/Out	CC13	X1B	21a	41	105	Referenzpunktschalter Achse 4	Pulldown
P9.4	nein	In/Out	CC14	X1B	20b	40	104	mit Stecker X10 / 1 kontaktiert	Pulldown
P9.5	nein	In/Out	CC15	X1B	20a	39	103	mit Stecker X10 / 2 kontaktiert	Pulldown
P9.6	nein	In/Out	CC16	X1B	19b	38	102	mit Stecker X10 / 3 kontaktiert	Pulldown
P9.7	nein	In/Out	CC17	X1B	19a	37	101	mit Stecker X10 / 4 kontaktiert	Pulldown
			Serial0 RxD 0	X1A	23a	45	45	Serielle Schnittstelle 0 RxD	
			Serial0 TxD 0	X1A	24a	47	47	Serielle Schnittstelle 0 TxD	
			Serial1 RxD 1	X1A	28a	55	55	Serielle Schnittstelle 1 RxD	
			Serial1 TxD 1	X1A	23b	46	46	Serielle Schnittstelle 1 TxD	
		AGND	AGND	X1C	1a	1	129		
		RESET	/Reset	X1A	31a	61	61	RESET	
		BOOT / D0	BOOT 537	X1B	5b	10	74	BOOT	
		VCC	VCC	X1B	1a	1	65	5V Power Supply for C509	
		VCC	VCC	X1B	1b	2	66	5V Power Supply for C509	
		GND	GND	X1B	32a	63	127	GND Power Supply for C509	
		GND	GND	X1B	32b	64	128	GND Power Supply for C509	

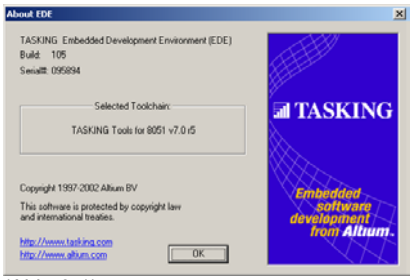
(Abb. 8-2)

9 Programmierung des Mikrocontrollers C509

9.1 Allgemeines

Mit dem Mikrocontroller C509 von Infineon wurde die Steuerung des Messautomaten verwirklicht. Der Mikrocontroller ist auf der Prozessorplatine integriert.

Als Programmiersprache wurde die Sprache C gewählt, die Entwicklungsumgebung von Altium.



(Abb. 9-1)

Tasking **E**mbedded **D**evelopment **E**nvironment, **EDE**, ist ein leistungsstarkes Tool, das zur Entwicklung des C-Programmes dient.

Das C-Programm des Mikrocontrollers stellt den Kern des Messautomaten dar. Hier werden die Schnittstellen des Controllers verwaltet, der Signalaustausch mit der Peripherie gesteuert.

Als Compiler dient der 8051 Cross Compiler von Tasking.

Um das generierte project.hex file in den Controller zu übertragen, wird das flashtool 98 von Phytec verwendet. Über eine RS232-Verbindung zur seriellen Schnittstelle S0 des Controllers werden die Daten in den EEprom-Speicher des Controllers geladen.

(Abb. 9-2)

Hierzu muß der Controller in den Boot-Strap-Mode gesetzt werden. Dies erreicht man über das gleichzeitige betätigen des Reset- und des Boottasters. Dabei muß der Boottaster länger als der Resettaster betätigt werden.

Die beiden Taster sind ebenso wie die serielle Schnittstelle S0 über die Adapterplatine auf die Frontplatte des Messautomaten herausgeführt.

Genauere Bedienungshinweise zum „flashen“ des EEproms finden Sie in den folgenden Kapiteln.

9.2 Tasking

Nach der Installation der Entwicklungsumgebung EDE 8051auf einem kompatiblen PC kann nun EDE 8051 gestartet werden.

Informationen zur Installation von EDE 8051 erhalten Sie aus dem Handbuch zu EDE 8051 oder sind über Tasking zu beziehen.

Nach dem Start der Entwicklungsumgebung ist zuerst ein neuer Arbeitsbereich, ein ProjectSpace, zu erstellen.

In einem ProjectSpace können verschiedene Projekte abgelegt oder bearbeitet werden. Das momentan zu bearbeitende Projekt ist als „current project“ zu deklarieren. Ein Projekt kann mehrere Dateien beinhalten.

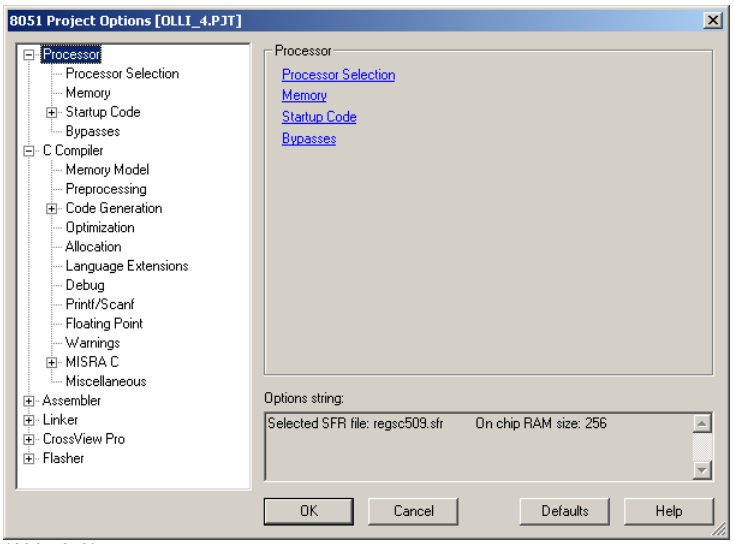
Das EDE 8051-Projekt des Messautomaten beinhaltet folgende Dateitypen :

- Dateien mit C-Quellcode : *.c
- Headerdateien : *.h
- Controllerspezifische Dateien : *.sfr
- Umgebungsspezifische Dateien: *.asm
- Die nach dem compilieren generierte *.hex Datei befindet sich im Stammverzeichnis des ProjectSpace

Die Dateien können über die Eigenschaften des Projektes verwaltet werden. Dort werden nicht mehr benötigte Dateien aus dem Projekt ausgegliedert oder neue Dateien dem EDE 8051 Projekt hinzugefügt.

Ebenso können dort Verzeichnisse eingestellt werden, in welche Dateien geschrieben werden oder in welchen auf Dateien zugegriffen werden.

Zu den Grundeinstellungen gehört auch die Einstellung der Projektoptionen. Diese Projektoptionen können auch von anderen Projekten, auch außerhalb des aktiven ProjectSpace, übernommen werden.



(Abb. 9-3)

In den Projektoptionen müssen Einstellungen zu verwendeter Hardware oder aber auch beispielsweise Compilereinstellungen vorgenommen werden.

Durch die Einstellung der Projektoptionen wird die Software an die Hardware angepasst.

Ein Auszug der Einstellungen :

- Einstellungen zum Prozessor
  - Prozessortyp
  - Größe des Speichers
- Einstellungen des C Compilers
  - Preprozessoreinstellungen
  - Codegenerierung
  - Optimierung
  - Ausgabe von Fehlern und Warnungen
  - Speichermodell
- Einstellungen zum Assembler
  - Angaben zu listfiles
  - Registerbänke
  - Einstellungen zum Linker
  - Ausgabeformat
  - Ausgabe von Fehlern und Warnungen
- Einstellungen zu Cross View Pro
  - Initialisierung
  - Verschiedene Einstellungen zum Debugger
- Flasher
  - Übertragungsrate
  - Flashertyp

Die gesamten Einstellungen werden in einer Datei abgelegt und können so jederzeit gegen eine andere Optionseinstellung ausgetauscht werden.

Die Oberfläche von EDE 8051 kann individuell an verschiedene Bedürfnisse angepasst werden. Schrift- und Hintergundfarbe, Schriftgröße und sämtliche Einstellungen, bezüglich der Darstellung, können wie bei Windows-Arbeitsoberflächen üblich, eingestellt werden.

9.3 Ablaufdiagramme C-Projekt

Die folgenden Seiten sollen einen Überblick über das C-Projekt verschaffen. Hier wird nicht der Zusammenhang der einzelnen C-Dateien zueinander dargestellt, sondern der Ablauf der Automation im Gesamten.

Auf eine Darstellung aller Details wird bewußt verzichtet. Die Ablaufdiagramme wurden so gestaltet, daß eine Übersicht über die Automation gegeben werden kann.

Das C-Programm wurde so gestaltet, daß ein zyklischer Ablauf gegeben ist. Die Hauptroutine ist der Ausgangspunkt der Ablaufdiagramme. Diese Hauproutine ist Koordinator des

gesamten Ablaufes. Von hier werden Funktionen des Messautomaten gesteuert, Unterprogramme aufgerufen und Abläufe organisiert.

Die einzelnen Schritte im Ablaufdiagramm stellen nicht die wirklich programmierten C-Funktionen dar, sondern wie schon im ersten Abschnitt dieses Kapitels erwähnt, sinngemäß die Abläufe im Programm. So können z. T. verschiedene C-Funktionen über mehrere Dateien zusammengefasst sein.

Solche Schritte werden hier als Modul bezeichnet. Jedes Modul erhält eine Modulnummer. Das Lupensymbol an einem Modul macht deutlich, daß es zu diesem Modul eine noch genauere Darstellung gibt. Dieses Prinzip ist über mehrere Ebenen gültig.

Die Querverweise sind logisch aufgebaut. Die Blattnummer des Querverweises ergibt sich aus der Modulnummer des entsprechenden Schrittes. Zur besseren Orientierung sind Querverweise auf externe Module farblich gestaltet.

Die Modulnummern sind ebenso logisch aufgebaut. Die Module einer tieferen Ebene erhalten eine Modulnummer, die sich auf das Ursprungsmodul der höheren Ebene bezieht.

Das Prinzip dieser Nummerierung finden Sie nicht nur hier, dieser Grundsatz wurde an allen neuen Komponenten des Messautomaten umgesetzt, wie Sie beispielsweise auch aus den Stromlaufplänen des Messautomaten entnehmen können.

Durch diese Gliederung lässt sich der Programmaufbau verständlich darstellen.

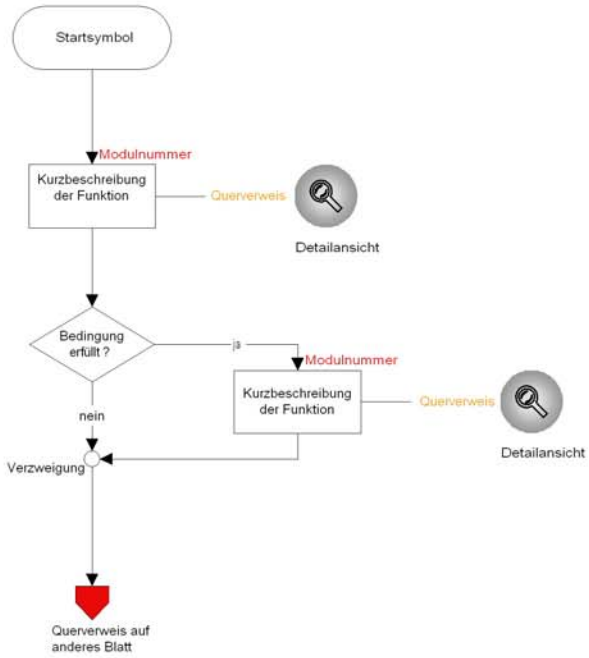
Im jeweiligen dargestellten Schritt erhalten Sie eine Kurzbeschreibung der Funktion des Moduls.

Diese Informationen können Sie nochmals zusammengefasst der Legende auf Blatt 0.1 des Ablaufdiagrammes entnehmen.

Auf Blatt 0.2 des Ablaufdiagrammes können Sie eine Übersicht des prinzipiellen Aufbaus des Programms erkennen. Die Struktur des Programmes entspricht einer Ablaufsteuerung mit Ereignisroutinen.

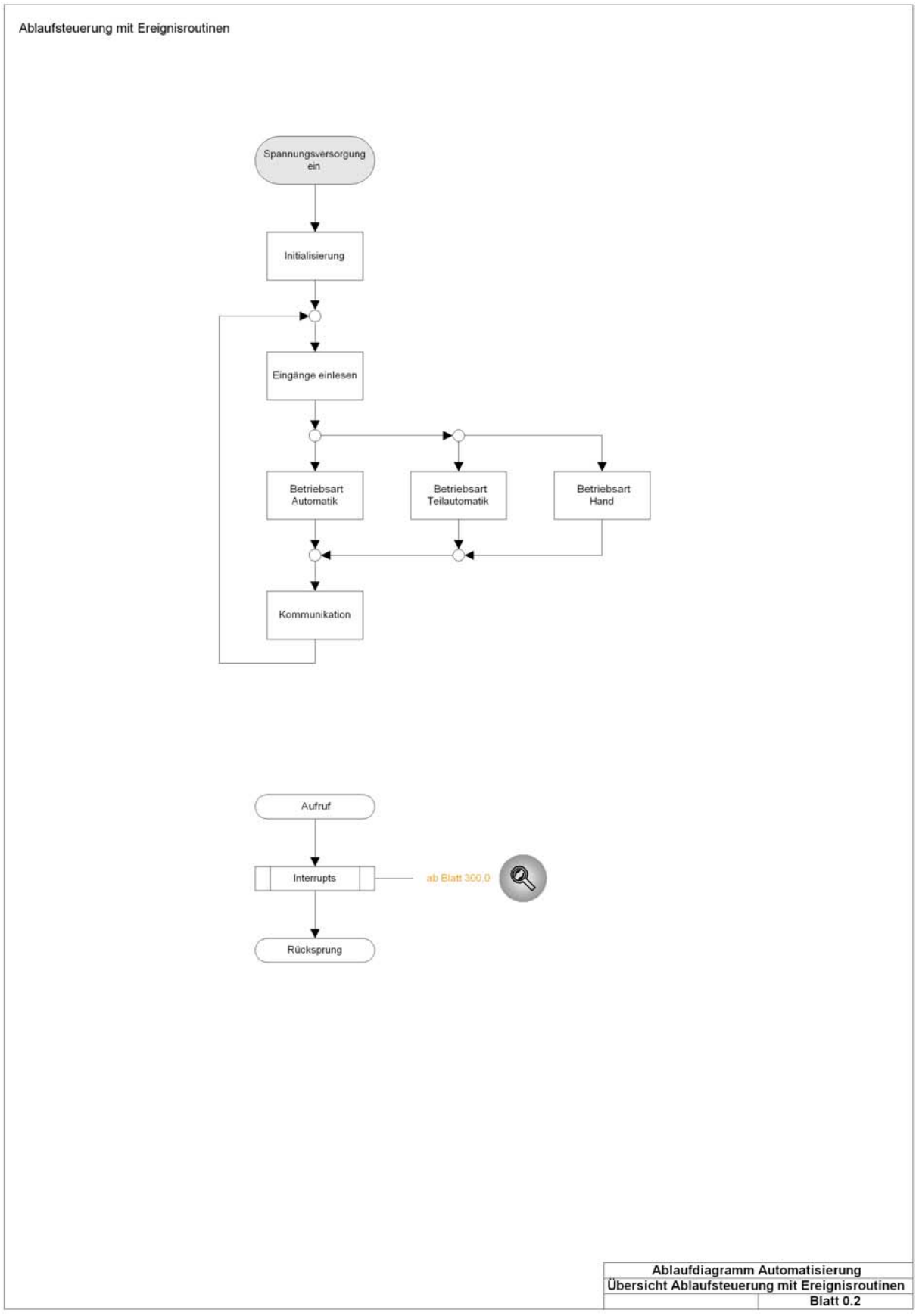
Ablaufdiagramm zur Programmierung der Steuerung Messautomat
Infineon C 509

Legende :

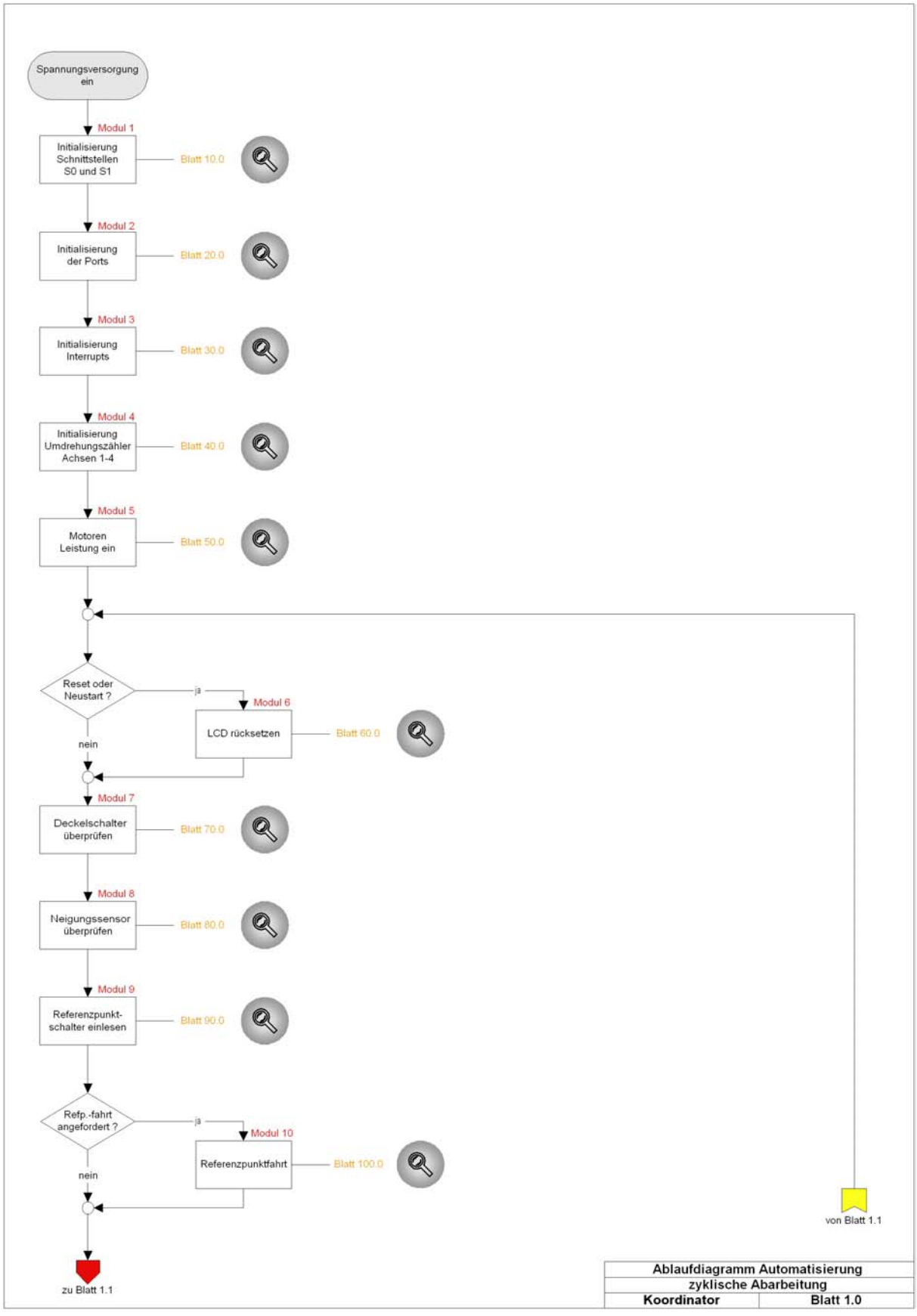


Ablaufdiagramm Automatisierung	
Titel	
	Blatt 0.1

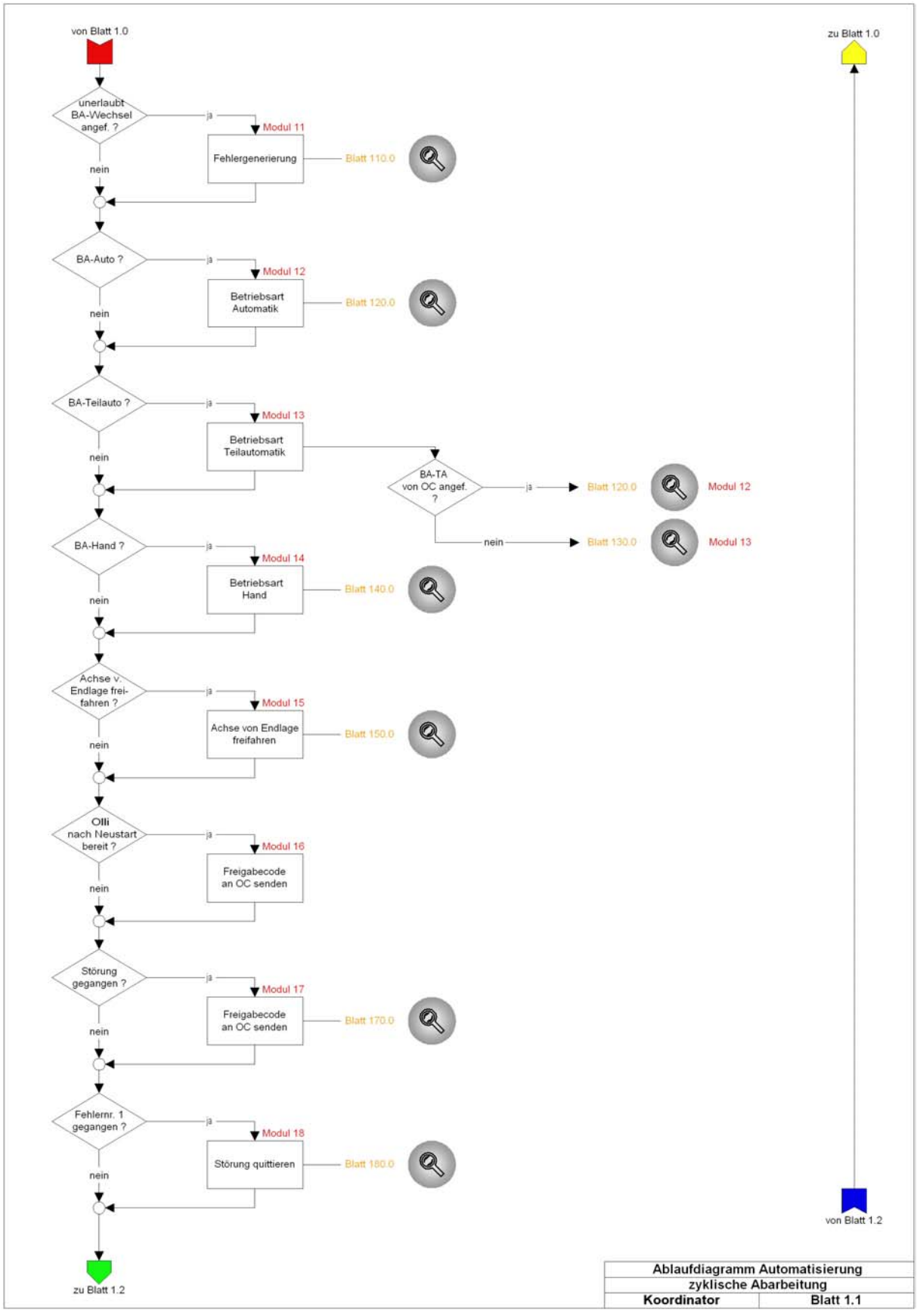
(Abb. 9-4)


Ablaufsteuerung mit Ereignisroutinen
Spannungsversorgung ein
Initialisierung
Eingänge einlesen
Betriebsart Automatik
Betriebsart Teilautomatik
Betriebsart Hand
Kommunikation
Aufruf
Interrupts
ab Blatt 300.0
Rücksprung
Ablaufdiagramm Automatisierung
Übersicht Ablaufsteuerung mit Ereignisroutinen
Blatt 0.2

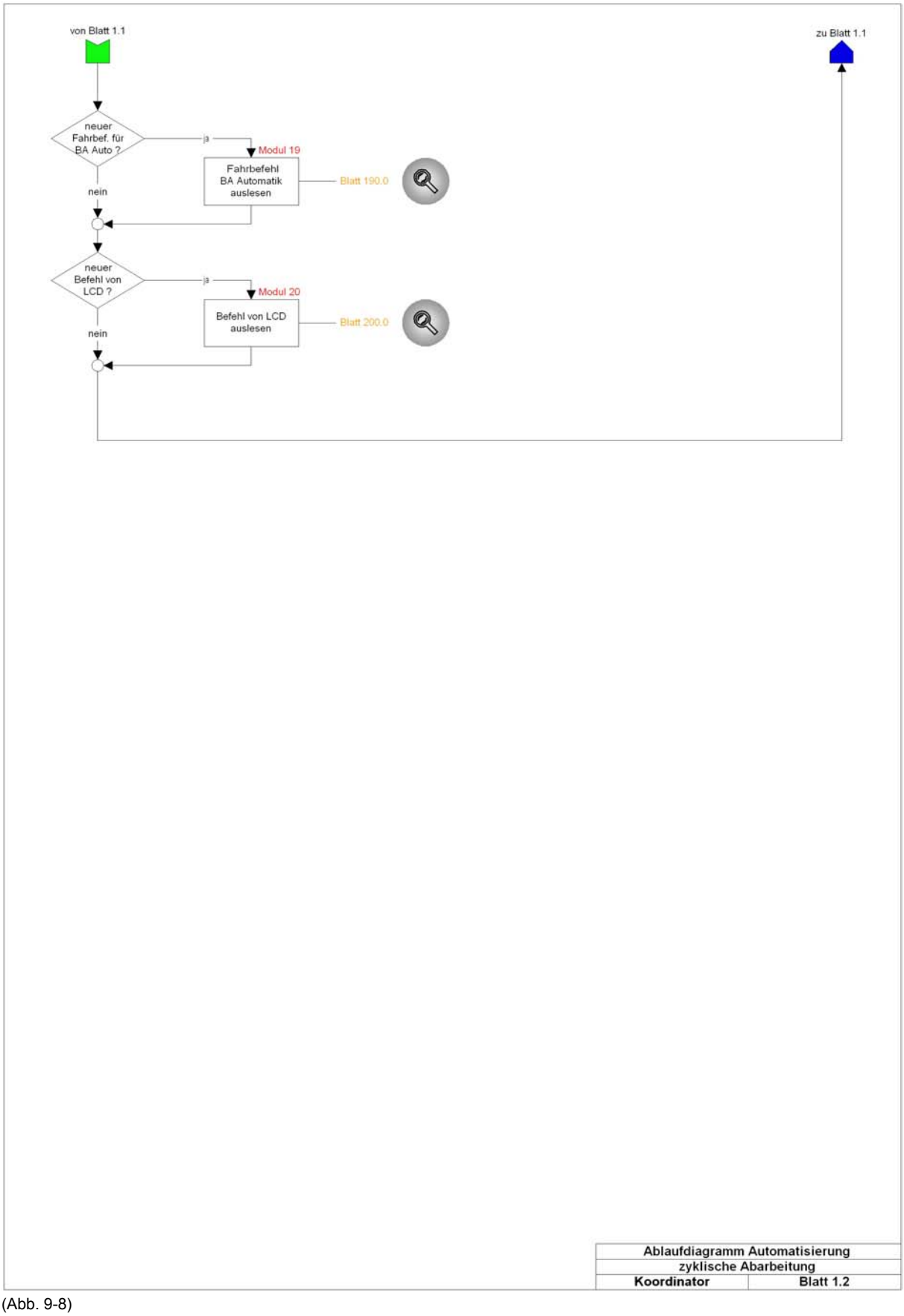
(Abb. 9-5)


Spannungsversorgung ein
Modul 1
Initialisierung Schnittstellen S0 und S1
Blatt 10.0
Modul 2
Initialisierung der Ports
Blatt 20.0
Modul 3
Initialisierung Interrupts
Blatt 30.0
Modul 4
Initialisierung Umdrehungszähler Achsen 1-4
Blatt 40.0
Modul 5
Motoren Leistung ein
Blatt 50.0
Reset oder Neustart ?
ja
Modul 6
LCD rücksetzen
Blatt 60.0
nein
Modul 7
Deckelschalter überprüfen
Blatt 70.0
Modul 8
Neigungssensor überprüfen
Blatt 80.0
Modul 9
Referenzpunkt-schalter einlesen
Blatt 90.0
Refp.-fahrt angefordert ?
ja
Modul 10
Referenzpunktfahrt
Blatt 100.0
nein
zu Blatt 1.1
von Blatt 1.1
Ablaufdiagramm Automatisierung
zyklische Abarbeitung
Koordinator
Blatt 1.0

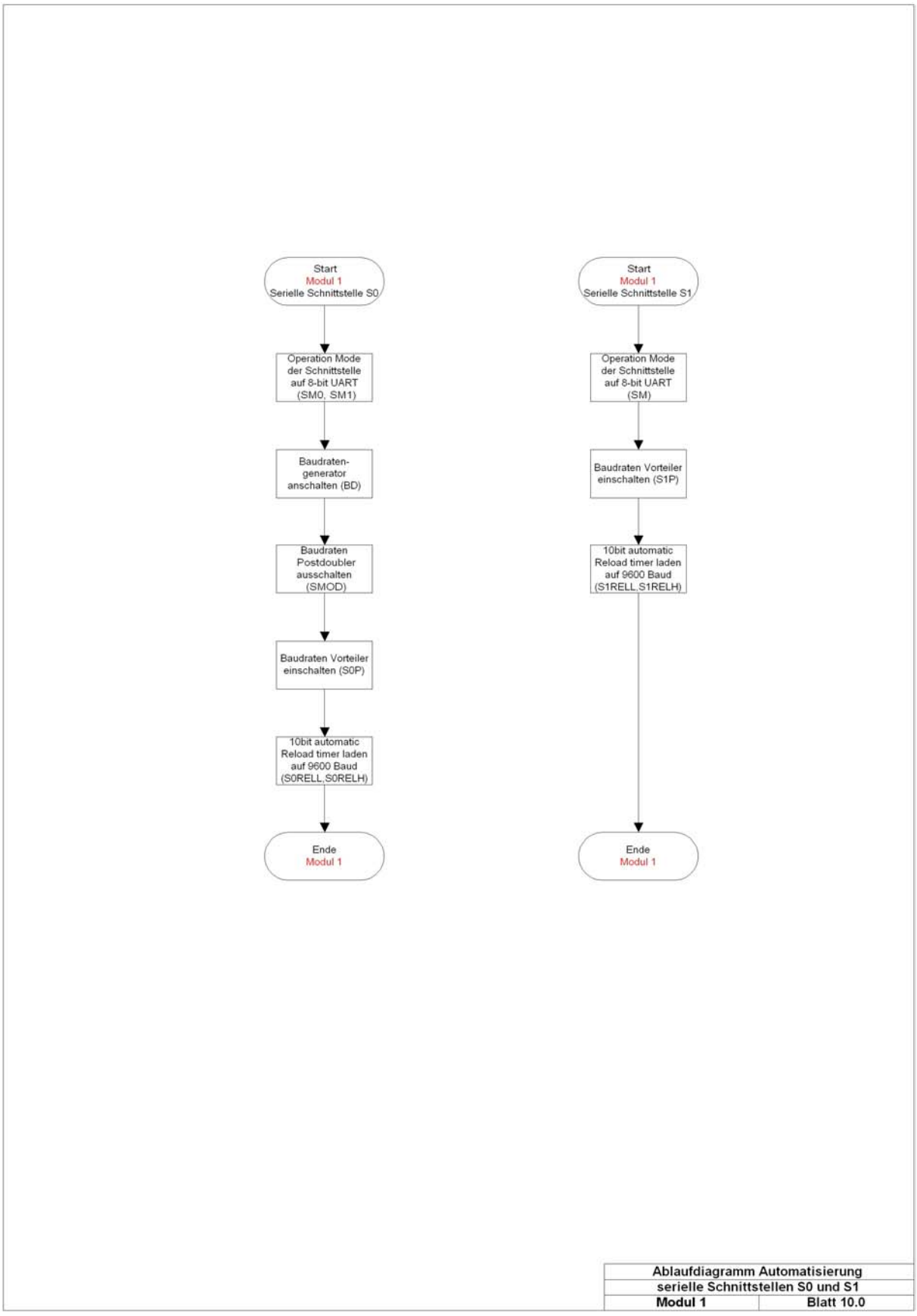
(Abb. 9-6)


von Blatt 1.0
zu Blatt 1.0
unerlaubt BA-Wechsel angef. ?
ja
Modul 11
Fehlergenerierung
Blatt 110.0
nein
BA-Auto ?
ja
Modul 12
Betriebsart Automatik
Blatt 120.0
nein
BA-Teilauto ?
ja
Modul 13
Betriebsart Teilautomatik
BA-TA von OC angef. ?
ja
Blatt 120.0
Modul 12
nein
Blatt 130.0
Modul 13
nein
BA-Hand ?
ja
Modul 14
Betriebsart Hand
Blatt 140.0
nein
Achse v. Endlage frei-fahren ?
ja
Modul 15
Achse von Endlage freifahren
Blatt 150.0
nein
Olli nach Neustart bereit ?
ja
Modul 16
Freigabecode an OC senden
nein
Störung gegangen ?
ja
Modul 17
Freigabecode an OC senden
Blatt 170.0
nein
Fehlernr. 1 gegangen ?
ja
Modul 18
Störung quittieren
Blatt 180.0
nein
zu Blatt 1.2
von Blatt 1.2
Ablaufdiagramm Automatisierung
zyklische Abarbeitung
Koordinator
Blatt 1.1

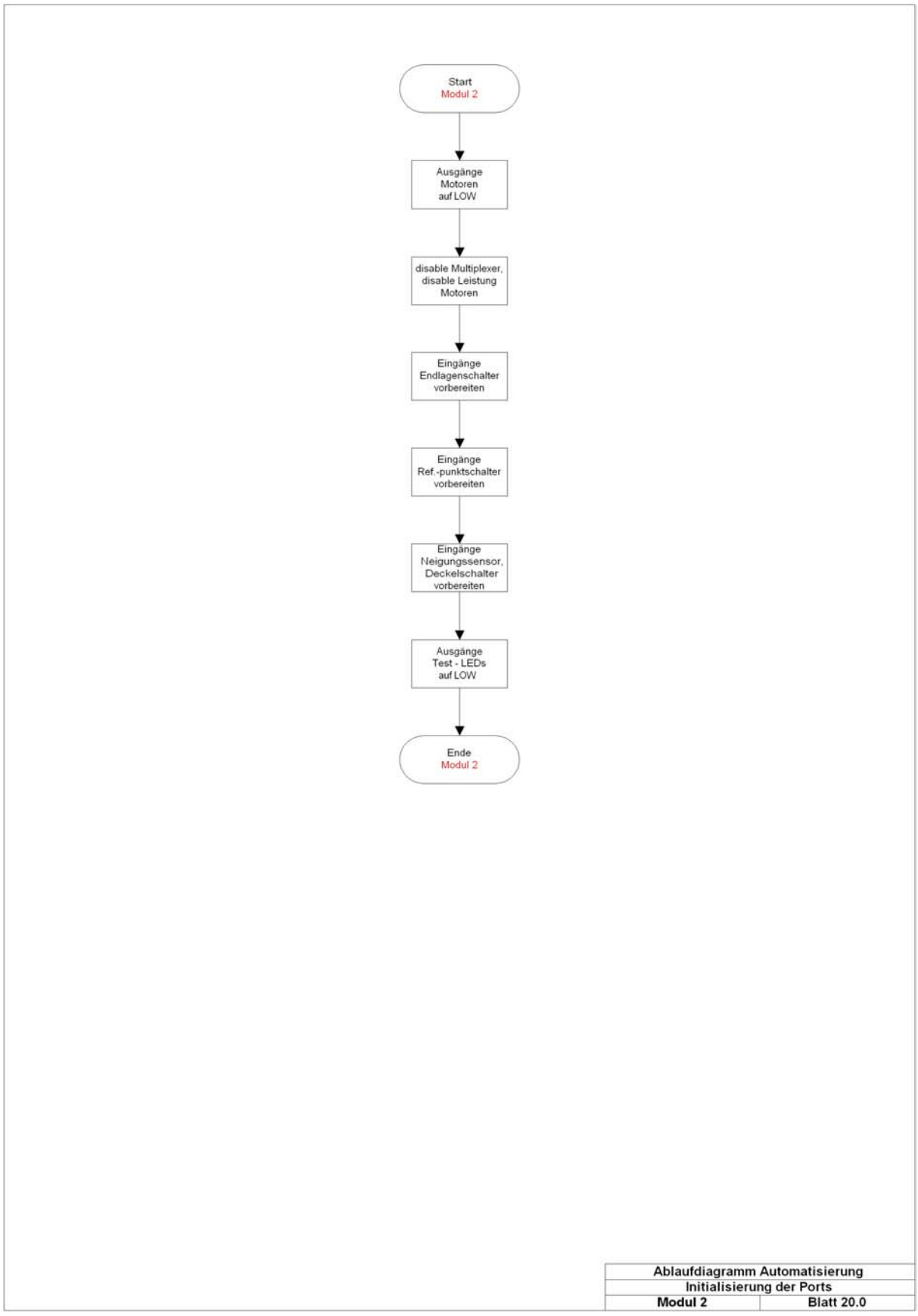
(Abb. 9-7)


von Blatt 1.1
zu Blatt 1.1
neuer Fahrbef. für BA Auto ?
ja
Modul 19
Fahrbefehl BA Automatik auslesen
Blatt 190.0
nein
neuer Befehl von LCD ?
ja
Modul 20
Befehl von LCD auslesen
Blatt 200.0
nein
Ablaufdiagramm Automatisierung
zyklische Abarbeitung
Koordinator
Blatt 1.2

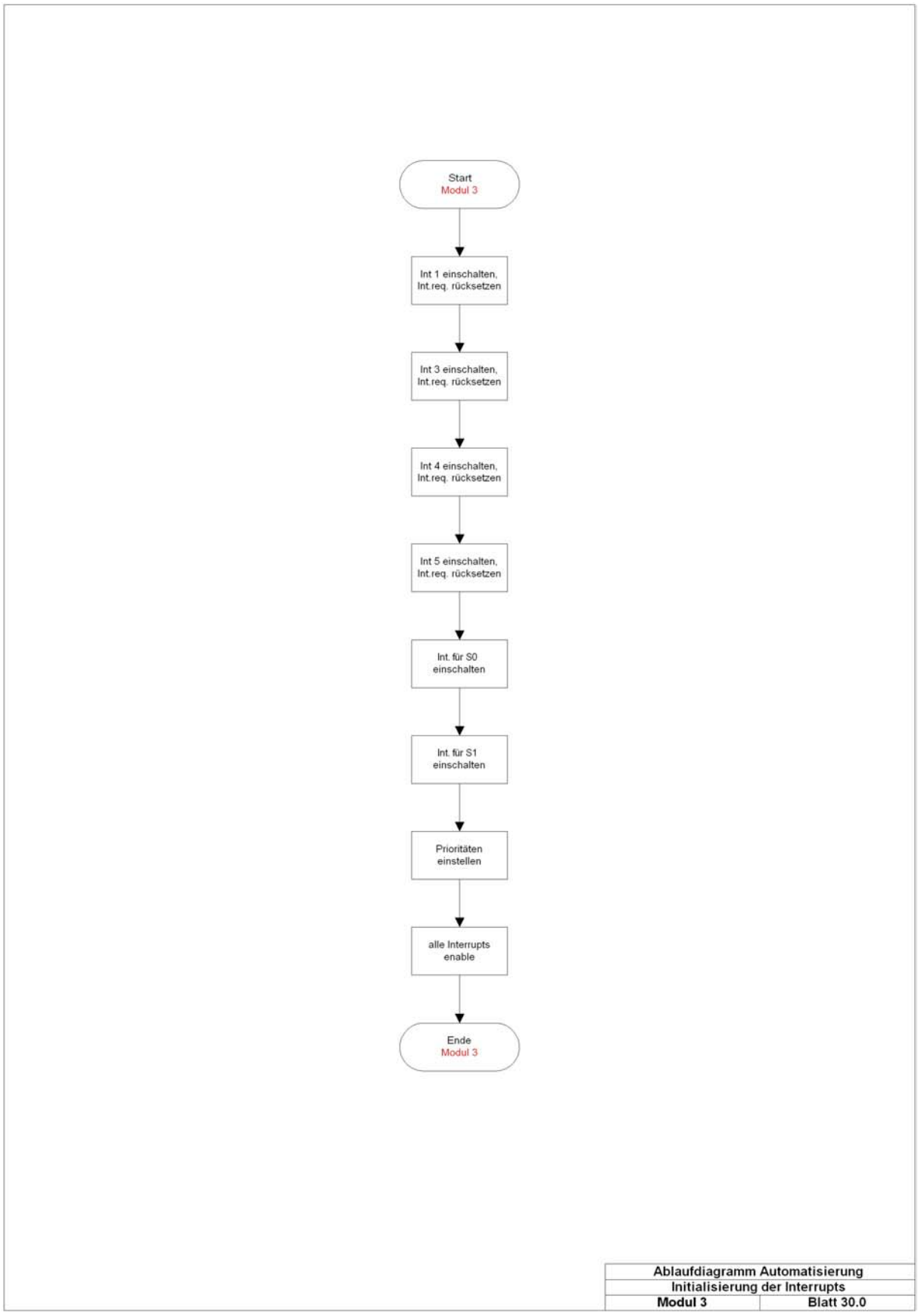
(Abb. 9-8)


Start
Modul 1
Serielle Schnittstelle S0
Operation Mode der Schnittstelle auf 8-bit UART (SM0, SM1)
Baudraten-generator anschalten (BD)
Baudraten Postdoubler ausschalten (SMOD)
Baudraten Vorteiler einschalten (S0P)
10bit automatic Reload timer laden auf 9600 Baud (S0RELL,S0RELH)
Ende
Modul 1
Start
Modul 1
Serielle Schnittstelle S1
Operation Mode der Schnittstelle auf 8-bit UART (SM)
Baudraten Vorteiler einschalten (S1P)
10bit automatic Reload timer laden auf 9600 Baud (S1RELL,S1RELH)
Ende
Modul 1
Ablaufdiagramm Automatisierung
serielle Schnittstellen S0 und S1
Modul 1
Blatt 10.0

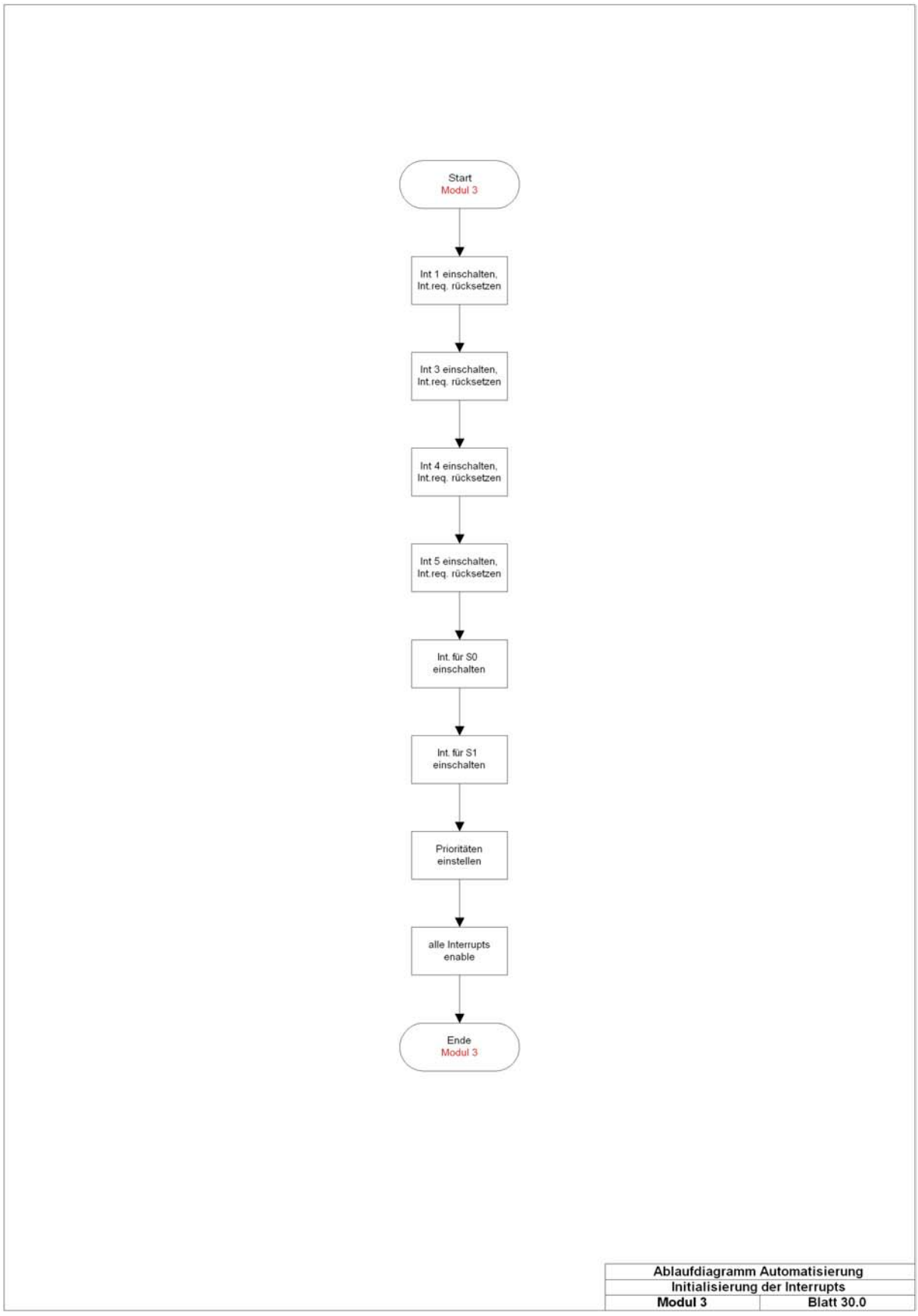
(Abb. 9-9)


Start
Modul 2
Ausgänge Motoren auf LOW
disable Multiplexer, disable Leistung Motoren
Eingänge Endlagenschalter vorbereiten
Eingänge Ref.-punktschalter vorbereiten
Eingänge Neigungssensor, Deckelschalter vorbereiten
Ausgänge Test - LEDs auf LOW
Ende
Modul 2
Ablaufdiagramm Automatisierung
Initialisierung der Ports
Modul 2
Blatt 20.0

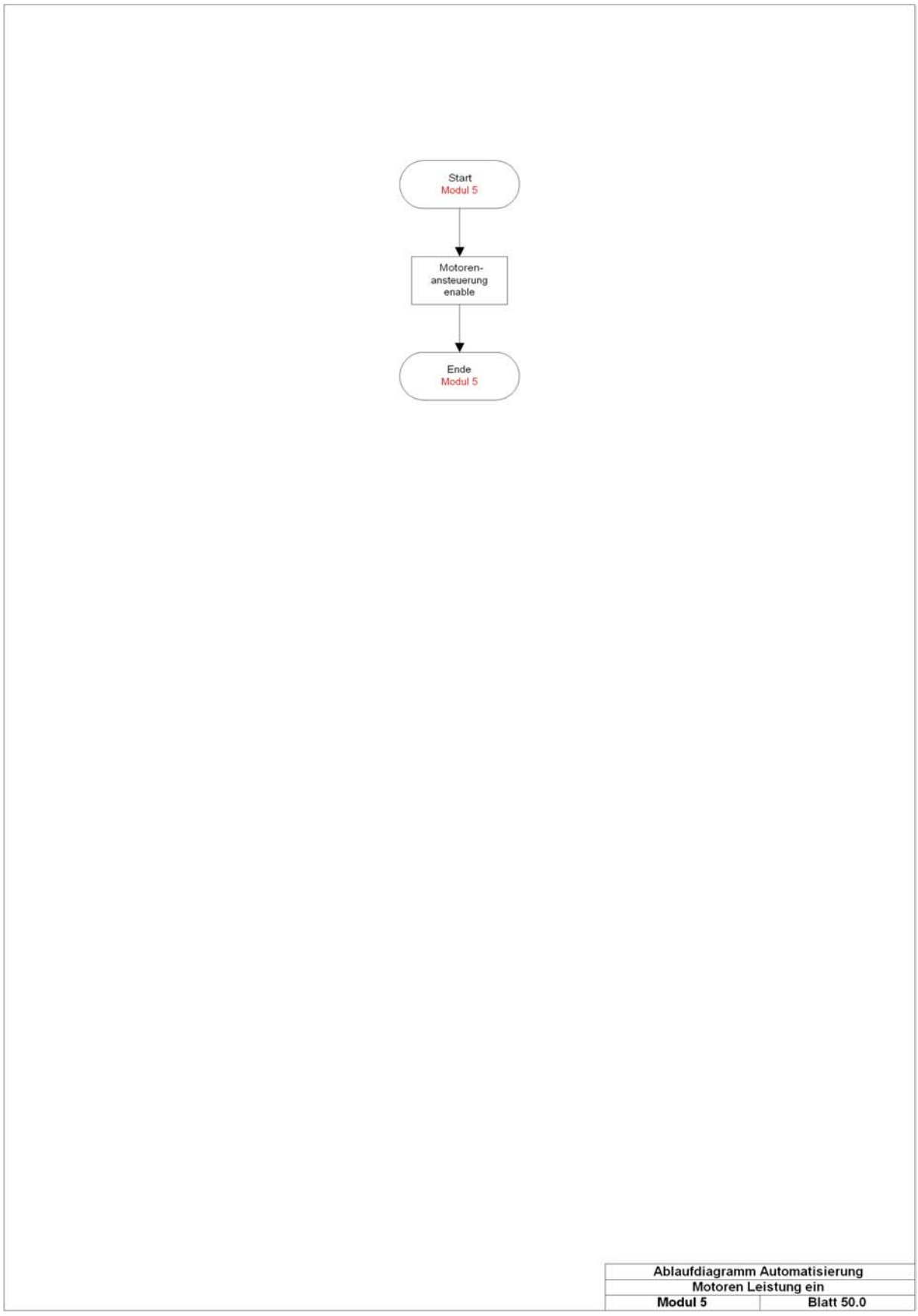
(Abb. 9-10)


Start
Modul 3
Int 1 einschalten, Int.req. rücksetzen
Int 3 einschalten, Int.req. rücksetzen
Int 4 einschalten, Int.req. rücksetzen
Int 5 einschalten, Int.req. rücksetzen
Int. für S0 einschalten
Int. für S1 einschalten
Prioritäten einstellen
alle Interrupts enable
Ende
Modul 3
Ablaufdiagramm Automatisierung
Initialisierung der Interrupts
Modul 3
Blatt 30.0

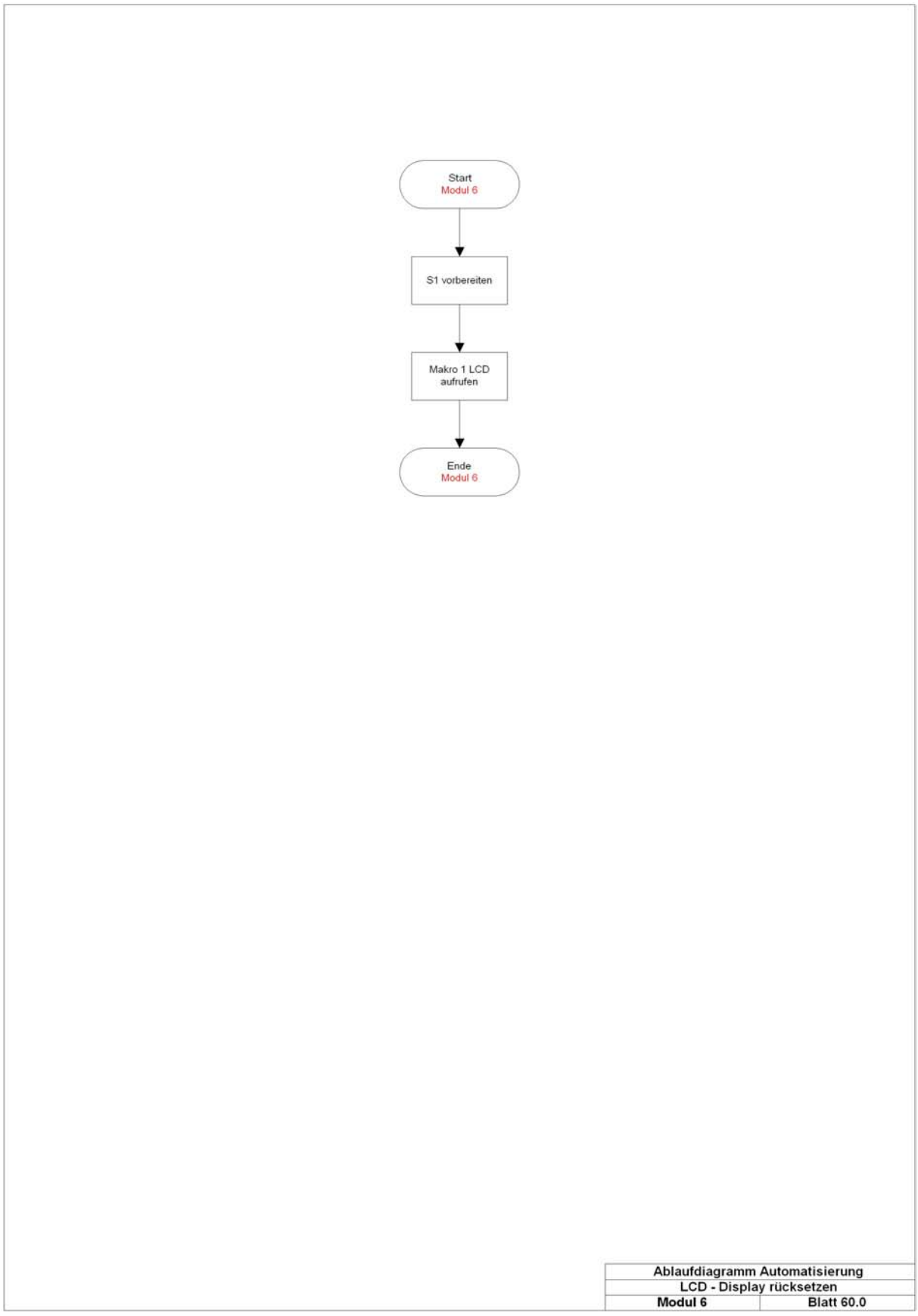
(Abb. 9-11)


Start
Modul 3
Int 1 einschalten, Int.req. rücksetzen
Int 3 einschalten, Int.req. rücksetzen
Int 4 einschalten, Int.req. rücksetzen
Int 5 einschalten, Int.req. rücksetzen
Int. für S0 einschalten
Int. für S1 einschalten
Prioritäten einstellen
alle Interrupts enable
Ende
Modul 3
Ablaufdiagramm Automatisierung
Initialisierung der Interrupts
Modul 3
Blatt 30.0

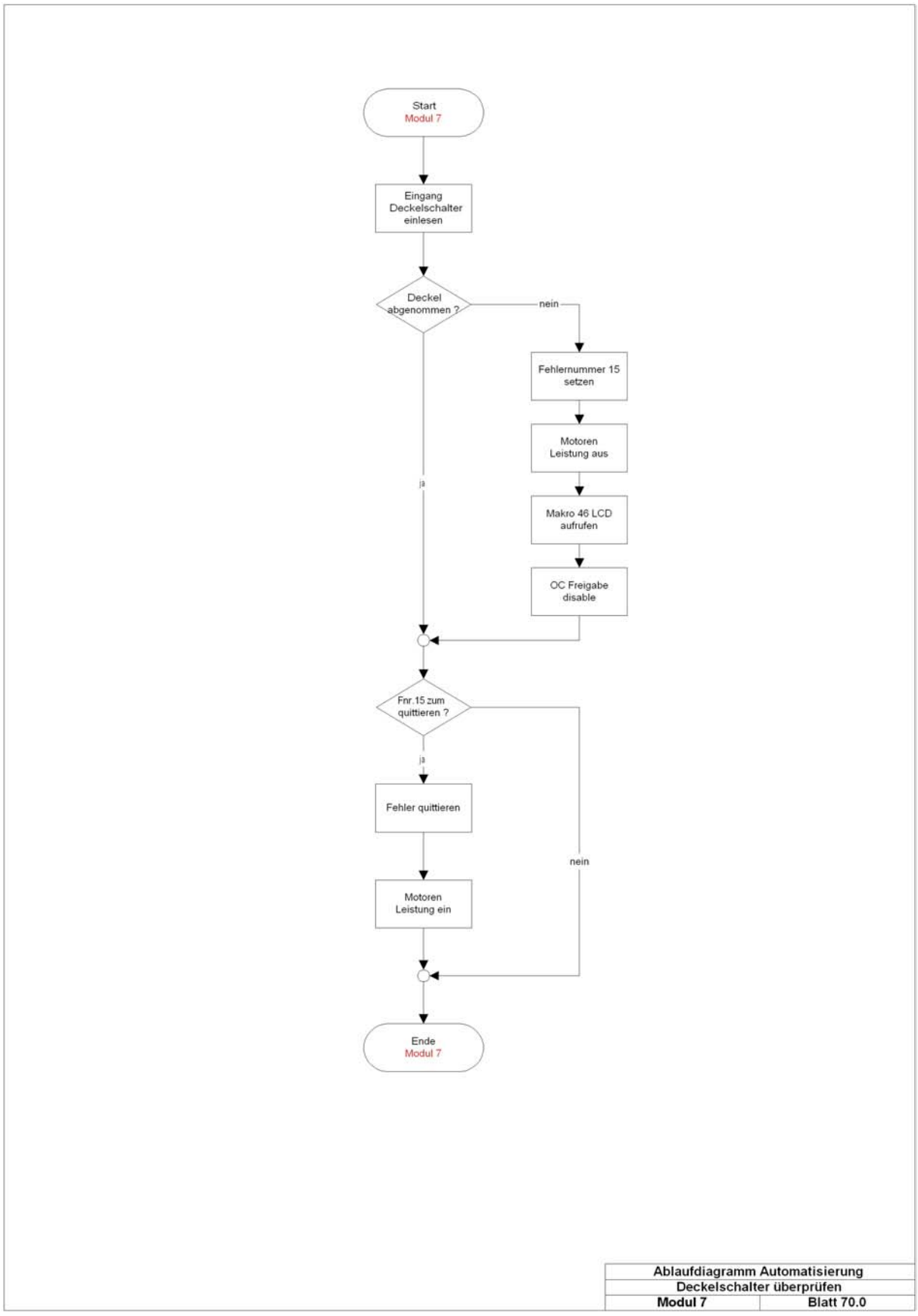
(Abb. 9-12)


Start
Modul 5
Motoren-
ansteuerung
enable
Ende
Modul 5
Ablaufdiagramm Automatisierung
Motoren Leistung ein
Modul 5
Blatt 50.0

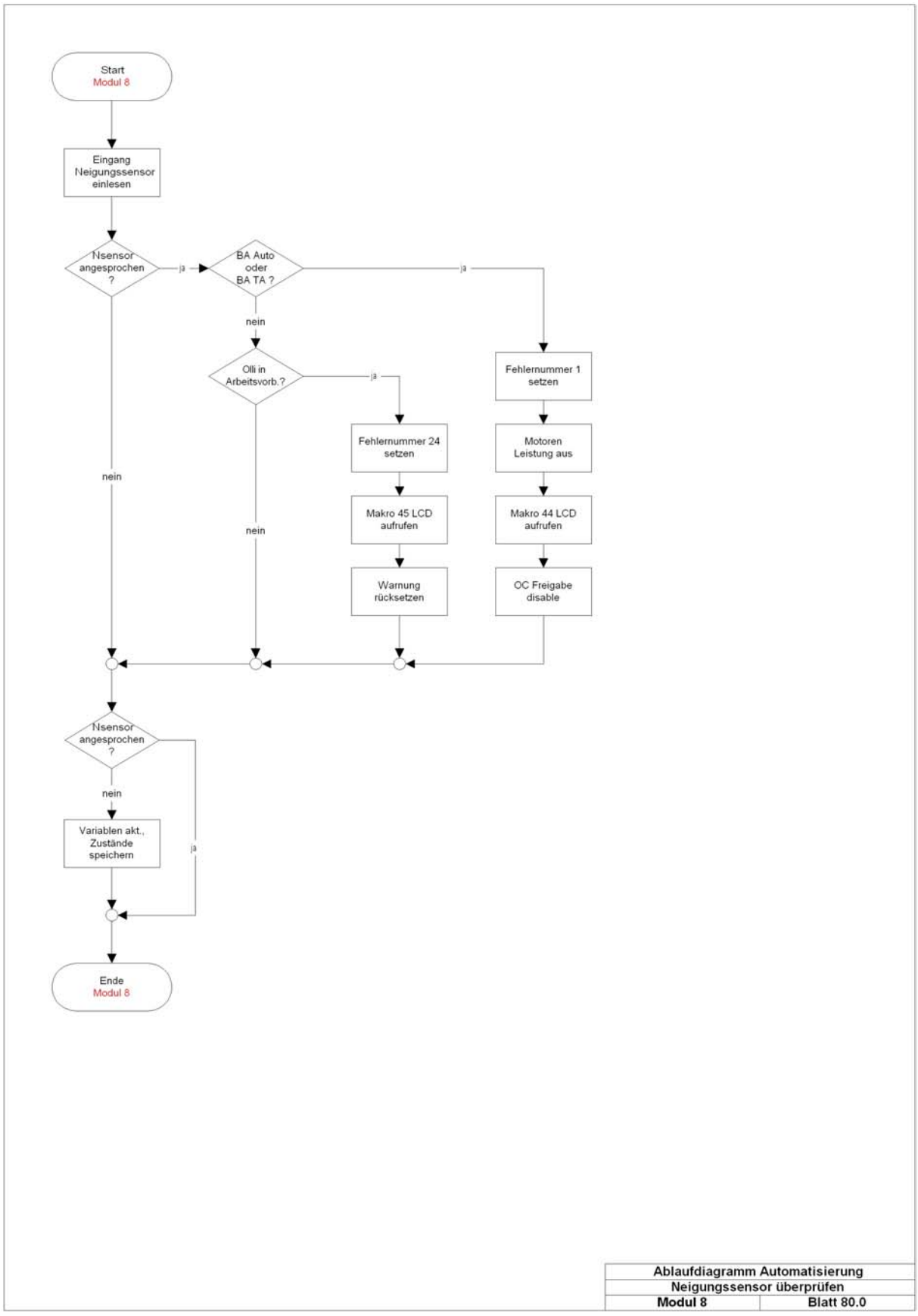
(Abb. 9-13)


Start
Modul 6
S1 vorbereiten
Makro 1 LCD aufrufen
Ende
Modul 6
Ablaufdiagramm Automatisierung
LCD - Display rücksetzen
Modul 6
Blatt 60.0

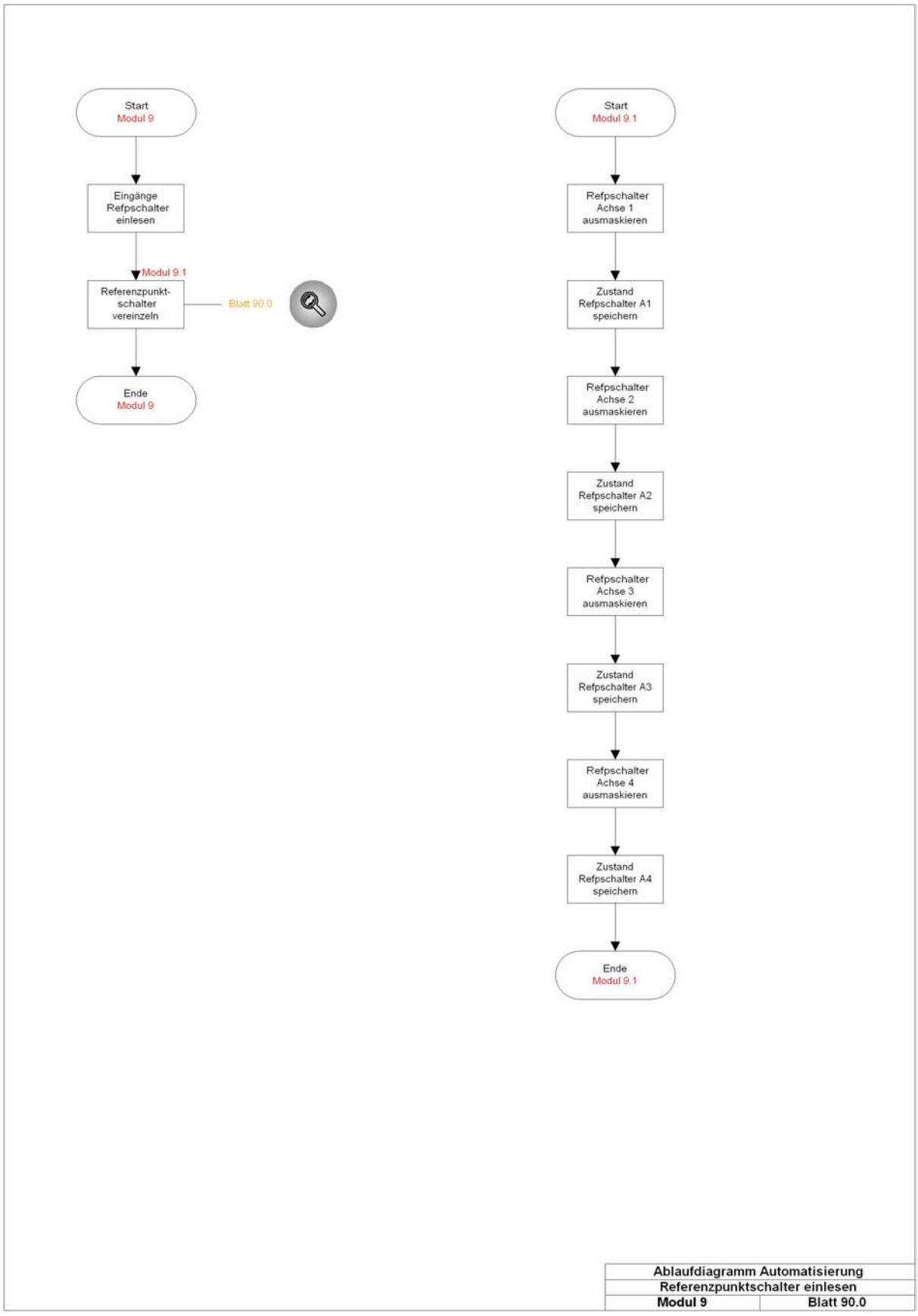
(Abb. 9-14)


Start
Modul 7
Eingang Deckelschalter einlesen
Deckel abgenommen ?
nein
ja
Fehlernummer 15 setzen
Motoren Leistung aus
Makro 46 LCD aufrufen
OC Freigabe disable
Fnr.15 zum quittieren ?
ja
nein
Fehler quittieren
Motoren Leistung ein
Ende
Modul 7
Ablaufdiagramm Automatisierung
Deckelschalter überprüfen
Modul 7
Blatt 70.0

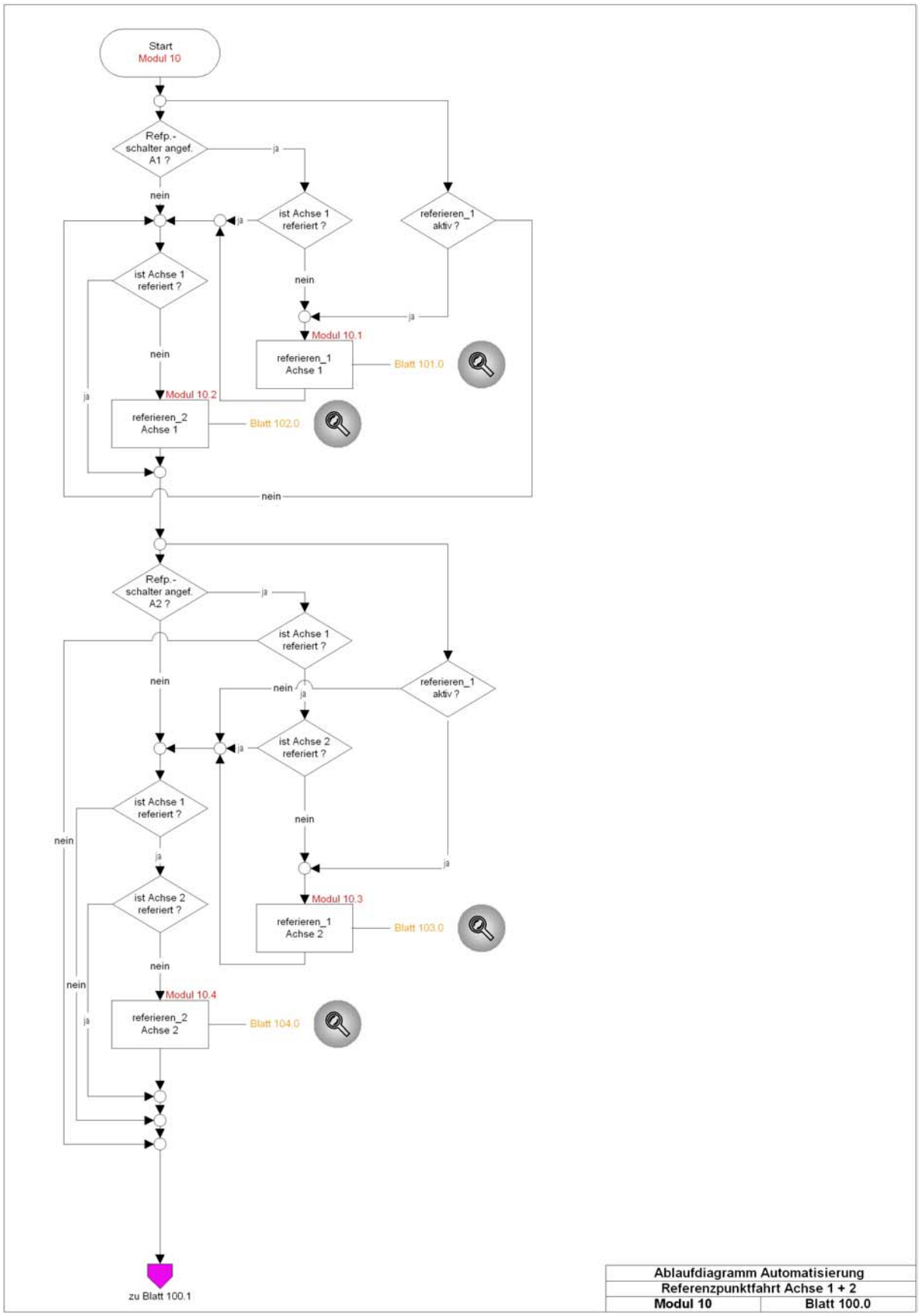
(Abb. 9-15)


Start
Modul 8
Eingang Neigungssensor einlesen
Nsensor angesprochen ?
ja
BA Auto oder BA TA ?
ja
nein
Olli in Arbeitsvorb.?
ja
Fehlernummer 1 setzen
Fehlernummer 24 setzen
Motoren Leistung aus
Makro 45 LCD aufrufen
Makro 44 LCD aufrufen
Warnung rücksetzen
OC Freigabe disable
nein
nein
Nsensor angesprochen ?
nein
Variablen akt., Zustände speichern
ja
Ende
Modul 8
Ablaufdiagramm Automatisierung
Neigungssensor überprüfen
Modul 8
Blatt 80.0

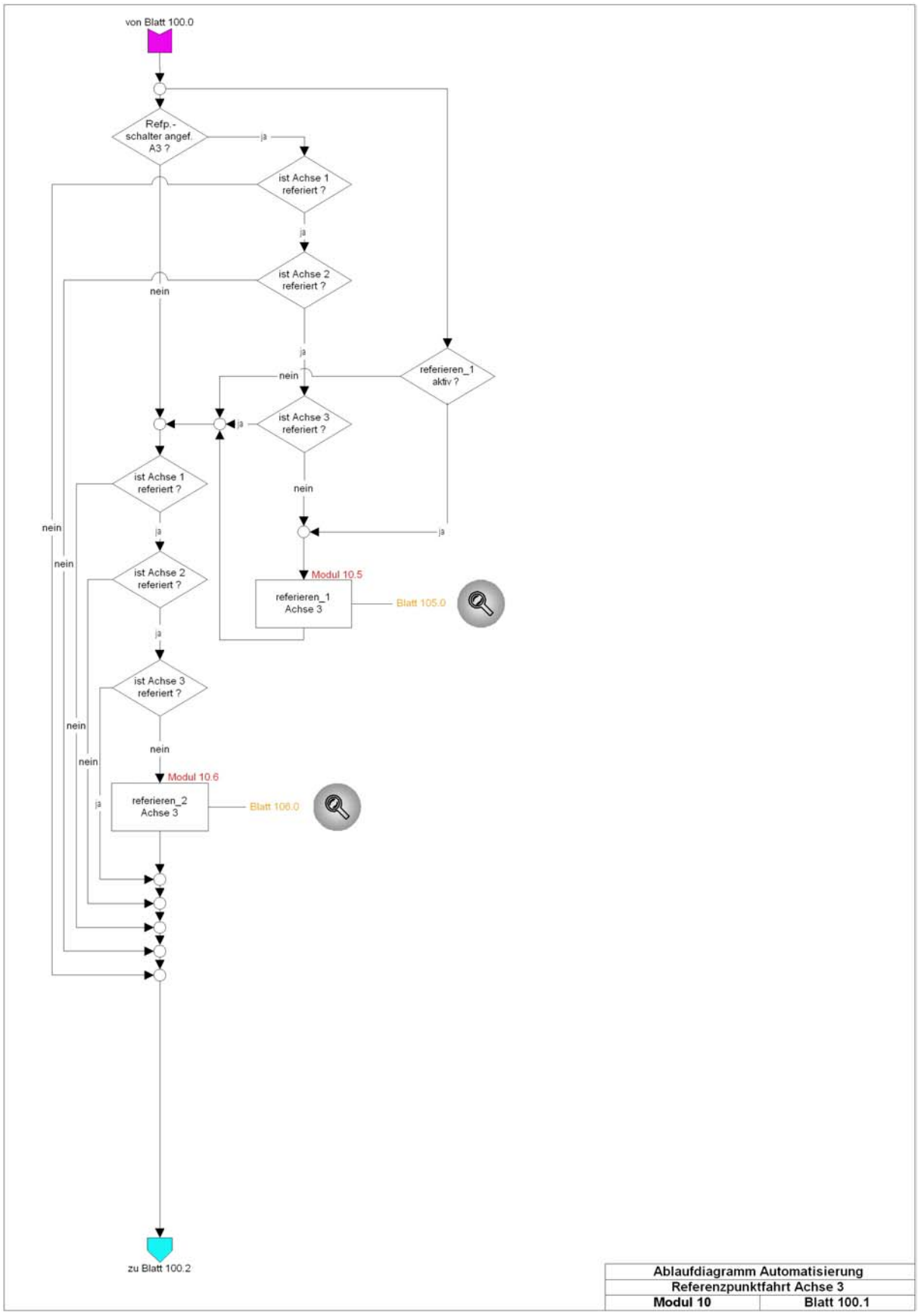
(Abb. 9-16)


Start
Modul 9
Eingänge Refpschalter einlesen
Modul 9.1
Referenzpunkt-schalter vereinzeln
Blatt 90.0
Ende
Modul 9
Start
Modul 9.1
Refpschalter Achse 1 ausmaskieren
Zustand Refpschalter A1 speichern
Refpschalter Achse 2 ausmaskieren
Zustand Refpschalter A2 speichern
Refpschalter Achse 3 ausmaskieren
Zustand Refpschalter A3 speichern
Refpschalter Achse 4 ausmaskieren
Zustand Refpschalter A4 speichern
Ende
Modul 9.1
Ablaufdiagramm Automatisierung
Referenzpunktschalter einlesen
Modul 9
Blatt 90.0

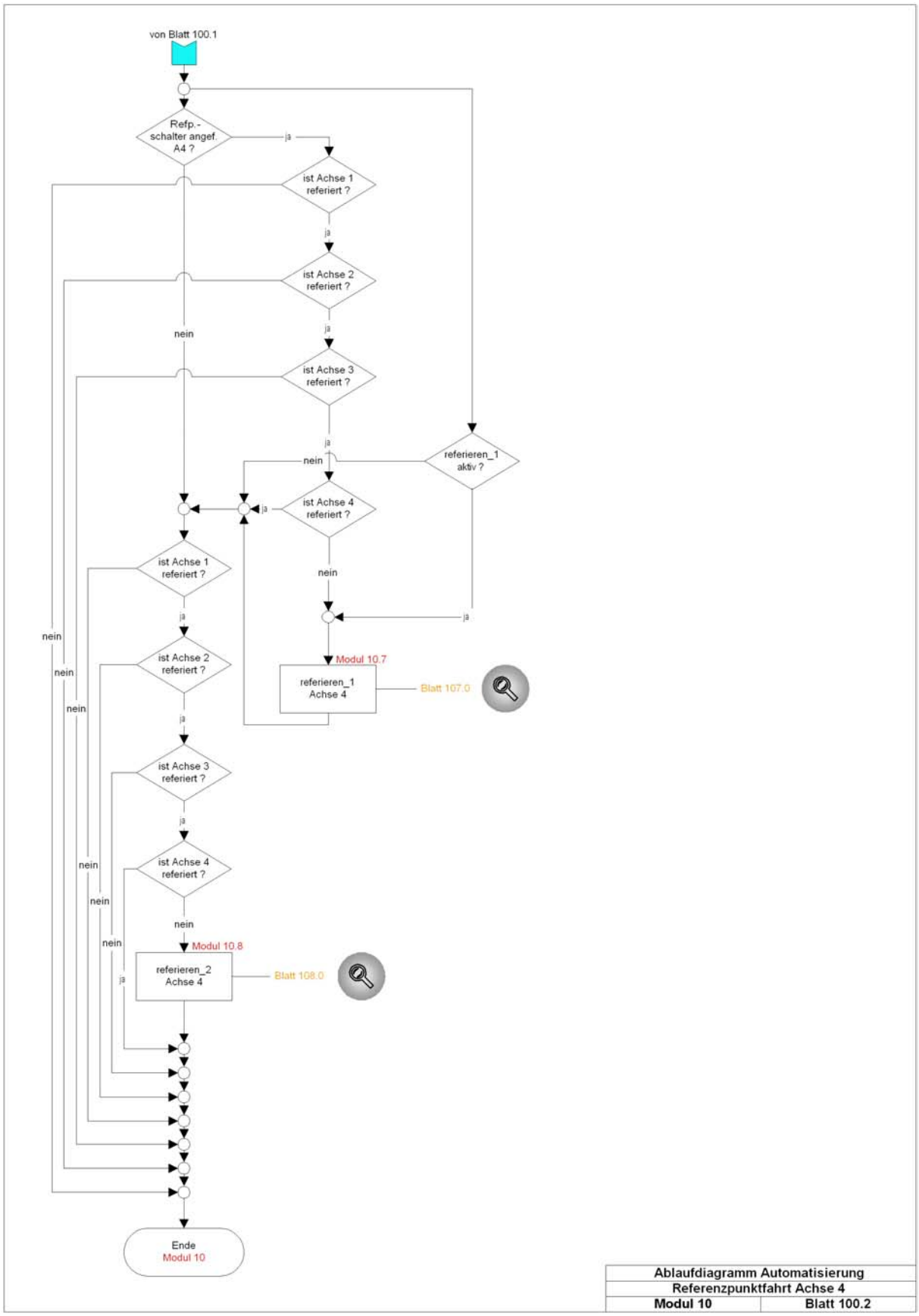
(Abb. 9-17)


Start
Modul 10
Refp.-schalter angef. A1 ?
ja
nein
ist Achse 1 referiert ?
referieren_1 aktiv ?
Modul 10.1
referieren_1 Achse 1
Blatt 101.0
Modul 10.2
referieren_2 Achse 1
Blatt 102.0
Refp.-schalter angef. A2 ?
ist Achse 2 referiert ?
Modul 10.3
referieren_1 Achse 2
Blatt 103.0
Modul 10.4
referieren_2 Achse 2
Blatt 104.0
zu Blatt 100.1
Ablaufdiagramm Automatisierung
Referenzpunktfahrt Achse 1 + 2
Modul 10
Blatt 100.0

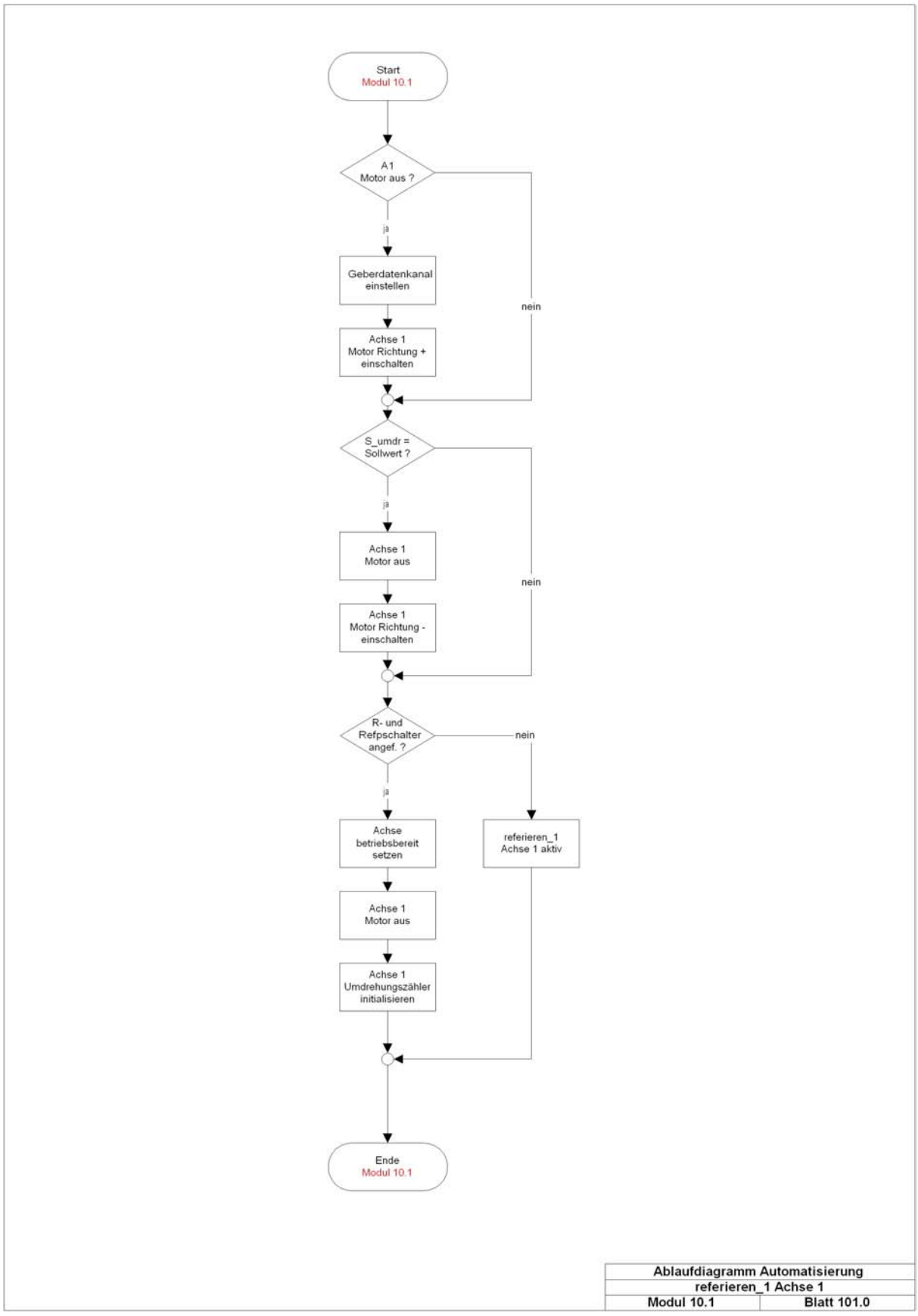
(Abb. 9-18)


von Blatt 100.0
Refp.-schalter angef. A3 ?
ja
ist Achse 1 referiert ?
ja
ist Achse 2 referiert ?
nein
ja
nein
referieren_1 aktiv ?
ja
ist Achse 3 referiert ?
ja
nein
ist Achse 1 referiert ?
nein
ja
nein
ja
ist Achse 2 referiert ?
Modul 10.5
referieren_1 Achse 3
Blatt 105.0
ja
ist Achse 3 referiert ?
nein
nein
nein
Modul 10.6
ja
referieren_2 Achse 3
Blatt 106.0
zu Blatt 100.2
Ablaufdiagramm Automatisierung
Referenzpunktfahrt Achse 3
Modul 10
Blatt 100.1

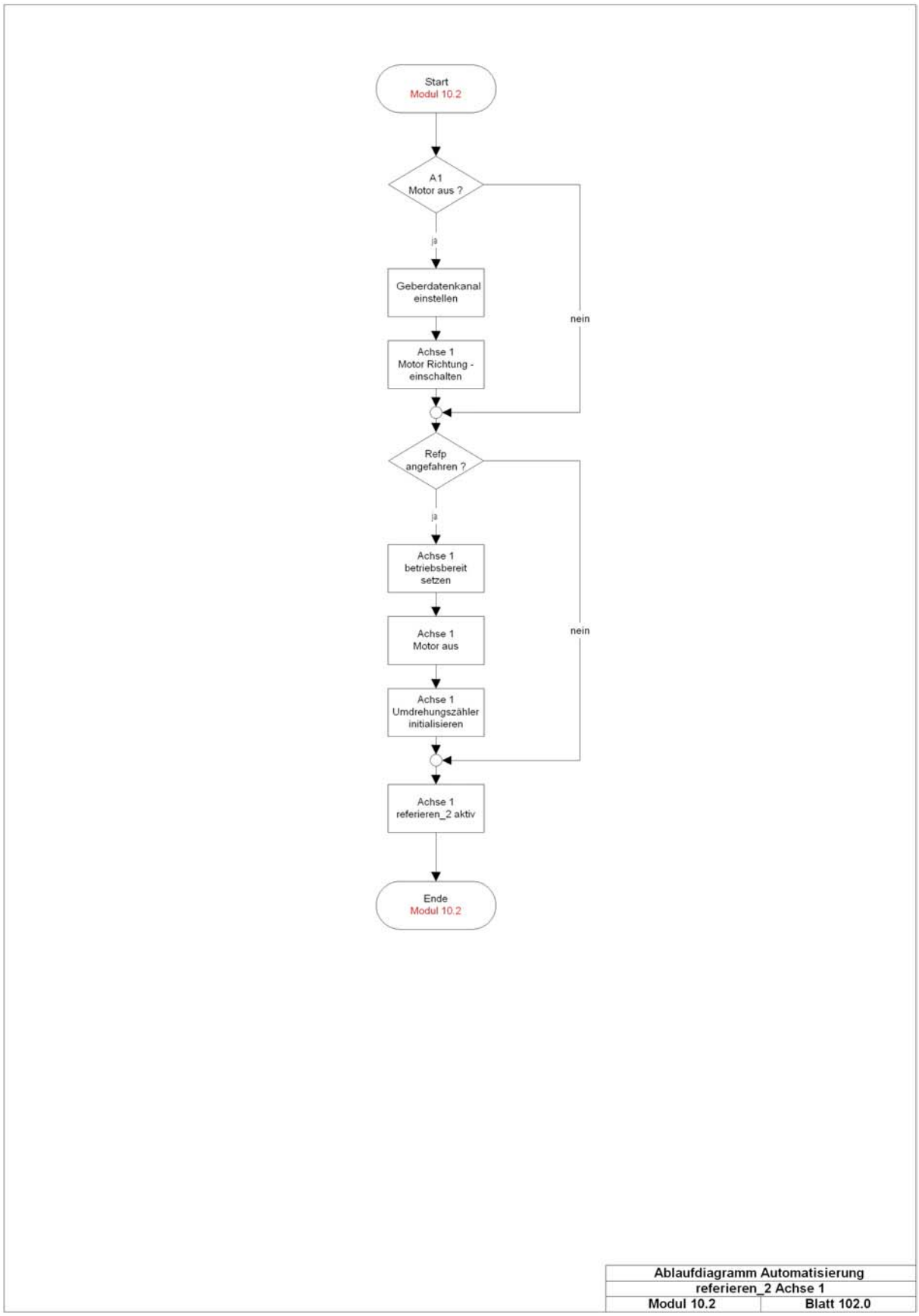
(Abb. 9-19)


von Blatt 100.1
Refp.-schalter angef. A4 ?
ja
ist Achse 1 referiert ?
ja
ist Achse 2 referiert ?
ja
ist Achse 3 referiert ?
ja
nein
referieren_1 aktiv ?
ja
ist Achse 4 referiert ?
nein
ja
ist Achse 1 referiert ?
ja
nein
ist Achse 2 referiert ?
Modul 10.7
referieren_1 Achse 4
Blatt 107.0
ja
nein
ist Achse 3 referiert ?
ja
nein
ist Achse 4 referiert ?
nein
Modul 10.8
referieren_2 Achse 4
Blatt 108.0
ja
Ende
Modul 10
Ablaufdiagramm Automatisierung
Referenzpunktfahrt Achse 4
Modul 10
Blatt 100.2

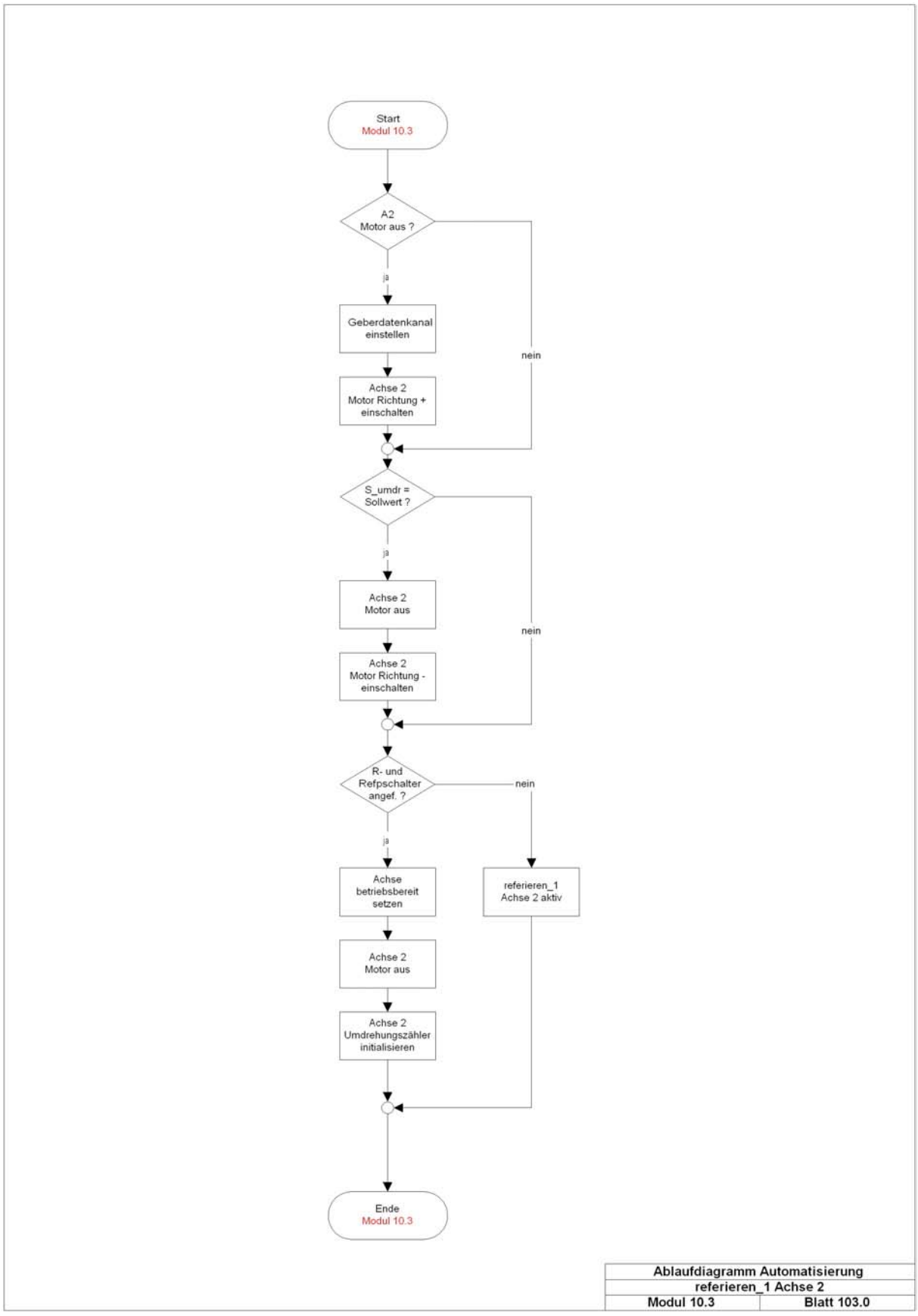
(Abb. 9-20)


Start
Modul 10.1
A1
Motor aus ?
ja
nein
Geberdatenkanal einstellen
Achse 1
Motor Richtung + einschalten
S_umdr =
Sollwert ?
ja
nein
Achse 1
Motor aus
Achse 1
Motor Richtung - einschalten
R- und
Refpschalter
angef. ?
nein
ja
Achse
betriebsbereit
setzen
referieren_1
Achse 1 aktiv
Achse 1
Motor aus
Achse 1
Umdrehungszähler initialisieren
Ende
Modul 10.1
Ablaufdiagramm Automatisierung
referieren_1 Achse 1
Modul 10.1
Blatt 101.0

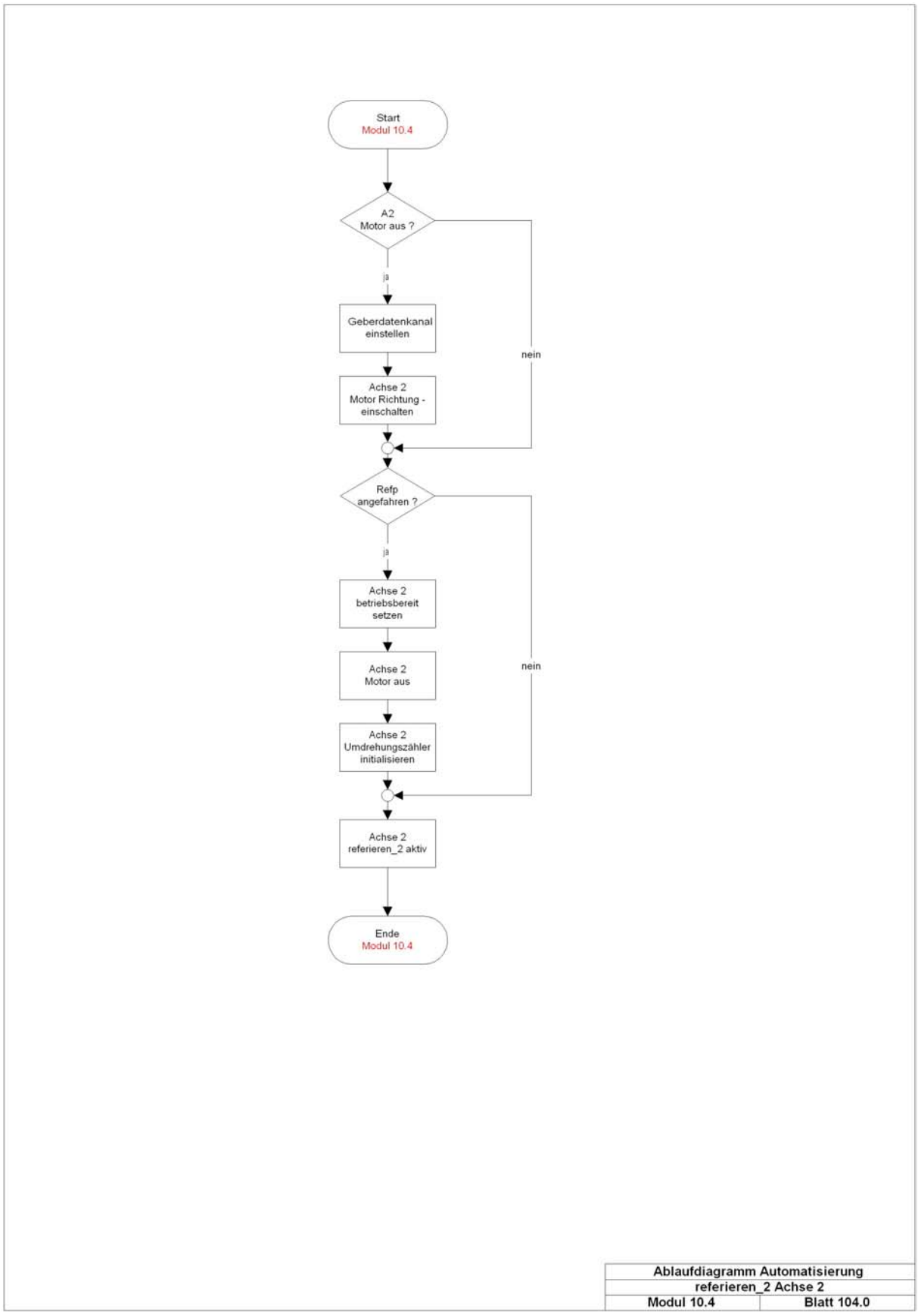
(Abb. 9-21)


Start
Modul 10.2
A1
Motor aus ?
ja
nein
Geberdatenkanal einstellen
Achse 1
Motor Richtung - einschalten
Refp
angefahren ?
ja
nein
Achse 1
betriebsbereit setzen
Achse 1
Motor aus
Achse 1
Umdrehungszähler initialisieren
Achse 1
referieren_2 aktiv
Ende
Modul 10.2
Ablaufdiagramm Automatisierung
referieren_2 Achse 1
Modul 10.2
Blatt 102.0

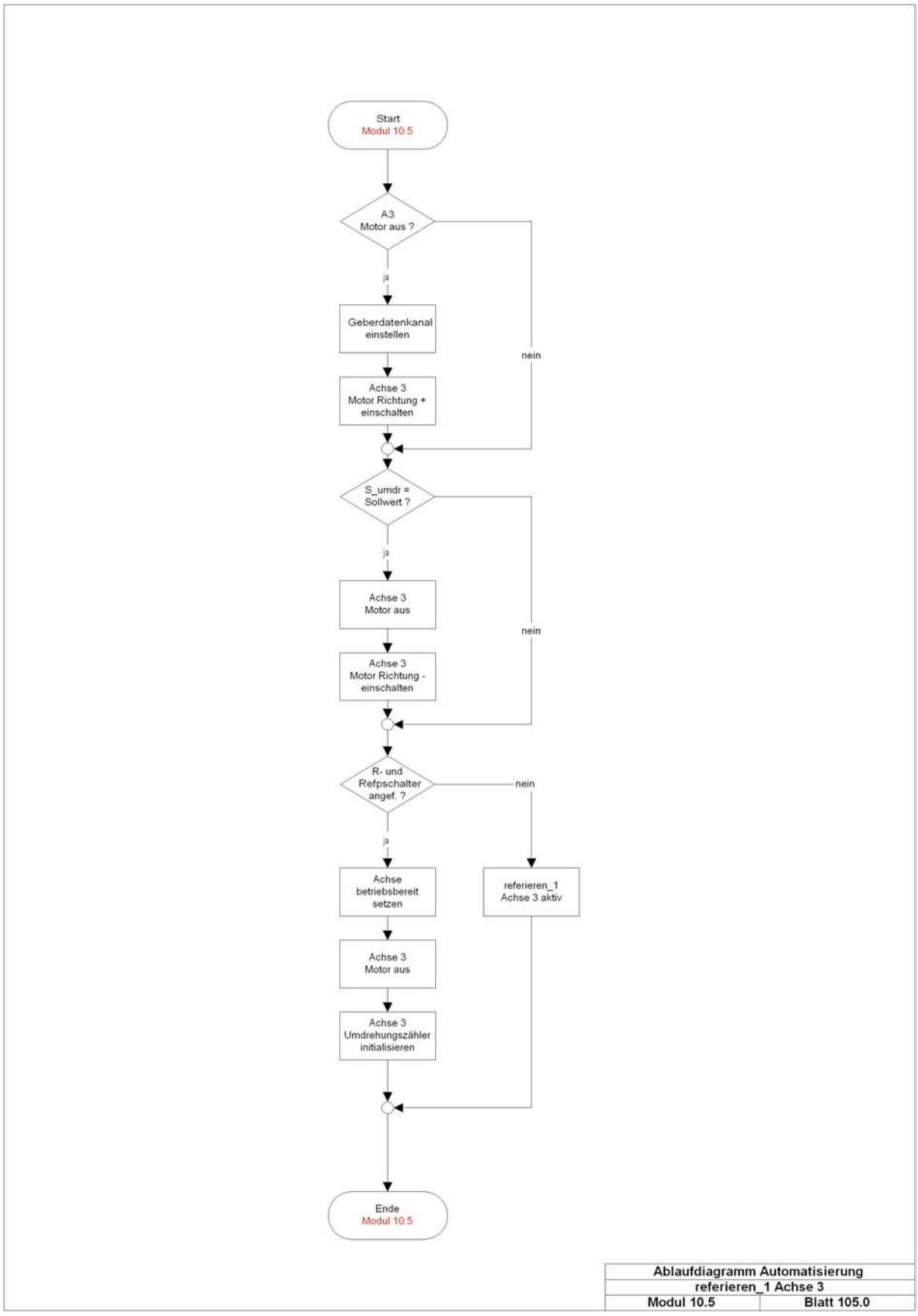
(Abb. 9-22)


Start
Modul 10.3
A2
Motor aus ?
ja
nein
Geberdatenkanal einstellen
Achse 2
Motor Richtung + einschalten
S_umdr =
Sollwert ?
ja
nein
Achse 2
Motor aus
Achse 2
Motor Richtung - einschalten
R- und
Refpschalter
angef. ?
nein
ja
Achse
betriebsbereit
setzen
referieren_1
Achse 2 aktiv
Achse 2
Motor aus
Achse 2
Umdrehungszähler
initialisieren
Ende
Modul 10.3
Ablaufdiagramm Automatisierung
referieren_1 Achse 2
Modul 10.3
Blatt 103.0

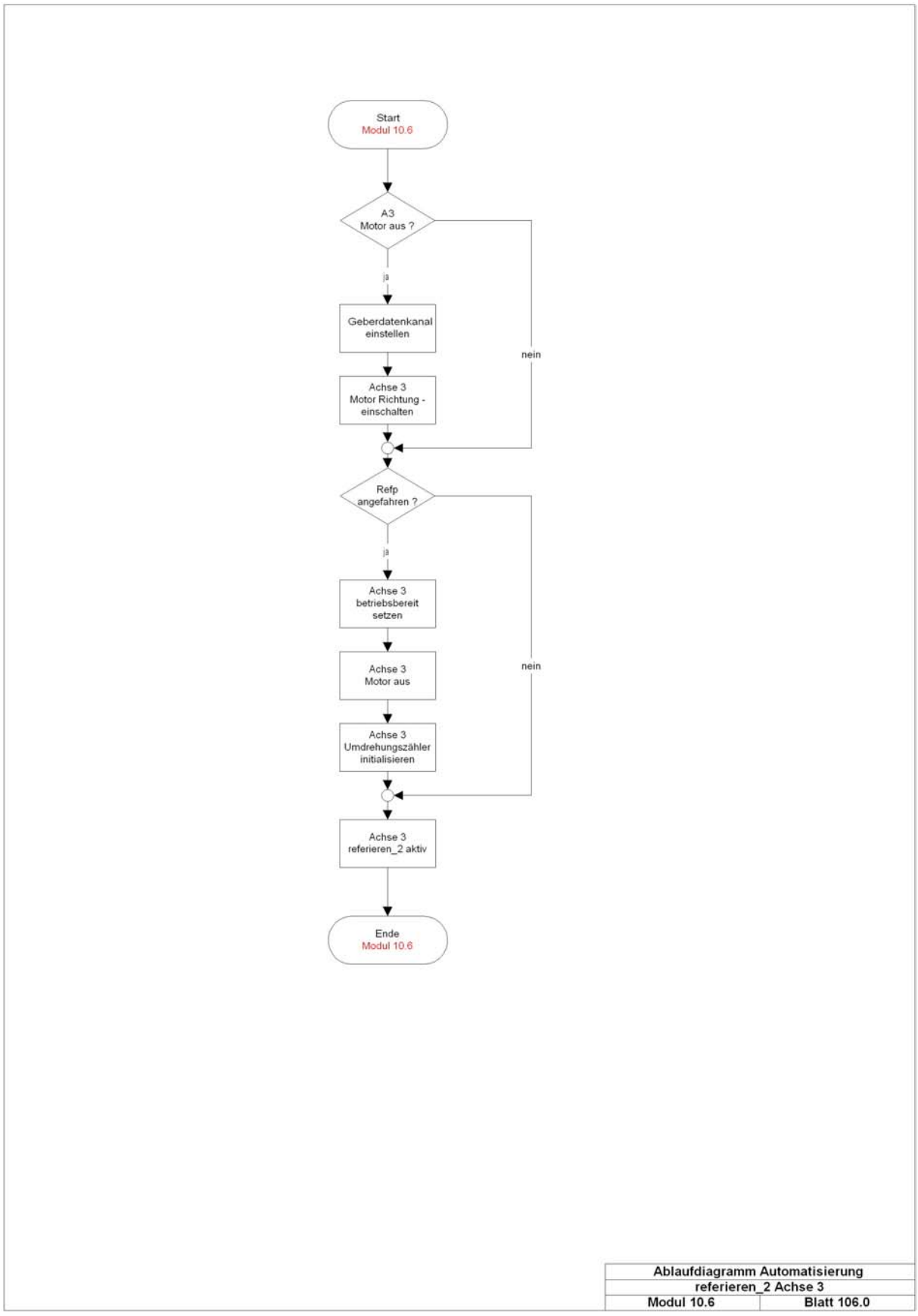
(Abb. 9-23)


Start
Modul 10.4
A2
Motor aus ?
ja
nein
Geberdatenkanal
einstellen
Achse 2
Motor Richtung -
einschalten
Refp
angefahren ?
ja
nein
Achse 2
betriebsbereit
setzen
Achse 2
Motor aus
Achse 2
Umdrehungszähler
initialisieren
Achse 2
referieren_2 aktiv
Ende
Modul 10.4
Ablaufdiagramm Automatisierung
referieren_2 Achse 2
Modul 10.4
Blatt 104.0

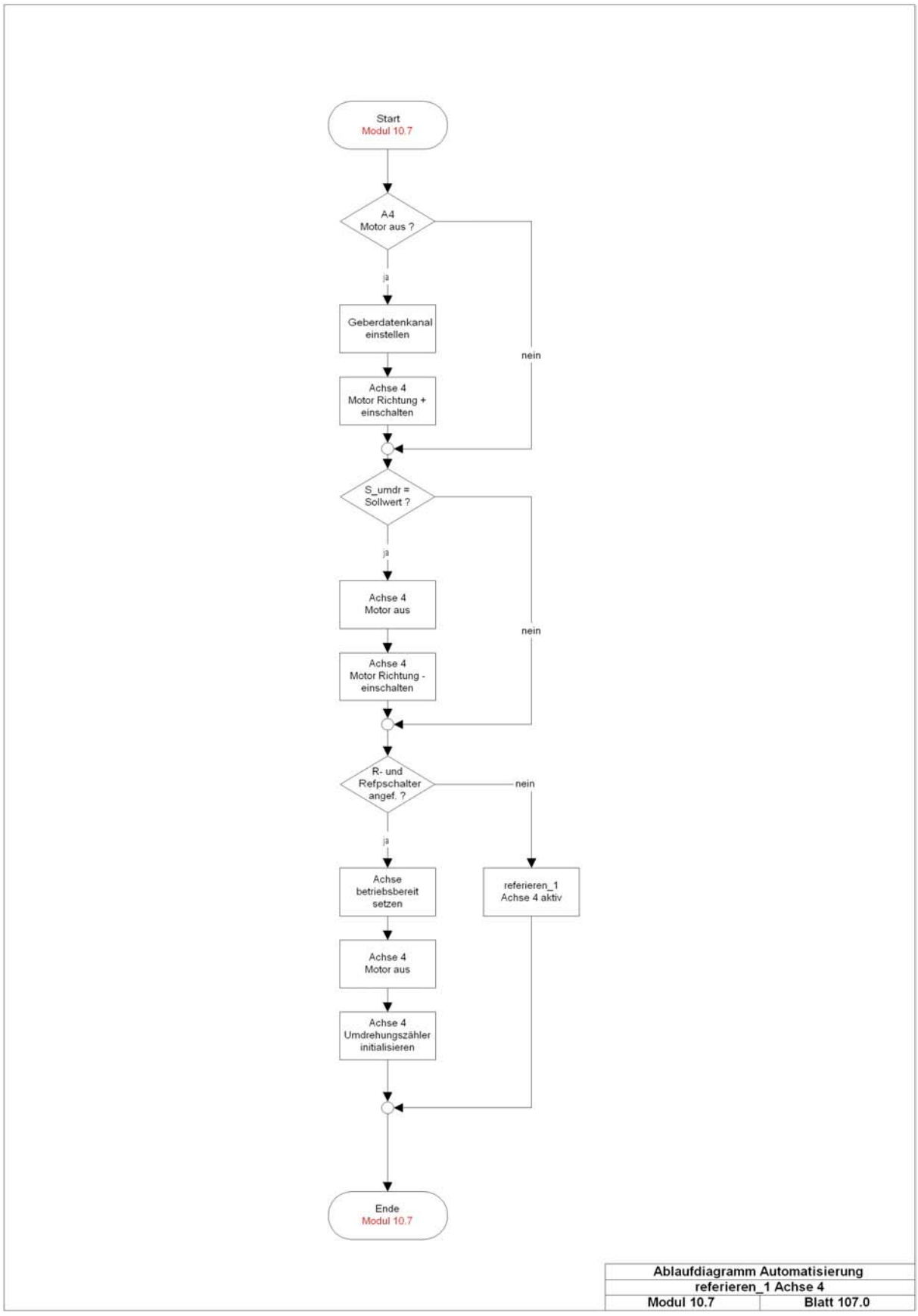
(Abb. 9-24)


Start
Modul 10.5
A3
Motor aus ?
ja
nein
Geberdatenkanal einstellen
Achse 3
Motor Richtung + einschalten
S_umdr =
Sollwert ?
ja
nein
Achse 3
Motor aus
Achse 3
Motor Richtung - einschalten
R- und
Refpschalter
angef. ?
nein
ja
Achse
betriebsbereit
setzen
referieren_1
Achse 3 aktiv
Achse 3
Motor aus
Achse 3
Umdrehungszähler
initialisieren
Ende
Modul 10.5
Ablaufdiagramm Automatisierung
referieren_1 Achse 3
Modul 10.5
Blatt 105.0

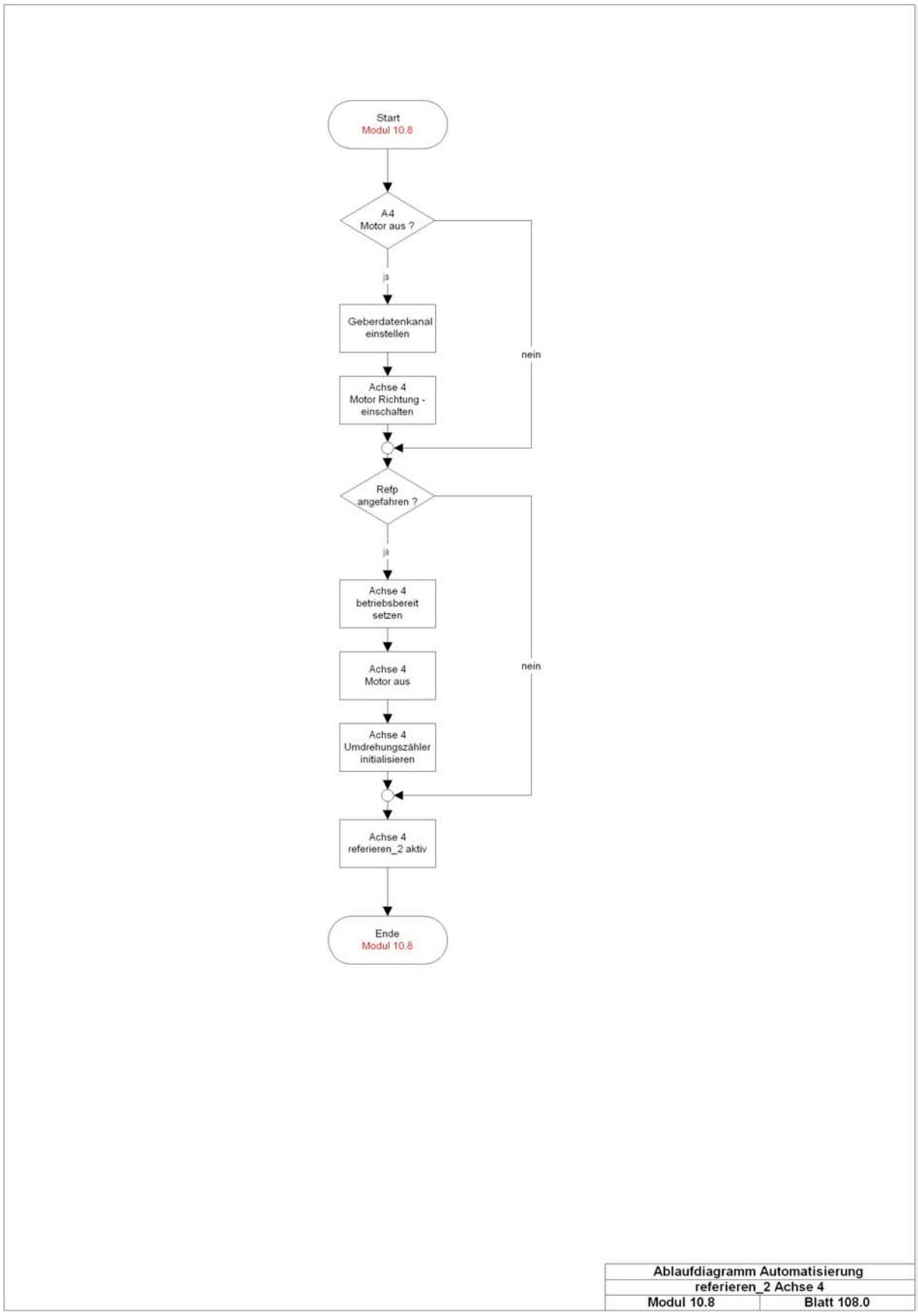
(Abb. 9-25)


Start
Modul 10.6
A3
Motor aus ?
ja
nein
Geberdatenkanal einstellen
Achse 3
Motor Richtung - einschalten
Refp
angefahren ?
ja
nein
Achse 3
betriebsbereit setzen
Achse 3
Motor aus
Achse 3
Umdrehungszähler initialisieren
Achse 3
referieren_2 aktiv
Ende
Modul 10.6
Ablaufdiagramm Automatisierung
referieren_2 Achse 3
Modul 10.6
Blatt 106.0

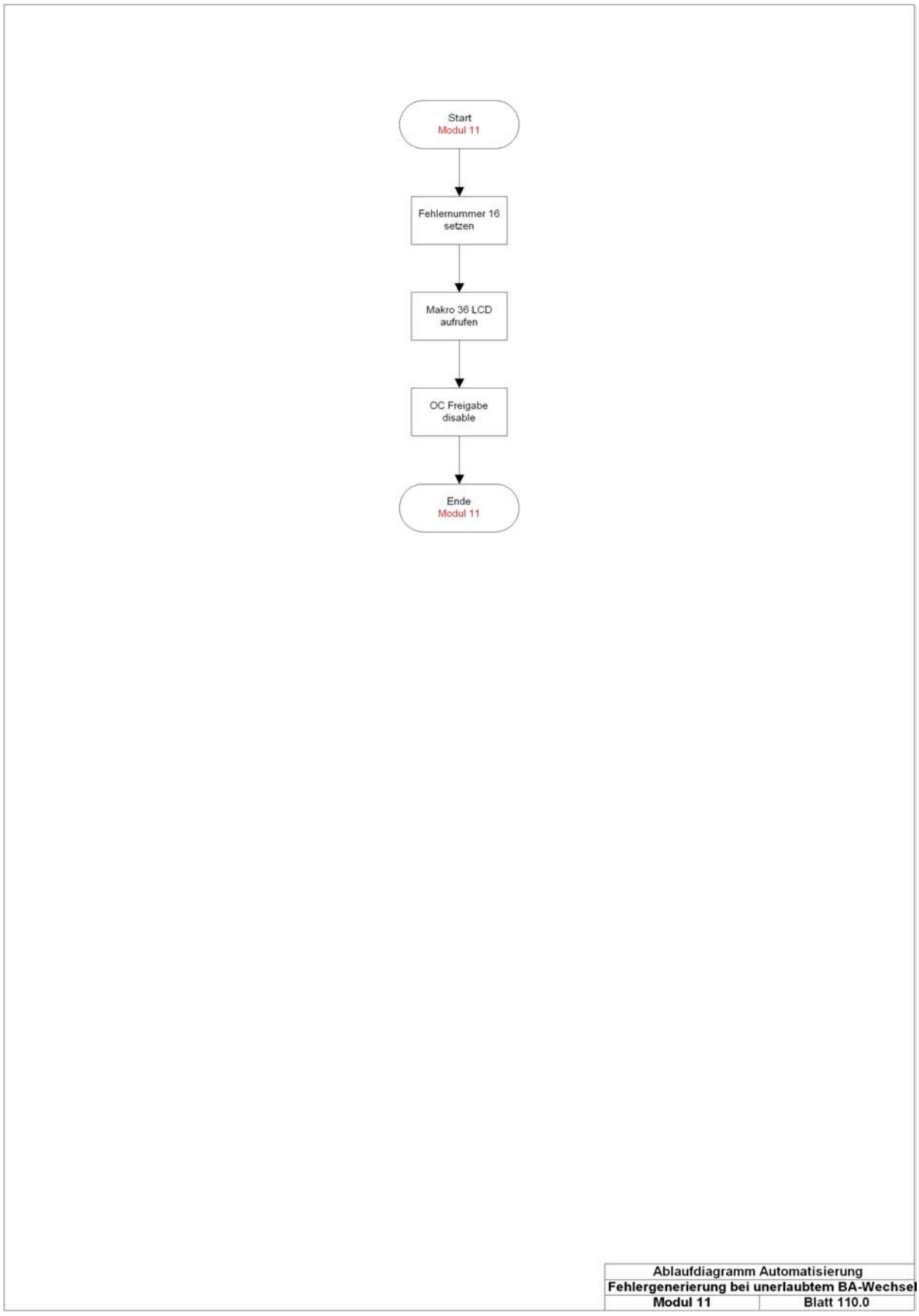
(Abb. 9-26)


Start
Modul 10.7
A4
Motor aus ?
ja
nein
Geberdatenkanal einstellen
Achse 4
Motor Richtung + einschalten
S_umdr = Sollwert ?
ja
nein
Achse 4
Motor aus
Achse 4
Motor Richtung - einschalten
R- und Refpschalter angef. ?
nein
ja
Achse betriebsbereit setzen
referieren_1 Achse 4 aktiv
Achse 4
Motor aus
Achse 4
Umdrehungszähler initialisieren
Ende
Modul 10.7
Ablaufdiagramm Automatisierung
referieren_1 Achse 4
Modul 10.7
Blatt 107.0

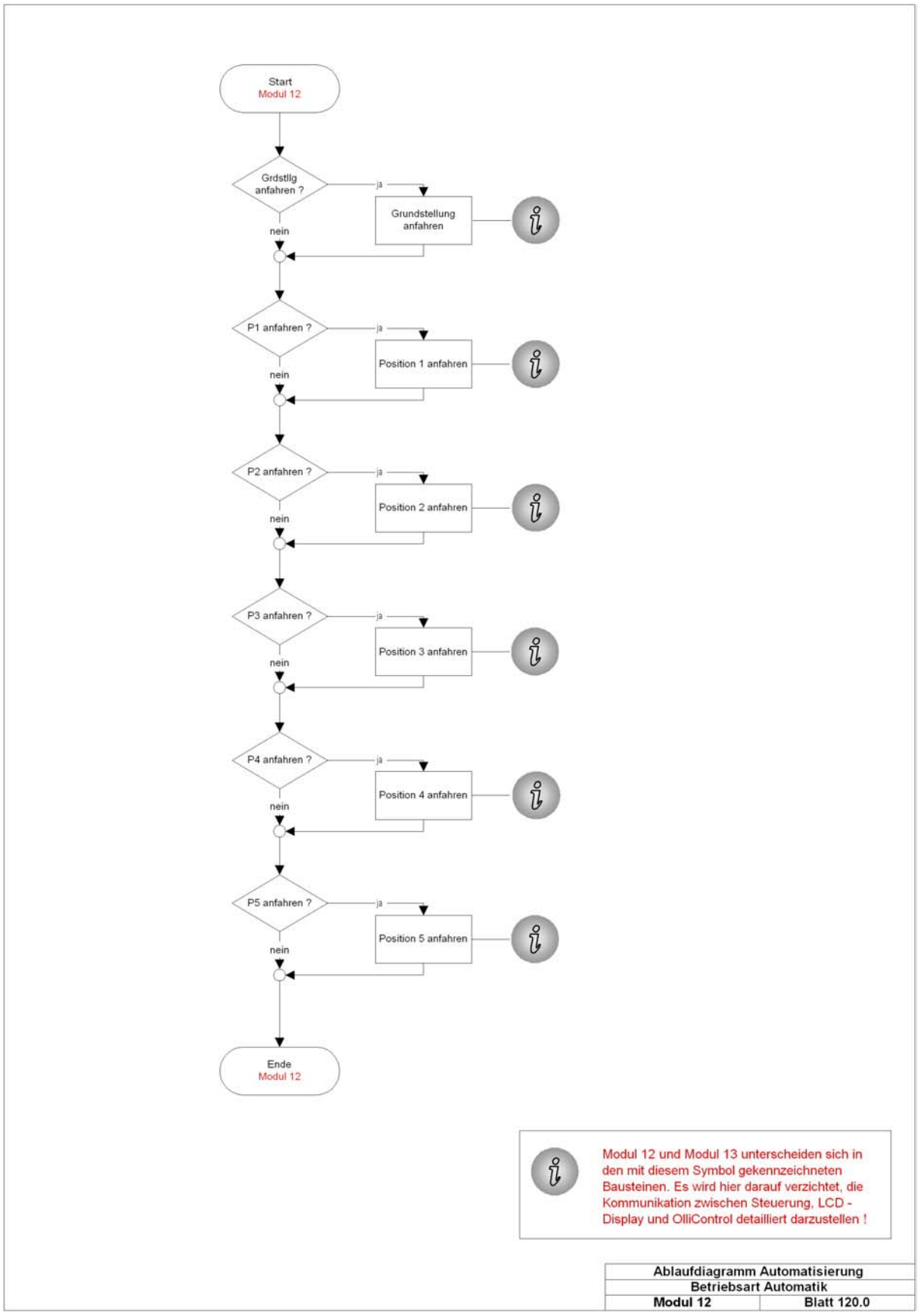
(Abb. 9-27)


Start
Modul 10.8
A4
Motor aus ?
ja
nein
Geberdatenkanal einstellen
Achse 4
Motor Richtung - einschalten
Refp
angefahren ?
ja
nein
Achse 4
betriebsbereit setzen
Achse 4
Motor aus
Achse 4
Umdrehungszähler initialisieren
Achse 4
referieren_2 aktiv
Ende
Modul 10.8
Ablaufdiagramm Automatisierung
referieren_2 Achse 4
Modul 10.8
Blatt 108.0

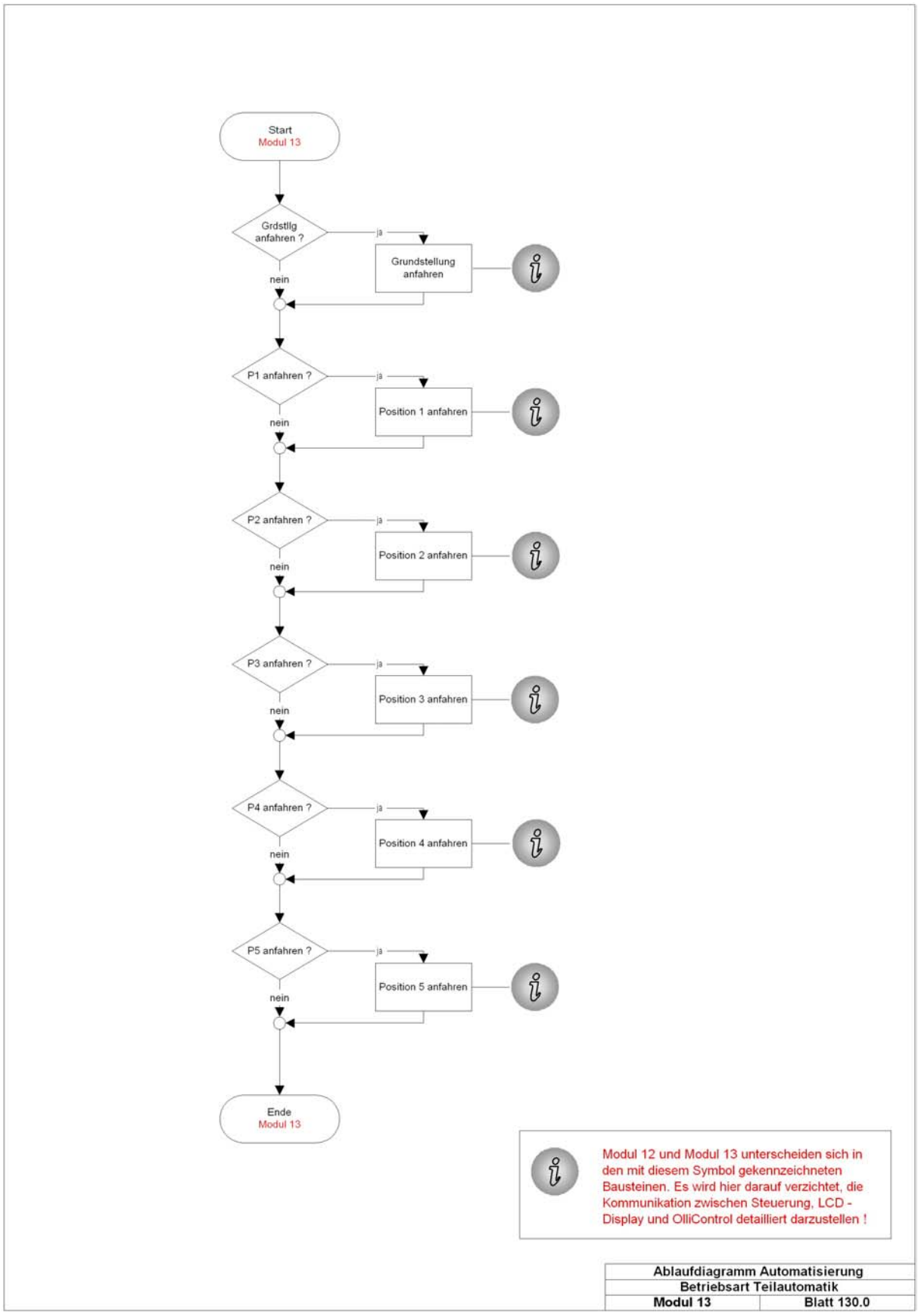
(Abb. 9-28)


Start
Modul 11
Fehlernummer 16 setzen
Makro 36 LCD aufrufen
OC Freigabe disable
Ende
Modul 11
Ablaufdiagramm Automatisierung
Fehlergenerierung bei unerlaubtem BA-Wechsel
Modul 11
Blatt 110.0

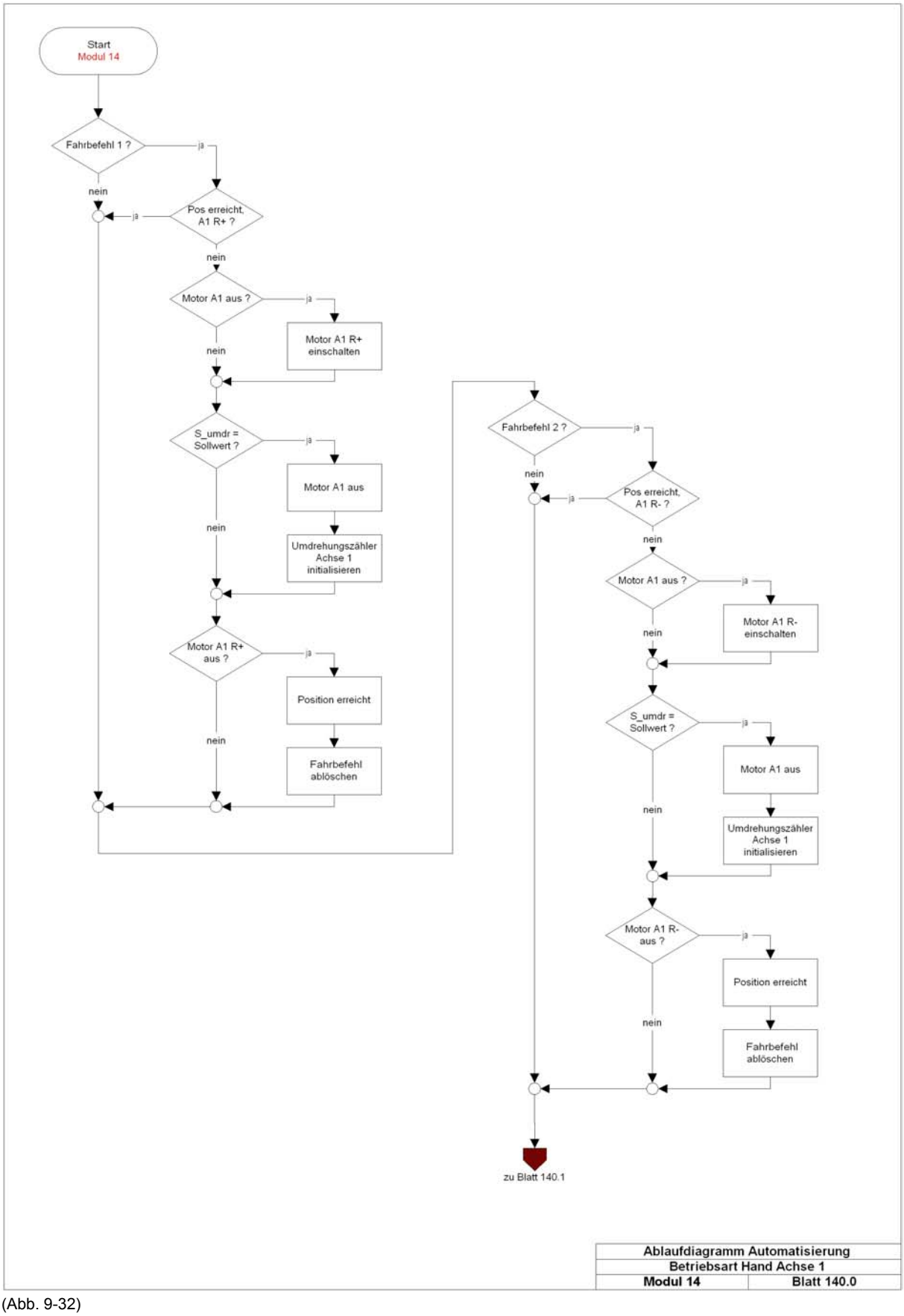
(Abb. 9-29)


Start
Modul 12
Grdstllg anfahren ?
ja
Grundstellung anfahren
nein
P1 anfahren ?
ja
Position 1 anfahren
nein
P2 anfahren ?
ja
Position 2 anfahren
nein
P3 anfahren ?
ja
Position 3 anfahren
nein
P4 anfahren ?
ja
Position 4 anfahren
nein
P5 anfahren ?
ja
Position 5 anfahren
nein
Ende
Modul 12
Modul 12 und Modul 13 unterscheiden sich in den mit diesem Symbol gekennzeichneten Bausteinen. Es wird hier darauf verzichtet, die Kommunikation zwischen Steuerung, LCD - Display und OlliControl detailliert darzustellen !
Ablaufdiagramm Automatisierung
Betriebsart Automatik
Modul 12
Blatt 120.0

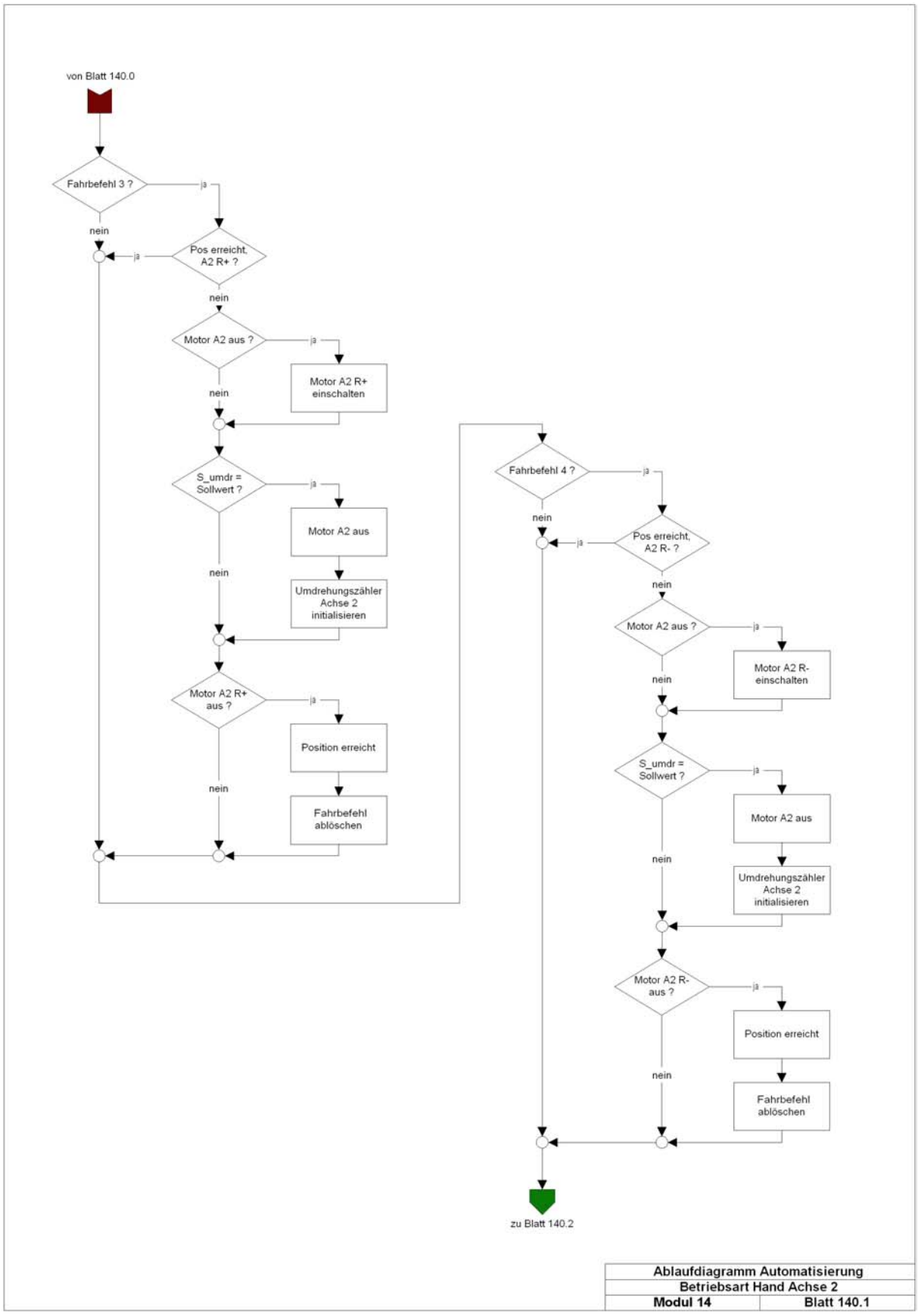
(Abb. 9-30)


Start
Modul 13
Grdstllg anfahren ?
ja
Grundstellung anfahren
nein
P1 anfahren ?
ja
Position 1 anfahren
nein
P2 anfahren ?
ja
Position 2 anfahren
nein
P3 anfahren ?
ja
Position 3 anfahren
nein
P4 anfahren ?
ja
Position 4 anfahren
nein
P5 anfahren ?
ja
Position 5 anfahren
nein
Ende
Modul 13
Modul 12 und Modul 13 unterscheiden sich in den mit diesem Symbol gekennzeichneten Bausteinen. Es wird hier darauf verzichtet, die Kommunikation zwischen Steuerung, LCD - Display und OlliControl detailliert darzustellen !
Ablaufdiagramm Automatisierung
Betriebsart Teilautomatik
Modul 13
Blatt 130.0

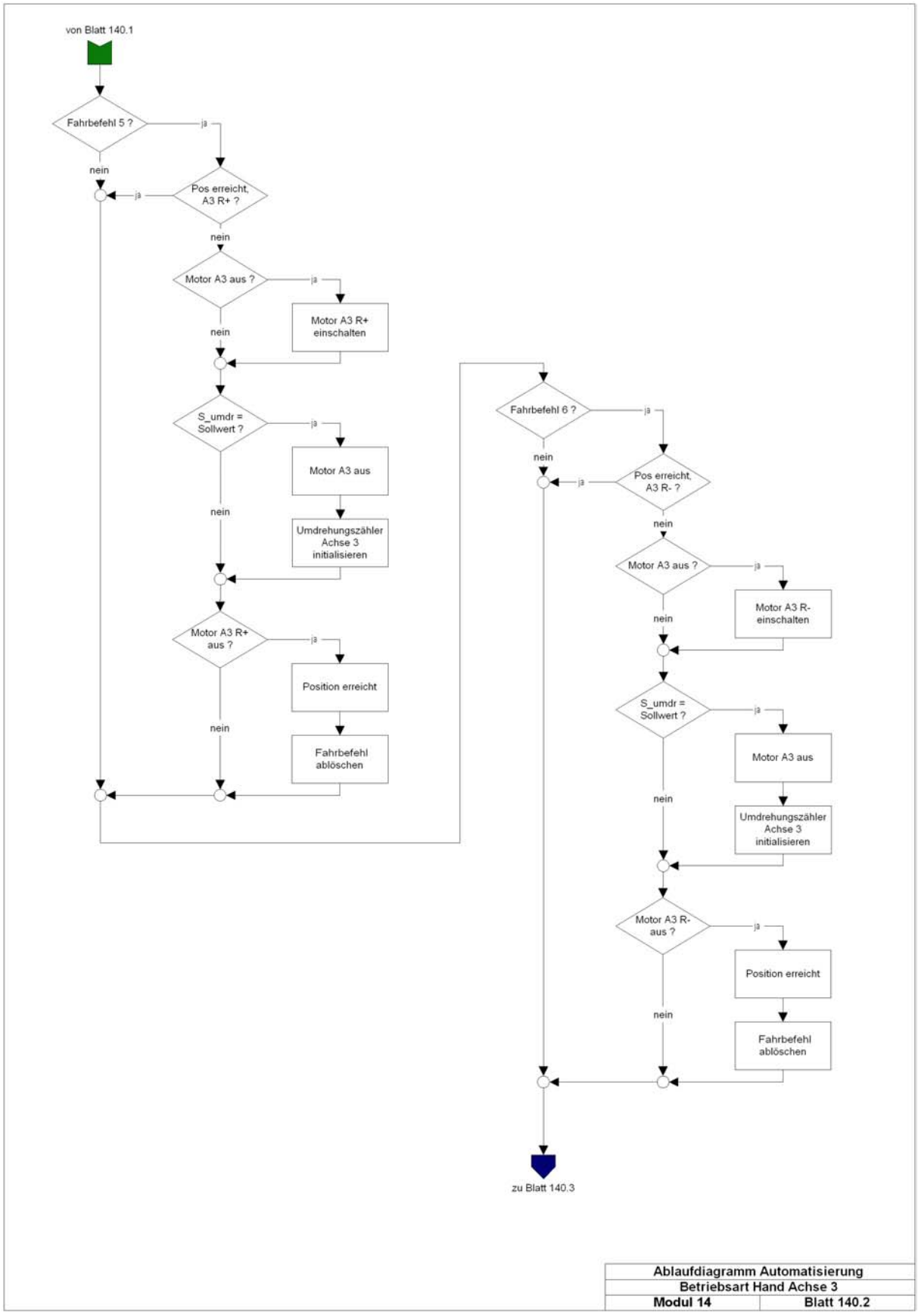
(Abb. 9-31)


Start
Modul 14
Fahrbefehl 1 ?
ja
nein
Pos erreicht, A1 R+ ?
ja
nein
Motor A1 aus ?
ja
Motor A1 R+ einschalten
nein
S_umdr = Sollwert ?
ja
Motor A1 aus
nein
Umdrehungszähler Achse 1 initialisieren
Motor A1 R+ aus ?
ja
Position erreicht
nein
Fahrbefehl ablöschen
Fahrbefehl 2 ?
ja
nein
Pos erreicht, A1 R- ?
ja
nein
Motor A1 aus ?
ja
Motor A1 R- einschalten
nein
S_umdr = Sollwert ?
ja
Motor A1 aus
nein
Umdrehungszähler Achse 1 initialisieren
Motor A1 R- aus ?
ja
Position erreicht
nein
Fahrbefehl ablöschen
zu Blatt 140.1
Ablaufdiagramm Automatisierung
Betriebsart Hand Achse 1
Modul 14
Blatt 140.0

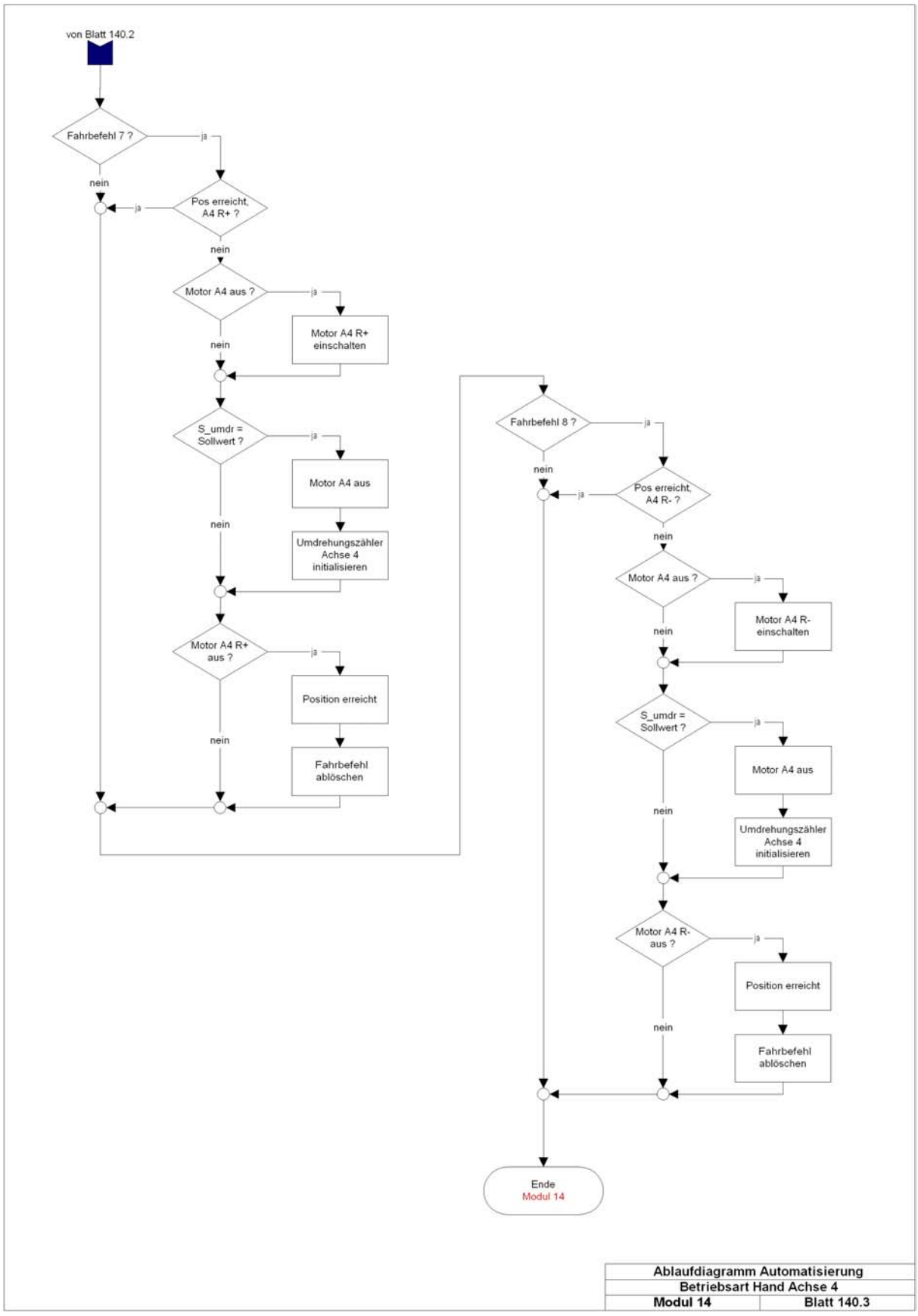
(Abb. 9-32)


von Blatt 140.0
Fahrbefehl 3 ?
ja
nein
Pos erreicht, A2 R+ ?
ja
nein
Motor A2 aus ?
ja
Motor A2 R+ einschalten
nein
S_umdr = Sollwert ?
ja
Motor A2 aus
nein
Umdrehungszähler Achse 2 initialisieren
Motor A2 R+ aus ?
ja
Position erreicht
nein
Fahrbefehl ablöschen
Fahrbefehl 4 ?
ja
nein
Pos erreicht, A2 R- ?
ja
nein
Motor A2 aus ?
ja
Motor A2 R- einschalten
nein
S_umdr = Sollwert ?
ja
Motor A2 aus
nein
Umdrehungszähler Achse 2 initialisieren
Motor A2 R- aus ?
ja
Position erreicht
nein
Fahrbefehl ablöschen
zu Blatt 140.2
Ablaufdiagramm Automatisierung
Betriebsart Hand Achse 2
Modul 14
Blatt 140.1

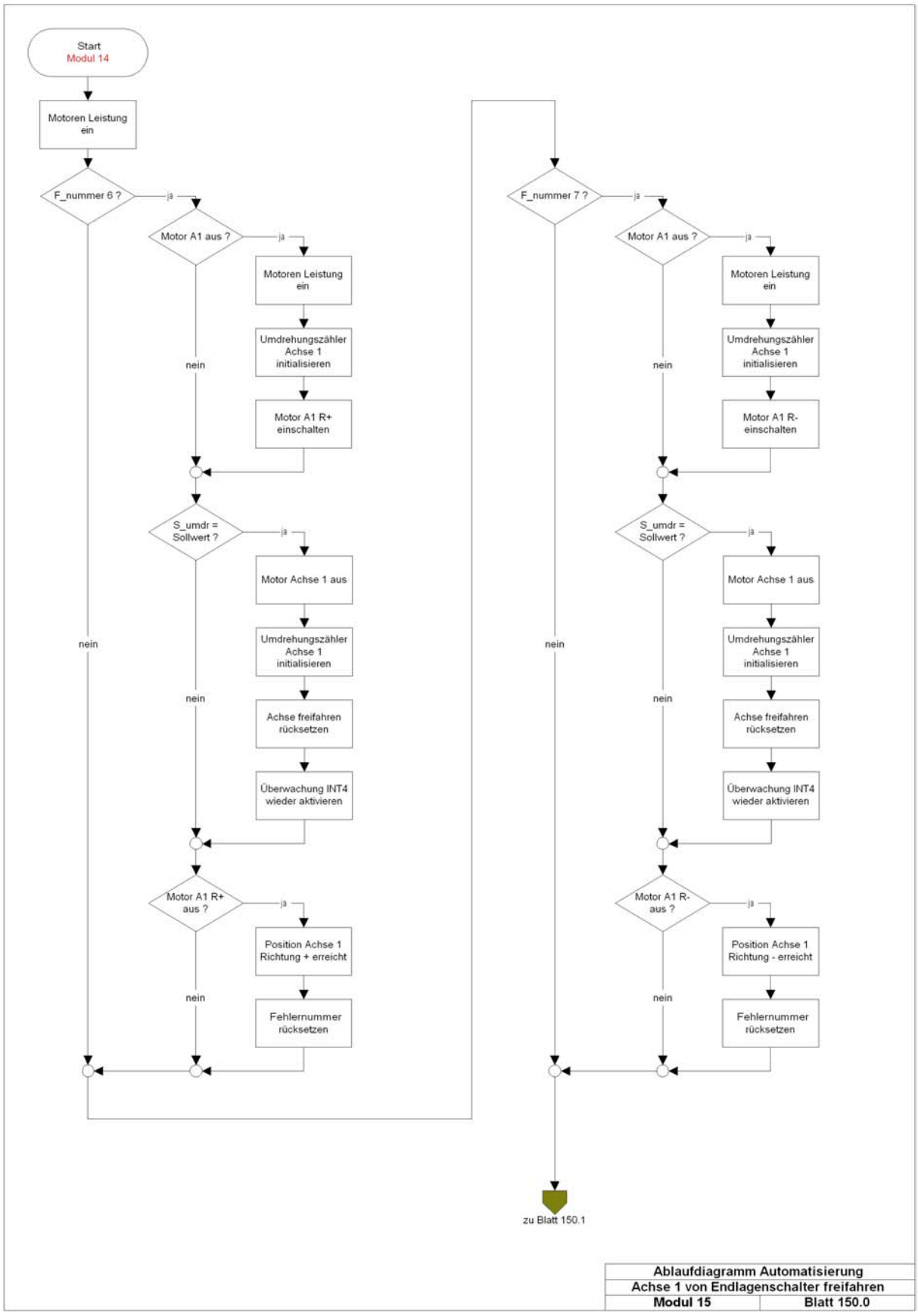
(Abb. 9-33)


von Blatt 140.1
Fahrbefehl 5 ?
ja
nein
Pos erreicht, A3 R+ ?
ja
nein
Motor A3 aus ?
ja
Motor A3 R+ einschalten
nein
S_umdr = Sollwert ?
ja
Motor A3 aus
nein
Umdrehungszähler Achse 3 initialisieren
Motor A3 R+ aus ?
ja
Position erreicht
nein
Fahrbefehl ablöschen
Fahrbefehl 6 ?
ja
nein
Pos erreicht, A3 R- ?
ja
nein
Motor A3 aus ?
ja
Motor A3 R- einschalten
nein
S_umdr = Sollwert ?
ja
Motor A3 aus
nein
Umdrehungszähler Achse 3 initialisieren
Motor A3 R- aus ?
ja
Position erreicht
nein
Fahrbefehl ablöschen
zu Blatt 140.3
Ablaufdiagramm Automatisierung
Betriebsart Hand Achse 3
Modul 14
Blatt 140.2

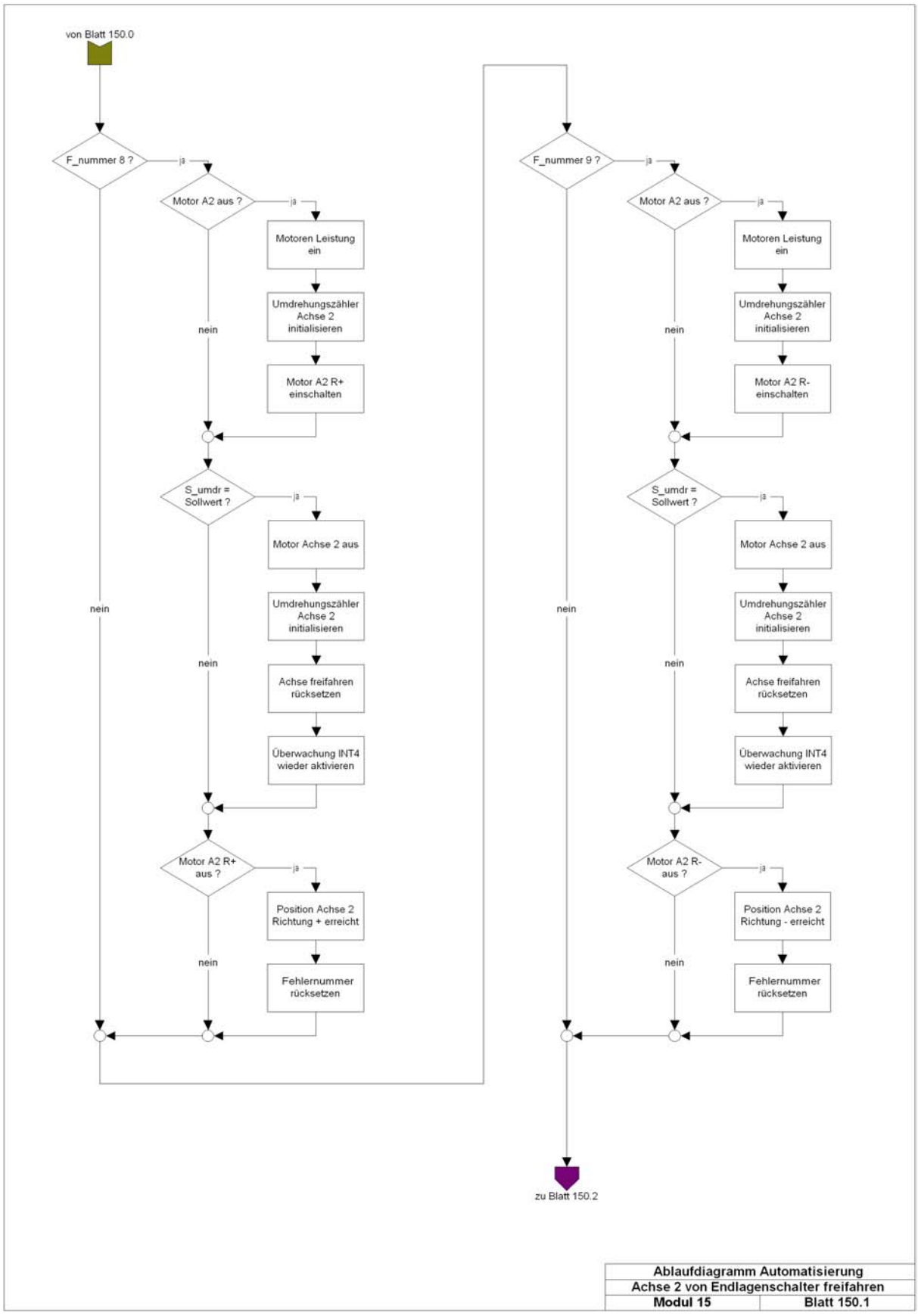
(Abb. 9-34)


von Blatt 140.2
Fahrbefehl 7 ?
ja
nein
Pos erreicht, A4 R+ ?
ja
nein
Motor A4 aus ?
ja
Motor A4 R+ einschalten
nein
S_umdr = Sollwert ?
ja
Motor A4 aus
nein
Umdrehungszähler Achse 4 initialisieren
Motor A4 R+ aus ?
ja
Position erreicht
nein
Fahrbefehl ablöschen
Fahrbefehl 8 ?
ja
nein
Pos erreicht, A4 R- ?
ja
nein
Motor A4 aus ?
ja
Motor A4 R- einschalten
nein
S_umdr = Sollwert ?
ja
Motor A4 aus
nein
Umdrehungszähler Achse 4 initialisieren
Motor A4 R- aus ?
ja
Position erreicht
nein
Fahrbefehl ablöschen
Ende
Modul 14
Ablaufdiagramm Automatisierung
Betriebsart Hand Achse 4
Modul 14
Blatt 140.3

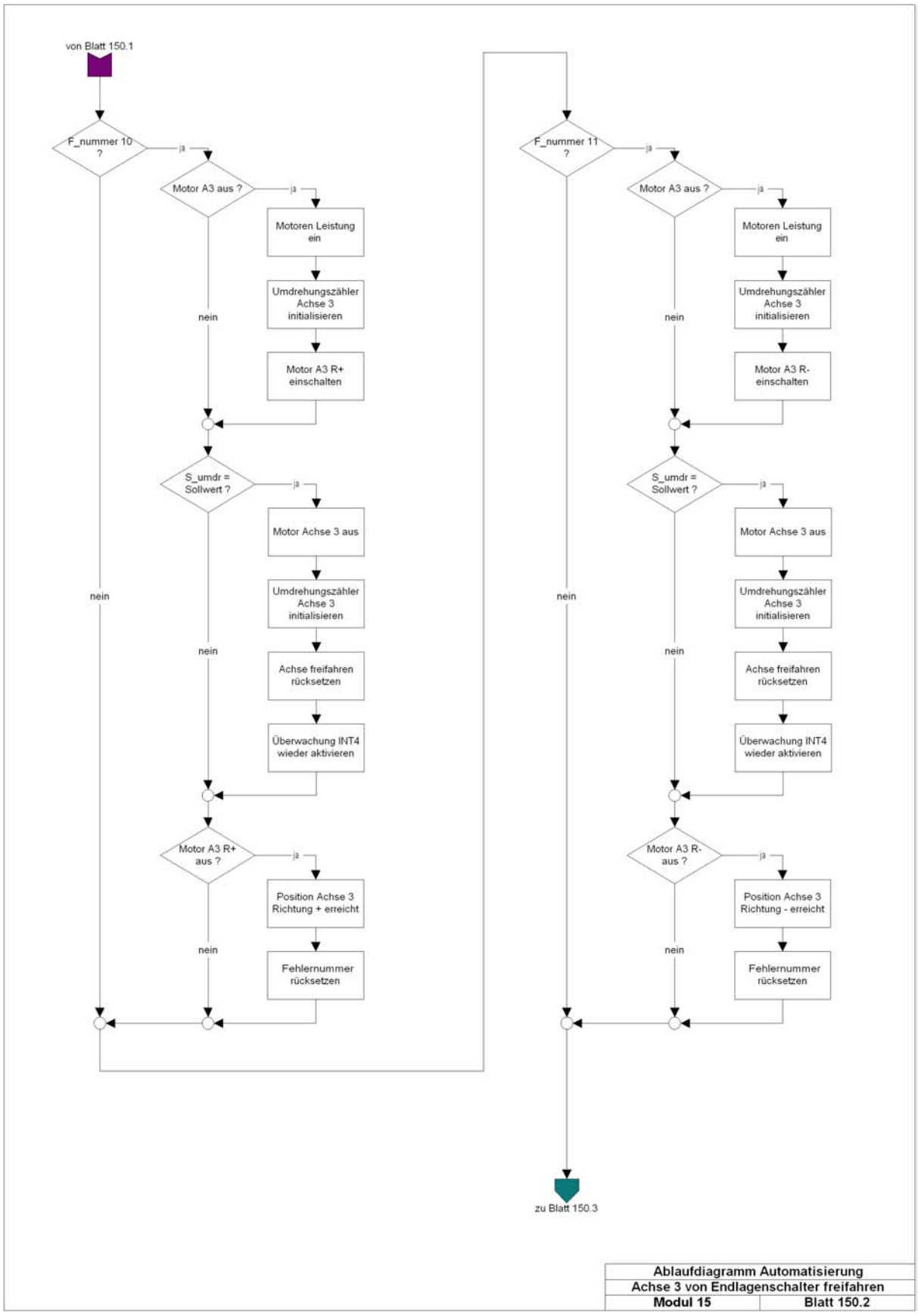
(Abb. 9-35)


Start
Modul 14
Motoren Leistung ein
F_nummer 6 ?
ja
Motor A1 aus ?
ja
Motoren Leistung ein
Umdrehungszähler Achse 1 initialisieren
Motor A1 R+ einschalten
nein
S_umdr = Sollwert ?
ja
Motor Achse 1 aus
Umdrehungszähler Achse 1 initialisieren
Achse freifahren rücksetzen
Überwachung INT4 wieder aktivieren
Motor A1 R+ aus ?
ja
Position Achse 1 Richtung + erreicht
Fehlernummer rücksetzen
nein
F_nummer 7 ?
ja
Motor A1 aus ?
ja
Motoren Leistung ein
Umdrehungszähler Achse 1 initialisieren
Motor A1 R- einschalten
nein
S_umdr = Sollwert ?
ja
Motor Achse 1 aus
Umdrehungszähler Achse 1 initialisieren
Achse freifahren rücksetzen
Überwachung INT4 wieder aktivieren
Motor A1 R- aus ?
ja
Position Achse 1 Richtung - erreicht
Fehlernummer rücksetzen
nein
zu Blatt 150.1
Ablaufdiagramm Automatisierung
Achse 1 von Endlagenschalter freifahren
Modul 15
Blatt 150.0

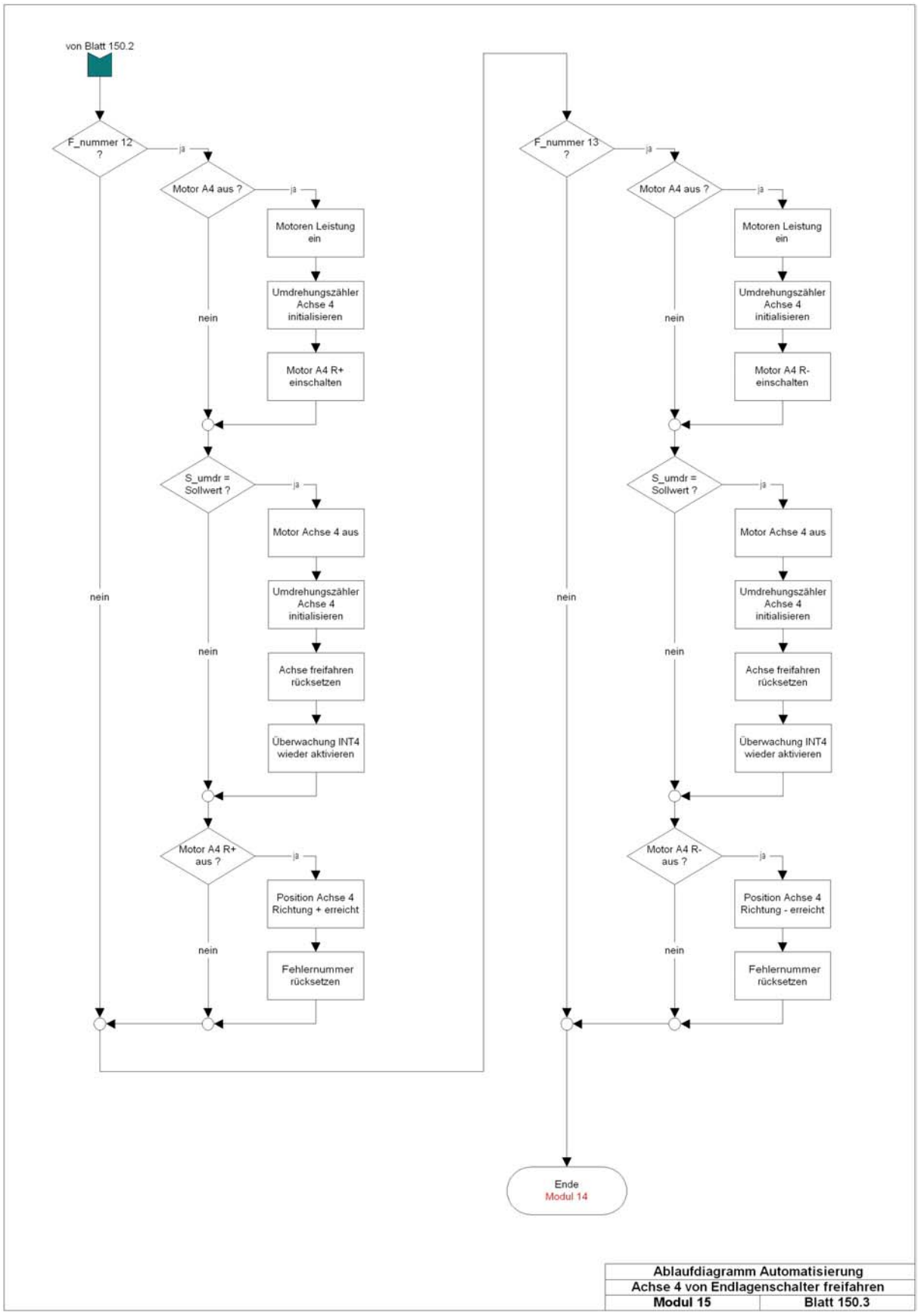
(Abb. 9-36)


von Blatt 150.0
F_nummer 8 ?
ja
Motor A2 aus ?
ja
Motoren Leistung ein
Umdrehungszähler Achse 2 initialisieren
Motor A2 R+ einschalten
nein
S_umdr = Sollwert ?
ja
Motor Achse 2 aus
Umdrehungszähler Achse 2 initialisieren
Achse freifahren rücksetzen
Überwachung INT4 wieder aktivieren
nein
Motor A2 R+ aus ?
ja
Position Achse 2 Richtung + erreicht
Fehlernummer rücksetzen
nein
F_nummer 9 ?
ja
Motor A2 aus ?
ja
Motoren Leistung ein
Umdrehungszähler Achse 2 initialisieren
Motor A2 R- einschalten
nein
S_umdr = Sollwert ?
ja
Motor Achse 2 aus
Umdrehungszähler Achse 2 initialisieren
Achse freifahren rücksetzen
Überwachung INT4 wieder aktivieren
nein
Motor A2 R- aus ?
ja
Position Achse 2 Richtung - erreicht
Fehlernummer rücksetzen
nein
zu Blatt 150.2
Ablaufdiagramm Automatisierung
Achse 2 von Endlagenschalter freifahren
Modul 15
Blatt 150.1

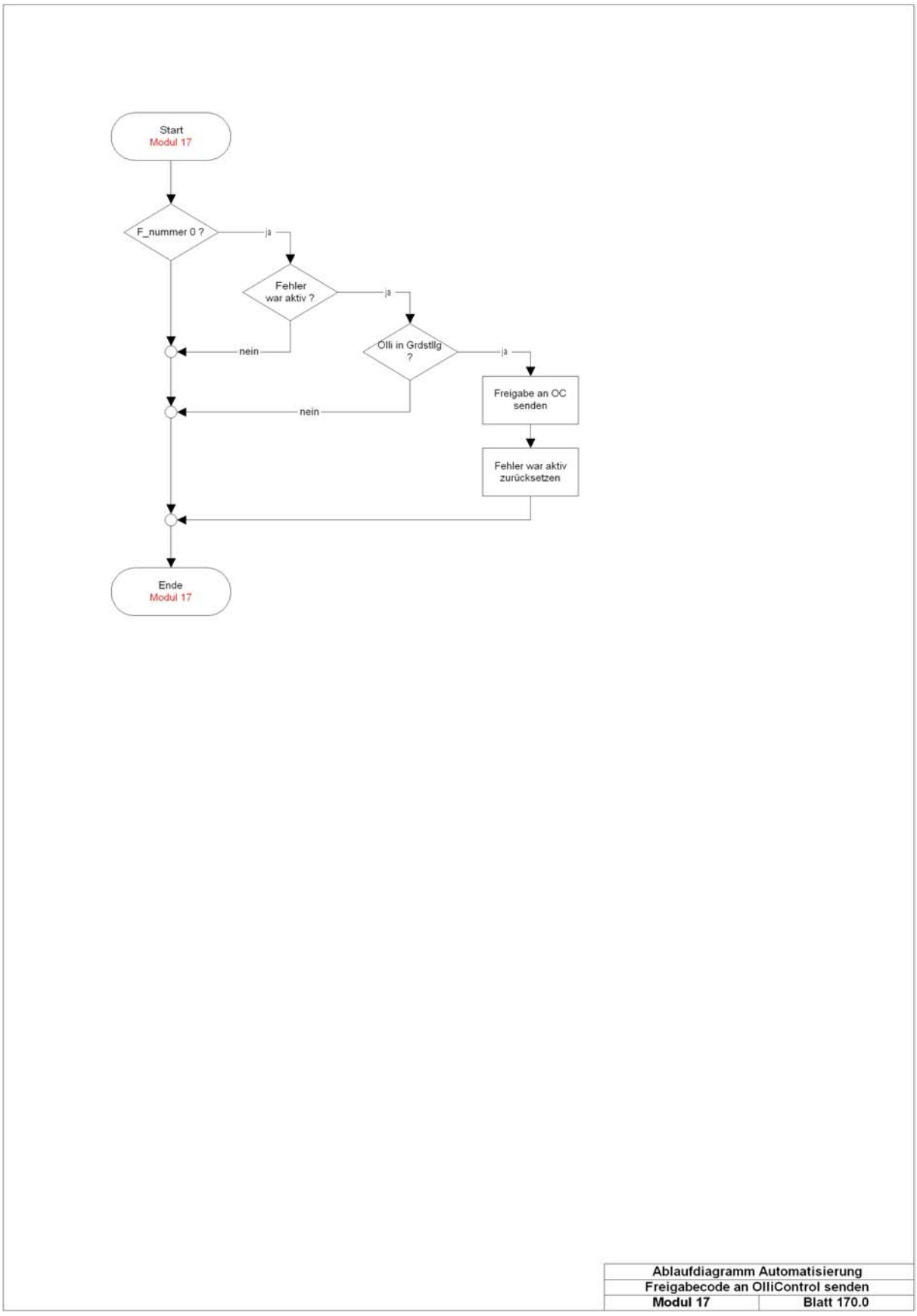
(Abb. 9-37)


von Blatt 150.1
F_nummer 10 ?
ja
Motor A3 aus ?
ja
Motoren Leistung ein
Umdrehungszähler Achse 3 initialisieren
Motor A3 R+ einschalten
nein
S_umdr = Sollwert ?
ja
Motor Achse 3 aus
Umdrehungszähler Achse 3 initialisieren
Achse freifahren rücksetzen
Überwachung INT4 wieder aktivieren
nein
Motor A3 R+ aus ?
ja
Position Achse 3 Richtung + erreicht
Fehlernummer rücksetzen
nein
F_nummer 11 ?
ja
Motor A3 aus ?
ja
Motoren Leistung ein
Umdrehungszähler Achse 3 initialisieren
Motor A3 R- einschalten
nein
S_umdr = Sollwert ?
ja
Motor Achse 3 aus
Umdrehungszähler Achse 3 initialisieren
Achse freifahren rücksetzen
Überwachung INT4 wieder aktivieren
nein
Motor A3 R- aus ?
ja
Position Achse 3 Richtung - erreicht
Fehlernummer rücksetzen
nein
zu Blatt 150.3
Ablaufdiagramm Automatisierung
Achse 3 von Endlagenschalter freifahren
Modul 15
Blatt 150.2

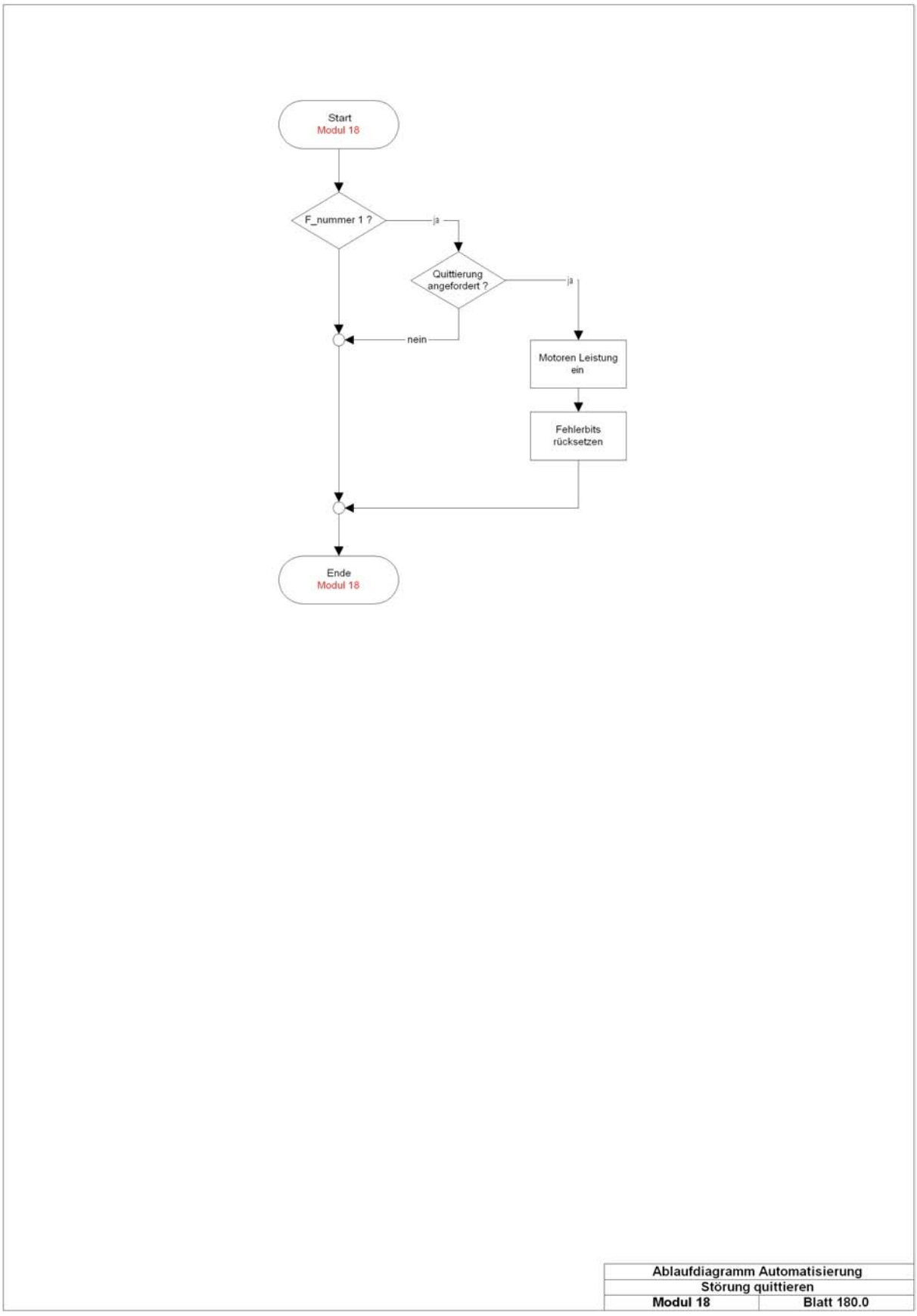
(Abb. 9-38)


von Blatt 150.2
F_nummer 12 ?
ja
Motor A4 aus ?
ja
Motoren Leistung ein
Umdrehungszähler Achse 4 initialisieren
Motor A4 R+ einschalten
nein
S_umdr = Sollwert ?
ja
Motor Achse 4 aus
Umdrehungszähler Achse 4 initialisieren
Achse freifahren rücksetzen
Überwachung INT4 wieder aktivieren
Motor A4 R+ aus ?
ja
Position Achse 4 Richtung + erreicht
Fehlernummer rücksetzen
nein
F_nummer 13 ?
ja
Motor A4 aus ?
ja
Motoren Leistung ein
Umdrehungszähler Achse 4 initialisieren
Motor A4 R- einschalten
nein
S_umdr = Sollwert ?
ja
Motor Achse 4 aus
Umdrehungszähler Achse 4 initialisieren
Achse freifahren rücksetzen
Überwachung INT4 wieder aktivieren
Motor A4 R- aus ?
ja
Position Achse 4 Richtung - erreicht
Fehlernummer rücksetzen
nein
Ende
Modul 14
Ablaufdiagramm Automatisierung
Achse 4 von Endlagenschalter freifahren
Modul 15
Blatt 150.3

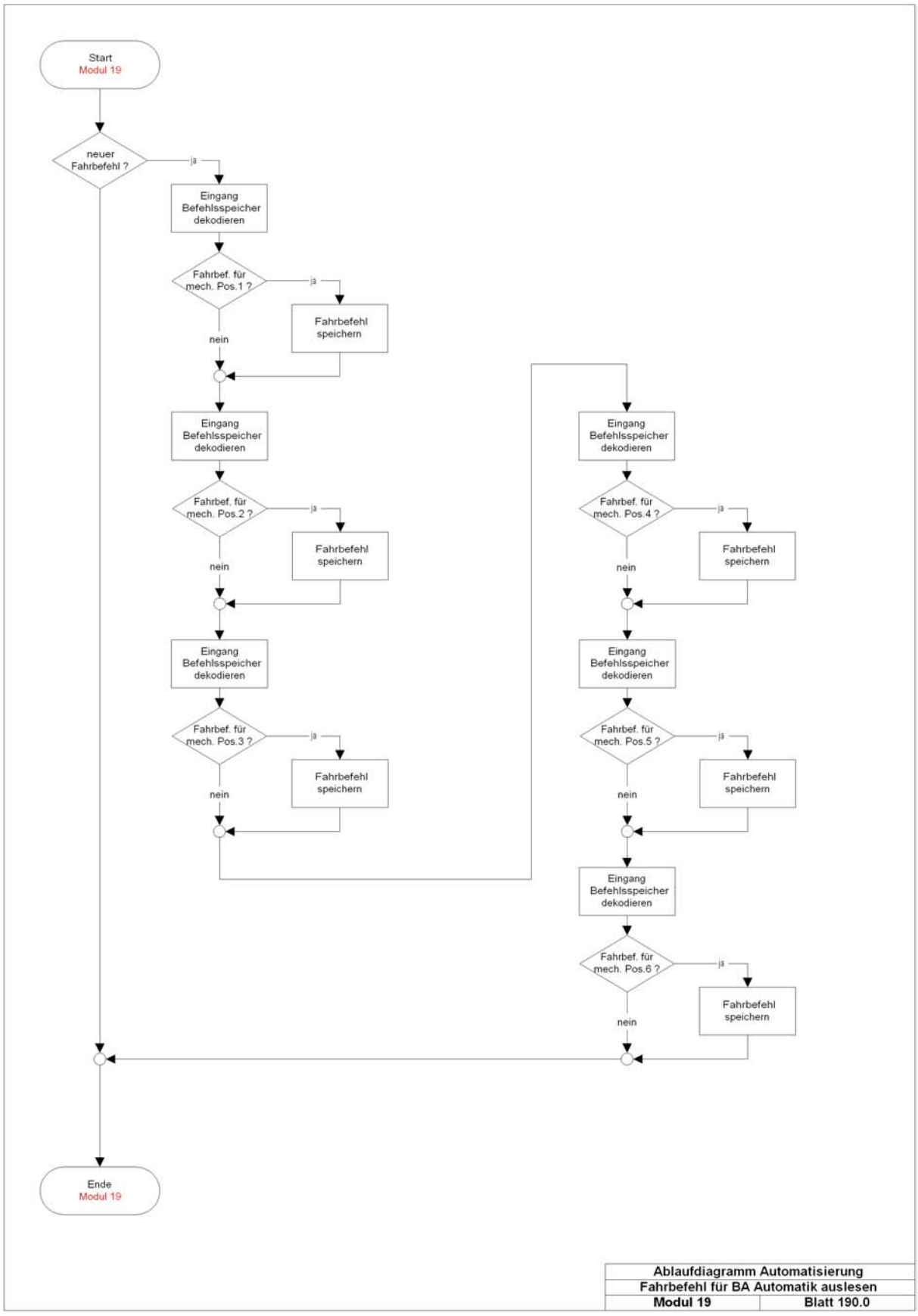
(Abb. 9-39)


Start
Modul 17
F_nummer 0 ?
ja
Fehler
war aktiv ?
ja
nein
Olli in Grdstllg
?
ja
Freigabe an OC
senden
nein
Fehler war aktiv
zurücksetzen
Ende
Modul 17
Ablaufdiagramm Automatisierung
Freigabecode an OlliControl senden
Modul 17
Blatt 170.0

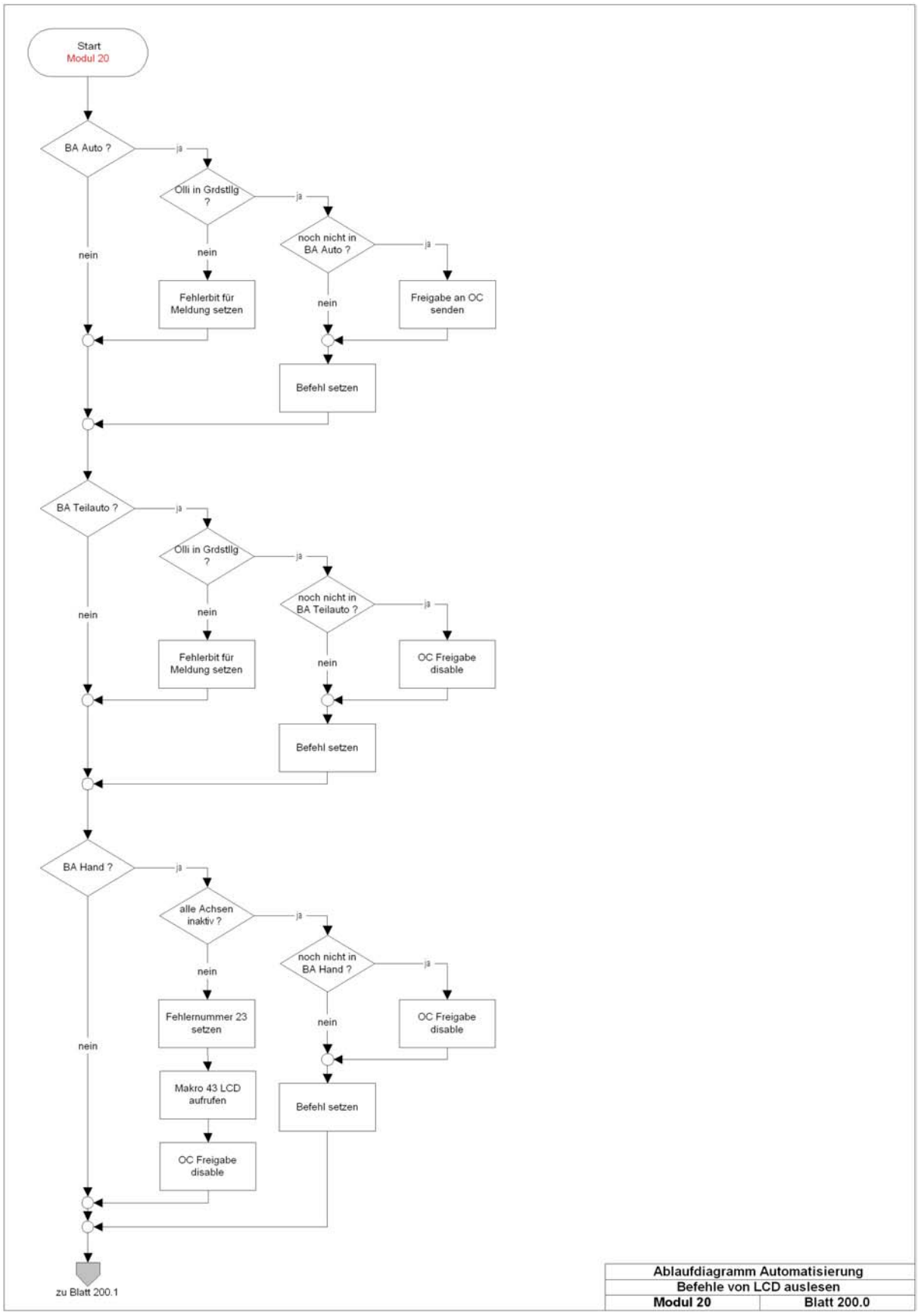
(Abb. 9-40)


Start
Modul 18
F_nummer 1 ?
ja
Quittierung angefordert ?
ja
nein
Motoren Leistung ein
Fehlerbits rücksetzen
Ende
Modul 18
Ablaufdiagramm Automatisierung
Störung quittieren
Modul 18
Blatt 180.0

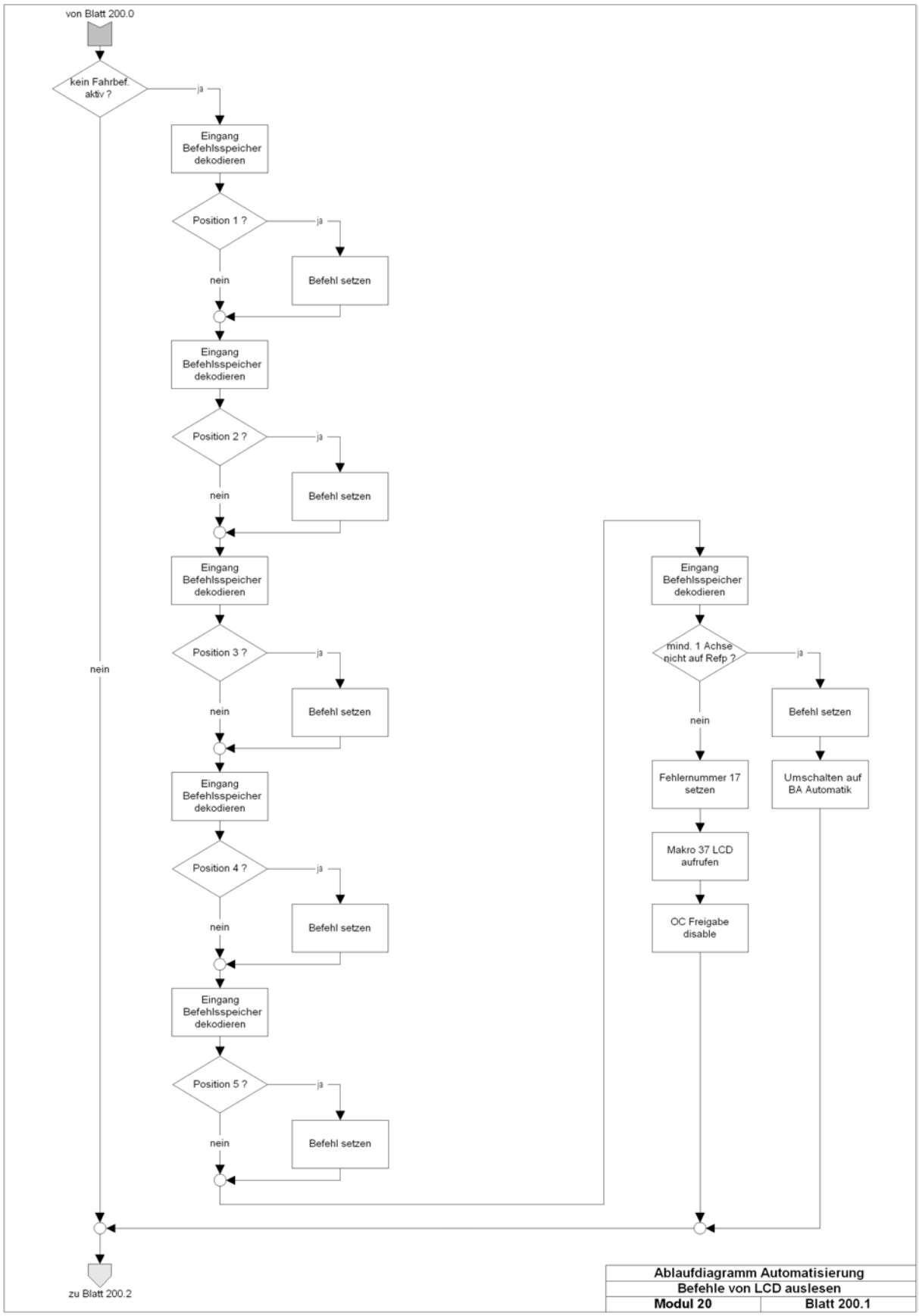
(Abb. 9-41)


Start
Modul 19
neuer Fahrbefehl ?
ja
Eingang Befehlsspeicher dekodieren
Fahrbef. für mech. Pos.1 ?
ja
Fahrbefehl speichern
nein
Eingang Befehlsspeicher dekodieren
Fahrbef. für mech. Pos.2 ?
ja
Fahrbefehl speichern
nein
Eingang Befehlsspeicher dekodieren
Fahrbef. für mech. Pos.3 ?
ja
Fahrbefehl speichern
nein
Eingang Befehlsspeicher dekodieren
Fahrbef. für mech. Pos.4 ?
ja
Fahrbefehl speichern
nein
Eingang Befehlsspeicher dekodieren
Fahrbef. für mech. Pos.5 ?
ja
Fahrbefehl speichern
nein
Eingang Befehlsspeicher dekodieren
Fahrbef. für mech. Pos.6 ?
ja
Fahrbefehl speichern
nein
Ende
Modul 19
Ablaufdiagramm Automatisierung
Fahrbefehl für BA Automatik auslesen
Modul 19
Blatt 190.0

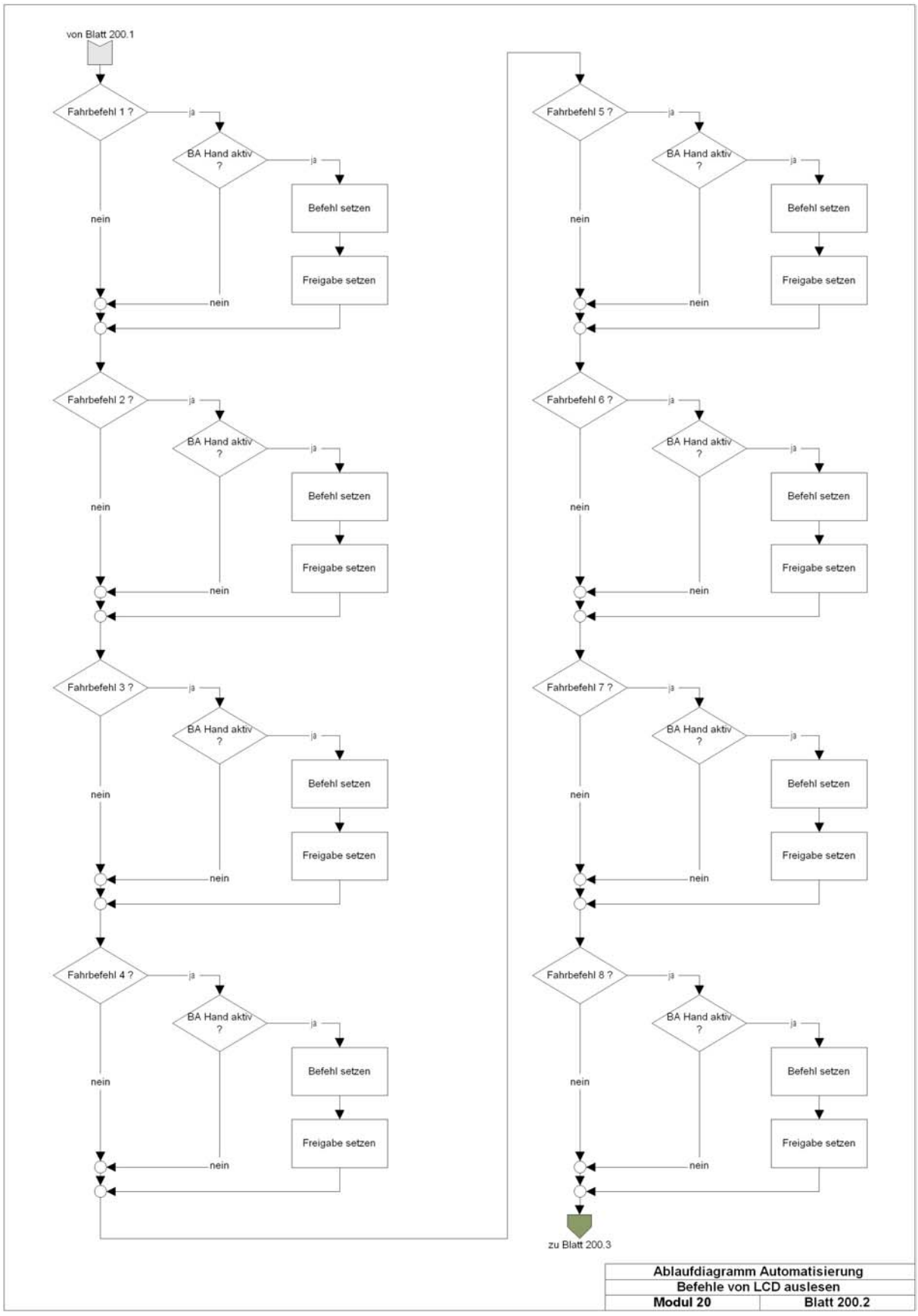
(Abb. 9-42)


Start
Modul 20
BA Auto ?
ja
nein
Olli in Grdstllg ?
ja
nein
noch nicht in BA Auto ?
ja
nein
Fehlerbit für Meldung setzen
Freigabe an OC senden
Befehl setzen
BA Teilauto ?
ja
nein
Olli in Grdstllg ?
ja
nein
noch nicht in BA Teilauto ?
ja
nein
Fehlerbit für Meldung setzen
OC Freigabe disable
Befehl setzen
BA Hand ?
ja
nein
alle Achsen inaktiv ?
ja
nein
noch nicht in BA Hand ?
ja
nein
Fehlernummer 23 setzen
OC Freigabe disable
Makro 43 LCD aufrufen
Befehl setzen
OC Freigabe disable
zu Blatt 200.1
Ablaufdiagramm Automatisierung
Befehle von LCD auslesen
Modul 20
Blatt 200.0

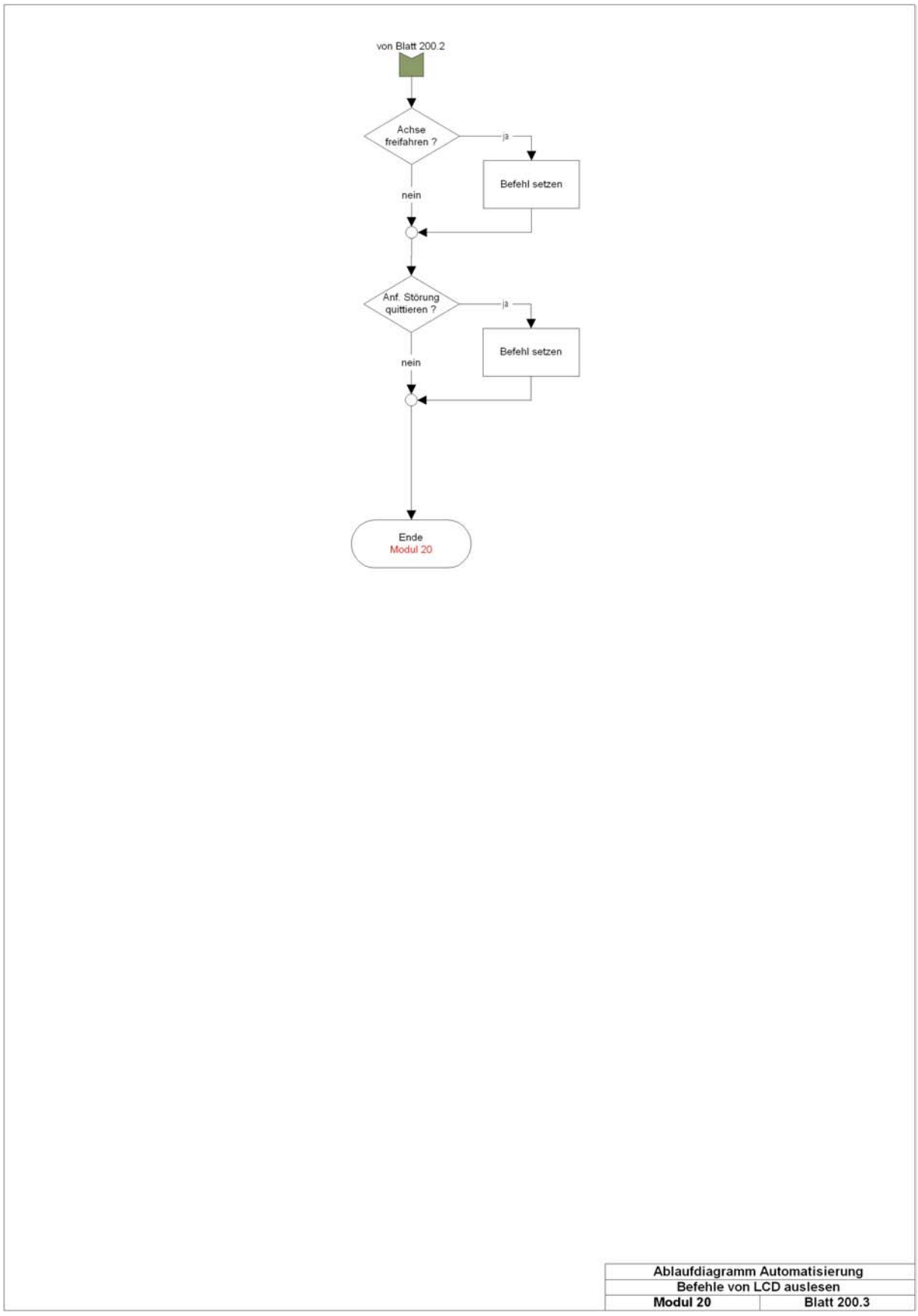
(Abb. 9-43)


von Blatt 200.0
kein Fahrbef. aktiv ?
ja
nein
Eingang Befehlsspeicher dekodieren
Position 1 ?
Befehl setzen
Position 2 ?
Position 3 ?
Position 4 ?
Position 5 ?
mind. 1 Achse nicht auf Refp ?
Fehlernummer 17 setzen
Umschalten auf BA Automatik
Makro 37 LCD aufrufen
OC Freigabe disable
zu Blatt 200.2
Ablaufdiagramm Automatisierung
Befehle von LCD auslesen
Modul 20
Blatt 200.1

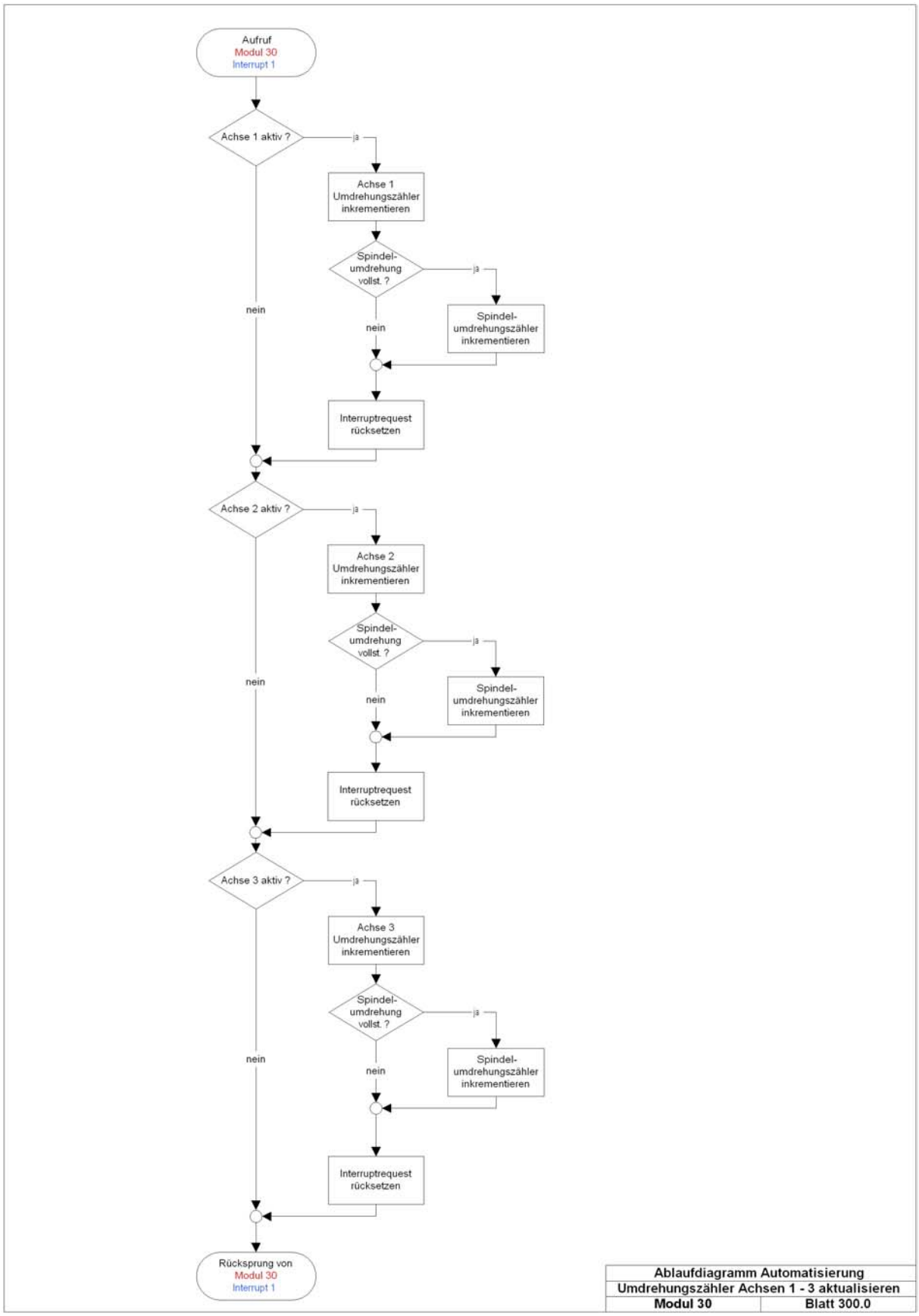
(Abb. 9-44)


von Blatt 200.1
Fahrbefehl 1 ?
ja
BA Hand aktiv ?
ja
Befehl setzen
Freigabe setzen
nein
nein
Fahrbefehl 2 ?
ja
BA Hand aktiv ?
ja
Befehl setzen
Freigabe setzen
nein
nein
Fahrbefehl 3 ?
ja
BA Hand aktiv ?
ja
Befehl setzen
Freigabe setzen
nein
nein
Fahrbefehl 4 ?
ja
BA Hand aktiv ?
ja
Befehl setzen
Freigabe setzen
nein
nein
Fahrbefehl 5 ?
ja
BA Hand aktiv ?
ja
Befehl setzen
Freigabe setzen
nein
nein
Fahrbefehl 6 ?
ja
BA Hand aktiv ?
ja
Befehl setzen
Freigabe setzen
nein
nein
Fahrbefehl 7 ?
ja
BA Hand aktiv ?
ja
Befehl setzen
Freigabe setzen
nein
nein
Fahrbefehl 8 ?
ja
BA Hand aktiv ?
ja
Befehl setzen
Freigabe setzen
nein
nein
zu Blatt 200.3
Ablaufdiagramm Automatisierung
Befehle von LCD auslesen
Modul 20
Blatt 200.2

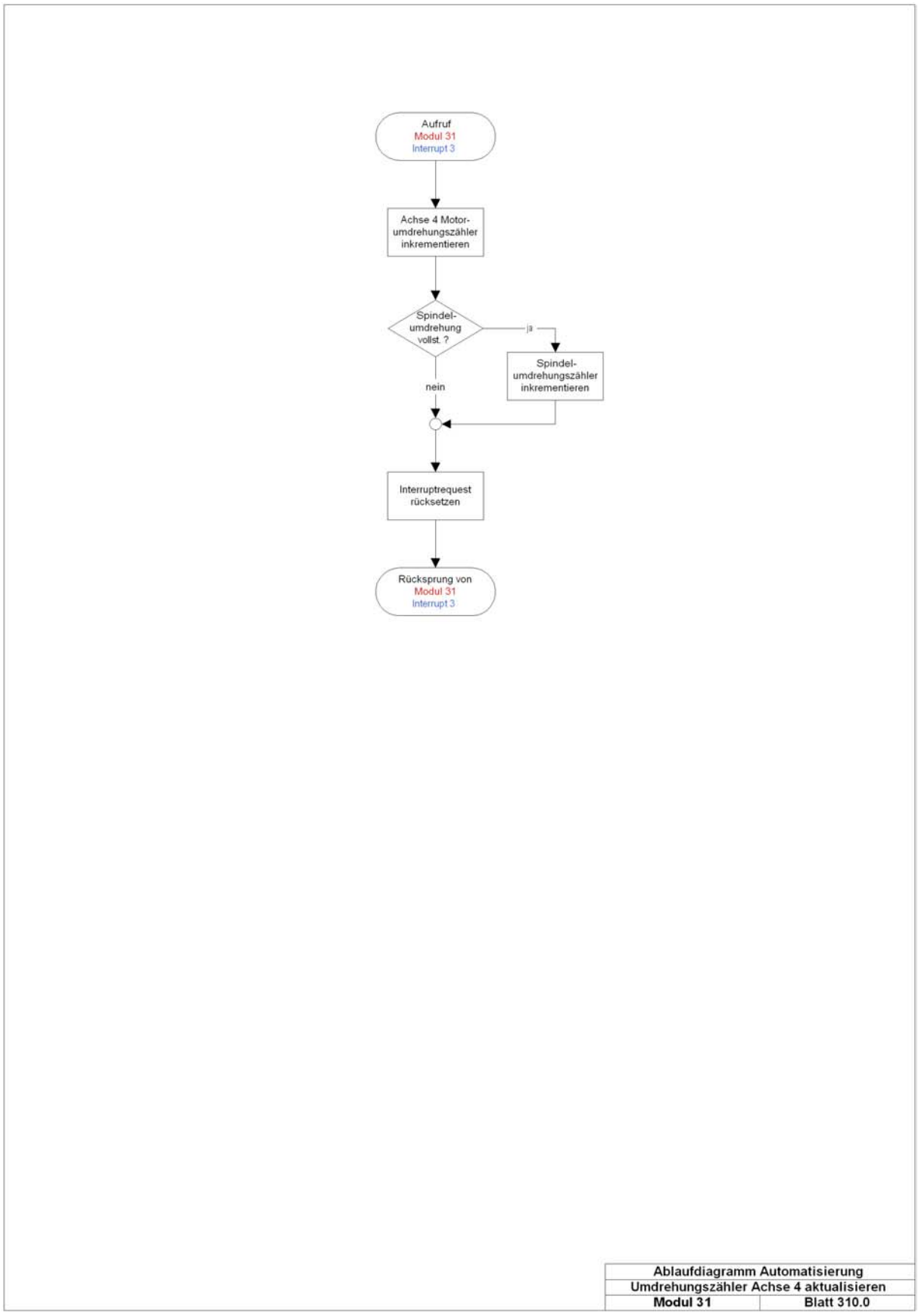
(Abb. 9-45)


von Blatt 200.2
Achse freifahren ?
ja
Befehl setzen
nein
Anf. Störung quittieren ?
ja
Befehl setzen
nein
Ende
Modul 20
Ablaufdiagramm Automatisierung
Befehle von LCD auslesen
Modul 20
Blatt 200.3

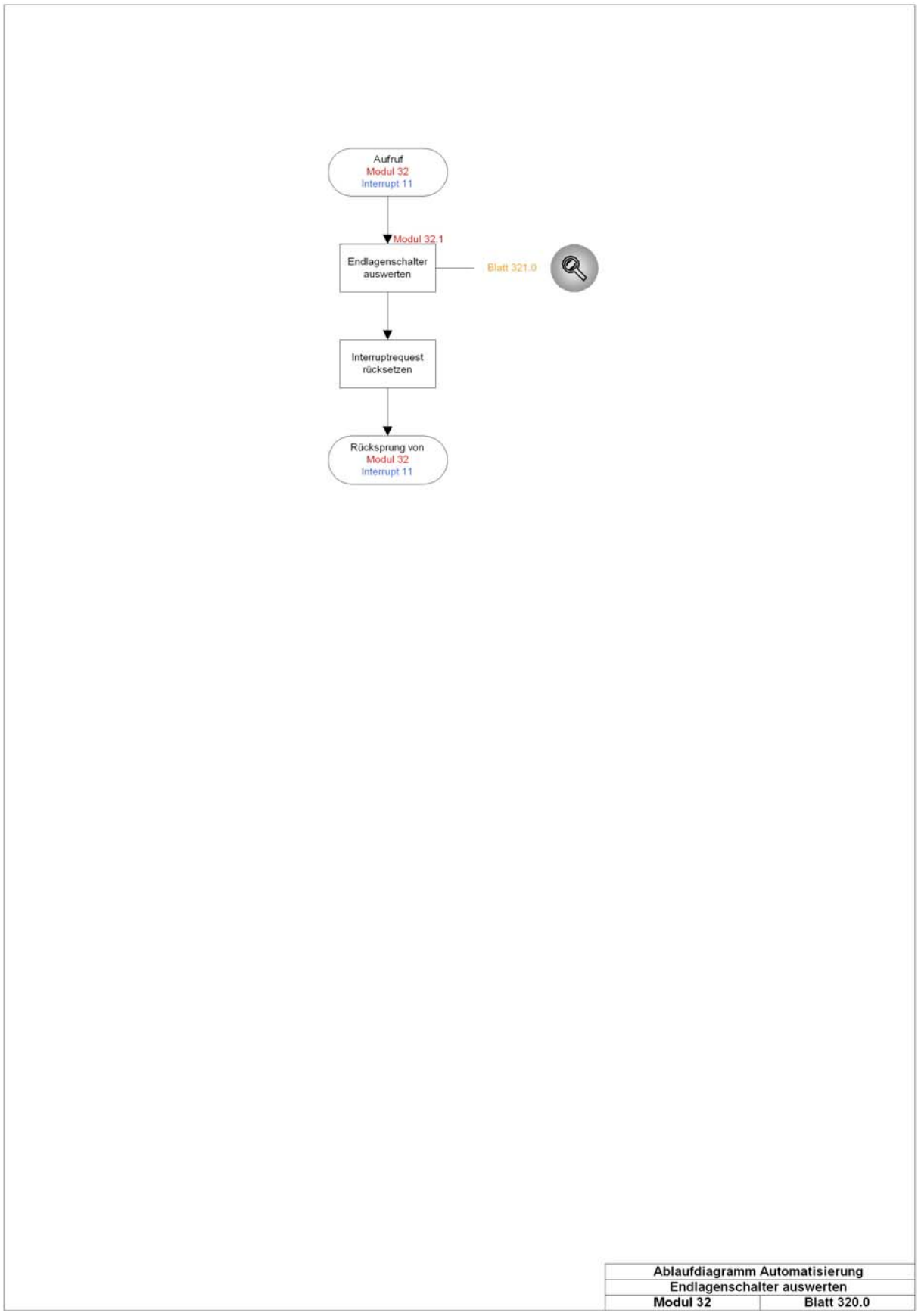
(Abb. 9-46)


Aufruf
Modul 30
Interrupt 1
Achse 1 aktiv ?
ja
nein
Achse 1 Umdrehungszähler inkrementieren
Spindel-umdrehung vollst. ?
ja
nein
Spindel-umdrehungszähler inkrementieren
Interruptrequest rücksetzen
Achse 2 aktiv ?
ja
nein
Achse 2 Umdrehungszähler inkrementieren
Spindel-umdrehung vollst. ?
ja
nein
Spindel-umdrehungszähler inkrementieren
Interruptrequest rücksetzen
Achse 3 aktiv ?
ja
nein
Achse 3 Umdrehungszähler inkrementieren
Spindel-umdrehung vollst. ?
ja
nein
Spindel-umdrehungszähler inkrementieren
Interruptrequest rücksetzen
Rücksprung von
Modul 30
Interrupt 1
Ablaufdiagramm Automatisierung
Umdrehungszähler Achsen 1 - 3 aktualisieren
Modul 30
Blatt 300.0

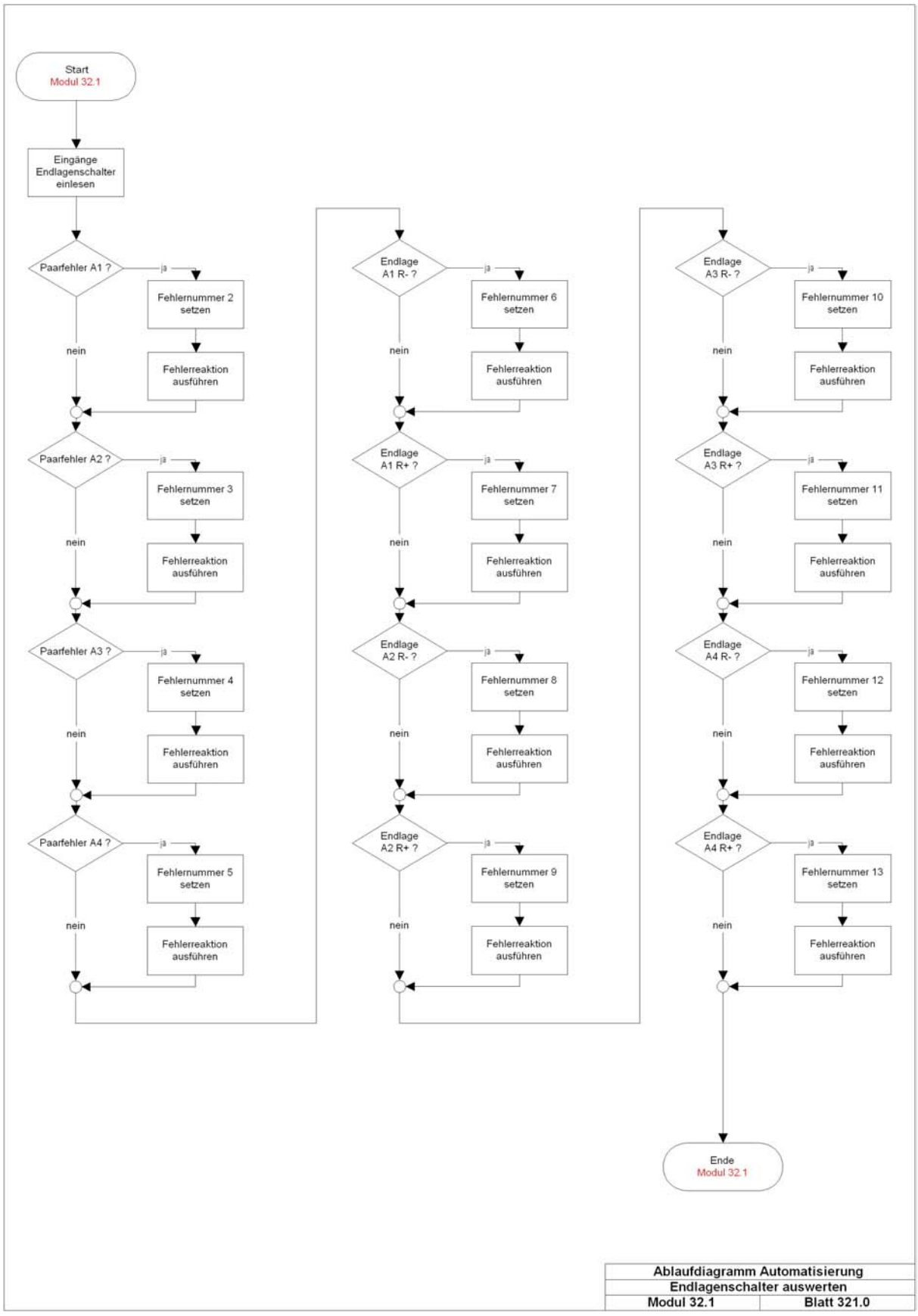
(Abb. 9-47)


Aufruf
Modul 31
Interrupt 3
Achse 4 Motor-umdrehungszähler inkrementieren
Spindel-umdrehung vollst. ?
ja
Spindel-umdrehungszähler inkrementieren
nein
Interruptrequest rücksetzen
Rücksprung von
Modul 31
Interrupt 3
Ablaufdiagramm Automatisierung
Umdrehungszähler Achse 4 aktualisieren
Modul 31
Blatt 310.0

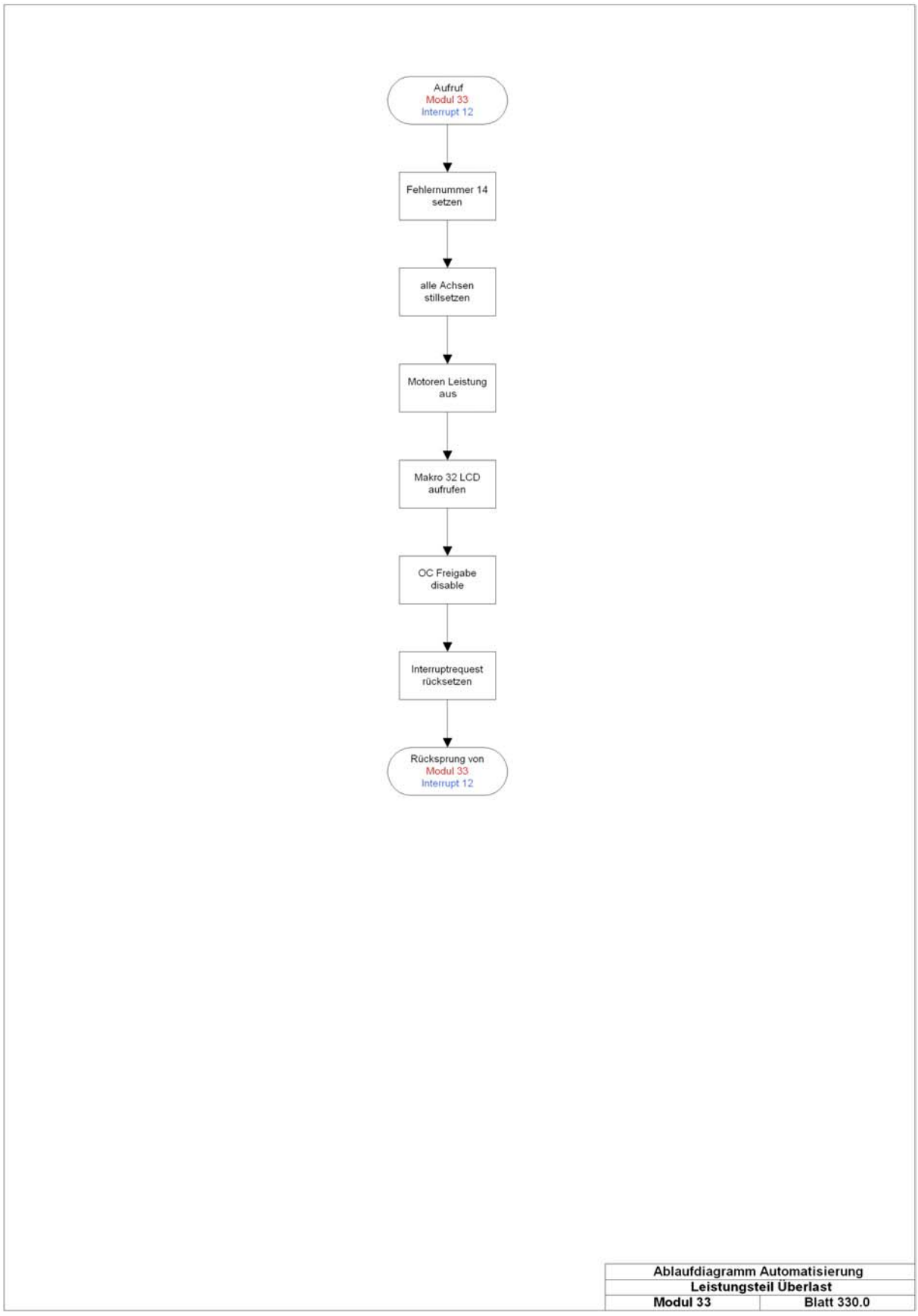
(Abb. 9-48)


Aufruf
Modul 32
Interrupt 11
Modul 32.1
Endlagenschalter auswerten
Blatt 321.0
Interruptrequest rücksetzen
Rücksprung von
Modul 32
Interrupt 11
Ablaufdiagramm Automatisierung
Endlagenschalter auswerten
Modul 32
Blatt 320.0

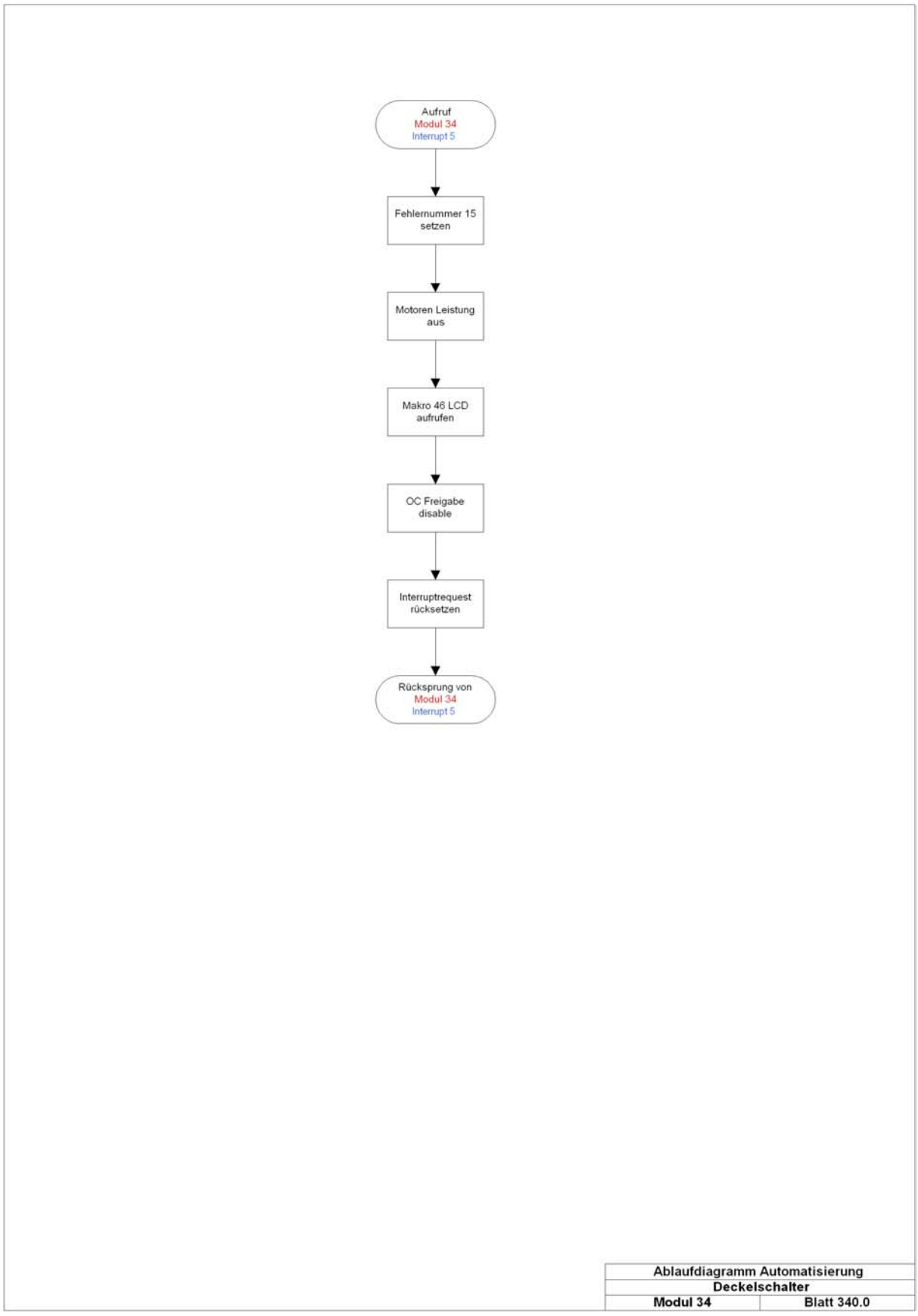
(Abb. 9-49)


Start
Modul 32.1
Eingänge Endlagenschalter einlesen
Paarfehler A1 ?
ja
Fehlernummer 2 setzen
nein
Fehlerreaktion ausführen
Paarfehler A2 ?
ja
Fehlernummer 3 setzen
nein
Fehlerreaktion ausführen
Paarfehler A3 ?
ja
Fehlernummer 4 setzen
nein
Fehlerreaktion ausführen
Paarfehler A4 ?
ja
Fehlernummer 5 setzen
nein
Fehlerreaktion ausführen
Endlage A1 R- ?
ja
Fehlernummer 6 setzen
nein
Fehlerreaktion ausführen
Endlage A1 R+ ?
ja
Fehlernummer 7 setzen
nein
Fehlerreaktion ausführen
Endlage A2 R- ?
ja
Fehlernummer 8 setzen
nein
Fehlerreaktion ausführen
Endlage A2 R+ ?
ja
Fehlernummer 9 setzen
nein
Fehlerreaktion ausführen
Endlage A3 R- ?
ja
Fehlernummer 10 setzen
nein
Fehlerreaktion ausführen
Endlage A3 R+ ?
ja
Fehlernummer 11 setzen
nein
Fehlerreaktion ausführen
Endlage A4 R- ?
ja
Fehlernummer 12 setzen
nein
Fehlerreaktion ausführen
Endlage A4 R+ ?
ja
Fehlernummer 13 setzen
nein
Fehlerreaktion ausführen
Ende
Modul 32.1
Ablaufdiagramm Automatisierung
Endlagenschalter auswerten
Modul 32.1
Blatt 321.0

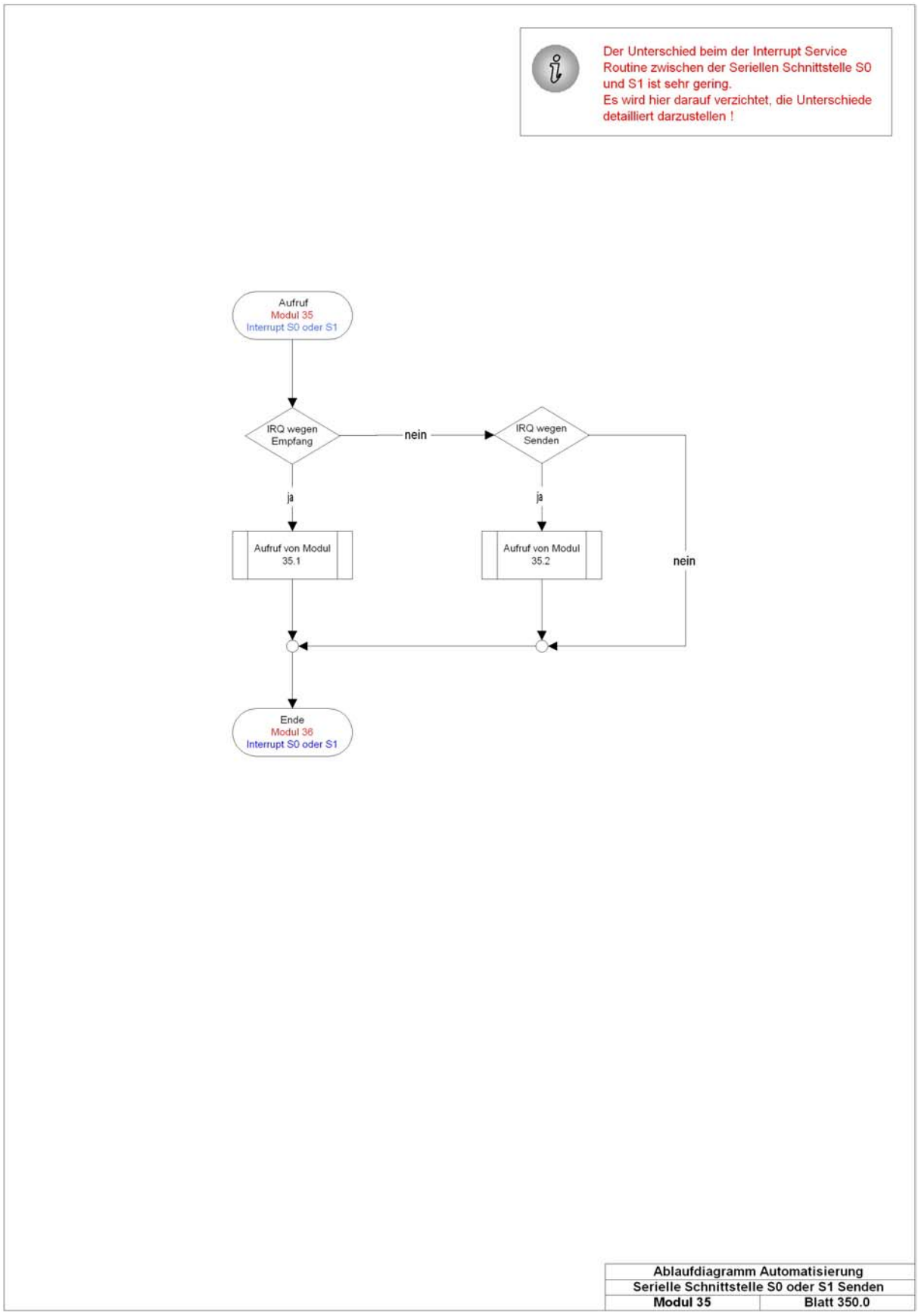
(Abb. 9-50)


Aufruf
Modul 33
Interrupt 12
Fehlernummer 14 setzen
alle Achsen stillsetzen
Motoren Leistung aus
Makro 32 LCD aufrufen
OC Freigabe disable
Interruptrequest rücksetzen
Rücksprung von
Modul 33
Interrupt 12
Ablaufdiagramm Automatisierung
Leistungsteil Überlast
Modul 33
Blatt 330.0

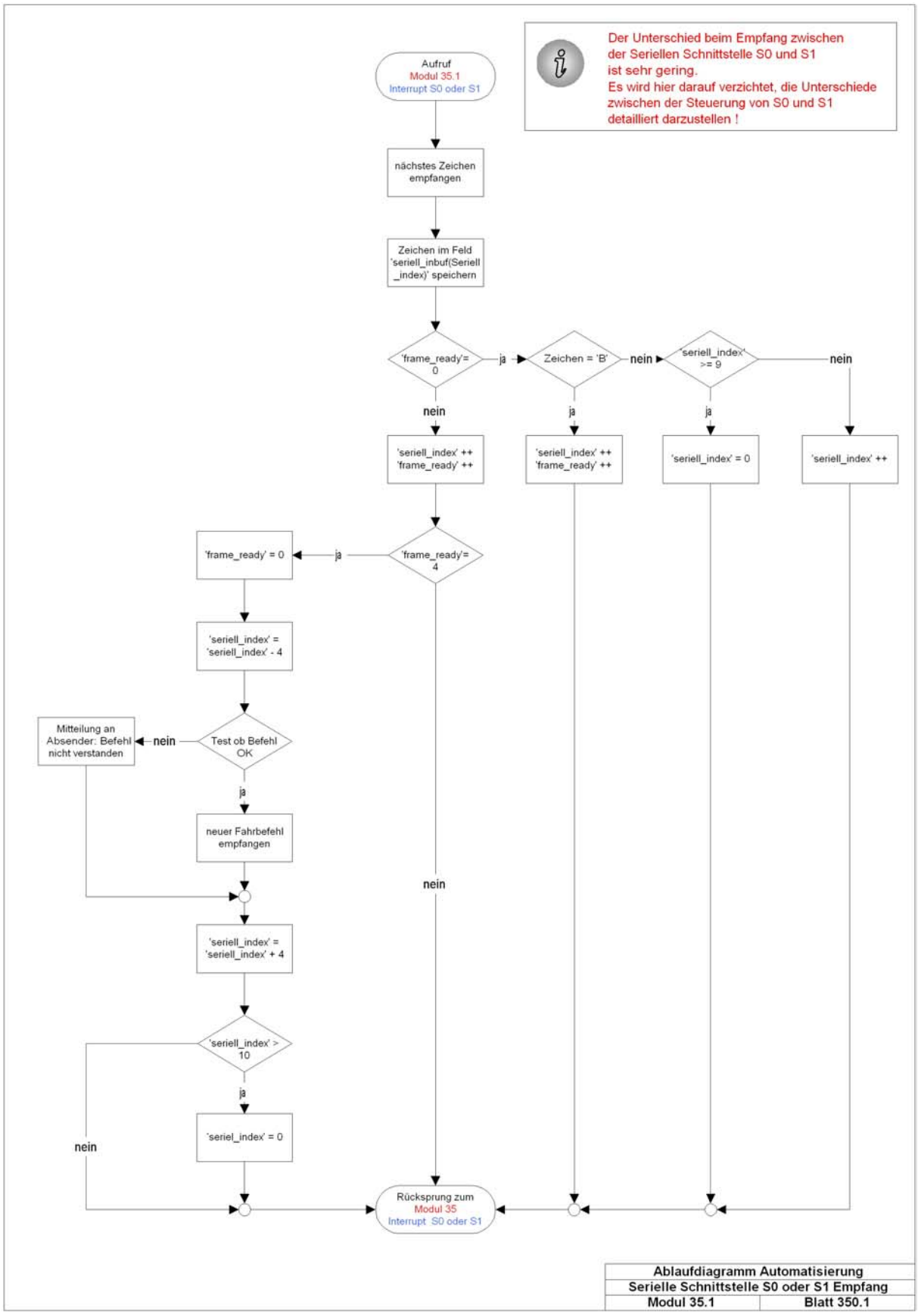
(Abb. 9-51)


Aufruf
Modul 34
Interrupt 5
Fehlernummer 15 setzen
Motoren Leistung aus
Makro 46 LCD aufrufen
OC Freigabe disable
Interruptrequest rücksetzen
Rücksprung von
Modul 34
Interrupt 5
Ablaufdiagramm Automatisierung
Deckelschalter
Modul 34
Blatt 340.0

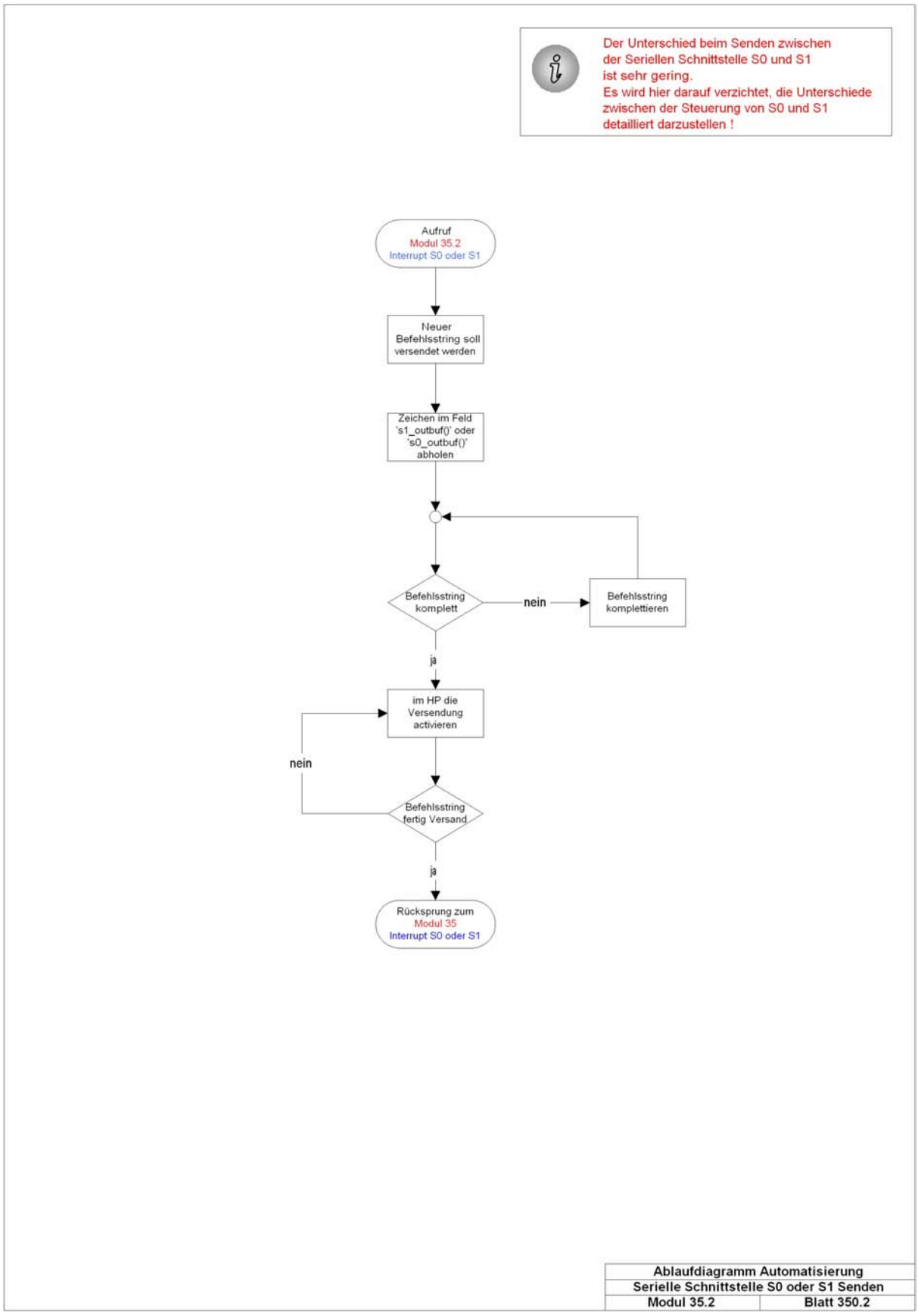
(Abb. 9-52)


Der Unterschied beim der Interrupt Service Routine zwischen der Seriellen Schnittstelle S0 und S1 ist sehr gering.
Es wird hier darauf verzichtet, die Unterschiede detailliert darzustellen !
Aufruf
Modul 35
Interrupt S0 oder S1
IRQ wegen Empfang
nein
IRQ wegen Senden
ja
ja
Aufruf von Modul 35.1
Aufruf von Modul 35.2
nein
Ende
Modul 36
Interrupt S0 oder S1
Ablaufdiagramm Automatisierung
Serielle Schnittstelle S0 oder S1 Senden
Modul 35
Blatt 350.0

(Abb. 9-53)


Der Unterschied beim Empfang zwischen der Seriellen Schnittstelle S0 und S1 ist sehr gering.
Es wird hier darauf verzichtet, die Unterschiede zwischen der Steuerung von S0 und S1 detailliert darzustellen !
Aufruf
Modul 35.1
Interrupt S0 oder S1
nächstes Zeichen empfangen
Zeichen im Feld 'seriell_inbuf(Seriell_index)' speichern
'frame_ready'= 0
ja
Zeichen = 'B'
nein
'seriell_index' >= 9
nein
nein
ja
ja
'seriell_index' ++ 'frame_ready' ++
'seriell_index' ++ 'frame_ready' ++
'seriell_index' = 0
'seriell_index' ++
'frame_ready'= 4
ja
'frame_ready' = 0
'seriell_index' = 'seriell_index' - 4
Test ob Befehl OK
nein
Mitteilung an Absender: Befehl nicht verstanden
ja
neuer Fahrbefehl empfangen
nein
'seriell_index' = 'seriell_index' + 4
'seriell_index' > 10
ja
'seriel_index' = 0
nein
Rücksprung zum
Modul 35
Interrupt S0 oder S1
Ablaufdiagramm Automatisierung
Serielle Schnittstelle S0 oder S1 Empfang
Modul 35.1
Blatt 350.1

(Abb. 9-54)


Der Unterschied beim Senden zwischen
der Seriellen Schnittstelle S0 und S1
ist sehr gering.
Es wird hier darauf verzichtet, die Unterschiede
zwischen der Steuerung von S0 und S1
detailliert darzustellen !
Aufruf
Modul 35.2
Interrupt S0 oder S1
Neuer Befehlsstring soll versendet werden
Zeichen im Feld 's1_outbuf()' oder 's0_outbuf()' abholen
Befehlsstring komplett
nein
Befehlsstring komplettieren
ja
im HP die Versendung activieren
nein
Befehlsstring fertig Versand
ja
Rücksprung zum
Modul 35
Interrupt S0 oder S1
Ablaufdiagramm Automatisierung
Serielle Schnittstelle S0 oder S1 Senden
Modul 35.2
Blatt 350.2

(Abb. 9-55)

9.4 Details zur Programmierung

9.4.1 Allgemeines

In den folgenden Abschnitten werden einige Details des Programmablaufes dargestellt und explizit verdeutlicht. Auf die Erklärung jedes Details zur Programmierung wird jedoch verzichtet.

Auf der Dokumentations-CDR finden Sie den gesamten Quellcode des C-Projektes. Durch ausreichende Kommentierung können hier weitere Detailinformationen gewonnen werden.

Die Erklärungen in den weiteren Abschnitten sind nicht nach ihrem Umfang oder ihrer Dominanz sortiert, sondern bezugnehmend auf die Gliederung der Programmablaufdiagramme in Kapitel 9.3.

Die Programmauszüge werden den Modulnummern der Programmablaufpläne in Kapitel 9.3 zugeordnet.

9.4.2 Modul 1: Initialisierung der seriellen Schnittstellen

In Modul 1 werden die seriellen Schnittstellen S0 und S1 initialisiert. Der folgende Programmauszug stellt die Initialisierung der seriellen Schnittstelle S0 dar.

```
    //Initialisierung der Seriellen Schnittstelle 0
    void seriell_S0_init (void)
    {
            SM0 = 0;
140         SM1 = 1;
            //im SFR S0CON(98h) sind die Bits 6 und 7 für die ModeEinstellung
            //zuständig. SM0 = 0; SM1 =1 für Mode 1 oder: S0CON = 0xC0h
            REN0 = 1;
            //Receiver Enable Bit im S0CON Byte, ermöglicht seriellen Empfang, wenn
145         //gesetzt. Wird nur durch Software beeinflußt
            BD = 1;
            //im SFR ADCON0(D8h) BD=1 (7Bit) um den Baudratengenerator anzuschalten:
            //muß auf "1" gesetzt werden
            //Nach Reset ist diese Bit auf "0"
150         //Diese Bit ist direkt ansprechbar
            PCON = (PCON & 0x7F);
            //im SFR PCON(87h) ist SMOD das 7Bit,
            //damit kann die Baudrate verdoppelt werden.
            //wenn Bit SMOD: "1" ist, dann wird Baudrate verdoppelt.
155         //Nach Reset ist dieses Bit und das Byte auf "0"
            //Diese Bit ist nicht direkt ansprechbar
            PRSC  = (PRSC | 0x40);
            //im SFR PRSC(B4h) ist S0P das 6Bit, damit wird die Baudrate verdoppelt
            //Nach Reset ist diese Bit auf "1"
160         //Nach Reset ist dieses Byte auf "1101 0101b"
            //Diese Bit ist nicht direkt ansprechbar.
            S0RELH  = 0x03;
            //damit werden die 2 oberen MSBits vom 10Bit Zählregister eingestellt
            //diese sind das 0 und 1 Bit im S0RELH SFR
165         //nach Reset ist dieses Byte auf "0000 0011b" = "0x03h" eingestellt.
            //für 9600Baud muß diese Byte auf "xxxx xx11b" = "0x03h"
            //eingestellt sein.
            S0RELL  = 0xE6;
            //damit werden die 8 unteren LSBits vom 10Bit Zählregister eingestellt.
170         //nach Reset ist dieses Byte auf "1101 1001b" = "0xD9h" eingestellt.
            //für 9600Baud muß diese Byte auf "1110 0110b" = "0xE6h" eingestellt.
            //Die Formel zu Berechung der 10Bits (SOREL)ist:
            //[1024 - ( (16Mhz) / (64*Bdrate) ) ]= "998d"
            //lt. SiemensBuch ist die Formel:
175         //[(2hochSM0D x 16hoch6) / (64 x (1024-S0REL) x 2hochS0P)]
    }
```

Durch ausführliche Kommentierung der einzelnen Befehlszeilen des Quellcodes wird eine weitere Erklärung überflüssig.

Die Initialisierung der seriellen Schnittstellen S0 und S1 erfolgt in der Datei *seriell_olli_seriell.c*. Der Aufruf der entsprechenden Funktionen erfolgt, wie Sie auch aus dem Programmablaufplan entnehmen können, vor dem zyklisch ablaufenden Teil des Programmes.

Die Steuerung der Funktionsaufrufe geschieht hauptsächlich aus der Datei *main.c*. Diese Datei ist der Koordinator der zyklischen Abarbeitung der verschiedenen Programmteile.

Wie die Initialisierung der seriellen Schnittstelle S0 ist auch die Initialisierung der seriellen Schnittstelle S1 durch Kommentierung der einzelnen Befehlszeilen des Quellcodes nachvollziehbar.

```
435 //Initialisierung der Seriellen Schnittstelle 1
    void seriell_S1_init (void)
    {
            S1CON = 0xD0;
            //ist gesetzt auf "1101 0000b" oder "0xD0h"
440         //im SFR S1CON sind folgende Bits:
            //Nach Reset ist diese Byte auf "0100 0000b" oder "0x40h"
            //bit 0 (RI1): Rx(receiver)Interrupt flag
            //bit 1 (TI1): Tx(transmitt)Interrupt flag
            //bit 2 (RB81): 9 tes receiver bit
445         //bit 3 (TB81): 9 tes transmitter bit
            //bit 4 (REN1): wenn 0 dann ist kein Serieller Empfang möglich
            //bit 5 (SM21): Mutiprocessor enable (bleibt immer auf "0")
            //bit 6 (S1P):  wenn 1(nach Reset)prescaler is active
            //              (für Baudraten Generator)
450         //bit 7 (SM):   Mode select bit:0=9bit Uart; 1=8bit Uart
            //Diese Bits in diesem Byte sind nicht direkt ansprechbar !!
            S1RELH  = 0x03;
            //damit werden die 2 oberen MSBits vom 10Bit Zählregister eingestellt
            //diese sind das 0 und 1 Bit im S1RELH SFR
455         //nach Reset ist dieses Byte auf "00000011b" = "0x03h" eingestellt.
            //für 9600Baud muß diese Byte auf "xxxxxx11b" ="0x03h" eingestellt sein
            S1RELL  = 0xCC;
            //damit werden die 8 unteren LSBits vom 10Bit Zählregister eingestellt.
            //nach Reset ist dieses Byte auf "00000000b" = "0x00h" eingestellt.
460         //für 9600Baud muß diese Byte auf "1100 1100b" ="0xCC" eingestellt sein
            //entspricht "972d"
            //Die Formel zu Berechung der 10Bits (S1REL)ist:
            //[ 1 / {(Bdrate*32) / 16hoch6} ] + 1024
            //lt. SiemensBuch ist die Formel:
465         //[ 16hoch6 / (32 x (1024-S1REL) x 2 hoch S1P)]
    }
```

9.4.3 Modul 2: Initialisierung der Ports

Die Initialisierung der Ports geschieht hauptsächlich in der Funktion *main { }* der Datei *main.c*. DO-Ports (digitale Ausgangsports) werden mit einer logischen 0 initialisiert, DI-Ports (digitale Eingangsports) werden mit einer logischen 1 initialisiert.

```
        // Ausgänge rücksetzen

        // Ausgang Motoren auf LOW
        P4=0x00;
145
        // Bit 0,1 setzen; Port 5
        // disable Multiplexer, disable Motorenansteuerung
        P5=0x03;

150     // Port 7 zum Einlesen vorbereiten
        // Eingangsbyte Achsen Endlagenschalter
        P7=0xff;

        // Port 9 zum Einlesen vorbereiten
155     // Referenzpunktschalter P9.0 - P9.3
        P9=0xff;

        // Portl.3 + P1.5 zum Einlesen vorbereiten
        P1=0x18;
160
        // Ausgänge für "Test-LEDs" rücksetzen
        P6 = (P6 & 0x0f);

        // ---------------------------------------------------------------------
165     // Aufruf Interruptinitialisierung
        interrupt_init();

        // ---------------------------------------------------------------------
        // Überwachung auf angefahrene Endlage ist mit INT4, (in C Nr.11) realisiert
170
        // Überwachung auf Motorsteuer-IC Überlast ist mit INT5 (in C 12) realisiert

        // ---------------------------------------------------------------------
        // Initialisierung Umdrehungszähler Achsen 1-4, Initialisierung Counter
175     umdr_zaehler_init_a1();
        umdr_zaehler_init_a2();
        umdr_zaehler_init_a3();
        umdr_zaehler_init_a4();

180     // ---------------------------------------------------------------------
        // Aufruf Motorenansteuerung enable
        mot_pwr_enable();
```

Nach der Initialisierung der Ports erfolgen verschiedene Funktionsaufrufe zur Initialisierung diverser Interrupts und der Umdrehungszähler.

Details hierzu können Sie Kapitel 9.4.4 und Kapitel 9.4.5 entnehmen.

9.4.4 Modul 3: Initialisierung der Interrupts

In Modul 3 ist die Initialisierung der Interrupts realisiert. Ein Auszug der Datei *intrrpt_init.c* stellt dies dar.

```
    // Interruptinitialisierung
    void interrupt_init (void)
    {
30      // Interruptnr. 1 (Port 3.4, Countereingang Achse 1-3)
        // Überlauf Counter Geberinkremente
        ET0=1;

        // Interrupt-Request-Bit ist TF0 und wird von Timer0 gesetzt
35      TF0=0;

        // ---------------------------------------------------------------------
        // Interruptnr. 3 (C), Port 3.5, Countereingang Achse 4
        ET1=1;
40      TF1=0;

        // ---------------------------------------------------------------------
        // Interrupt 4, Interruptnr. 11, Port 1.1, Endlage einer Achse angefahren
        IEX4=0;
45      EX4=1;

        // ---------------------------------------------------------------------
        // Interrupt 5, Interruptnr. 12, Port 1.2, MotorsteuerIC - Overload/Überlast
        IEX5=0;
50      EX5=1;
```

Die Nummern der Interrupts sind diejenigen, die unter Verwendung der Programmiersprache C unter EDE 8051 verwendet werden müssen.

Folgende Interrupts werden verwendet:

Interruptnummer 1

- Countereingang der Achsen 1-3 an Port 3.4
- Überlauf des Counters der Geberinkremente

Interruptnummer 3

- Countereingang der Achse 4

Interruptnummer 4

- Auswertung Endlagenstörung der Antriebsachsen

Interruptnummer 5

- Überlast der Motorsteuer-ICs

Im Folgenden die Initialisierung der Interrupts der seriellen Schnittstellen S0 und S1.

```
        //
        // Serielle Schnittstelle 0
        TI0 = 0;
55      // Das Transmit Interrupt Request Flag wird gelöscht.
        // (TI0 wird nach erfolgter Sendung über S0 gesetzt).
        // TI0 wird automatisch gesetzt muß aber manuell zurückgesetzt werden.
        RI0 = 0;
        // Das Receive Interrupt Request Flag wird gelöscht.
60      // (RI0 wird nach einem Empfang über S0 gesetzt).
        // RI0 wird automatisch gesetzt muß aber manuell zurückgesetzt werden.
        ES0 = 1;
        // Interrupt Enable fuer S0 wird eingeschaltet  (altern: SET_ES0)
        // Prioritätstufe sollte 2 sein (zweit Höchste)
65
        // ---------------------------------------------------------------------
        // Serielle Schnittstelle 1
        S1CON = (S1CON & 0xFC);
        // Interrupt Request Flags der S1 werden gelöscht
70      // Maskierung von Bit 0 und Bit 1 (auf "0" setzten)
        // bit 0 in S1CON ist: (RI1): Rx(receiver)Interrupt Request Flag
        // bit 1 in S1CON ist: (TI1): Tx(transmit)Interrupt Request Flag
        IEN2 = (IEN2 | 0x01);
        // Interrupt Enable fuer S1 wird eingeschaltet
75      // Im SFR IEN2 ist das Bit0 ES1.
        // Prioritätstufe sollte 2 sein (zweit Höchste)
        // Prioritaets Stufen Einstellungen
        // IP generelle Einstellungen    543210b
```

Der Initialisierung inbegriffen, ist die Festlegung der Prioritäten. Diese werden über die SFR (Special Function Register) IP0 und IP1 festgelegt.

Im Anschluss erfolgt die generelle Freigabe aller Interrupts über das Setzen von EAL.

```
80      // ---------------------------------------------------------------------
        // Prioritäten einstellen
        IP0=0x0A;    // 00001010
        IP1=0x9B;    // 10011011

85      // ---------------------------------------------------------------------
        // Intrptnr.    Funktion                  Priorität   Bit in IP0/IP1
        // 16           serielle Schnittstelle    2           IP1.0/IP0.0
        // 1            Impulszählung Geber       3           IP1.1/IP0.1
        // 3            Impulszählung Geber       3           IP1.3/IP0.3
90      // 11           Endlagenüberwachung       3           -----"-----
        // 4            serielle Schnittstelle    2           IP1.4/IP0.4
        // 12           Motoren Überlast          2           -----"-----
        // 13           Deckelüberwachung         0           IP1.5/IP0.5
        // ---------------------------------------------------------------------
95      // um P9 einlesen zu können muss IP1.7 mit 1 beschrieben werden

        // alle Interrupts enable
        EAL=1;
    }
```

Detailliertere Informationen können Sie dem Kommentar des Quellcodes entnehmen.

9.4.5 Modul 4: Initialisierung der Umdrehungszähler

Die Lageregelung der Antriebseinheiten wird mit Inkrementalgebern realisiert. Diese Inkrementalgeber sind motorseitig montiert und liefern pro Umdrehung 16 Impulse. Jeder Impuls löst einen Interrupt aus. Die Achsen 1-3 arbeiten mit dem Interrupt Nummer 1, Achse 4 arbeitet mit Interrupt Nummer 3.

Aus der Auflösung des Gebers von 16 Impulsen pro Umdrehung folgt, daß 16 Impulse eine Motorumdrehung ergeben müssen. Somit wird alle 16 Impulse der Umdrehungszähler der entsprechenden Achse inkrementiert.

Aus der Getriebeübersetzung von 23:1 ergibt sich, daß alle 23 Motorumdrehungen der Umdrehungszähler der entsprechenden Spindel inkrementiert wird.

Diese Umdrehungszähler werden jedes Mal initialisiert, wenn die Antriebseinheit auf Referenzpunkt fährt. Die Grundstellung einer Achse ist ebenso Referenzpunktstellung.

Mit diesem Verfahren werden Positionierungenauigkeiten durch z.B. Umkehrlose oder mechanischen Verschleiß kompensiert.

Theoretische Positionierungenauigkeiten würden auch bei Drehrichtungsumkehr einer Achse entstehen, wenn die Drehrichtungsumkehr bei einer nicht vollendeter Motorumdrehung eingeleitet würde. Diese Abweichung würde aber weniger als 0,04 mm Weg am Schlitten der Antriebseinheit betragen. Um aber auch dies auszuschließen, findet eine Counterinitialisierung statt.

```
// ------------------------------------------------------------------------
// Counterinitialisierung
void counter_init (void)
{

    TR0 = 1;        // Timer 0 einschalten in SFR->TCON
    TR1 = 1;        // Timer 1 einschalten in SFR->TCON


    TMOD = 0x66;    // 0110 0110b in TMOD
                    // Timer 0 und Timer 1 in Counterbetrieb und Mode 2

    TH0 = 0xF0;     // Timer 0: reload-Register setzen (auf Wert 240d)
    TH1 = 0xF0;     // Timer 1: reload-Register setzen (auf Wert 240d)


    TL0 = 0xF0;     // Timer 0:
                    // Zählregister einmalig auf Startwert setzen (240d)

    TL1 = 0xF0;     // Timer 1:
                    // Zählregister einmalig auf Startwert setzen (240d)

    // ------------------------------------------------------------------------
    // Weitere Informationen:
    // Geber gibt 16 Impulse/Umdrehung, deshalb 255-16=239.
    // Beim 16. Impuls ist eine Umdrehung vollständig -> 239+1=240
    // Umdrehungszähler Motor +1
    // Somit eine Interruptauslösung pro Motorumdrehung
    // -> mit Interrupt Umdrehungszähler aktualisieren
    // ------------------------------------------------------------------------
}
```

Nach Initialisierung der Variablen für die Aktualwerte der Umdrehungszähler in der Datei *umdrehungszaehler.c* wird die Funktion *counter_init { }* in der Datei *counter_init.c* aufgerufen.

In dieser Funktion werden Timer 0 und Timer 1 eingeschaltet durch Setzen von TR0 und TR1 im SFR TCON. Timer 0 und Timer 1 arbeiten im Counterbetrieb Mode 2.

Die Zählregister von Timer 0 und Timer 1 werden mit dem Startwert 240d geladen. Nach 16 Impulsen des Inkrementalgebers ist eine Motorumdrehung vollständig und es findet ein Überlauf des Counters statt. Durch diesen Counterüberlauf wird ein Interrupt ausgelöst, welcher die Aktualwerte der entsprechenden Umdrehungszähler inkrementiert.

```
    // ------------------------------------------------------------------------
    // Umdrehungszähler Achse 1 aktualisieren
    // wird aufgerufen durch Interrupt 1 aus Überlauf Counter 0
105 void count_umdr_a1 (void)
    {
            // Umdrehungszähler Motor inkrementieren
            // jeden 16. Impuls wird Umdrehungszähler Motor akt.
            m_umdr_a1++;
110         m_umdr_hilfsm_a1++;

            // Umdrehungszähler Spindel inkrementieren
            // alle GEAR_N Motorumdrehungen wird Umdrehungszähler Spindel aktualisiert.
            // Hilfsmerker Umdrehungszähler Motor wird gelöscht.
115         if (m_umdr_hilfsm_a1 == GEAR_N_23 )
            {
                s_umdr_a1++;
                m_umdr_hilfsm_a1=0;
            }
120 }
```

Weitere Informationen

Nicht zu diesem Modul gehörend, aber der Vollständigkeit und dem besseren Verständnis wegen, wird hier noch verdeutlicht, wie die Umsetzung zum Erlangen der Aktualwerte der Umdrehungszähler für die Spindeln realisiert ist.

Nach jedem Inkrementieren der Motorumdrehungszähler wird überprüft, ob eine komplette Spindelumdrehung erreicht wurde. Das Übersetzungsverhältnis des Getriebes ist als DEFINE-Wert in der Headerdatei *umdrehungszaehler.h* hinterlegt.

Eine Änderung der Getriebeübersetzung bringt somit keinen großen Programmieraufwand mit sich. Es muß lediglich der DEFINE-Wert in *umdrehungszaehler.h* an die entsprechende Getriebeübersetzung angepasst werden.

9.4.6 Modul 5: Motoren Leistung ein

In diesem Modul wird aus der Datei *main.c* die Funktion *mot_pwr_enable { }* in der Datei *mot_outp.c* aufgerufen.

Hier werden die Motorsteuer-ICs generell eingeschaltet. Zu beachten ist allerdings, daß ein enable-Signal mit einer logischen 0 erreicht wird.

9.4.7 Modul 6: LCD rücksetzen

Dieses Modul wird aus der Hauptroutine angestoßen. Als Hauptroutine wird hier eine while (1) - Schleife bezeichnet.

Zuerst wird überprüft, ob die Hauptroutine nach dem Einschalten der Netzversorgung oder dem Auslösen eines Neustarts des Mikrocontrollers das erste Mal durchlaufen wird. Dies wird mit Hilfe eines flags „first_turn_lcd“ realisiert.

Als Sicherheit werden weitere Bedingungen mit diesem flag verknüpft.

```
        // Hauptroutine
        while(1)
        {
            // ------------------------------------------------------------------
190         // nach Reset der Steuerung Display zurücksetzen
            // Makro 1 aufrufen
            if((!first_turn_lcd) || ((!P1_3) && (!m_all_ax_bb) && (err_neig_ntact == 1)))
            {
                befehl_s1_in[1] = 0;
195             transmit_S1_vorbereitung('N',0x01);
                first_turn_lcd=1;
            }
```

Ist diese Bedingung erfüllt, wird über die serielle Schnittstelle S1 das Normalmakro 1 des LCD-Displays aufgerufen. Dies hat einen Neustart der HMI zur Folge.

Weitere Informationen zur Programmierung der HMI erhalten Sie in den entsprechenden Kapiteln. Zum besseren Verständnis sollten diese Kapitel eventuell parallel zu Kapitel 9.4 eingesehen werden. Dort erhalten Sie u. A. Informationen zur Programmierung mit Makros der HMI.

9.4.8 Modul 7: Überprüfung Deckelschalter

Die Überprüfung des Deckelschalters wird bei jedem Zyklus der Hauptroutine durchgeführt. Über den Aufruf der Funktion *check_hut { }* wird diese Überprüfung gestartet.

Die Funktion *check_hut { }* finden Sie in der Datei *input.c*.

Im oberen Rahmen des Messautomaten finden Sie den Reedkontakt mit der Bezeichnung SE1.5. Die Bezeichnung ist normgerecht gewählt und verweist auf den Eingang 1.5 der Steuerung.

In diesem Fall entspricht der Eingang 1.5 der Steuerung, dem Portpin 1.5 des Mikrocontrollers. Dieser Portpin ist ein digitaler Eingang DI.

Im Deckel des Messautomaten sind zwei Magnete eingearbeitet, die auch ein versehentlich falsches Aufsetzen des Deckels auf den Automaten erkennen.

Eine entsprechende Visualisierung erfolgt über die HMI.

Ist der Eingang 1.5 des Mikrocontrollers auf logisch 0, wird ein Fehlerbit gesetzt. Über dieses Fehlerbit wird der Aufruf zur Fehlerdiagnose und den entsprechenden Reaktionen auf den Fehler realisiert.

```
//
// Deckelschalter überprüfen
void check_hut (void)
{
    if(P1_5 == 0)
    {
        err_hut=1;
    }
    else
    {
        err_hut=0;
    }

    if(err_hut == 1)
    {
        error_without_hut();
    }

    if((err_number == 15) && (err_qtt == 1))
    {
        // Fehlernummer rücksetzen
        err_number=0;

        // Anforderung Störung quittieren rücksetzen
        err_qtt=0;

        // Motoransteuerungen enable
        mot_pwr_enable();

        first_turn_hut=0;
        first_turn_err=0;
    }
}
```

Die Fehlerauswertung findet in der Datei *error_1.c* statt. Die entsprechenden Reaktionen auf die analysierten Fehler werden dort ebenso ausgelöst.

Ist in diesem Fall die Fehlernummer 15 aktiv, wird bei einer Anforderung zur Quittierung der Störung, die Fehlernummer zurückgesetzt, die Anforderung zur Quittierung wird zurückgesetzt und der Leistungsteil der Hauptplatine zur Ansteuerung der Antriebe wird wieder aktiviert.

Detailliertere Informationen zur Fehlerdiagnose erhalten Sie in den folgenden Kapiteln.

9.4.9 Modul 8: Überprüfung Neigungssensor

Die Überprüfung des Neigungssensors wird bei jedem Zyklus der Hauptroutine durchgeführt. Über den Aufruf der Funktion *check_neigung { }* wird diese Überprüfung gestartet.

Die Funktion *check_neigung { }* finden Sie in der Datei *input.c*.

```
//-------------------------------------------------------------------------
// Neigungssensor überprüfen
void check_neigung(void)
{
    if(P1_3 == 1)
    {
        lage_nio=1;
        error_switch_drop();
        err_neig_ntact=1;
    }

    if(!P1_3)
    {
        lage_nio=0;

        if(err_neig_ntact == 1)
        {
            err_neig_ntact=0;
        }
    }
}
```

Der Neigungssensor ist mit SE 1.3 bezeichnet. Er liegt somit an Portpin 1.3 des Mikrocontrollers.

Ist der DI 1.3 auf logisch 1, wird ein Fehlerbit gesetzt und die Diagnose in der Datei *error_1.c* gestartet.

In der Funktion *check_neigung { }* werden zusätzlich noch mehrere flags wie „err_neig_ntact“ (Fehler Neigungssensor nicht aktiv) oder „lage_nio“ (Lage des Messautomaten nicht in Ordnung) verwaltet.

Die Reaktion auf das Ansprechen des Neigungssensors wird in der Datei *error_1.c* ausgelöst.

9.4.10 Modul 9: Referenzpunktschalter einlesen

Das Einlesen der Referenzpunktschalter wird gestartet über den Aufruf der Funktion *check_refp { }* in der Datei *main.c.*

Das Einlesen der Referenzpunktschalter erfolgt zyklisch bei jedem Durchlauf der Hauptroutine.

Die Funktion *check_refp { }* finden Sie in der Datei *input_1.c.*

Die Referenzpunktschalter liegen, wie Sie auch dem Pinbelegungsplan in Kapitel 8.2 oder dem Schaltplan der Hauptplatine bzw. dem Stromlaufplan des Messautomaten entnehmen können, an den Portpins 9.0 bis 9.3.

Der Referenzpunktschalter der Achse 1 ist mit SE 9.0, der Referenzpunktschalter der Achse 2 mit SE 9.1, der Referenzpunktschalter an Achse 3 mit SE 9.2 und der Referenzpunktschalter für Achse 4 ist mit SE 9.3 bezeichnet.

Da Port 9 des Mikrocontrollers nicht bitadressierbar ist, gestaltet sich das Auswerten der einzelnen Referenzpunktschalter etwas umfangreicher.

```
//-------------------------------------------------------------------------
// Refpunktschalter aus P9 vereinzeln und in Speicher schreiben (bitadressierbar
// machen)
// Diese Kacke ist notwendig, da P9 nicht bitadressierbar ist
void check_refp (void)
{
    // Refpunktschalter an P9 einlesen
    a1234_refp=P9;

    // Refp Achse 1 ausmaskieren
    tmp=(a1234_refp & 0x01);

    // Zustand Refpschalter Achse 1 in Speicher schreiben
    if(tmp == 0x01)
    {
        refp[0]=1;
    }
    else
    {
        refp[0]=0;
    }
```

Zuerst wird Port 9 eingelesen und das PAE (Prozessabbild der Eingänge) in eine Variable geschrieben.

Das Bit des Referenzpunktschalters der Achse 1, DI 9.0, wird ausmaskiert und das Bitmuster in eine temporäre Variable geschrieben.

Ist das Bitmuster der temporären Variablen 0000 0001, wird der logische Zustand 1 des Referenzpunktschalters der Achse 1 in das array *refp[]* geschrieben.

Bei jedem anderem Bitmuster der temporären Variablen, wird im array *refp[]* der logische Zustand 0 des Referenzpunktschalters der Achse 1 gespeichert.

Das array *refp[]* wird wie folgt genutzt :

- der Zustand des Referenzpunktschalters der Achse 1 wird abgelegt in refp[0].
- der Zustand des Referenzpunktschalters der Achse 2 wird abgelegt in refp[1].
- der Zustand des Referenzpunktschalters der Achse 3 wird abgelegt in refp[2].
- der Zustand des Referenzpunktschalters der Achse 4 wird abgelegt in refp[3].

Nach dem speichern des PAE und der Aktualisierung des Zustandes von DI 9.0 in refp[0], wird die Aktualisierung des Zustandes für den Referenzpunktschalter der Achse 2 in refp[1] vorgenommen.

```
    // Refp Achse 2 ausmaskieren
    tmp=(a1234_refp & 0x02);

    // Zustand Refpschalter Achse 2 in Speicher schreiben
    if(tmp == 0x02)
    {
        refp[1]=1;
    }
    else
    {
        refp[1]=0;
    }
```

Anschließend wird mit dem aktualisieren der Schaltzustände der Referenzpunktschalter von Achse 3 und 4 verfahren, wie bereits bei den Achsen 1 und 2 geschehen.

```
        // Refp Achse 3 ausmaskieren
        tmp=(a1234_refp & 0x04);

        // Zustand Refpschalter Achse 3 in Speicher schreiben
 95     if(tmp == 0x04)
        {
            refp[2]=1;
        }
        else
100     {
            refp[2]=0;
        }


105     // Refp Achse 4 ausmaskieren
        tmp=(a1234_refp & 0x08);

        // Zustand Refpschalter Achse 4 in Speicher schreiben
        if(tmp == 0x08)
110     {
            refp[3]=1;
        }
        else
        {
115         refp[3]=0;
        }

    }
```

9.4.11 Modul 10: Referenzpunktfahrt

In der Hauptroutine in *main.c* wird überprüft, ob eine Referenzpunktfahrt einzuleiten ist. Ist der Merker „m_all_ax_bb“ (alle Achsen betriebsbereit) im Zustand logisch 0, wird eine Referenzpunktfahrt eingeleitet.

Um zu verhindern, daß der Messautomat während des Aufbauens zur Messung durch Verbinden mit der Versorgungsspannung eine Referenzpunktfahrt durchführt, ist diese Anforderung zusätzlich mit der Bedingung verbunden, daß der Neigungssensor den Messautomat in Arbeitsstellung lokalisiert.

```
        // ------------------------------------------------------------------
        // Wenn noch nicht alle Achsen betriebsbereit sind, Referenzpunktfahrt
        // anfordern in grundstellung_1.c
        if((!m_all_ax_bb) && (!lage_nio))
215     {
            // Aufruf Referenzpunktfahrt
            referieren_haupt();
        }
```

Ist diese Bedingung erfüllt, wird der Aufruf zur Referenzpunktfahrt in *grundstellung_1.c* ausgelöst.

In der Funktion *referieren_haupt { }* wird zuerst überprüft, welche Achsen eine Referenzpunktfahrt durchzuführen haben.

Eine Referenzpunktfahrt ist notwendig, um eine eindeutige mechanische Position zu einem Bezugspunkt herzustellen. Die Positionierungen der Achsen werden in Bezug auf diesen Referenzpunkt hergestellt.

Für die Referenzpunktfahrt ist außerdem entscheidend, ob die Achse, bzw. der Schlitten der Antriebseinheit bereits den entsprechenden Referenzpunktschalter bedämpft hat.

Entsprechend dieser Überprüfung wird eine von zwei möglichen Arten zum Justieren der Antriebseinheit gewählt.

Möglichkeit 1:

Der Schlitten der Antriebseinheit hat den Referenzpunktschalter bereits bedämpft.

Justieren 1 wird eingeleitet.

- Der Schlitten wird um den DEFINE – Wert S_U_SOLL in *grundstellung_1.h* (zum Zeitpunkt des Schreibens der Dokumentation entspricht S_U_SOLL dem Wert 5) Spindelumdrehungen in Richtung (+) verfahren.
- Nach dem Erreichen dieser Position läuft eine kurze Wartezeit ab, bevor die Richtungsumkehr eingeleitet wird. Diese Wartezeit ist notwendig, um die Überlastüberwachung durch die Motorsteuer-ICs nicht ansprechen zu lassen. Die Wartezeit entspricht etwa 0,2 s.
- Der Schlitten der Antriebseinheit wird nun in Richtung (-) bis zum Erreichen des Referenzpunktschalters bewegt.
- Beim Erreichen des Referenzpunktschalters wird die Achse als betriebsbereit deklariert.

```
    //------------------------------------------------------------------------
    // Achse 1

115     if(((refp[0] == true)
            &&(!referiert[0])
            &&(!flag_ref2_a1))
            ||(flag_ref1_a1 == true))
        {referieren_1_a1(s_umdr_a1);}
120
        if( (!referiert[0])
            &&(!flag_ref1_a1))
        {referieren_2_a1();}
```

Möglichkeit 2:

Der Schlitten der Antriebseinheit hat den Referenzpunktschalter zum Zeitpunkt der Anforderung der Referenzpunktfahrt nicht bedämpft.

Justieren 2 wird eingeleitet.

- Der Schlitten der Antriebseinheit wird in Richtung (-) bis zum Erreichen des Referenzpunktschalters bewegt.
- Beim Erreichen des Referenzpunktschalters wird die Achse als betriebsbereit deklariert

Beim Referieren wird grundsätzlich eine Reihenfolge eingehalten, in der die Achsen referiert werden.

Reihenfolge der Achsen beim Referieren:

- Achse 1
- Achse 2
- Achse 3
- Achse 4

Um diese Reihenfolge einhalten zu können, sind mehrere Bedingungen abzufragen. Die Funktion *referieren_haupt { }* stellt somit eine Sprungtabelle dar.

```
    // Achse 2

        if(((refp[1] == true)
            &&(!referiert[1])
130         &&(!flag_ref2_a2)
            &&(referiert[0] == true))
            ||(flag_ref1_a2 == true))
        {referieren_1_a2(s_umdr_a2);}

135     if((!referiert[1])
            &&(!flag_ref1_a2)
            &&(referiert[0]))
        {referieren_2_a2();}

140 //---------------------------------------------------------------------------
    // Achse 3

        if(((refp[2] == true)
            &&(!referiert[2])
145         &&(!flag_ref2_a3)
            &&(referiert[0] == true)
            &&(referiert[1] == true))
            ||(flag_ref1_a3))
        {referieren_1_a3(s_umdr_a3);}
150
        if((!referiert[2])
            &&(!flag_ref1_a3)
            &&(referiert[0] == true)
            &&(referiert[1] == true))
155     {referieren_2_a3();}

    //---------------------------------------------------------------------------
    // Achse 4

160     if(((refp[3] == true)
            &&(!referiert[3])
            &&(!flag_ref2_a4)
            &&(referiert[0] == true)
            &&(referiert[1] == true)
165         &&(referiert[2] == true))
            ||(flag_ref1_a4))
        {referieren_1_a4(s_umdr_a4);}

        if((!referiert[3])
170         &&(!flag_ref1_a4)
            &&(referiert[0] == true)
            &&(referiert[1] == true)
            &&(referiert[2] == true))
        {referieren_2_a4();}
175
```

Aus dem Programmauszug ist ersichtlich, daß eine Achse nur die Freigabe zur Referenzpunktfahrt erhält, wenn die Achsen, die in der Sprungtabelle vorrangig sind, bereits als referiert deklariert sind.

Um die Achsen als referiert zu deklarieren, werden folgende Einträge in das eindimensionale array *referiert []* notwendig:

- logisch 1 in referiert [0] : Achse 1 ist referiert.
- logisch 1 in referiert [1] : Achse 2 ist referiert.
- logisch 1 in referiert [2] : Achse 3 ist referiert.
- logisch 1 in referiert [3] : Achse 4 ist referiert.

Ein Programmauszug zur Justage 1 der Antriebseinheit 1. Nachfolgend die einzelnen Schritte des Ablaufes.

```
    // ------------------------------------------------------------------------
185 void referieren_1_a1 (signed char s_umdr_a1)
    {
        // Wenn Motor A1 R+ noch nicht läuft, dann einschalten
        // Geberdatenkanal wird autom. über mot_a1_plus() ausgewählt
        if ((!m_mot_a1_plus) && (!m_mot_a1_minus))
190     {
            // Motor A1 Richtung+ ein
            mot_a1_plus();
        }

195     // Wenn Spindelumdrehungen = Sollwert, dann Motor stop,
        // anschließend Motor A1 R-
        if (s_umdr_a1 == S_U_SOLL)
        {
            mot_a1_stp();
200
            if (change_pm_a1 == 0)
            {
                change_pm_a1=1;
                ax_wait();
205         }

            // Motor A1 Richtung- ein
            mot_a1_minus();
        }
210
        // Wenn Referenzpunktschalter angefahren ist, dann Bit für Achse "ist
        // referiert" setzen und Motor ausschalten
        if ((refp[0] == 1) && (m_mot_a1_minus))
        {
215         // Achse 1 ist referiert
            referiert[0]=1;

            // Flag "referieren_1_Achse_1_läuft" rücksetzen
            flag_ref1_a1=0;
220
            // Achse 1 ist betriebsbereit setzen
            ax_bb[0]=1;

            // Motor A1 ausschalten
225         mot_a1_stp();

            // alle Zähler rücksetzen
            umdr_zaehler_init_a1();
        }
230     else
        {
            // Flag "referieren_1_Achse_1_läuft" setzen
            flag_ref1_a1=1;
        }
235 }
```

- Wenn der Antrieb noch nicht in Bewegung ist, dann wird der Antriebsmotor in Richtung (+) eingeschaltet.
- Der Istwert des Spindelumdrehungszählers wird mit dem vorgegebenen Sollwert verglichen.
- Ist der Sollwert erreicht, wird der Motor gestoppt.
- Überprüfung, ob Wartezeit zur Drehrichtungsumkehr bereits angefordert wurde.
- Nach Ablauf der Wartezeit wird der Motor in Richtung (-) eingeschaltet.
- Ist der Motor in Richtung (-) eingeschaltet und der Referenzpunktschalter wird erreicht, dann wird die Achse als referiert deklariert.
- Flag : Justage 1 - Achse 1 aktiv, wird zurückgesetzt
- Achse 1 wird betriebsbereit gesetzt
- Motor Achse 1 wird gestoppt
- Umdrehungszähler der Achse 1 wird initialisiert

Wie dem nachfolgenden Programmauszug zu entnehmen ist, ist die Justage 2 weniger umfangreich als das Verfahren Justage 1.

```
    // ---------------------------------------------------------------------------
    void referieren_2_a1 (void)
    {
240         // falls Motor in R- noch nicht läuft, einschalten
            if(m_mot_a1_minus == 0)
            {
                // Motor A1 Richtung- ein
                mot_a1_minus();
245         }


            // Wenn Referenzpunktschalter angefahren ist, dann Bit für Achse ist
            // referiert setzen und Motor ausschalten
250         if (refp[0] == 1)
            {
                referiert[0]=1;

                // Achse 1 ist betriebsbereit setzen
255             ax_bb[0]=1;
                flag_ref2_a1=0;
                mot_a1_stp();

                // alle Zähler rücksetzen
260             umdr_zaehler_init_a1();
            }

            // flag setzen, daß nicht referieren1_a1 angewählt wird
            // rücksetzen von flag_ref2_a1 erst, wenn referieren 1/2 _a2 läuft
265         flag_ref2_a1=1;
    }

    // ---------------------------------------------------------------------------
```

Auf eine Erklärung der einzelnen Schritte wird verzichtet, diese sind ähnlich dem Verfahren der Justage 1 und können deshalb der Erklärung zu Justage 1 entnommen werden.

Eine ausreichende Kommentierung der einzelnen Befehlszeilen wurde ebenso vorgenommen.

Diese Programmblöcke wiederholen sich natürlich für Achse 2, Achse 3 und Achse 4 entsprechend.

```
    // Merkerbit für "alle Achsen sind referiert" setzen + rücksetzen
    // Merkerbit für "alle Achsen betriebsbereit" setzen + rücksetzen
485 void all_ax_bb_ref(void)
    {
        if  ((referiert[0])
            &&(referiert[1])
            &&(referiert[2])
490         &&(referiert[3]))
        {
            m_all_ax_ref=1;
            change_pm_a1=0;
            change_pm_a2=0;
495         change_pm_a3=0;
            change_pm_a4=0;
        }

        if  ((m_all_ax_ref)
500         &&(ax_bb[0])
            &&(ax_bb[1])
            &&(ax_bb[2])
            &&(ax_bb[3]))
        {
505         m_all_ax_bb=1;

            // Bild Betriebsartenwahl aufrufen
            transmit_S1_vorbereitung('N',0x03);
        }
510
    }

    //---------------------------------------------------------------------------
    // Wartezeit für Richtungsumkehr der Spindel bei referieren_1 zur
515 // Strombegrenzung
    void ax_wait(void)
    {
        for(ax_wait_time=0;ax_wait_time<=20000;ax_wait_time++)
        {}
520 }
```

Nach dem Referieren werden Merker aktualisiert, die zur weiteren Verwendung in anderen Programmabschnitten zur Verfügung stehen.

In der Funktion *ax_wait { }* ist die erforderliche Wartezeit zur Strombegrenzung bei Drehrichtungsumkehr der Spindel programmiert.

9.4.12 Modul 11: Fehlergenerierung

Das System der Fehlergenerierung beruht auf drei Schritten:

- Das Erkennen des Fehlers
- Das Erzeugen einer Fehlernummer
- Die auszulösende Reaktion auf den anstehenden Fehler

Grundsätzlich ist dieses Modul zur Fehlergenerierung einfach gestaltet. Die einzelnen Funktionen, die entweder zyklisch oder durch Interrupts aufgerufen werden, überprüfen durch einlesen von Ports oder durch auslesen von Werten in Variablen, den Messautomat auf Fehlerzustände.

Ein Programmauszug zur Generierung der Fehlernummer 1 und Fehlernummer 24:

```
    // ------------------------------------------------------------------------
    // Fehlernummer: 1 + 24
    // Überprüfen, ob Olli umgefallen ist
    // Wenn Neigungsschalter an P1.3=true, dann wird error_break() aufgerufen
105 // und Fehlernummer übergeben.
    void error_switch_drop(void)
    {
        // Fehlernummer 1:
        // Wird gebildet, wenn alle Achsen referiert sind und Neigungssensor in
110     // BA Auto oder BA TeilAuto anspricht
        if  ((m_all_ax_ref == 1) && (first_turn_neig == 0)
            &&((m_ba_auto == 1) || (m_ba_teilauto == 1)))
        {
            err_number = 1;
115         error_break(err_number);
        }


        // Fehlernummer 24:
120     // Wird nur gebildet, um eine Warnung auf dem LCD auszugeben, falls Olli
        // beim Aufbau zur Messung mit Spannung versorgt wird aber noch nicht in
        // Arbeitsposition steht
        // Fehlernummer ist selbstquittierend
        if((!m_all_ax_ref) && (m_ba_auto))
125     {
            err_number = 24;
            error_break(err_number);
        }
    }
130
```

Fehlernummer 1 wird generiert, wenn der Messautomat in Betriebsart Automatik oder Betriebsart Teilautomatik in unzulässigen Neigungswinkeln verfährt. Der zulässige Arbeitsbereich wird durch einen Neigungssensor überprüft.

Verlässt der Messautomat seinen zulässigen Arbeitsbereich, so wird in die Variable „err_number“ der Wert 1, also die Fehlernummer 1 eingetragen.

Die Variable „err_number“, die nun die Fehlernummer enthält, wird an die Funktion *error_break { }* übergeben.

Dieses Verfahren wird für die Generierung aller Fehler im Projekt angewendet. Kernstück dieser Fehlergenerierung ist die Datei *error_1.c*.

Nach Übergabe der Fehlernummer an die Funktion *error_break { }* wird die Reaktion auf die entsprechende Fehlernummer vorbereitet und anschließend ausgeführt.

Die Reaktion auf die ausgewertete Fehlernummer ist entsprechend dem Fehler angepasst. So wird als Reaktion auf einen erkannten Fehler nur eine Meldung auf der HMI ausgegeben oder es werden weitere Funktionen, die zur Reaktion auf die generierte Fehlernummer benötigt werden, aufgerufen.

```
        // ------------------------------------------------------------------------
        // Paarfehler, Achse 4
        if (number == 5)
        {
425         // LCD-Bild aufrufen
            // Makro 23
            b1='N';
            b2=0x17;
            transmit_S1_vorbereitung(b1,b2);
430     }

        // ------------------------------------------------------------------------
        // Achse 1 Endlage -
        if (number == 6)
435     {
            // Interrupt 4 (Endlagenüberwachung) ausschalten
            EX4=0;

            // Ausgänge Motoren zurücksetzen
440         mot_al_stp();

            // Motoransteuerungen disable
            mot_pwr_disable();

445         // Variable initialisieren
            init_var();

            // Ausgänge initialisieren
            init_outp();
450
            // LCD-Bild aufrufen
            // Makro 28
            b1='N';
            b2=0x1C;
455         transmit_S1_vorbereitung(b1,b2);

        }
```

Im Programmauszug wird die Reaktion auf die Fehlernummer 5 und Fehlernummer 6 dargestellt.

Fehlernummer 5 wurde erzeugt durch einen Paarfehler der Endlagenüberwachungen von Achse 4 und löst als Reaktion einen Aufruf des Makros 23 der HMI aus.

Fehlernummer 6 wurde erzeugt durch einen erkannten Endlagenfehler an Achse 1. Der Schlitten der Antriebseinheit 1 bedämpfte den Endlagenschalter Richtung (-) der Achse 1.

- Als Reaktion auf diesen erkannten Fehler wird die Überwachung zuerst deaktiviert. Dieser Vorgang lässt ein späteres Freifahren der Achse zu.
- Der Antriebsmotor der Achse 1 wird stillgesetzt.
- Alle Achsbewegungen werden durch Abschalten des Leistungsteils verriegelt.
- Entsprechende Variablen werden initialisiert, um flags von aktiven Positionierbewegungen rückzusetzen.
- Ausgänge werden initialisiert, um selbsttätiges Anlaufen eines Antriebes bei Wiederkehr der Freigabe des Leistungsteils zu verhindern.
- Eine Meldung zur Bedienerführung wird auf der HMI durch Aufruf des Makros 28 angefordert.

Bei allen erzeugten Fehlern wird ebenso dem Tool OLLICONTROL (OC) die Freigabe zur Bedienung des Messautomaten entzogen.

Dies wird selbstverständlich auf OC als zu quittierende Meldung angezeigt.

```
        // Bei Störung Freigabe von Olli-Control entziehen
795     if(err_number!=0)
        {
            error_activ=1;

            if((error_activ == 1) && (first_turn_err == 0))
800         {
                olli_control_disable();
                first_turn_err=1;
            }
        }
```

Nach Beseitigung und Quittierung des Fehlers über die HMI des Messautomaten wird OC, unter Berücksichtigung weiterer Bedingungen, die Freigabe zum Bedienen des Messautomaten wieder erteilt.

```
    // ------------------------------------------------------------------------
// tool Olli-Control Freigabe erteilen
    void olli_control_enable(void)
    {
        transmit_S0_vorbereitung(8);
    }
    // ------------------------------------------------------------------------
    // tool Olli-Control Freigabe entziehen
    void olli_control_disable(void)
    {
        transmit_S0_vorbereitung(7);
    }
```

Für die Aktivierung bzw. die Deaktivierung der Freigabe wird ein entsprechender Code über die serielle Schnittstelle S0 zu OC gesendet.

9.4.13 Modul 12: Betriebsart Automatik

In der Hauptroutine in *main.c* wird überprüft, ob die Betriebsart Automatik angefordert wurde.

```
            // Betriebsart Automatik angewählt
            // Springe nur ein, wenn alle Achsen referiert und in Grundstellung
            // sind.
            if(m_all_ax_ref == 1)
            {
                // BA Auto angewählt
                if(m_ba_auto == 1)
                {
                    // Aufruf Betriebsart Automatik in ba_auto.c
                    ba_auto();
                }
            }
```

Die Betriebsart Automatik wird erst aktiv, wenn alle Achsen referiert und somit in Betriebsbereitschaft sind.

Der eigentliche Automatikablauf wird in der Datei *ba_auto.c* verwaltet.

In der Betriebsart Automatik werden die 5 Messpositionen automatisch nacheinander angefahren. Das Anfordern der jeweiligen nächsten Messposition wird von OC bzw. dem Hasheraccelerator übernommen. Details hierzu erhalten Sie im Kapitel Kommunikation.

Das Positionieren des Messautomaten auf die jeweils nächste Messposition erfolgt über das Fahren auf die Grundstellung.

Folgende Positionen werden in der Betriebsart Automatik nacheinander angefahren:

- Grundstellung
- Position 1
- Grundstellung
- Position 2
- Grundstellung
- Position 3
- Grundstellung
- Position 4
- Grundstellung
- Position 5

- Grundstellung

Nach jedem Erreichen der Grundstellung initialisieren sich die verschiedenen Umdrehungszähler der entsprechenden Achsen. Durch diese Maßnahme wird die größtmöglichste Positioniergenauigkeit erreicht.

Zum Erreichen einer Messposition verfahren maximal 2 Achsen gleichzeitig. Im folgenden Programmauszug wird die Positionierung des Messautomaten zu Messposition 4 dargestellt. Um diese Messposition zu erreichen, muß lediglich Achse 2 in Richtung (+) auf die geforderte Position bewegt werden.

```
        // Position 4: Achse 2 in R+
        if((move_order_pos == 4) && (!p6_fahrt_aktiv))
        {
            if(pos_erreicht_p4 == 0)
320         {
                if((!m_mot_a2_plus) && (!m_mot_a2_minus))
                {
                    p4_fahrt_aktiv=1;

325                 // Motor A2 Richtung+ ein
                    mot_a2_plus();

                    // Meldung LCD ausgeben
                    ba_a_msg_p4_aktiv();
330             }

                // Wenn Spindelumdrehungen = Sollwert, dann Motor stop
                if (s_umdr_a2 == S_U_SOLL_A2)
                {
335                 mot_a2_stp();
                    p4_fahrt_aktiv=0;
                    move_order_pos=0;
                    transmit_S0_vorbereitung(4);

340                 // LCD-Meldung aufrufen
                    ba_a_msg_p4();
                }

                if(m_mot_a2_plus == 0)
345             {
                    pos_erreicht_p4=1;
                }
            }
        }
350
```

Die angeforderte Position wird in der Variablen „move_order_pos“ hinterlegt. Die Bewegung zur Positionierung auf die geforderte Messposition Nummer 4 wird erst gestartet, wenn die Grundstellungsfahrt nicht mehr aktiv ist.

Ist Position 4 noch nicht erreicht und der Antriebsmotor der Achse 2 noch nicht eingeschaltet, wird zuerst das flag „p4_fahrt_aktiv“ gesetzt. Das Setzen dieses flags bedeutet, daß die Fahrt auf Position 4 nun aktiv ist.

Anschließend wird der Antriebsmotor der Achse 2 in Drehrichtung (+) eingeschaltet. Zusätzlich wird eine Meldung zur Bedienerführung auf der HMI abgesetzt.

Der Istwert des Spindelumdrehungszählers wird mit dem Sollwert verglichen. Ist der Sollwert erreicht, wird der Antriebsmotor der Achse 2 stillgesetzt und das Flag, welches signalisiert, daß die Fahrt auf Position 4 aktiv ist, zurückgesetzt.

Der aktive Fahrbefehl, der in der Variablen „move_order_pos“ hinterlegt ist, wird zurückgesetzt.

Anschließend erfolgt eine Bestätigung an OC bzw. den Hasheraccelerator, daß die angeforderte Position erreicht ist. Hierzu wird ein Code über die serielle Schnittstelle S0 an den entsprechenden Adressaten gesendet.

Es erfolgt ebenso nach dem Erreichen der Messposition 4 eine Meldung zur Bedienerführung auf der HMI. Hierzu wird die Funktion *ba_a_msg_p4 { }* (Betriebsart Automatik message Position 4) aufgerufen.

Diese Funktion bewirkt einen Makroaufruf in der HMI, welcher zur Folge hat, daß ein entsprechendes Bild auf dem LCD-Touchpanel angezeigt wird.

Unterschiedlich zum vorherigen Programmauszug stellt sich der Folgende dar. Zur Fahrt auf Messposition 2 werden zwei Achsen bewegt. Prinzipiell ist der Ablauf der Gleiche wie zur Positionierung mit einer Achse.

```
        // Position 2: Achse 1 in R+, Achse 4 Mitte
        if((move_order_pos == 2) && (!p6_fahrt_aktiv))
        {
            if(pos_erreicht_p2 == 0)
220         {
                if  (((!m_mot_a4_plus)
                    &&(!m_mot_a4_minus))
                    &&((!m_mot_a1_plus)
                    &&(!m_mot_a1_minus)))
225             {
                    p2_fahrt_aktiv=1;

                    // Motor A4, A1 Richtung+ ein
                    mot_a4_plus();
230                 mot_a1_plus();

                    // Meldung LCD ausgeben
                    ba_a_msg_p2_aktiv();
                }
235
                // Wenn Spindelumdrehungen = Sollwert, dann Motor stop
                if (s_umdr_a4 == S_U_SOLL_A4_M)
                {
                    mot_a4_stp();
240             }

                if (s_umdr_a1 == S_U_SOLL_A1)
                {
                    mot_a1_stp();
245
                    // Achtung !
                    // folgende 2 Zeilen müssen eventuell nach mot_a4_stp();
                    // an Statt hier eingefügt werden.
                    // Die Achse, die später am Ziel ist, muß diese flags auf
250                 // false setzen.
                    p2_fahrt_aktiv=0;
                    move_order_pos=0;
                    transmit_S0_vorbereitung(2);

255                 // LCD-Meldung aufrufen
                    ba_a_msg_p2();
                }

                if((m_mot_a4_plus == 0) && (m_mot_a1_plus == 0))
260             {
                    pos_erreicht_p2=1;
                }
            }
        }
```

Im Beispiel wird Messposition 2 angefahren über die Bewegung durch Achse 1 in Richtung (+) und die Bewegung der Achse 4 ebenso in Richtung (+).

Die Verriegelungen gestalten sich ähnlich dem Ablauf durch Positionierung über eine Achse, deshalb wird hier nicht mehr detailliert darauf eingegangen. Es ist lediglich darauf zu achten, daß Ein-/Ausschaltbedingungen nun für beide zu bewegende Achsen notwendig sind.

Das Flag, das die aktive Fahrt zur Messposition signalisiert, wird von der Achse zurückgesetzt, die als letzte Achse ihre Sollposition erreicht. Somit wird gewährleistet, daß entsprechende Meldungen zur Bedienerführung an die HMI erst abgesetzt werden, wenn der Messautomat richtig positioniert ist.

Eine Akustikmessung zu einem zu frühen Zeitpunkt und somit einer fehlerhaften Akustikmessung wird somit verhindert.

9.4.14 Modul 13: Betriebsart Teilautomatik

Wie für die Betriebsart Automatik, wird für die Betriebsart Teilautomatik in der Hauptroutine in *main.c* überprüft, ob die Betriebsart Teilautomatik angefordert wurde.

```
        // ------------------------------------------------------------------
245     // Betriebsart Teilautomatik angewählt
        // Springe nur ein, wenn alle Achsen referiert und in Grundstellung
        // sind.
        if(m_all_ax_ref == 1)
        {
250         // BA Teilauto angewählt
            if(m_ba_teilauto == 1)
            {
                // Aufruf Betriebsart Teilautomatik in ba_teilauto.c
                ba_teilauto();
255         }
        }
```

Wie auch in die Betriebsart Automatik darf in die Betriebsart Teilautomatik nur gewechselt werden, wenn alle Achsen des Messautomaten referiert und somit betriebsbereit sind. Eine entsprechende Bedienerführung wird auf der HMI ausgegeben.

Wird die Betriebsart Teilautomatik von OC bzw. dem Hasheraccelerator angefordert, findet die Koordination des Ablaufes ebenso wie bei der Betriebsart Automatik in der Funktion *ba_auto { }* der Datei *ba_auto.c* statt. Es wird somit das Modul 12 mit in Anspruch genommen.

Findet die Anforderung der Betriebsart Teilautomatik durch die HMI statt, wird die Funktion ba_teilauto { } in Datei ba_teilauto.c verwendet.

```
        // Position 1: Olli Mitte, Achse 4 Mitte
        if((neu_anzufahrende_pos == 1) && (!ta_p6_fahrt_aktiv))
        {
            if(pos_erreicht_p1 == 0)
185         {
                if  ((!m_mot_a4_plus)
                    &&(!m_mot_a4_minus)
                    &&(!m_mot_a1_minus)
                    &&(!m_mot_a2_minus)
190                 &&(!m_mot_a3_minus))
                {
                    ta_p1_fahrt_aktiv=1;

                    // Motor A4 Richtung+ ein
195                 mot_a4_plus();

                    // Meldung LCD ausgeben
                    ba_ta_msg_p1_aktiv();
                }
200
                // Wenn Spindelumdrehungen = Sollwert, dann Motor stop
                if (s_umdr_a4 == S_U_SOLL_A4_M)
                {
                    mot_a4_stp();
205                 ta_p1_fahrt_aktiv=0;
                    neu_anzufahrende_pos=0;

                    // Meldung auf LCD : Position 1 erreicht
                    msg_p1();
210             }

                if(m_mot_a4_plus == 0)
                {
                    pos_erreicht_p1=1;
215             }
            }
        }
```

Der Quellcode ist prinzipiell gleich dem Quellcode der Betriebsart Automatik. Dies wird deutlich im Programmauszug zur Bewegung des Messautomaten zur Position 1.

Unterscheidungen ergeben sich hauptsächlich in der Namensgebung der Variablen. Dies ist eine Folge der Kommunikation, die in weiteren Kapiteln detailliert dargestellt wird.

Meldungen zur Bedienerführung werden ebenso auf der HMI wie in der Betriebsart Automatik ausgegeben.

Eine Anforderung zum Positionswechsel erfolgt nicht mehr automatisch nach dem Erreichen der vorherigen Position. In der Betriebsart Teilautomatik wurde eine Möglichkeit geschaffen, die verschiedenen Messpositionen in beliebiger Reihenfolge anzufahren.

Ein Anfahren der Messpositionen erfolgt jedoch immer, wie auch in Betriebsart Automatik, über die Grundstellung.

Die Anforderung zum Positionswechsel erfolgt entweder über OC oder den Hasheraccelerator, außerdem kann ein Positionswechsel über die HMI angefordert werden. Informationen hierzu erhalten Sie in den Kapiteln zur HMI.

Ebenso wie in der Betriebsart Automatik wird auf die Rückmeldungen an die entsprechenden Adressaten beim Erreichen einer angeforderten Position nicht verzichtet.

Eine Fehlergenerierung und Diagnose wie bereits im Kapitel zu Modul 11 beschrieben, findet ebenso statt.

9.4.15 Modul 14: Betriebsart Hand

Wie auch bei anderen Betriebsarten, wird in der Hauptroutine in *main.c* überprüft, ob eine Anforderung der Betriebsart Hand ansteht.

```
            // ------------------------------------------------------------------
            // Betriebsart Hand angewählt
260         if(m_ba_hand == 1)
            {
                // Aufruf Betriebsart Hand in ba_hand.c
                ba_hand();
            }
```

Die einzelnen Fahrbefehle, die in der Betriebsart Hand möglich sind, sind in der Funktion *ba_hand { }* der Datei *ba_hand.c* hinterlegt.

Die Betriebsart Hand ist ausschließlich von der HMI anzufordern. Die Freigaben zur Bedienung des Messautomaten durch OC bzw. den Hasheraccelerator werden dadurch gesperrt.

Die Betriebsart Hand dient dazu, einzelne Achsen des Messautomaten schrittweise zu verfahren. Die Schrittgröße wird über einen DEFINE-Wert in der Headerdatei ba_hand.h eingestellt. Eine Schrittgröße von 2 Spindelumdrehungen, dies entspricht einem linearen Verfahrweg von 2 mm, wird hier für sinnvoll gehalten.

```
    // ------------------------------------------------------------------------
    // Ablauf Tippen in Betriebsart Hand
55  void ba_hand (void)
    {
        // Fahrbefehl 1, Codierung in Kommunikation_Auswertung.c
        if(ba_h_fahrbef == 1)
        {
60          if(pos_erreicht_tipp_a1_plus == 0)
            {
                if((!m_mot_a1_plus) && (!m_mot_a1_minus))
                {
                    mot_a1_plus();
65              }

                // Wenn Spindelumdrehungen = Sollwert, dann Motor stop
                if (s_umdr_a1 == S_U_SOLL_A1_TIPP)
                {
70                  mot_a1_stp();
                    umdr_zaehler_init_a1();
                }

                if(m_mot_a1_plus == 0)
75              {
                    pos_erreicht_tipp_a1_plus=1;
                    ba_h_fahrbef=0;
                }
            }
80      }
```

Im Programmauszug wird das Verfahren der Achse 1 in Drehrichtung (+) dargestellt. Die Fahrbefehle werden in der Variablen „ba_h_fahrbef“ gespeichert. Detaillierte Informationen hierzu in den Kapiteln zur Kommunikation.

- Der Verfahrbefehl Nummer 1 signalisiert die Anforderung der Bewegung der Achse 1 in Drehrichtung (+).
- Ist die angeforderte Position noch nicht erreicht und der Antriebsmotor nicht angesteuert, wird der Antrieb in Drehrichtung (+) bewegt.
- Der Istwert des Spindelumdrehungszählers wird mit dem Sollwert in „S_U_SOLL_A1_TIPP“ (Spindelumdrehungen Sollwert Achse 1 Tippbetrieb) verglichen.
- Ist der Sollwert erreicht, wird der Antrieb gestoppt und die Umdrehungszähler der Achse 1 werden initialisiert.
- Ist die angeforderte Position erreicht, wird dies in einem Flag gespeichert und der aktuelle Fahrbefehl wird gelöscht.

Die Fahrbefehle werden in der Datei *Kommunikation_Auswertung.c* decodiert. Um alle vier Achsen in jeweils Drehrichtung (+) und Drehrichtung (-) verfahren zu können, stehen acht Fahrbefehle zur Verfügung.

Je nach ausgelesenem Fahrbefehl, wird die entsprechende Achse in die geforderte Drehrichtung bewegt.

In der Betriebsart Hand steht wie auch in der Betriebsart Automatik und Betriebsart Teilautomatik die Bedienerführung der HMI zur Verfügung.

Eine Diagnose findet ebenso über die Datei *error_1.c* statt.

9.4.16 Modul 15: Achse freifahren

Das Freifahren der Achsen von einer Endlagenposition ist in der Datei *achse_freifahren.c* realisiert.

Der Aufruf der Funktion *ax_free { }* findet in der Hauptroutine in *main.c* statt.

```
270         // Anforderung von LCD
            if(achse_freifahren == 1)
            {
                // Aufruf Achse von Endlage freifahren in achse_freifahren.c
                ax_free();
            }
```

Das Freifahren wurde programmiert, um eine Achse, die eventuell durch ein defektes Wegmesssystem, auf eine Arbeitsbereichsendlage gefahren ist, automatisiert von dieser Endlage wieder freizufahren.

Ebenso ist es möglich, in der Betriebsart Hand, die Achsen auf die Endlagen zu fahren. Um eine mechanische Beschädigung durch Fehlbedienung auszuschließen, wurde das Freifahren der Achse automatisiert.

```
        // von Endlage+, Achse 1 freifahren
        if(err_number == 7)
        {
            if(!pos_erreicht_ax_free_a1_minus)
100         {
                if((!m_mot_a1_plus) && (!m_mot_a1_minus))
                {
                    // Achse 1 in Richtung - freifahren
                    mot_pwr_enable();
105                 umdr_zaehler_init_a1();
                    mot_a1_minus();
                }

                // Wenn Spindelumdrehungen = Sollwert, dann Motor stop
110             if (s_umdr_a1 == S_U_SOLL_AX_FREE)
                {
                    mot_a1_stp();
                    umdr_zaehler_init_a1();
                    achse_freifahren=0;
115
                    // enable INT4
                    EX4=1;
                }

120             if(!m_mot_a1_minus)
                {
                    pos_erreicht_ax_free_a1_minus=1;
                    err_number=0;
                }
125         }
        }
```

Die Funktion *ax_free { }* wird aufgerufen, nachdem ein entsprechender Fehler in der Datei *error_1.c* diagnostiziert wurde.

Die im Modul 11 generierte Fehlernummer wird in der Funktion *ax_free { }* aus der Variablen „err_number“ ausgelesen.

Die Betriebsarten Automatik und Teilautomatik werden gesperrt. Eine Bedienung durch OC und den Hasheraccelerator ist nicht mehr möglich.

Der Entzug der Freigabe zur Bedienung wird entsprechend durch ein Meldungsfenster in OC angezeigt. Die Meldung ist quittierpflichtig.

Eine Bedienung des Messautomaten ist ausschließlich durch die HMI möglich. Detailinformationen hierzu im Kapitel zur HMI.

Der Ablauf des Freifahrens:

- Die Fehlernummer wird ausgelesen.
- ist der entsprechende Antriebsmotor noch nicht eingeschaltet, wird zuerst der Leistungsteil der Hauptplatine wieder freigegeben. Dieser wurde als Reaktion auf den diagnostizierten Fehler ausgeschaltet.
- Die Umdrehungszähler der entsprechenden Achse werden initialisiert.
- Der Antriebsmotor wird in die entsprechende Drehrichtung bewegt.
- Der Istwert des Spindelumdrehungszählers wird mit dem Sollwert verglichen.
- Ist der Sollwert erreicht, wird der Antriebsmotor gestoppt.
- Die Umdrehungszähler werden erneut initialisiert.
- Die Anforderung zum Freifahren der Achse wird zurückgesetzt.
- Die Überwachung der Endlagen wird wieder aktiviert. Dazu wird die Freigabe des Interrupts Nummer 4 wieder gesetzt. Die Überwachung wurde als Reaktion auf den diagnostizierten Fehler deaktiviert.
- Die Fehlernummer wird zurückgesetzt.

Als mögliche Fehlernummer sind die Nummern 6 bis 13 reserviert.

Folgender Programmauszug macht deutlich, daß auch ein automatisiertes Freifahren der Achsen in Drehrichtung (+) möglich ist. Hierzu muß eine Antriebsachse die negative Endlage angefahren haben.

```
        // von Endlage-, Achse 2 freifahren
130     if(err_number == 8)
        {
            if(!pos_erreicht_ax_free_a2_plus)
            {
                if((!m_mot_a2_plus) && (!m_mot_a2_minus))
135             {
                    // Achse 2 freifahren in Richtung +
                    mot_pwr_enable();
                    umdr_zaehler_init_a2();
                    mot_a2_plus();
140             }

                // Wenn Spindelumdrehungen = Sollwert, dann Motor stop
                if (s_umdr_a2 == S_U_SOLL_AX_FREE)
                {
145                 mot_a2_stp();
                    umdr_zaehler_init_a2();
                    achse_freifahren=0;

                    // enable INT4
150                 EX4=1;
                }

                if(m_mot_a2_plus == 0)
                {
155                 pos_erreicht_ax_free_a2_plus=1;
                    err_number=0;
                }
            }
        }
160
```

Grundsätzlich funktioniert das Freifahren der Achsen in Drehrichtung (+) auf gleiche Art und Weise wie das Freifahren der Achsen in Drehrichtung (-).

Auf eine detaillierte Beschreibung hierzu wird deshalb verzichtet.

9.4.17 Modul 16 + 17: Freigabecode an OC senden

Bei Störungen wird dem Tool OLLICONTROL bzw. Hasheraccelerator die Freigabe zur Bedienung des Messautomaten entzogen.

Um den Messautomat wieder über externe Software bedienen zu können, muß dieser Software die Freigabe zur Bedienung wieder erteilt werden.

```
300         // ------------------------------------------------------------------
            // Nachdem Fehler angestanden ist, Olli-Control wieder Freigabe senden
            // Störung wurde quittiert und Olli ist in Grundstellung
            if  ((err_number == 0)
                &&(error_activ == 1)
305             &&(refp[0] == 1)
                &&(refp[1] == 1)
                &&(refp[2] == 1)
                &&(refp[3] == 1))
            {
310             olli_control_enable();
                error_activ=0;
                first_turn_err=0;
            }
```

Dies darf erst geschehen, wenn keine Störung mehr diagnostiziert wird, alle vorherigen Störungen beseitigt und quittiert sind, und der Messautomat sich in Grundstellung befindet.

Über den Aufruf der Funktion *olli_control_enable { }* wird ein Freigabecode an den entsprechenden Adressaten gesendet.

```
// ------------------------------------------------------------------------
// tool Olli-Control Freigabe erteilen
void olli_control_enable(void)
{
    transmit_S0_vorbereitung(8);
}

// ------------------------------------------------------------------------
// tool Olli-Control Freigabe entziehen
void olli_control_disable(void)
{
    transmit_S0_vorbereitung(7);
}
```

Die Anbindung ist über die serielle Schnittstelle S0 realisiert.

Modul 16 und Modul 17 können hier zusammengefasst dargestellt werden, sie unterscheiden sich in der Funktion nicht voneinander.

Modul 16 dient der Erteilung der Freigabe an externe Software zur Bedienung des Messautomaten nach dem Start und der Initialisierung des Systems.

Modul 17 dient der Wiedererteilung der Freigabe an externe Software nach aufgetretenen Störungen im System.

9.4.18 Modul 18: Störung quittieren

Nach dem Grundsatz des Aufteilens des C-Projekts in Module, mußte hier ein Modul deklariert werden, um explizit die Fehlernummer 1 zu quittieren, da dies in der Hauptroutine in *main.c* stattfindet.

Eine separate Behandlung dieses Fehlers findet aber nicht statt. Es wird hiermit auf das Kapitel zu Modul 20 verwiesen.

Eine Quittierung von quittierpflichtigen Störungen findet nur durch die Bedienung der HMI statt. Dazu finden Sie Hinweise in den Kapiteln zur HMI.

9.4.19 Modul 19: Fahrbefehle in Betriebsart Automatik auslesen

Die Datei move_order.c stellt den Fahrbefehlsmanager der Betriebsart Automatik dar. In der Funktion check_move_order { } wird überprüft, ob ein neuer Fahrbefehl ansteht. Anschließend wird der Fahrbefehl ausgelesen.

```
// ------------------------------------------------------------------------
void check_move_order(void)
{
    // Überprüfen, ob neuer Fahrbefehl vorhanden ist
    if(m_new_move_order == true)
    {
        // Fahrbefehle auslesen und abspeichern

        // Fahrbefehl für mechanische Position 1 auslesen
        if  (((befehl_s0_in[1] == 0) && (befehl_s0_in[2] == 1))
            ||
            ((befehl_s0_in[1] == 1) && (befehl_s0_in[2] == 1))
            ||
            ((befehl_s0_in[1] == 2) && (befehl_s0_in[2] == 1))
            ||
            ((befehl_s0_in[1] == 3) && (befehl_s0_in[2] == 1)))

            {
                move_order_pos=1;
            }

        // Fahrbefehl für mechanische Position 2 auslesen
        if  (((befehl_s0_in[1] == 0) && (befehl_s0_in[2] == 2))
            ||
            ((befehl_s0_in[1] == 1) && (befehl_s0_in[2] == 2))
            ||
            ((befehl_s0_in[1] == 2) && (befehl_s0_in[2] == 2))
            ||
            ((befehl_s0_in[1] == 3) && (befehl_s0_in[2] == 2)))

            {
                move_order_pos=2;
            }

        // Fahrbefehl für mechanische Position 3 auslesen
        if  (((befehl_s0_in[1] == 0) && (befehl_s0_in[2] == 3))
            ||
            ((befehl_s0_in[1] == 1) && (befehl_s0_in[2] == 3))
            ||
            ((befehl_s0_in[1] == 2) && (befehl_s0_in[2] == 3))
            ||
            ((befehl_s0_in[1] == 3) && (befehl_s0_in[2] == 3)))

            {
                move_order_pos=3;
            }
```

Der Programmauszug stellt das Auslesen der Fahrbefehle 1 bis 3 dar. Insgesamt werden 6 Fahrbefehle deklariert.

Es wird jeweils für eine mechanische Messposition des Messautomaten ein Fahrbefehl hinterlegt. Dieser wird abgelegt in der Variablen „move_order_pos“.

Die Fahrbefehle für den Automatikbetrieb werden über die serielle Schnittstelle S0 empfangen und im eindimensionalen array „befehl_s0_in []“ abgelegt.

Im Befehlsstring steht die für die Messposition relevante Information an zweiter und dritter Stelle, somit wird hierfür folgende Stelle im array ausgelesen:

- befehl_s0_in [1]
- befehl_s0_in [2]

Detaillierte Informationen zu den Befehlscodierungen erhalten Sie im Kapitel Kommunikation.

Verwirrend erscheint hier, daß z. B. der Fahrbefehl Nummer 2 über 4 Möglichkeiten im array „befehl_s0_in []“ zustande kommt. Dies ist aber eine Folge davon, daß der Messautomat auf 4 verschiedenen Sitzpositionen im Kraftfahrzeug die Messungen durchführt.

Die Messposition 2 ist immer ein zur Fahrzeugmitte geneigter Messautomat. Unterschieden werden folgende Sitzpositionen im Kraftfahrzeug :

- Fahrersitz
- Beifahrersitz
- hinter Beifahrersitz
- hinter Fahrersitz

Somit wäre nicht nur eine Spiegelung der Positionen notwendig, sondern auch eine Unterscheidung zwischen den Sitzen vorne und den Sitzplätzen hinten.

Um eindeutig unterscheiden zu können, wurde für jede Messposition auf dem entsprechenden Sitzplatz ein Befehl der externen Software festgelegt, der hier entsprechend decodiert und den mechanischen Positionen zugeordnet wird.

9.4.20 Modul 20: Befehle der HMI auslesen

Die Befehle der HMI werden in der Datei *kommunikation_auswertung.c* decodiert und ausgewertet.

Der Dialog findet über die serielle Schnittstelle S1 statt.

Die ankommenden Befehle werden im Befehlspuffer „befehl_s1_in []“ gespeichert. Die relevanten Informationen stehen im array an dritter und vierter Stelle, deshalb werden grundsätzlich Informationen an folgender Stelle ausgelesen:

- befehl_s1_in [2]
- befehl_s1_in [3]

Die Befehle werden nach der Decodierung weiter zur Verwendung aufbereitet, d. h. Verriegelungen werden eingefügt, die Verschaltung mit weiteren Bedingungen wird ebenso in dieser Funktion realisiert.

```
100     // BA Teilautomatik setzen
        if((befehl_s1_in[2] == 0) && (befehl_s1_in[3] == 2))
        {
            // nur bei Grundstellung umschalten auf BA TeilAuto zulassen, sonst
            // Fehlerbit setzen
105         if  ((refp[0] == 1)
                &&(refp[1] == 1)
                &&(refp[2] == 1)
                &&(refp[3] == 1))
                {
110                 // Bei Bedienung über das LCD-Display Olli_Control Freigabe entziehen
                    if(m_ba_teilauto == 0)
                    {
                        olli_control_disable();
                    }
115
                    m_ba_hand=0;
                    m_ba_auto=0;
                    m_ba_teilauto=1;
                }
120             else
                {
                    // Fehlerbit "keine Grundstellung" setzen
                    m_grdstllg_nio=1;
                }
125     }
```

Im Programmauszug das Beispiel zur Umschaltung der Betriebsart. Die Betriebsart Teilautomatik soll gesetzt werden.

- Das array „befehl_s1_in []“ wird ausgelesen.
- Ergibt die Decodierung die dezimalen Werte 0 und 2 ist die Anforderung zum Betriebsartenwechsel auf die Betriebsart Teilautomatik gegeben.

- Als weitere Bedingung zum Setzen der Betriebsart muß der Messautomat in Grundstellung sein.
- Wird der Betriebsartenwechsel über die HMI angefordert, muß OC die Freigabe zur Bedienung des Messautomaten entzogen werden.
- Die Betriebsarten Automatik und Hand werden zurückgesetzt.
- Wurde der Betriebsartenwechsel angefordert, obwohl der Messautomat nicht in Grundstellung war, wird ein Fehlerbit gesetzt. Über dieses Fehlerbit wird die Bedienerführung auf der HMI aktualisiert.

Ähnlich wird auch verfahren, um die Fahrbefehle im Teilautomatikbetrieb auszulesen. Im Programmauszug werden hier lediglich 4 Fahrbefehle dargestellt. Insgesamt stehen in diesem Bereich 6 Fahrbefehle zur Verfügung.

```
    //------------------------------------------------------------------------------
    // Teilautomatik Fahrbefehl auslesen
        // neuen Fahrbefehl nur akzeptieren, wenn der letzte nicht mehr aktiv ist
155     if(neu_anzufahrende_pos == 0)
        {
            if((befehl_s1_in[2] == 0) && (befehl_s1_in[3] == 6))
            {
                neu_anzufahrende_pos=1;
160         }

            if((befehl_s1_in[2] == 0) && (befehl_s1_in[3] == 7))
            {
                neu_anzufahrende_pos=2;
165         }

            if((befehl_s1_in[2] == 0) && (befehl_s1_in[3] == 8))
            {
                neu_anzufahrende_pos=3;
170         }

            if((befehl_s1_in[2] == 0) && (befehl_s1_in[3] == 9))
            {
                neu_anzufahrende_pos=4;
175         }
```

Ein neuer Fahrbefehl wird nur ausgelesen, wenn der letzte Fahrbefehl bereits abgearbeitet wurde. Ansonsten wird der Fahrbefehl ignoriert.

Der Fahrbefehl wird aus dem Befehlspuffer „befehl_s1_in []“ ausgelesen und anschließend in die Variable „neu_anzufahrende_pos“ zur weiteren Verarbeitung geschrieben.

Mit der Anforderung der Grundstellungsfahrt wird ebenso verfahren.

```
            if((befehl_s1_in[2] == 1) && (befehl_s1_in[3] == 1))
            {
                // Grundstellung nur anfordern, wenn mindestens eine Achse nicht
185             // auf Referenzpunktschalter steht
                if  (!(refp[0])
                    ||(!refp[1])
                    ||(!refp[2])
                    ||(!refp[3]))
190             {
                    // Position 6 entspricht Position Grundstellung
                    neu_anzufahrende_pos=6;
                    m_ba_hand=0;
                    m_ba_auto=0;
195                 m_ba_teilauto=1;
                }
                else
                {
                    error_grdstllg_ok();
200             }
            }
```

Die Grundstellungsfahrt wird allerdings nur angefordert, wenn sich mindestens eine Achse nicht in Grundstellung befindet.

Eine entsprechende Bedienerführung wird über die HMI ausgegeben.

Zur Grundstellungsfahrt wird die Betriebsart automatisch in Teilautomatik gewechselt und die Betriebsarten Automatik und Hand zurückgesetzt.

Mit dem Erreichen der Grundstellung werden entsprechende Umdrehungszähler der Achsen initialisiert.

Auf gleiche Art und Weise werden die Befehle zur Betriebsart Hand decodiert und weiter verarbeitet.

```
    //------------------------------------------------------------------------------
    // BA Hand
205
        if((befehl_s1_in[2] == 1) && (befehl_s1_in[3] == 6))
        {
            if(m_ba_hand == 1)
            {
210             ba_h_fahrbef=1;
                pos_erreicht_tipp_a1_plus=0;
            }
        }

215     if((befehl_s1_in[2] == 1) && (befehl_s1_in[3] == 7))
        {
            if(m_ba_hand == 1)
            {
                ba_h_fahrbef=2;
220             pos_erreicht_tipp_a1_minus=0;
            }
        }

        if((befehl_s1_in[2] == 1) && (befehl_s1_in[3] == 8))
225     {
            if(m_ba_hand == 1)
            {
                ba_h_fahrbef=3;
                pos_erreicht_tipp_a2_plus=0;
230         }
        }

        if((befehl_s1_in[2] == 1) && (befehl_s1_in[3] == 9))
        {
235         if(m_ba_hand == 1)
            {
                ba_h_fahrbef=4;
                pos_erreicht_tipp_a2_minus=0;
            }
240     }
```

Der Befehlspuffer wird ausgelesen. Entspricht die ausgelesene Information dem entsprechenden Befehl, wird dieser gesetzt.

Acht Fahrbefehle stehen insgesamt in der Betriebsart Hand zur Verfügung. Das ist ausreichend, um die 4 Achsen in jeweils beide Drehrichtungen verfahren zu können.

```
    //------------------------------------------------------------------------------
280 // sonstige Befehle auslesen

        // nach Endlagenfahrt über LCD Achse freifahren
        if((befehl_s1_in[2] == 2) && (befehl_s1_in[3] == 5))
        {
285         achse_freifahren=1;
        }


        // Befehl "Störung quittieren" auslesen
290     if((befehl_s1_in[2] == 2) && (befehl_s1_in[3] == 4))
        {
            err_qtt=1;
        }
```

Im Modul 20 werden noch andere Befehle ausgelesen. Die Anforderung zum Freifahren der Achsen nach einer Endlagenfahrt wird ebenso hier decodiert wie die Anforderung der Störungsquittierung seitens der HMI.

9.4.21 Modul 30: Umdrehungszähler der Achsen 1 – 3 aktualisieren

In Modul 30 wird durch einen Funktionsaufruf, ausgelöst durch einen Interrupt gesprungen. Interrupt 1 wird ausgelöst durch einen Überlauf von Counter 0.

Mit diesem Interrupt wird die Impulszählung der Inkrementalgeber von Achse 1 bis 3 realisiert. Nach dem Einsprung in die ISR (Interrupt Service Routine) wird überprüft, welche der 3 Achsen im Moment aktiv ist.

Entsprechend der aktiven Achse wird die Funktion zur Aktualisierung der Umdrehungszähler der jeweiligen Achse aufgerufen. In diesen Funktionen wie *count_umdr_a1 { }* werden die Umdrehungszähler anschließend inkrementiert.

Hierbei handelt es sich um einen Umdrehungszähler für die Motorumdrehungen, einem Hilfsumdrehungszähler und einem Spindelumdrehungszähler.

```
// Interrupt aus Überlauf Counter 0, Impulszählung Achsen 1-3
_interrupt(1) void intrpt_1 (void)
{
    if((m_mot_a1_plus == 1) || (m_mot_a1_minus == 1))
    {
        count_umdr_a1();
        TF0=0;
    }

    if((m_mot_a2_plus == 1) || (m_mot_a2_minus == 1))
    {
        count_umdr_a2();
        TF0=0;
    }

    if((m_mot_a3_plus == 1) || (m_mot_a3_minus == 1))
    {
        count_umdr_a3();
        TF0=0;
    }
}
```

Im Programmauszug ist die ISR des Interrupts 1 dargestellt. Diese Funktion ist in der Datei *intrrpt_functions.c* programmiert.

Im nachfolgenden Programmauszug ist die Inkrementierung der Umdrehungszähler für Achse 1 und Achse 2 dargestellt.

```
// ------------------------------------------------------------------------
// Umdrehungszähler Achse 1 aktualisieren
// wird aufgerufen durch Interrupt 1 aus Überlauf Counter 0
void count_umdr_a1 (void)
{
        // Umdrehungszähler Motor inkrementieren
        // jeden 16. Impuls wird Umdrehungszähler Motor akt.
        m_umdr_a1++;
        m_umdr_hilfsm_a1++;

        // Umdrehungszähler Spindel inkrementieren
        // alle GEAR_N Motorumdrehungen wird Umdrehungszähler Spindel aktualisiert.
        // Hilfsmerker Umdrehungszähler Motor wird gelöscht.
        if (m_umdr_hilfsm_a1 == GEAR_N_23 )
        {
            s_umdr_a1++;
            m_umdr_hilfsm_a1=0;
        }
}

// ------------------------------------------------------------------------
// Umdrehungszähler Achse 2 aktualisieren
// wird aufgerufen durch Interrupt 1 aus Überlauf Counter 0
void count_umdr_a2 (void)
{
        // Umdrehungszähler Motor inkrementieren
        // jeden 16. Impuls wird Umdrehungszähler Motor akt.
        m_umdr_a2++;
        m_umdr_hilfsm_a2++;

        // Umdrehungszähler Spindel inkrementieren
        // alle GEAR_N Motorumdrehungen wird Umdrehungszähler Spindel aktualisiert.
        // Hilfsmerker Umdrehungszähler Motor wird gelöscht.
        if (m_umdr_hilfsm_a2 == GEAR_N_23 )
        {
            s_umdr_a2++;
            m_umdr_hilfsm_a2=0;
        }
}
```

Nach dem Aufruf der Funktion wird der Umdrehungszähler des Motors inkrementiert. Der Hilfsumdrehungszähler wird ebenso inkrementiert.

Anschließend wird überprüft, ob eine vollständige Spindelumdrehung vollzogen wurde. Dazu wird der Istwert des Motorumdrehungszählers mit dem DEFINE-Wert „GEAR_N_23“ verglichen. In diesem Wert ist die Getriebeübersetzung hinterlegt.

„GEAR_N_23“ wird in der Headerdatei *umdrehungszähler.h* deklariert.

Ist eine Spindelumdrehung vollständig, wird der Umdrehungszähler der Spindel inkrementiert und der Hilfsumdrehungszähler des Motors zurückgesetzt.

Mit diesem Verfahren wird die Positionierung der einzelnen Achsen im System realisiert.

9.4.22 Modul 31: Umdrehungszähler der Achse 4 aktualisieren

Das Aktualisieren des Umdrehungszählers der Achse 4 wird ebenso wie das Aktualisieren der Umdrehungszähler für Achse 1 bis 3 realisiert.

Hierzu wird der Interrupt 3 durch einen Überlauf des Counters 1 angefordert. Dieser ist für die Zählung der Impulse des Inkrementalgebers der Achse 4 zuständig.

```
    // Interrupt aus Überlauf Counter 1, Impulszählung Achse 4
    _interrupt(3) void intrpt_3 (void)
    {
70      count_umdr_a4();
        TF1=0;
    }
```

Informationen zur Initialisierung der Counter erhalten Sie in der Dokumentation zu Modul 4.

In der ISR zu Interrupt 3 wird lediglich die Funktion *count_umdr_a4 { }* zur Aktualisierung der Umdrehungszähler der Achse 4 aufgerufen.

Ein Programmauszug explizit zur Inkrementierung der Umdrehungszähler der Achse 4 wird hier nicht separat dargestellt.

Informationen hierzu können der Dokumentation zu Modul 30 entnommen werden. Die Unterschiede liegen lediglich in der Namensgebung der Variablen.

9.4.23 Modul 32: Störung durch Endlagenschalter auswerten

Die Endlagenschalter des Systems sind über die Basisplatine hardwaremäßig mit dem Eingang des Interrupts 11 an Port 1.1 verbunden.

Auf diese Art und Weise muß nicht der gesamte Port, an dem die Endlagenschalter angeschlossen sind, zyklisch abgefragt werden.

```
    // ------------------------------------------------------------------------
75  // Fehler: Achse Endlage
    // Überprüfen, welche Achse auf Endlage gefahren ist
    _interrupt(11) void intrpt_error_endlage (void)
    {
        // Funktionsaufruf, Endlagenschalter auswerten
80      error_endlage();

        // Interruptrequest rücksetzen
        IEX4=0;
    }
```

Mit diesem Verfahren wird die Zykluszeit der Hauptroutine nicht unnötig verlängert. Sobald eine Antriebsachse auf Endlage gefahren ist, wird dies bemerkt und eine entsprechende Reaktion in Verbindung mit Stillsetzung der Antriebe ausgelöst.

Für eine exakte Diagnose wird nun über den Aufruf der Funktion *error_endlage { }* der Port 7, an welchem die Endlagenschalter angeschlossen sind, ausgewertet.

In der ISR des Interrupts 11 steht allerdings nur der Aufruf der Funktion *error_endlage { }*. Nach dem Rücksprung von *error_endlage { }* wird der Interruptrequest zurückgesetzt.

Die Funktion *error_endlage { }* wird in der Datei *error_1.c*, in der nahezu die gesamte Fehlerauswertung stattfindet, ausgeführt.

In der Funktion *error_endlage { }* werden nun nicht nur die Endlagenschalter auf Paarfehler ausgewertet, sondern auch einzeln auf Bedämpfung überprüft.

Diese Auswertung wird u. A. auch für die Freifahrfunktion, die in Modul 15 dokumentiert wird, benötigt.

Eine Bedienerführung durch die HMI findet statt. Die Verwaltung der Freigaben für die Steuerung des Messautomaten durch externe Software wird ebenso aktualisiert.

```
        /// Überprüfe auf Endlage+, Achse 1
180     // Bit 7.1 = 1 ?
        if (achse_endlage == 0x02)
        {
            err_number = 7;
            error_break(err_number);
185     }

        // Fehlernummer: 8
        // Überprüfe auf Endlage-, Achse 2
        // Bit 7.2 = 1 ?
190     if (achse_endlage == 0x04)
        {
            err_number = 8;
            error_break(err_number);
        }
195
        // Fehlernummer: 9
        // Überprüfe auf Endlage+, Achse 2
        // Bit 7.3 = 1 ?
        if (achse_endlage == 0x08)
200     {
            err_number = 9;
            error_break(err_number);
        }

205     // Fehlernummer: 10
        // Überprüfe auf Endlage-, Achse 3
        // Bit 7.4 = 1 ?
        if (achse_endlage == 0x10)
        {
210         err_number = 10;
            error_break(err_number);
        }

        // Fehlernummer: 11
215     // Überprüfe auf Endlage+, Achse 3
        // Bit 7.5 = 1 ?
        if (achse_endlage == 0x20)
        {
            err_number = 11;
220         error_break(err_number);
        }
```

Im Programmauszug nur ein kleiner Ausschnitt der Diagnose. Auf die komplette Darstellung der Diagnose der Endlagenüberwachung wird verzichtet.

Port 7 wird eingelesen und in die Variable „achse_endlage“ geschrieben. Anschließend erfolgt eine Überprüfung auf Paarfehler.

Von Paarfehlern spricht man, wenn das unzulässige Ansprechen beider Endlagenschalter einer Achse diagnostiziert wird.

Nach der Überprüfung auf Paarfehler erfolgt die explizite Auswertung der einzelnen Schalter. Ist ein Fehler diagnostiziert, wird eine entsprechende Fehlernummer generiert.

Die Fehlernummer wird an die Funktion *error_break { }* übergeben und löst dort die für diesen Fehler hinterlegte Reaktion aus.

Über die HMI kann dieser Fehler nun visualisiert, beseitigt und quittiert werden. Endlagenfehler sind stets quittierpflichtig.

9.4.24 Modul 33: Leistungsteil Überlast

Die OCD-Ausgänge der Motorsteuer-ICs sind über die Basisplatine hardwaremäßig an den Port 1.2 des Mikrocontrollers angeschlossen.

Diese Signale wurden zusammengeführt an den Port angeschlossen und lösen bei Überlast der ICs einen Interrupt aus. Hierfür wird der Interrupt Nummer 12 verwendet.

In der ISR (Interrupt Service Routine) des Interrupts Nummer 12 folgt ein Aufruf der Funktion *„error_overload { }“*, welche in der Datei *error_1.c* abgearbeitet wird.

Eine Visualisierung der Störung über die HMI ist selbstverständlich. Die Störung ist quittierpflichtig.

Eine entsprechende Reaktion auf die generierte Fehlernummer in error_1.c wird ausgelöst. Diese Reaktion ist mit dem Stillsetzen der Antriebe verbunden, um mechanische Beschädigungen der Antriebsachsen und Schäden an der Elektronik durch thermische Überlastung zu verhindern.

Auf die Darstellung eines Programmauszuges wird verzichtet.

Für diesen Fehler, Überlast Leistungsteil, wird die Fehlernummer 14 generiert.

9.4.25 Modul 34: Deckelschalter

Der Deckelschalter wird überprüft, durch den Anschluss eines Reedkontaktes an Port 1.5. Die Auswertung dieser Störung erfolgt in der Funktion *error_without_hut { }* in der Datei *error_1.c*.

```
255 // ------------------------------------------------------------------------
    // Fehlernummer: 15
    // Überprüfen, ob Olli seinen Hut genommen hat (der Deckel abgenommen wurde)
    void error_without_hut (void)
    {
260     err_number = 15;
        error_break(err_number);
    }
```

Durch Übergabe der generierten Fehlernummer 15 an die Funktion *error_break { }* wird eine entsprechende Reaktion auf die Fehlernummer ausgelöst.

Es erfolgt u. A. eine Visualisierung der Störung und weitere Bedienerführung auf der HMI.

Die Störung ist quittierpflichtig.

9.4.26 Modul 35: Serielle Schnittstellen

Die Programmierung der seriellen Schnittstelle S0 und S1 unterscheidet sich nur geringfügig voneinander, daher wird in den folgenden Abschnitten nur die Programmierung der seriellen Schnittstelle S0 dokumentiert.

Die Initialisierung der beiden seriellen Schnittstellen wurde bereits im Kapitel 9.4.2 zu Modul 1 behandelt.

Die beiden Schnittstellen sind nach dem Prinzip der Interruptsteuerung programmiert.

Jeder Interrupt-Request in der Datei *seriell_olli_seriell.c,* wird in der ISR *interrupt_seriell_S0 { }* abgearbeitet. In dieser Funktion wird unterschieden, ob es sich um eine Sende- oder Empfangsanforderung handelt. Je nach Anforderung wird in die entsprechende Funktion weitergesprungen.

```
//*****************************************************************************
//interrupt(4)void interrupt_seriell_S0 (void)
//Das ist die Interrupt Service Routine für Interrupts der Seriellen
//Schnittstelle S0
_interrupt(4) void interrupt_seriell_S0 (void)
{
    if ( RI0 )
    {
        seriell_S0_int_receive();
    }
    if ( TI0 )
    {
        seriell_S0_int_transmit();
    }
}
```

Das Empfangen über S0 :

Beim Empfangen wird die Funktion *seriell_S0_int_receive { }* aufgerufen. In dieser Funktion wird ein Ringpuffer verwaltet, in dem die neu empfangenen Zeichen zwischengespeichert werden.

Sobald dieser Puffer 4 Bytes Daten gespeichert hat, werden diese an die Funktion *interpreter_S0_receive { }* weitergegeben. In dieser Funktion wird der 4 Byte lange Befehlsframe auf Plausibilität geprüft.

Wenn durch diese Überprüfung kein Fehler diagnostiziert wird, wird das Bit *m_new_move_order* gesetzt.

Dem Absender wird als Bestätigung zum erfolgreichen Empfang des Befehlsframes ein „B00x“ gesandt.

Über diese Befehle erhalten Sie mehr Informationen im Kapitel 10.4.4 – Sicherheit bei der Datenübertragung.

Mit dem global gültigen Bit *m_new_move_order* wird in der Hauproutine der Datei *main.c* angezeigt, daß ein neuer Befehl zur Abarbeitung bereit steht.

Wenn bei der Plausibilitätsprüfung jedoch ein Fehler diagnostiziert wird, dann wird das dem Absender mitgeteilt. Dazu wird dem Absender der Befehlsframe „B09x“ gesandt.

Auch an dieser Stelle nochmals der Hinweis auf Kapitel 10.4.4.

Das Senden über S0 :

Bevor das Versenden eines Befehls stattfinden kann, wird in der Funktion transmit_S0_vorbereitung { } der Befehlsframe komplettiert.

Um der Hauptroutine anzuzeigen, daß ein Befehlsframe zum absenden bereit ist, wird das Bit *m_frame_s0_fertiggesendet* auf „false“ gesetzt.

```
370 //transmit_S0_vorbereitung
    //Der Aufruf dieser Funktion erfolgt aus der Funktion: "interpreter_S0_receive"
    //und "main". Der Wert der an die Funktion übergeben wird, ist der Inhalt vom
    //Code Byte2. Ein Daten-Frame besteht aus 4 Byte.
    void transmit_S0_vorbereitung(unsigned char wert)
375 {
        seriell_S0_clear_out_buffer();
        if ((wert == 0)|(wert == 7)|(wert == 8)|(wert == 9))
        {
            s0_outbuf[0] = 'B';
380         s0_outbuf[1] = '0';
            s0_outbuf[2] = wert + '0';
            s0_outbuf[3] = paritaetsberechnung(s0_outbuf[1],s0_outbuf[2]);
            m_frame_s0_fertiggesendet = false;
            index_s0_outbuf = 0;
385     }
        else
        {
            s0_outbuf[0] = 'B';
            s0_outbuf[1] = befehl_s0_in[1] + '0';
390         s0_outbuf[2] = wert + '0';
            s0_outbuf[3] = paritaetsberechnung(s0_outbuf[1],s0_outbuf[2]);
            m_frame_s0_fertiggesendet = false;
            index_s0_outbuf = 0;
        }
395 }
```

Jetzt kann mit dem eigentlichen Versand des Befehls begonnen werden. Dazu wird die Funktion *seriell_S0_int_transmit { }* aufgerufen. In dieser Funktion wird überprüft, ob der Befehlsframe bereits vollständig versandt wurde.

Kommt diese Überprüfung zu einem negativen Ergebnis, bleibt das Bit *m_frame_s0_fertiggesendet* auf „false“. Solange dieses Bit auf „false“ steht, wird über die Hauptroutine die Funktion *seriell_S0_int_transmit { }* zyklisch aufgerufen.

```
    //seriell_S0_int_transmit
180 //Der Aufruf der Funktion erfolgt aus der Funktion: "interrupt_seriell_S0"
    //Der Aufruf der "interrupt_seriell_S0" Funktion erfolgt automatisch, nachdem
    //über S0 Daten gesendet werden. Die gesendeten oder zusendeten Daten sind im
    //Array "s0_outbuf" gespeichert. Es wird überprüft ob schon alle Daten gesendet
    //wurden, wenn nicht wird der Zeiger des Arrays hochgesetzt.
185 void seriell_S0_int_transmit(void)
    {
        TI0 = 0;
        sendebussy = 0;
        if (m_frame_s0_fertiggesendet == false)
190     {
            if (s0_outbuf[index_s0_outbuf+1] == 0x00)
            //Überprüfung, ob das nächste Zeichen im Array 0x00 ist. Wenn es 0x00
            //ist, dann sind die Daten fertig gesendet worden, egal wieviel Bytes
            //es waren (max 4). Wenn das nächste Zeichen nicht 0x00 ist, wird der
195         //Zeiger (index_s0_outbuf) erhöht, und es kann weiter gesendet werden.
            //Diese Überprüfung auf das 0x00 Zeichen geht nur, da ich in der
            //"transmit_S0_vorbereitung" jedesmal das ganze s0_outbuf Array leere.
            {
                m_frame_s0_fertiggesendet = true;
200             REN0 = 1;
                //Der Empfang über S0 ist jetzt wieder eingeschaltet.
            }
            else
            {
205             index_s0_outbuf ++;
            }
        }
    }
```

Mehr Informationen zum Thema Befehlframes und ihre Bedeutung erhalten Sie im Kapitel 10 – Kommunikation.

10 Kommunikation

10.1 Allgemein

Der Mikrocontroller der Prozessorplatine verfügt über zwei serielle Schnittstellen. Beide Schnittstellen sind RS 232 – Schnittstellen.

- Serielle RS 232 – Schnittstelle S0
- Serielle RS 232 – Schnittstelle S1

Über die serielle Schnittstelle S0 ist die Anbindung von externer Software wie OLLICONTROL oder das Matlab-Programm Hasheraccelerator an den Mikrocontroller realisiert.

Über die serielle Schnittstelle S1 wurde die Verbindung des Mikrocontrollers mit dem LCD-Touchpanel als HMI projektiert.

10.2 Kommunikation: HMI zu Mikrocontroller

Über das Betätigen der Softkeys des LCD-Panels werden codierte Befehle über die Schnittstelle S1 zum Mikrocontroller gesendet.

Die Befehle setzen sich aus einem ASCII-String zusammen, der 5 Byte lang ist. Eine Übersicht der Befehlsstrings, die in der HMI abgelegt sind, zeigt Abbildung 10-1.

Kommunikation HMI -> Mikrocontroller 11.09.2003

Code Byte 0 Empfängeradr. A -> Olli	Code Byte 1 Tastennummer Ziffer 1	Code Byte 2 Tastennummer Ziffer 2	Code Byte 3 Paritätskennzeichnung Addition Byte 1 + Byte 2 Anzahl "1" gerade -> x Anzahl "1" ungerade -> y	Bildname Display	Tastenbezeichnung	Tastennummer	Kommentar
Fettdruck -> Befehl verwendet							
A	**0**	**1**	**y**	**Betriebsartenauswahl**	**Auto**	**01**	**BA Auto auswählen**
A	**0**	**2**	**y**	**Betriebsartenauswahl**	**Teilauto**	**02**	**BA Teilauto auswählen**
A	**0**	**3**	**x**	**Betriebsartenauswahl**	**Hand**	**03**	**BA Hand auswählen**
A	0	4	y	Betriebsartenauswahl	restart	04	LCD Neuinitialisierung
A	**0**	**5**	**x**	**Automatik**	**Automatik Freigabe**	**05**	**Automatikablauf Freigabe**
A	**0**	**6**	**x**	**Teilautomatik**	**Position 1**	**06**	**BA TA Position 1 anfahren**
A	**0**	**7**	**y**	**Teilautomatik**	**Position 2**	**07**	**BA TA Position 2 anfahren**
A	**0**	**8**	**y**	**Teilautomatik**	**Position 3**	**08**	**BA TA Position 3 anfahren**
A	**0**	**9**	**x**	**Teilautomatik**	**Position 4**	**09**	**BA TA Position 4 anfahren**
A	**1**	**0**	**y**	**Teilautomatik**	**Position 5**	**10**	**BA TA Position 5 anfahren**
A	**1**	**1**	**y**	**Teilautomatik**	**Grundstellung**	**11**	**BA TA Grundstellung anfahren**
A	1	2	x	Handbetrieb	Achse 1	12	Bild BA Hand Achse 1 auswählen
A	1	3	y	Handbetrieb	Achse 2	13	Bild BA Hand Achse 2 auswählen
A	1	4	x	Handbetrieb	Achse 3	14	Bild BA Hand Achse 3 auswählen
A	1	5	x	Handbetrieb	Achse 4	15	Bild BA Hand Achse 4 auswählen
A	**1**	**6**	**y**	**Handbetrieb Achse 1**	**Richtung +**	**16**	**BA Hand A1 in Richtung + fahren**
A	**1**	**7**	**y**	**Handbetrieb Achse 1**	**Richtung -**	**17**	**BA Hand A1 in Richtung - fahren**
A	**1**	**8**	**x**	**Handbetrieb Achse 2**	**Richtung +**	**18**	**BA Hand A2 in Richtung + fahren**
A	**1**	**9**	**x**	**Handbetrieb Achse 2**	**Richtung -**	**19**	**BA Hand A2 in Richtung - fahren**
A	**2**	**0**	**y**	**Handbetrieb Achse 3**	**Richtung +**	**20**	**BA Hand A3 in Richtung + fahren**
A	**2**	**1**	**x**	**Handbetrieb Achse 3**	**Richtung -**	**21**	**BA Hand A3 in Richtung - fahren**
A	**2**	**2**	**y**	**Handbetrieb Achse 4**	**Richtung +**	**22**	**BA Hand A4 in Richtung + fahren**
A	**2**	**3**	**x**	**Handbetrieb Achse 4**	**Richtung -**	**23**	**BA Hand A4 in Richtung - fahren**
A	**2**	**4**	**x**	**Stoerung 1**	**Störung quittieren**	**24**	**Störung quittieren**
A	**2**	**5**	**y**	**Stoerung 2**	**Achse freifahren**	**25**	**Achse von Endlage fahren**

(Abb. 10-1)

Die Übersicht der Befehlsstrings finden Sie auch auf der Dokumentations – CDR als Datei im pdf-Format.

Jedem Softkey der HMI ist ein Befehlsstring zugeordnet. Wird der Softkey betätigt, läuft in der HMI ein Makro ab, das den 4 Byte langen Befehlsstring über die serielle Schnittstelle S1 zum Mikrocontroller sendet.

Als Besonderheit sendet das LCD-Display als erstes Byte immer den Wert 0x04. Dieses Byte wird dem CodeByte 0 vorangestellt, so daß sich immer ein 5 Byte langer Befehl ergibt.

Die Software des Mikrocontrollers decodiert den empfangenen Befehl und löst eine entsprechende Reaktion aus.

Als eigentliches Informationsbyte dient CodeByte 1 und CodeByte 2. Zur Bedeutung der einzelnen CodeBytes im Befehlsstring lesen Sie Kapitel 10.2.1.

Zur Programmierung der HMI lesen Sie Kapitel 11 – Details zur Programmierung der HMI. Dort erhalten Sie auch Informationen zur Programmierung mit Makros.

10.2.1 Aufbau eines Befehls

Wie Sie Kapitel 10.2 entnehmen können, besteht ein Befehl, der von der HMI zum Mikrocontroller gesendet wird, aus 5 Byte.

Befehlsbyte START:

Das LCD-Touchpanel sendet als Startbyte zur Datenübertragung immer den Wert 0x04. Dieses Startbyte muß nicht explizit programmiert werden.

Im entsprechenden Makro zur Versendung der Befehlsstrings werden nur folgende 4 Bytes programmiert. Der Befehl wird im Makro als 4 Byte Befehl deklariert. Wie auch in Kapitel 10.2 hier der Verweis auf Kapitel 11 – Details zur Programmierung mit Makros.

Befehlsbyte 1 (entspricht in Abbildung 10-1 CodeByte 0) **:**

Ein Großbuchstabe wird als Empfängeradresse verwendet.

- Empfängeradresse A entspricht dem Mikrocontroller
- Empfängeradresse B entspricht OLLICONTROL / Hasheraccelerator

Die Empfängeradresse B wird hier nur der Vollständigkeit wegen aufgeführt. Ein Befehl seitens der HMI mit der Empfängeradresse ist nicht möglich. Diese Adresse wird nur in der Kommunikation über die serielle Schnittstelle S0 verwendet.

Befehlsbyte 2 (entspricht in Abbildung 10-1 CodeByte 1) **:**

Die Softkeys werden nach ihrer Programmierung durchnummeriert. Die HMI wurde projektiert mit Auslegung auf 99 Softkeys.

- Befehlsbyte 2 entspricht der Zehnerstelle des programmierten Softkeys, also der ersten Ziffer der Nummer des Softkeys.

Befehlsbyte 3 (entspricht in Abbildung 10-1 CodeByte 2) **:**

- Befehlsbyte 3 entspricht der Einerstelle des programmierten Softkeys, also der zweiten Ziffer der Nummer des Softkeys.

Befehlsbyte 4 (entspricht in Abbildung 10-1 CodeByte 3) **:**

Dieses Byte wird zur Überprüfung auf korrekt empfangenen Befehl mitgesendet. Die Software des Mikrocontrollers führt eine Paritätsprüfung durch. Folgende Festlegungen wurden zur Paritätsprüfung getroffen:

- Als Befehlsbyte 4 wird ein „x“ gesendet, wenn die Anzahl der Einser nach Addition von Befehlsbyte 2 und Befehlsbyte 3, gerade ist.

- Als Befehlsbyte 4 wird ein „y“ gesendet, wenn die Anzahl der Einser nach Addition von Befehlsbyte 2 und Befehlsbyte 3, ungerade ist.

Die Paritätsprüfung findet im Mikrocontroller statt. Wird ein empfangener Befehl als falsch erkannt, löst die Software des Mikrocontrollers eine entsprechende Reaktion aus. Mehr Informationen hierzu erhalten Sie im Kapitel 9 – Programmierung des Mikrocontrollers.

Die Berechnungen zur Paritätsprüfung wurden seitens der HMI bei der Projektierung durchgeführt und bereits als Befehlsstring im EEprom der HMI abgelegt.

10.3 Kommunikation: Mikrocontroller zu HMI

Die Oberflächen der HMI zur Bedienung des Messautomaten wurden in Form von Makros im EEprom der HMI gespeichert.

Der Aufruf einer Oberfläche geschieht nun durch das Starten eines Makros. Die Oberflächen sind als sogenannte Normalmakros in der HMI abrufbar.

Lesen Sie hierzu Kapitel 11 – Details zur Programmierung der HMI.

10.3.1 Aufbau eines Befehls

Zum Aufruf eines Makros in der HMI wird ein 5 Byte langer Befehlsstring benötigt. Der Befehl zum Aufruf eines Normalmakros setzt sich wie folgt zusammen:

Befehlsbyte 0 :

- Als Startzeichen wird immer der Wert 0x1B gesendet. Dieser Wert entspricht dem „esc“ bzw. # - Zeichen, mit dem grundsätzlich ein Befehl der textbasierenden Programmiersprache, in der das LCD – Touchpanel zu programmieren ist, eingeleitet wird.

Befehlsbyte 1 :

- Als Befehlsbyte 1 wird ein ‚M’ gesendet. ‚M’ steht für den Aufruf eines Makros.

Befehlsbyte 2 :

- Als Befehlsbyte 2 wird ein ‚N’ gesendet. ‚N’ steht in Kombination mit dem Befehlsbyte 1 für den Aufruf eines Normalmakros.

Befehlsbyte 3 :

- Als Befehlsbyte 3 wird die Nummer des aufzurufenden Makros gesendet. Befehlsbyte 3 entspricht der Nummer, mit welcher das Makro im EEprom des LCD-Displays abgelegt wurde.

- Befehlsbyte 3 ist eine Zahl zwischen 0x00 und 0xFF. 256 Normalmakros können im Speicher des EEproms abgelegt werden.

Befehlsbyte 4 :

- Als Abschlusszeichen wird im Befehlsbyte 4 immer der Wert 0x0D gesendet.

10.4 Kommunikation: Externe Software und Mikrocontroller

10.4.1 Allgemein

Die Kommunikation zwischen externer Software wie OLLICONTROL oder dem Matlabprogramm Hasheraccelerator mit dem Mikrocontroller findet über die serielle Schnittstelle S0 statt.

Der Befehlsframe ist nach dem gleichen Prinzip wie der Frame zur seriellen Schnittstelle S1 aufgebaut. Der Unterschied besteht darin, daß bei der Schnittstelle S0 ein Befehlsframe 4 Byte lang ist.

10.4.2 Übersicht der Befehle

Die Übersicht der möglichen Befehle, die von einer externen Software wie OC oder dem Hasheraccelerator gesendet werden, ist in Abbildung 10-2 dargestellt.

		Olli					Hals			Achsen fahren				mechanische Position					
		in Fahrtrichtung		Seitenneigung															
Fahrersitz		vorne	hinten	links	mittig	rechts	oben	mittig	unten	A 1	A 2	A 3	A 4	P 1	P 2	P 3	P 4	P 5	Grd/P6
1	A01y		x		x			x					x	x					
2	A02y		x			x		x		x			x		x				
3	A03x		x	x				x				x	x			x			
4	A04y	x			x				x		x						x		
5	A05x		x		x		x						x					x	
6	A06x		x		x				x				x						x
Beifahrersitz																			
1	A11y		x		x			x					x	x					
2	A12x		x			x		x		x			x		x				
3	A13y		x	x				x				x	x			x			
4	A14x	x							x		x						x		
5	A15x		x		x		x						x					x	
6	A16y		x		x				x				x						x
hinter Fahrersitz																			
1	A21x		x		x			x					x	x					
2	A22y		x			x		x		x			x		x				
3	A23x		x	x				x				x	x			x			
4	A24x	x							x		x						x		
5	A25y		x		x		x						x					x	
6	A26y		x		x				x				x						x
hinter Beifahrersitz																			
1	A31y		x		x			x					x	x					
2	A32x		x			x		x		x			x		x				
3	A33x		x	x				x				x	x			x			
4	A34y	x							x		x						x		
5	A35y		x		x		x						x					x	
6	A36x		x		x				x				x						x

(Abb. 10-2)

Welche mechanische Position der Messautomat in Abhängigkeit der Befehle anfährt und welche Achsen dazu Verfahrbewegungen auszuführen haben, ist in Abbildung 10-2 ebenso ersichtlich.

10.4.3 Aufbau eines Befehls

Wie bereits in Kapitel 10.4.1 darauf hingewiesen, ist ein Befehlframe 4 Byte lang. Der folgende Abschnitt informiert über die Bedeutung der einzelnen Bytes im Frame.
Befehlsbyte 1 :

Ein Großbuchstabe wird als Empfängeradresse verwendet.

- Empfängeradresse A entspricht dem Mikrocontroller
- Empfängeradresse B entspricht OLLICONTROL / Hasheraccelerator

Die Empfängeradresse B ist in der Übersicht der Befehle in Abbildung 10-2 nicht ersichtlich. Mehr Informationen dazu erhalten Sie in Kapitel 10.4.4 – Sicherheit bei der Datenübertragung.

Befehlsbyte 2 :

Im Befehlsbyte 2 ist der Sitzplatz im Kraftfahrzeug hinterlegt, auf dem der Messautomat positioniert ist.

- Wert „0“ entspricht dem Fahrersitz
- Wert „1“ entspricht dem Beifahrersitz
- Wert „2“ entspricht dem Sitz hinter dem Fahrersitz
- Wert „3“ entspricht dem Sitz hinter dem Beifahrersitz

Befehlsbyte 3 :

Im Befehlsbyte 3 ist die angeforderte Position hinterlegt, die der Messautomat anfahren soll. 6 Positionen sind zur Auswahl möglich.

- Wert „1“ entspricht der mechanischen Position 1
- Wert „2“ entspricht der mechanischen Position 2
- Wert „3“ entspricht der mechanischen Position 3
- Wert „4“ entspricht der mechanischen Position 4
- Wert „5“ entspricht der mechanischen Position 5
- Wert „6“ entspricht der mechanischen Position 6, der Grundstellung

Befehlsbyte 4 :

Dieses Byte wird zur Überprüfung auf korrekt empfangenen Befehl mitgesendet. Durch dieses Befehlsbyte findet eine Paritätsprüfung statt. Folgende Festlegungen wurden zur Paritätsprüfung getroffen:

- Als Befehlsbyte 4 wird ein „x“ gesendet, wenn die Anzahl der Einser nach Addition von Befehlsbyte 2 und Befehlsbyte 3, gerade ist.
- Als Befehlsbyte 4 wird ein „y“ gesendet, wenn die Anzahl der Einser nach Addition von Befehlsbyte 2 und Befehlsbyte 3, ungerade ist.

Informationen zu den Folgen eines diagnostizierten Datenübertragungsfehler erhalten Sie in Kapitel 10.4.4 – Sicherheit bei der Datenübertragung.

Im Kapitel 9.4.26 erhalten Sie Detailinformationen zur Programmierung der seriellen Schnittstellen S0 und S1.

10.4.4 Sicherheit bei der Datenübertragung

Befehlsframe				
Byte 1	**Byte 2**	**Byte 3**	**Byte 4**	**Funktion**
B	0	0	x	Mikrocontroller an ext. SW: "Nachricht verstanden"
B	0	7	y	Mikrocontroller an ext. SW: "Freigabe entzogen"
B	0	8	y	Mikrocontroller an ext. SW: "Freigabe erteilt"
B	0	9	x	Mikrocontroller an ext. SW: "Nachricht nicht verstanden"

(Abb. 10-3)

Jeder Befehl, der an den Mikrocontroller gesendet wurde, wird auf seine Plausibilität und einer Paritätsprüfung unterzogen.

Die Tabelle in Abbildung 10-3 gibt Aufschluss darüber, mit welchen Befehlen auf eine Bewegungsanforderung geantwortet wird.

Zwei Möglichkeiten stehen zur Auswahl:

- Der Mikrocontroller hat den Befehl verstanden und bestätigt das mit dem Befehlsframe „B00x“.
- Der Mikrocontroller hat den Befehl nicht verstanden und meldet das dem Absender mit dem Befehlsframe „B09x“.

Der Aufbau dieses Befehlsframe ist gleich dem Aufbau des dargestellten Frames in Kapitel 10.4.2.

Befehlsbyte 1 :

Ein Großbuchstabe wird als Empfängeradresse verwendet.

- Empfängeradresse B entspricht OLLICONTROL / Hasheraccelerator

Befehlsbyte 2 :

- Erste Stelle des Informationsteils.

Befehlsbyte 3 :

- Zweite Stelle des Informationsteils.

Befehlsbyte 4 :

Dieses Byte wird zur Überprüfung auf korrekt empfangenen Befehl mitgesendet. Durch dieses Befehlsbyte findet eine Paritätsprüfung statt. Folgende Festlegungen wurden zur Paritätsprüfung getroffen:

- Als Befehlsbyte 4 wird ein „x“ gesendet, wenn die Anzahl der Einser nach Addition von Befehlsbyte 2 und Befehlsbyte 3, gerade ist.

Als Befehlsbyte 4 wird ein „y“ gesendet, wenn die Anzahl der Einser nach Addition von Befehlsbyte 2 und Befehlsbyte 3, ungerade ist.

10.4.5 Sicherheit beim Prozess

In Kapitel 12.3 erhalten Sie Informationen zur Bedienung des Messautomaten durch externe Software. Dort wird darauf hingewiesen, daß die HMI die dominante Schnittstelle zur Bedienung des Automaten ist.

Bei diagnostizierten Fehlern wird externer Software die Freigabe zur Bedienung des Messautomaten entzogen. Ist die Störung erfolgreich beseitigt und quittiert, wird die Freigabe zur Bedienung wieder erteilt.

Dieser Vorgang des Entzugs und der Erteilung der Freigabe dient der Prozesssicherheit des Systems und gelingt mit folgenden zwei Befehlen:

- „B07y“ entspricht „Freigabe entzogen“
- „B08y“ entspricht „Freigabe erteilt“

Es wird darauf verzichtet, den Befehlsframe hier nochmals darzustellen. Dieser Befehlsframe entspricht dem in Kapitel 10.4.4 dargestelltem Frame.

11 Details zur Programmierung der HMI

11.1 Allgemeines

Erste Informationen zum LCD-Touchpanel erhalten Sie in Kapitel 6.2.5. Auf die Darstellung technischer Daten wird hier deshalb verzichtet.

Das Display ist mit hochsprachenähnlichen Grafikbefehlen zu programmieren. Die Programmierung erfolgt textbasierend, eine grafische Programmierung ist nicht möglich.

Zum Programmieren des Displays ist ein ASCII - Editor ausreichend. Dieses Projekt wurde mit dem Editor UltraEdit 32 programmiert.

Die Programmierung erfolgt ausschließlich durch Programmierung von Makros. Drei Arten von Makros werden hierzu unterschieden :

- Normalmakros
- Touchmakros
- Portmakros

Portmakros wurden in diesem Projekt nicht verwendet.

11.2 Ablaufdiagramme der HMI

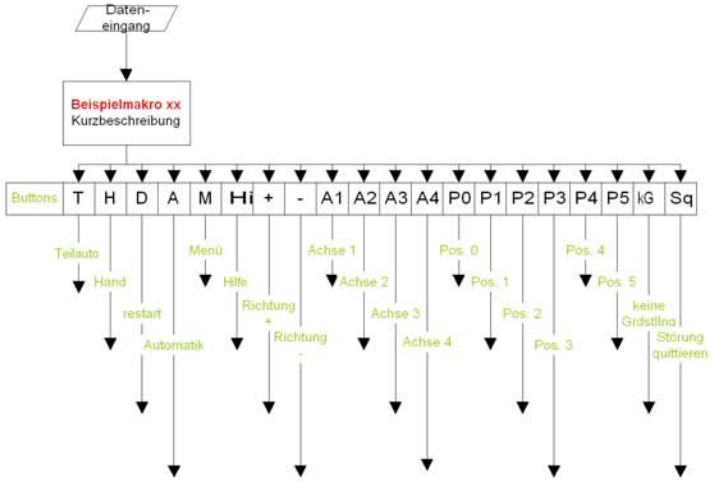
Auf den folgenden Seiten werden die Zusammenhänge der einzelnen Makros in Ablaufdiagrammen dargestellt.

Das Prinzip ist bereits aus den Ablaufdiagrammen zum C-Projekt bekannt und wird hier nicht weiter erläutert.

LCD - Display

Ablaufdiagramm der Makros

Legende :



LCD - Display	
Ablaufdiagramm der Makros	
	Blatt 1.0

(Abb. 11-1)

Kurzbeschreibung der Makros :

Normalmakros

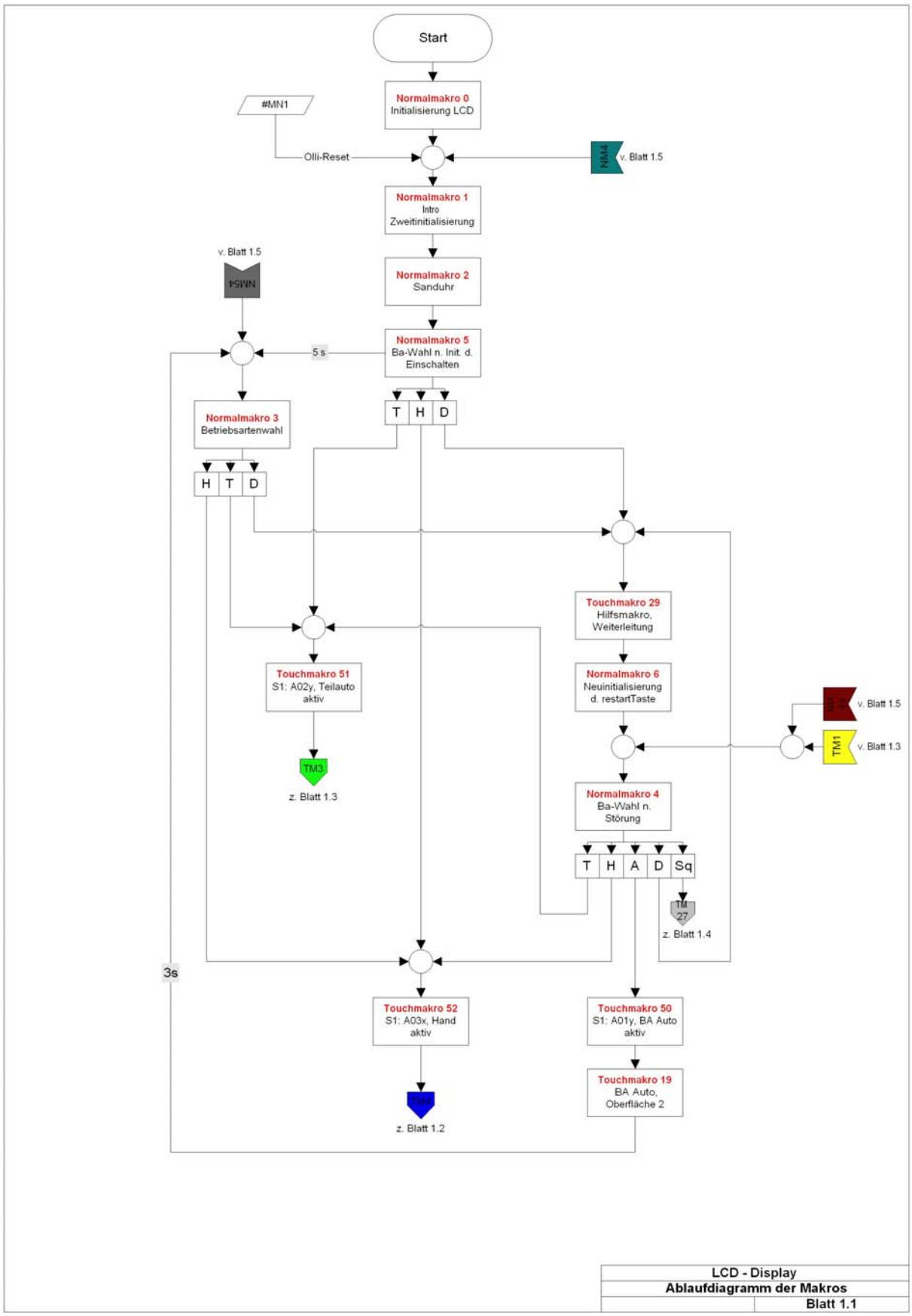
Makro 0:	Startmakro, Initialisierung
Makro 1:	Intro, Startbildschirm
Makro 2:	Intro Sanduhr
Makro 3:	Aufruf Betriebsartenwahl
Makro 4:	Aufruf Betriebsartenwahl nach Störung
Makro 5:	Betriebsartenwahl nach Initialisierung des Displays
Makro 6:	Neuinitialisierung des Displays durch "restart-Taste"
Makro 10:	Grundmaske 1 Störbild
Makro 11:	Grundmaske 2 Störbild
Makro 12:	Grundmaske 3 Störbild
Makro 20:	Störung Paarfehler Endschalter Achse 1
Makro 21:	Störung Paarfehler Endschalter Achse 2
Makro 22:	Störung Paarfehler Endschalter Achse 3
Makro 23:	Störung Paarfehler Endschalter Achse 4
Makro 24:	Störung Endlage + angefahren, Achse 1
Makro 25:	Störung Endlage + angefahren, Achse 2
Makro 26:	Störung Endlage + angefahren, Achse 3
Makro 27:	Störung Endlage + angefahren, Achse 4
Makro 28:	Störung Endlage - angefahren, Achse 1
Makro 29:	Störung Endlage - angefahren, Achse 2
Makro 30:	Störung Endlage - angefahren, Achse 3
Makro 31:	Störung Endlage - angefahren, Achse 4
Makro 32:	Störung Überlast, Achse 1
Makro 33:	Störung Überlast, Achse 2
Makro 34:	Störung Überlast, Achse 3
Makro 35:	Störung Überlast, Achse 4
Makro 36:	Störung keine Grundstellung
Makro 37:	Meldung Grundstellung schon vorhanden
Makro 38:	Meldung BA TeilAuto Position 1 erreicht
Makro 39:	Meldung BA TeilAuto Position 2 erreicht
Makro 40:	Meldung BA TeilAuto Position 3 erreicht
Makro 41:	Meldung BA TeilAuto Position 4 erreicht
Makro 42:	Meldung BA TeilAuto Position 5 erreicht
Makro 43:	Meldung BA umschalten nur bei inaktiven Achsen möglich
Makro 44:	Störung Olli aufrichten
Makro 45:	Warnung Olli aufrichten
Makro 46:	Störung Deckel nicht abgenommen
Makro 50:	Meldung BA Auto, P1 erreicht
Makro 51:	Meldung BA Auto, P2 erreicht
Makro 52:	Meldung BA Auto, P3 erreicht
Makro 53:	Meldung BA Auto, P4 erreicht
Makro 54:	Meldung BA Auto, P5 erreicht
Makro 55:	Meldung BA Auto, P1-Fahrt aktiv
Makro 56:	Meldung BA Auto, P2-Fahrt aktiv
Makro 57:	Meldung BA Auto, P3-Fahrt aktiv
Makro 58:	Meldung BA Auto, P4-Fahrt aktiv
Makro 59:	Meldung BA Auto, P5-Fahrt aktiv
Makro 60:	Meldung BA Auto, P6-Fahrt aktiv
Makro 61:	Meldung BA TeilAuto, P1-Fahrt aktiv
Makro 62:	Meldung BA TeilAuto, P2-Fahrt aktiv
Makro 63:	Meldung BA TeilAuto, P3-Fahrt aktiv
Makro 64:	Meldung BA TeilAuto, P4-Fahrt aktiv
Makro 65:	Meldung BA TeilAuto, P5-Fahrt aktiv
Makro 66:	Meldung BA TeilAuto, P6-Fahrt aktiv

Touchmakros

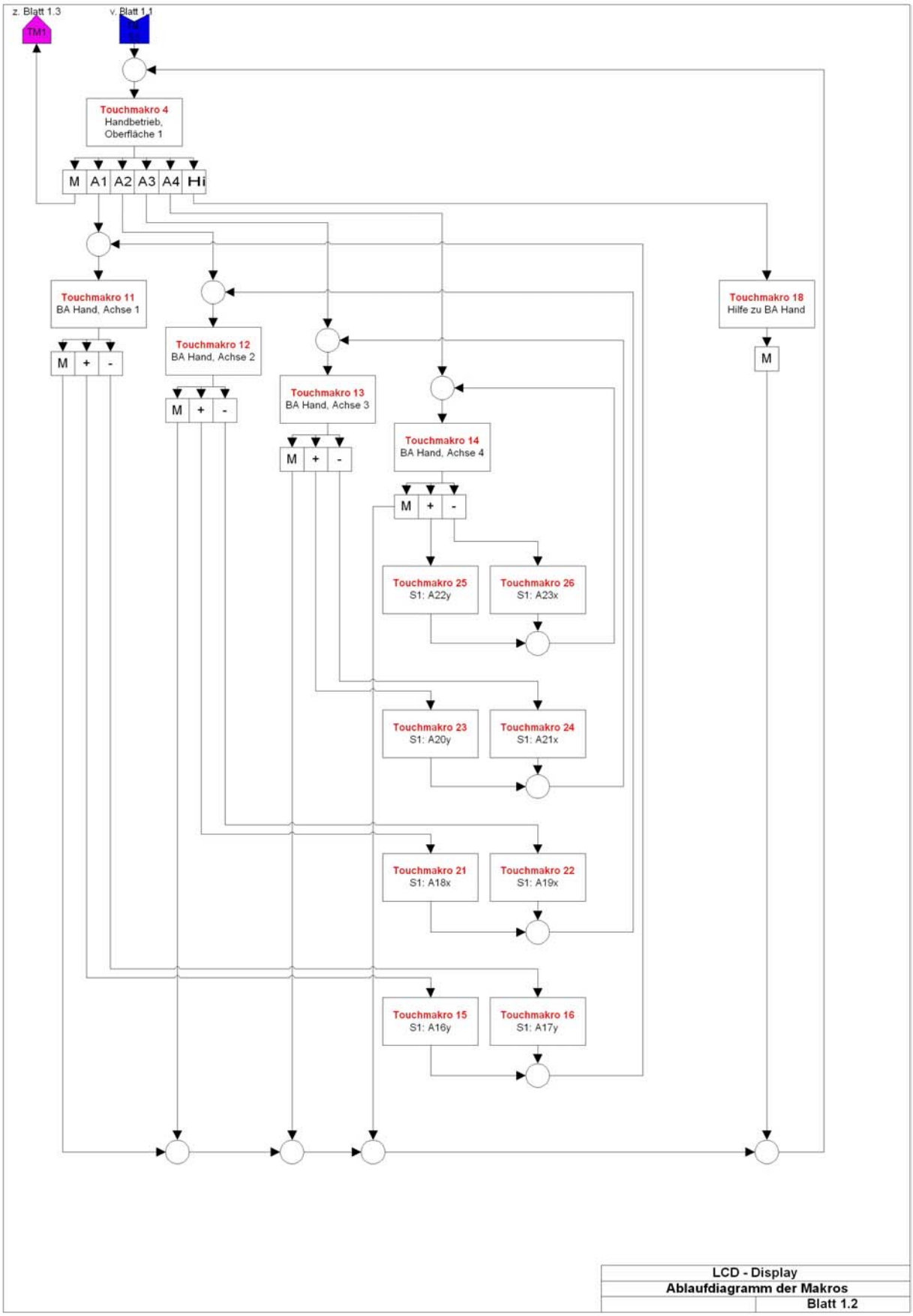
Touchmakro 1:	Hilfsmakro zu Aufruf Makro2, Betriebsartenwahl
Touchmakro 2:	Automatikablauf Oberfläche 1
Touchmakro 3:	Teilautomatik Oberfläche
Touchmakro 4:	Handbetrieb Oberfläche 1
Touchmakro 5:	Teilautomatik Taste Position 4 anfahren
Touchmakro 6:	Teilautomatik Taste Position 1 anfahren
Touchmakro 7:	Teilautomatik Taste Position 5 anfahren
Touchmakro 8:	Teilautomatik Taste Position 3 anfahren
Touchmakro 9:	Teilautomatik Taste Position 2 anfahren
Touchmakro 10:	Teilautomatik Taste Position 0, Grundstellung anfahren
Touchmakro 11:	Handbetrieb Oberfläche 2 - Achse 1
Touchmakro 12:	Handbetrieb Oberfläche 3 - Achse 2
Touchmakro 13:	Handbetrieb Oberfläche 4 - Achse 3
Touchmakro 14:	Handbetrieb Oberfläche 5 - Achse 4
Touchmakro 15:	Handbetrieb Taste Richtung +, Achse 1
Touchmakro 16:	Handbetrieb Taste Richtung -, Achse 1
Touchmakro 17:	Hilfe zu Teilautomatikoberfläche 1
Touchmakro 18:	Hilfe zu Handbetrieboberfläche 1
Touchmakro 19:	Automatikablauf Oberfläche 2
Touchmakro 20:	Hilfe zu Teilautomatikoberfläche 2
Touchmakro 21:	Handbetrieb Taste Richtung +, Achse 2
Touchmakro 22:	Handbetrieb Taste Richtung -, Achse 2
Touchmakro 23:	Handbetrieb Taste Richtung +, Achse 3
Touchmakro 24:	Handbetrieb Taste Richtung -, Achse 3
Touchmakro 25:	Handbetrieb Taste Richtung +, Achse 4
Touchmakro 26:	Handbetrieb Taste Richtung -, Achse 4
Touchmakro 27:	Störung quittieren
Touchmakro 28:	serielle Schnittstelle an Olli, Grdstllg fahren
Touchmakro 29:	Aufruf reset/restart, Neuinitialisierung
Touchmakro 30:	Achse nach Endlagenfahrt freifahren
Touchmakro 50:	serielle Schnittstelle an Olli, BA Auto Freigabe aktiv
Touchmakro 51:	serielle Schnittstelle an Olli, BA TeilAuto aktiv
Touchmakro 52:	serielle Schnittstelle an Olli, BA Hand aktiv

LCD - Display	
Ablaufdiagramm der Makros	
	Blatt 1.0.1

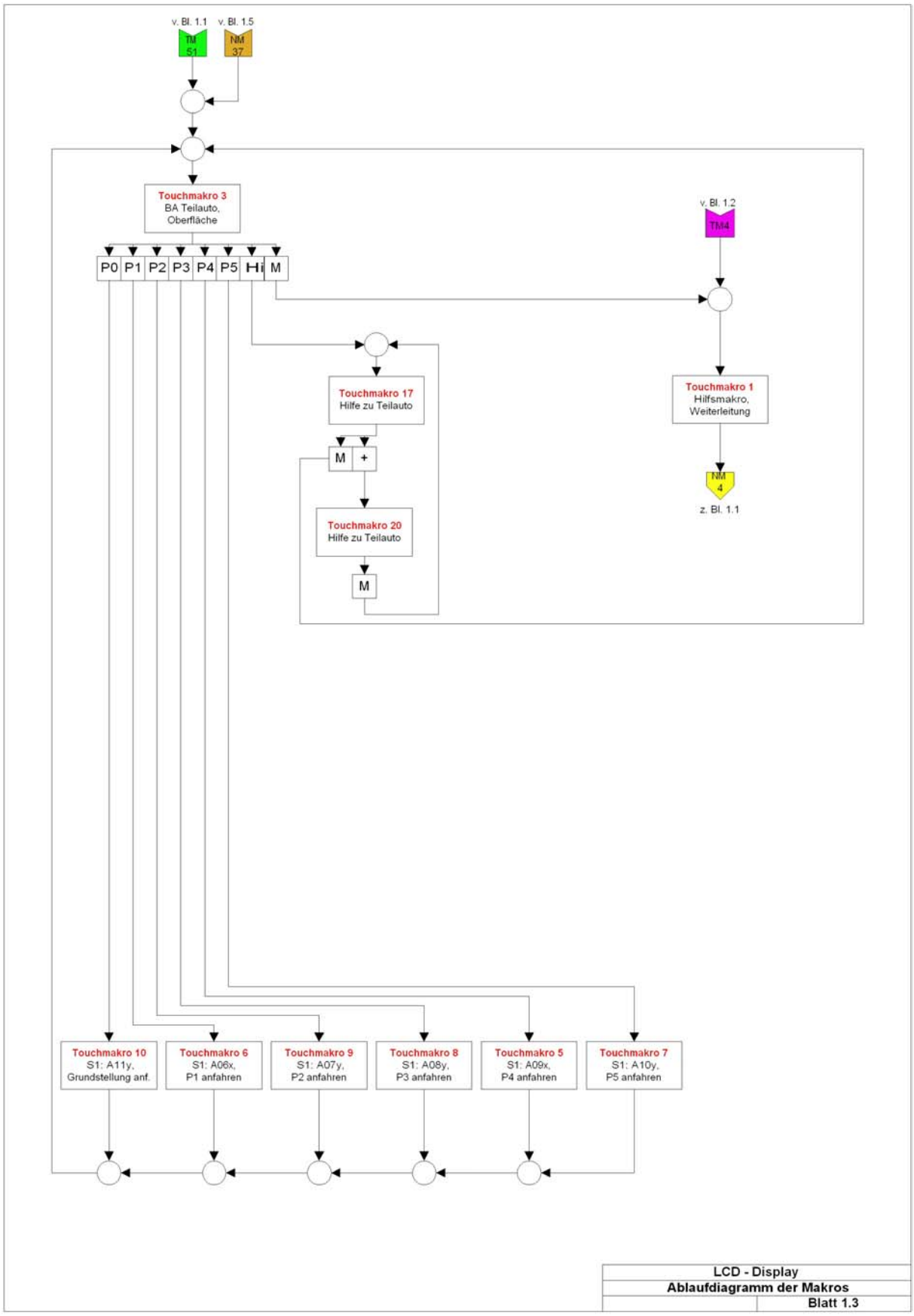
(Abb. 11-2)


Start
Normalmakro 0
Initialisierung LCD
#MN1
Olli-Reset
v. Blatt 1.5
Normalmakro 1
Intro
Zweitinitialisierung
v. Blatt 1.5
Normalmakro 2
Sanduhr
Normalmakro 5
Ba-Wahl n. Init. d.
Einschalten
5 s
T H D
Normalmakro 3
Betriebsartenwahl
H T D
Touchmakro 29
Hilfsmakro,
Weiterleitung
Touchmakro 51
S1: A02y, Teilauto
aktiv
TM3
z. Blatt 1.3
Normalmakro 6
Neuinitialisierung
d. restartTaste
v. Blatt 1.5
v. Blatt 1.3
Normalmakro 4
Ba-Wahl n.
Störung
T H A D Sq
TM
27
z. Blatt 1.4
3s
Touchmakro 52
S1: A03x, Hand
aktiv
Touchmakro 50
S1: A01y, BA Auto
aktiv
Touchmakro 19
BA Auto,
Oberfläche 2
z. Blatt 1.2
LCD - Display
Ablaufdiagramm der Makros
Blatt 1.1

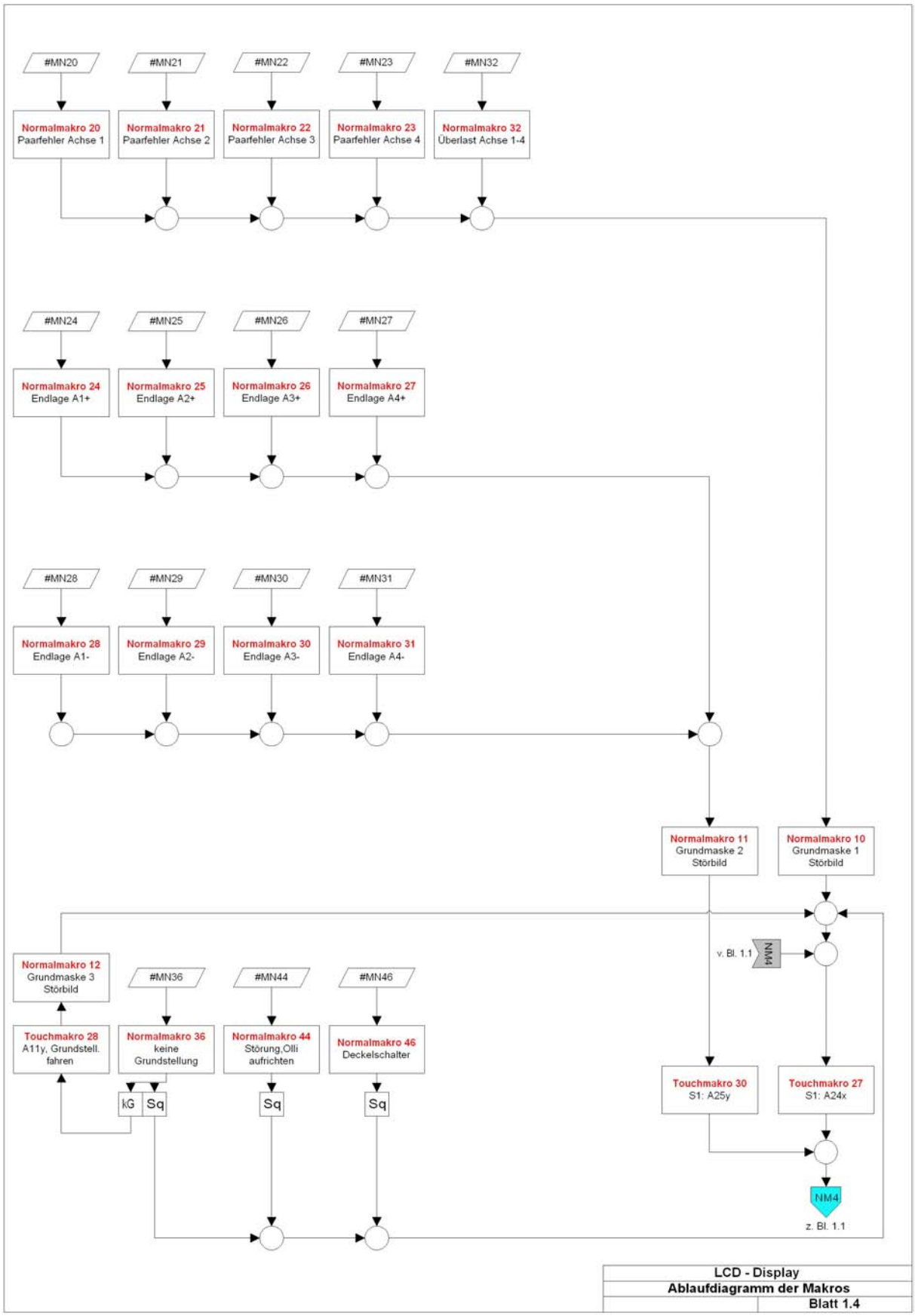
(Abb. 11-3)


z. Blatt 1.3
TM1
v. Blatt 1.1
Touchmakro 4
Handbetrieb,
Oberfläche 1
M A1 A2 A3 A4 Hi
Touchmakro 11
BA Hand, Achse 1
M + -
Touchmakro 12
BA Hand, Achse 2
M + -
Touchmakro 13
BA Hand, Achse 3
M + -
Touchmakro 14
BA Hand, Achse 4
M + -
Touchmakro 18
Hilfe zu BA Hand
M
Touchmakro 25
S1: A22y
Touchmakro 26
S1: A23x
Touchmakro 23
S1: A20y
Touchmakro 24
S1: A21x
Touchmakro 21
S1: A18x
Touchmakro 22
S1: A19x
Touchmakro 15
S1: A16y
Touchmakro 16
S1: A17y
LCD - Display
Ablaufdiagramm der Makros
Blatt 1.2

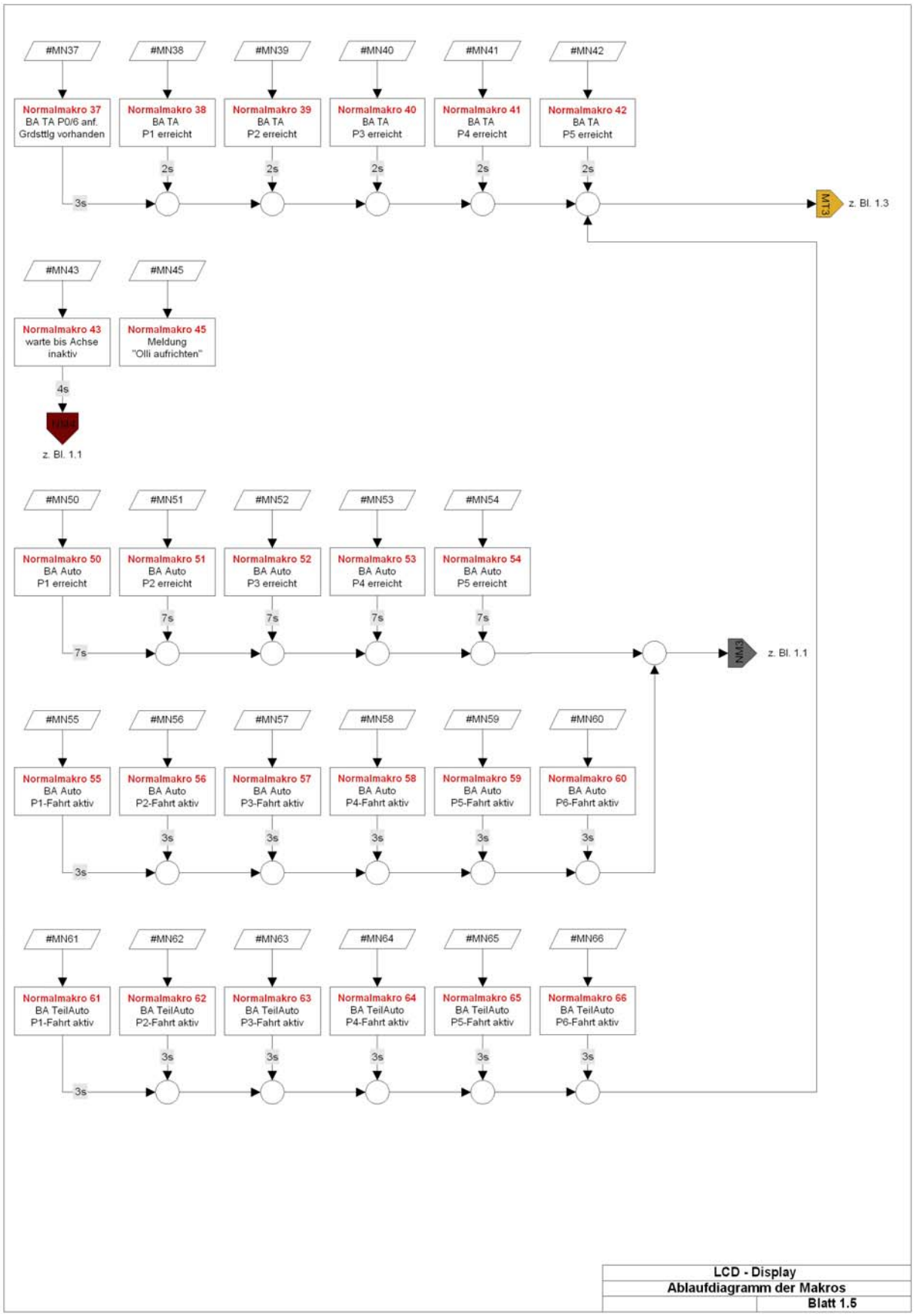
(Abb. 11-4)


v. Bl. 1.1
TM 51
v. Bl. 1.5
NM 37
Touchmakro 3
BA Teilauto, Oberfläche
P0 P1 P2 P3 P4 P5 Hi M
v. Bl. 1.2
TM4
Touchmakro 17
Hilfe zu Teilauto
M +
Touchmakro 20
Hilfe zu Teilauto
M
Touchmakro 1
Hilfsmakro, Weiterleitung
NM 4
z. Bl. 1.1
Touchmakro 10
S1: A11y, Grundstellung anf.
Touchmakro 6
S1: A06x, P1 anfahren
Touchmakro 9
S1: A07y, P2 anfahren
Touchmakro 8
S1: A08y, P3 anfahren
Touchmakro 5
S1: A09x, P4 anfahren
Touchmakro 7
S1: A10y, P5 anfahren
LCD - Display
Ablaufdiagramm der Makros
Blatt 1.3

(Abb. 11-5)


#MN20
#MN21
#MN22
#MN23
#MN32
Normalmakro 20
Paarfehler Achse 1
Normalmakro 21
Paarfehler Achse 2
Normalmakro 22
Paarfehler Achse 3
Normalmakro 23
Paarfehler Achse 4
Normalmakro 32
Überlast Achse 1-4
#MN24
#MN25
#MN26
#MN27
Normalmakro 24
Endlage A1+
Normalmakro 25
Endlage A2+
Normalmakro 26
Endlage A3+
Normalmakro 27
Endlage A4+
#MN28
#MN29
#MN30
#MN31
Normalmakro 28
Endlage A1-
Normalmakro 29
Endlage A2-
Normalmakro 30
Endlage A3-
Normalmakro 31
Endlage A4-
Normalmakro 11
Grundmaske 2
Störbild
Normalmakro 10
Grundmaske 1
Störbild
v. Bl. 1.1
NM4
Normalmakro 12
Grundmaske 3
Störbild
#MN36
#MN44
#MN46
Touchmakro 28
A11y, Grundstell. fahren
Normalmakro 36
keine Grundstellung
Normalmakro 44
Störung,Olli aufrichten
Normalmakro 46
Deckelschalter
kG
Sq
Sq
Sq
Touchmakro 30
S1: A25y
Touchmakro 27
S1: A24x
NM4
z. Bl. 1.1
LCD - Display
Ablaufdiagramm der Makros
Blatt 1.4

(Abb. 11-6)


#MN37
Normalmakro 37
BA TA P0/6 anf. Grdsttlg vorhanden
3s
#MN38
Normalmakro 38
BA TA P1 erreicht
2s
#MN39
Normalmakro 39
BA TA P2 erreicht
2s
#MN40
Normalmakro 40
BA TA P3 erreicht
2s
#MN41
Normalmakro 41
BA TA P4 erreicht
2s
#MN42
Normalmakro 42
BA TA P5 erreicht
2s
MT3
z. Bl. 1.3
#MN43
Normalmakro 43
warte bis Achse inaktiv
4s
MN44
z. Bl. 1.1
#MN45
Normalmakro 45
Meldung "Olli aufrichten"
#MN50
Normalmakro 50
BA Auto P1 erreicht
7s
#MN51
Normalmakro 51
BA Auto P2 erreicht
7s
#MN52
Normalmakro 52
BA Auto P3 erreicht
7s
#MN53
Normalmakro 53
BA Auto P4 erreicht
7s
#MN54
Normalmakro 54
BA Auto P5 erreicht
7s
MN3
z. Bl. 1.1
#MN55
Normalmakro 55
BA Auto P1-Fahrt aktiv
3s
#MN56
Normalmakro 56
BA Auto P2-Fahrt aktiv
3s
#MN57
Normalmakro 57
BA Auto P3-Fahrt aktiv
3s
#MN58
Normalmakro 58
BA Auto P4-Fahrt aktiv
3s
#MN59
Normalmakro 59
BA Auto P5-Fahrt aktiv
3s
#MN60
Normalmakro 60
BA Auto P6-Fahrt aktiv
3s
#MN61
Normalmakro 61
BA TeilAuto P1-Fahrt aktiv
3s
#MN62
Normalmakro 62
BA TeilAuto P2-Fahrt aktiv
3s
#MN63
Normalmakro 63
BA TeilAuto P3-Fahrt aktiv
3s
#MN64
Normalmakro 64
BA TeilAuto P4-Fahrt aktiv
3s
#MN65
Normalmakro 65
BA TeilAuto P5-Fahrt aktiv
3s
#MN66
Normalmakro 66
BA TeilAuto P6-Fahrt aktiv
3s
LCD - Display
Ablaufdiagramm der Makros
Blatt 1.5

(Abb. 11-7)

11.3 Details zum Quelltext

Zum besseren Verständnis des Quelltextes, in Abbildung 11-8 und Abbildung 11-9 zuerst eine Befehlsübersicht.

11.3.1 Befehlstabelle

Befehlstabelle für EA KIT160-7

Befehl	Codes									Anmerkung
Befehle für den Terminal Betrieb										
Formfeed FF (dez:12)	^L									Bildschirm wird gelöscht und der Cursor nach Pos. (1,1) gesetzt
Carriage Return CR(13)	^M									Cursor ganz nach links zum Zeilenanfang
Linefeed LF (dez:10)	^J									Cursor 1 Zeile tiefer, falls Cursor in letzter Zeile dann auf 1. Zeile setzen
Cursor On / Off	ESC	Q	C	n1						n1=0: Cursor ist unsichtbar; n1=1: Cursor blinkt (invers 6/10s);
Cursor positionieren	ESC	O	n1	n2						n1=Spalte; n2=Zeile; Ursprung links oben ist (1,1)
Terminal Font einstellen	ESC	F	T	n1						n1=1: Font Nr. n1 (1..16) für Terminal Betrieb einstellen
Befehle zur Textausgabe										
Text-Modus	ESC	L	n1	mst						Modus n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace; mst: Muster Nr. 0..7 verwenden;
Font einstellen	ESC	F	n1	n2	n3					Font mit der Nummer n1 (1..16) einstellen; n2=X- n3=Y-Zommfaktor (1x..8x);
Zeichenkette horizontal ausgeben	ESC	Z	L / Z / R	x1	y1	Text ...	NUL			Eine Zeichenkette (...) an x1,y1 ausgeben. 'NUL' ($00)=Zeichenkettenende; Mehrere Zeilen werde durch das Zeichen '|' ($7C, dez: 124) getrennt; 'L':= Linkbündig an x1; 'Z':= Zentriert an x1; 'R':= Rechtsbündig an x1; y1 ist immer die Oberkannte der Zeichenkette
Zeichenkette 90° gedreht (vertikal) ausgeben	ESC	Z	O / M / U	x1	y1	Text ...	NUL			Eine Zeichenkette (...) um 90° gedreht an x1,y1 ausgeben; 'NUL' ($00)=Ende; Mehrere Zeilen werde durch das Zeichen '|' ($7C, dez: 124) getrennt; 'O':= Oben-Bündig an y1; 'M':= Mittig an y1; 'U':= Unten-Bündig an y1; x1 ist immer die Rechte Kannte der Zeichenkette
Zeichen definieren	ESC	E	n1	daten ...						n1=Zeichen Nr.; daten=Anzahl Bytes je nach akt. Font
Befehle zum Zeichnen										
Grafik-Modus	ESC	V	n1							Zeichenmodus einstellen für die Befehle: 'Punkt setzen', 'Gerade zeichnen', 'Rechteck', 'Rundeck' und 'Bereich mit Füllmuster' n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace;
Punkt setzen	ESC	P	x1	y1						Ein Pixel an die Koordinaten x1, y1 setzen
Gerade zeichnen	ESC	G	x1	y1	x2	y2				Eine Gerade von x1,y1 nach x2,y2 zeichnen
Gerade weiter zeichnen	ESC	W	x1	y1						Eine Gerade vom letzten Endpunkt bis x1, y1 zeichnen
Rechteck Befehle										
Rechteck zeichnen	ESC	R	R	x1	y1	x2	y2			Ein Rechteck (Rahmen) von x1,y1 nach x2,y2 zeichnen
Rundeck zeichnen			N	x1	y1	x2	y2			Ein Rechteck mit runden Ecken von x1,y1 nach x2,y2 zeichnen
Bereich löschen			L	x1	y1	x2	y2			Einen Bereich von x1,y1 nach x2,y2 löschen (alle Pixel aus)
Bereich invertieren			I	x1	y1	x2	y2			Einen Bereich von x1,y1 nach x2,y2 invertieren (alle Pixel umkehren)
Bereich füllen			S	x1	y1	x2	y2			Einen Bereich von x1,y1 nach x2,y2 füllen (alle Pixel ein)
Bereich m. Füllmuster			M	x1	y1	x2	y2	mst		Einen Bereich von x1,y1 nach x2,y2 mit Muster mst (0..7) zeichnen
Box zeichnen			O	x1	y1	x2	y2	mst		Ein Rechteck mit Füllmuster mst (0..7) zeichnen; (immer Replace)
Rundbox zeichnen			J	x1	y1	x2	y2	mst		Ein Rundeck mit Füllmuster mst (0..7) zeichnen; (immer Replace)
Bitmap Bilder Befehle										
Bild aus EEPROM	ESC	U	E	x1	y1	nr				internes Bild mit der nr (0..255) aus dem EEPROM nach x1,y1 laden
Bild laden			L	x1	y1	daten ...				Ein Bild nach x1,y1 laden; daten des Bildes siehe Bildaufbau
Hardcopy senden		H	x1	y1	x2	y2				Es wird ein Bild angefordert. Zuerst werden die Breite und Höhe in Pixel und dann die eigentlichen Bilddaten über RS232 gesendet.
Display-Befehle (Wirkung auf das gesamte Display)										
Display löschen	ESC	D	L							Displayinhalt löschen (alle Pixel aus)
Display invertieren			I							Displayinhalt invertieren (alle Pixel umkehren)
Display füllen			S							Displayinhalt füllen (alle Pixel ein)
Display ausschalten			A							Displayinhalt wird unsichtbar bleibt aber erhalten, Befehle weiterhin möglich
Display einschalten			E							Displayinhalt wird wieder sichtbar
Display Clipboard			C							Inhalt des Clipboards wird dargestellt. Displayausgaben sind nicht mehr sichtbar
Disp. Normaldarstellung			N							Aktuelles Bild wird dargestellt (Normalbetrieb). Alle Ausgaben wieder sichtbar
Display Reset			R							Der Displaykontroller wird per Befehl rückgesetzt und neu initialisiert
Makro Befehle										
Makro ausführen	ESC	M	N	n1						Das (Normal-)Makro mit der Nummer n1 aufrufen (max. 7 Ebenen)
Touch Makro ausführen			T	n1						Das Touch-Makro mit der Nummer n1 aufrufen (max. 7 Ebenen)
Port Makro ausführen			P	n1						Das Port-Makro mit der Nummer n1 aufrufen (max. 7 Ebenen)
autom. Makro zyklisch			A	n1	n2	n3				Makros n1..n2 automatisch zyklisch abarbeiten; n3=Pause in 1/10s
autom. Makro pingpong			J	n1	n2	n3				Makros autom. von n1..n2..n1 (PingPong) abarbeiten; n3=Pause in 1/10s

(Abb. 11-8)

Bargraph Befehle												
Bargraph definieren	ESC	B	R L O U	nr	x1	y1	x2	y2	aw	ew	mst	Bargraph nach L(inks), R(echts), O(ben), U(nten) mit der ´nr´ (1..16) definieren. x1,y1,x2,y2 sind das umschließende Rechteck des Bargraphs. aw,ew sind die Werte für 0% und 100%. mst=Muster (0..7)
Bargraph zeichnen			nr	wer								Den Bargraph mit der Nummer nr (1..16) auf den neuen Benutzer-´wert´ setzen
Clipboard Befehle (Zwischenspeicher für Bildbereiche)												
Displayinhalt sichern	ESC	C	B									Der gesamte Displayinhalt wird als Bildbereich ins Clipboard kopiert
Bereich sichern			S	x1	y1	x2	y2					Der Bildbereich von x1, y1 bis nach x2, y2 wird ins Clipboard kopiert
Display restaurieren			R									Der Bildbereich im Clipboard wird wieder ins Display zurückkopiert
Bereich kopieren			K	x1	y1							Der Bildbereich im Clipboard wird ins Display nach x1, y1 kopiert
Tastatur / Touch-Panel Befehle												
Touch-Taste mit horizontaler Beschriftung definieren	ESC	T	H	f1	f2	Ret. Code	Form	Text ...	NUL			Die Touch-Felder f1 bis f2 (gegenüberliegenden Eckfelder), werden zu einer Touch-Taste mit dem Rückgabewert ´Ret. Code´ (=1..255) zusammengefasst (Ret.Code=0 Touch-Taste nicht aktiv). ´Form´:Touch-Taste (=0 nichts; =1 löschen; =2 mit Rahmen) zeichnen ´Text´: es folgt eine Zeichenkette die zentriert mit dem akt. Font in der Touch-Taste plaziert wird, mehrzeilige Texte werden mit dem Zeichen '|' ($7C, dez: 124) getrennt; Zeichen NUL ($00) = Zeichenkettenende
Touch-Taste mit vertikaler (90° gedreht) Beschriftung definieren			V									
Touch-Tasten (P)Reset			P									Alle Touch-Tasten werden aufsteigend aktiviert (Felder mit Code 1..60)
			R									Alle Touch-Tasten werden deaktiviert (alle Felder mit Code 0)
Touch-Tasten Reaktion			I	n1								n1=0: kein invertieren beim Berühren der Touch-Taste n1=1: Touch-Taste wird beim Berühren automatisch invertiert
			S	n1								n1=0: kein Summer beim Berühren einer (Touch-)Taste n1=1: Summer piepst kurz beim Berühren einer (Touch-)Taste
Touch-Taste Invertieren			M	n1								Die Touch-Taste mit dem zugeordnetem Return-Code n1 wird manuell Invertiert
Taste manuell abfragen			W									Die momentan gedrückte (Touch-)Taste wird auf der RS-232/RS-422 gesendet
Tasten-Abfrage Ein/Aus			A	n1								Tastaturabfrage wird n1=0:deaktiviert; n1=1:aktiviert, Tastendrücke werden automatisch gesendet; n1=2:aktiviert, Tastendrücke werden nicht gesendet (mit ESC T W abfragen)
Menü / Popup Befehle												
Menü mit horizoltalen Einträgen definieren	ESC	N	H	x1	y1	nr	Text ...	NUL				Ein Menü wird ab der Ecke x1,y1 (Horizotales Menü = linke obere Ecke; Vertikales menü = rechte obere Ecke) mit dem akt. Font gezeichnet. nr:= aktuell invertierter Eintrag (z.B: 1 = 1. Eintrag) Text:= Zeichenkette mit den Menüeinträgen. Die einzelnen Einträge sind durch Zeichen '|' ($7C,dez:124) getrennt z.B. "Eintrag1|Eintrag2|Eintrag3" Der Hintergrund des Menüs wird automatisch ins Clipboard gesichert. Ist bereits ein Menü definiert, wird dieses automatisch abgebrochen+entfernt.
Menü mit vertikalen (90° gedrehten) Einträgen definieren			V									
Menübox invertieren			I									Die gesamte Menübox wird invertiert. Sinnvoll für negative Darstellung
nächster Eintrag			N									Der nächste Eintrag wird invertiert oder bleibt am Ende stehen
vorheriger Eintrag			P									Der vorherige Eintrag wird invertiert oder bleibt am Anfang stehen
Menüende / Senden			S									Das Menü wird vom Display entfernt und durch den Clipboardinhalt ersetzt der aktuelle Eintrag wird als Nummer (1..n) gesendet (0=kein Menü dargestellt)
Menüende / Makro			M	nr								Das Menü wird vom Display entfernt und durch den Clipboardinhalt ersetzt Für Eintrag 1 wird Makro 'nr' aufgerufen; für Eintrag 2 Makro nr+1 usw.
Menüende / Abbrechen			A									Das Menü wird vom Display entfernt und durch den Clipboardinhalt ersetzt
Kontroll- / Definitions-Befehle												
Automatisch blinkender Bereich (Cursor-Funktion)	ESC	Q	D	x1	y1	x2	y2					Definiert einen Blinkbereich x1,y1 bis x2,y2; Blinkfunktion aktivieren
			Z	n1								Einstellen der Blinkzeit n1= 1..15 in 1/10s; 0=Blinkfunktion deaktivieren
			M	I								Invers-Modus (Blinkbereich wird invertiert); Blinkfunktion aktivieren
				mst								Clipboard-Modus mst=Muster (0..7) des Blockcursors; Blinken aktivieren
			C	n1								Automatisch blinkender Bereich als Cursor für den Terminal Betrieb n1=0: Blinkfunktion deaktivieren; n1=1: Blinkfunktion aktivieren (Invers, 6/10s)
Selekt / Deselekt	ESC	K	S	adr								Kit mit Adresse n1 aktivieren (n1=255: alle)
			D	adr								Kit mit Adresse n1 deaktivieren (n1=255: alle)
			A	adr								Neue Adresse adr zuweisen (z.B. im Power-On Makro)
Warten (Pause)	ESC	X	n1									n1 Zehntel-Sekunden abwarten bevor der nächste Befehl ausgeführt wird.
Summer Ein / Aus	ESC	J	n1									n1=0:Summer Aus; n1=1:Summer Ein; n1=2..255:für n1 1/10s lang Ein
Bytes senden	ESC	S	anz	daten ...								Es werden anz (1..255; 0=256) Bytes auf der RS-232/RS-422 gesendet daten ... = anz Bytes (z.B Ansteuerung eines externen seriellen Druckers)
I2C-Bus lesen	ESC	I	R	adr	anz							Von dem Baustein am I2C-Bus mit der Device Adresse adr werden anz (1..255; 0=256) Bytes angefordert und über die RS-232/RS-422 gesendet.
I2C-Bus schreiben	ESC	I	W	adr	anz	daten ...						Auf dem I2C-Bus für den Baustein mit der Device Adresse adr werden anz (1..255; 0=256) Bytes gesendet. daten ... = anz Bytes
Port-Befehle												
Output-Port schreiben	ESC	Y	W	n1	n2							n1=0: Alle 8 Ausgabe-Ports entsprechend n2 (=8-Bit Binärwert) einstellen n1=1..8: Ausgabe-Port n1 rücksetzen (n2=0); setzen (n2=1); invertieren (n2=2)
Eingabe-Port lesen			R	n1								n1=0: Alle 8 Eingabe-Ports als 8-Bit Binärwert einlesen n1=1..8: Eingabe-Port <n1> einlesen (1=H-Pegel=5V, 0=L-Pegel=0V)
Port Scan Ein/Aus			A	n1								Der automatiche Scan des Eingabe-Port wird n1=0: deaktiviert; n1=1: aktiviert
Eingabe-Port invers			I	n1								Der Eingabe-Port wird n1=0: normal; n1=1: invertiert ausgewertet
Beleuchtung Ein/Aus			L	n1								CFL/LED-Beleuchtung n1=0: AUS; n1=1: EIN; n1=2: INVERTIEREN; n1=3..255: Beleuchtung für n1 Zehntel Sek.. lang einschalten

(Abb. 11-9)

11.3.2 Makroprogrammierung

Einzelne oder mehrere Befehlsfolgen können als sogenannte Makros zusammengefasst und im EEprom der HMI fest abgespeichert werden. Diese Makros können dann mit dem Befehl *Makro ausführen* gestartet werden.

Wie bereits in Kapitel 11.1 beschrieben, gibt es drei unterschiedliche Makroarten. Zwei dieser Makroarten werden in diesem Projekt verwendet.

Touchmakro:

- Start bei Berührung eines Touchfeldes.
- 255 Touchmakros können im EEprom der HMI abgelegt werden.

Normalmakro:

- Start durch Befehl „ESC MN nr“ über die serielle Schnittstelle.
- Start durch Aufruf von einem anderen Makro aus
- 255 Normalmakros können im EEprom der HMI abgelegt werden
- Normalmakro 0 wird zur Initialisierung der HMI verwendet

Nachfolgend eine Übersicht der verwendeten Makros und ihrer Kurzbeschreibung:

- Makro 0: Startmakro, Initialisierung
- Makro 1: Intro, Startbildschirm
- Makro 2: Intro Sanduhr
- Makro 3: Aufruf Betriebsartenwahl
- Makro 4: Aufruf Betriebsartenwahl nach Störung
- Makro 5: Betriebsartenwahl nach Initialisierung des Displays
- Makro 6: Neuinitialisierung des Displays durch "restart-Taste"
- Makro 10: Grundmaske 1 Störbild
- Makro 11: Grundmaske 2 Störbild
- Makro 12: Grundmaske 3 Störbild
- Makro 20: Störung Paarfehler Endschalter Achse 1
- Makro 21: Störung Paarfehler Endschalter Achse 2
- Makro 22: Störung Paarfehler Endschalter Achse 3
- Makro 23: Störung Paarfehler Endschalter Achse 4
- Makro 24: Störung Endlage + angefahren, Achse 1
- Makro 25: Störung Endlage + angefahren, Achse 2
- Makro 26: Störung Endlage + angefahren, Achse 3
- Makro 27: Störung Endlage + angefahren, Achse 4
- Makro 28: Störung Endlage - angefahren, Achse 1
- Makro 29: Störung Endlage - angefahren, Achse 2
- Makro 30: Störung Endlage - angefahren, Achse 3
- Makro 31: Störung Endlage - angefahren, Achse 4
- Makro 32: Störung Überlast, Achsen 1-4
- Makro 36: Störung keine Grundstellung
- Makro 37: Meldung Grundstellung schon vorhanden
- Makro 38: Meldung BA TeilAuto Position 1 erreicht
- Makro 39: Meldung BA TeilAuto Position 2 erreicht
- Makro 40: Meldung BA TeilAuto Position 3 erreicht
- Makro 41: Meldung BA TeilAuto Position 4 erreicht
- Makro 42: Meldung BA TeilAuto Position 5 erreicht
- Makro 43: Meldung BA umschalten nur bei inaktiven Achsen möglich
- Makro 44: Störung Olli aufrichten

- Makro 45: Warnung Olli aufrichten
- Makro 46: Störung Deckel nicht abgenommen
- Makro 50: Meldung BA Automatik Position 1 erreicht
- Makro 51: Meldung BA Automatik Position 2 erreicht
- Makro 52: Meldung BA Automatik Position 3 erreicht
- Makro 53: Meldung BA Automatik Position 4 erreicht
- Makro 54: Meldung BA Automatik Position 5 erreicht
- Makro 55: Meldung BA Automatik, P1-Fahrt aktiv
- Makro 56: Meldung BA Automatik, P2-Fahrt aktiv
- Makro 57: Meldung BA Automatik, P3-Fahrt aktiv
- Makro 58: Meldung BA Automatik, P4-Fahrt aktiv
- Makro 59: Meldung BA Automatik, P5-Fahrt aktiv
- Makro 60: Meldung BA Automatik, P6-Fahrt aktiv
- Makro 61: Meldung BA TeilAutomatik, P1-Fahrt aktiv
- Makro 62: Meldung BA TeilAutomatik, P2-Fahrt aktiv
- Makro 63: Meldung BA TeilAutomatik, P3-Fahrt aktiv
- Makro 64: Meldung BA TeilAutomatik, P4-Fahrt aktiv
- Makro 65: Meldung BA TeilAutomatik, P5-Fahrt aktiv
- Makro 66: Meldung BA TeilAutomatik, P6-Fahrt aktiv
- Touchmakro 1: Hilfsmakro zu Aufruf Makro2, Betriebsartenwahl
- Touchmakro 2: Automatikablauf Oberfläche 1
- Touchmakro 3: Teilautomatik Oberfläche
- Touchmakro 4: Handbetrieb Oberfläche 1
- Touchmakro 5: Teilautomatik Taste Position 4 anfahren
- Touchmakro 6: Teilautomatik Taste Position 1 anfahren
- Touchmakro 7: Teilautomatik Taste Position 5 anfahren
- Touchmakro 8: Teilautomatik Taste Position 3 anfahren
- Touchmakro 9: Teilautomatik Taste Position 2 anfahren
- Touchmakro 10: Teilautomatik Taste Position 0, Grundstellung anfahren
- Touchmakro 11: Handbetrieb Oberfläche 2 - Achse 1
- Touchmakro 12: Handbetrieb Oberfläche 3 - Achse 2
- Touchmakro 13: Handbetrieb Oberfläche 4 - Achse 3
- Touchmakro 14: Handbetrieb Oberfläche 5 - Achse 4
- Touchmakro 15: Handbetrieb Taste Richtung +, Achse 1
- Touchmakro 16: Handbetrieb Taste Richtung -, Achse 1
- Touchmakro 17: Hilfe zu Teilautomatikoberfläche 1
- Touchmakro 18: Hilfe zu Handbetrieboberfläche 1
- Touchmakro 19: Automatikablauf Oberfläche 2
- Touchmakro 20: Hilfe zu Teilautomatikoberfläche 2
- Touchmakro 21: Handbetrieb Taste Richtung +, Achse 2
- Touchmakro 22: Handbetrieb Taste Richtung -, Achse 2
- Touchmakro 23: Handbetrieb Taste Richtung +, Achse 3
- Touchmakro 24: Handbetrieb Taste Richtung -, Achse 3
- Touchmakro 25: Handbetrieb Taste Richtung +, Achse 4
- Touchmakro 26: Handbetrieb Taste Richtung -, Achse 4
- Touchmakro 27: Störung quittieren
- Touchmakro 28: serielle Schnittstelle an Olli, Grdstllg fahren
- Touchmakro 29: Aufruf reset/restart, Neuinitialisierung
- Touchmakro 30: Achse nach Endlagenfahrt freifahren
- Touchmakro 50: serielle Schnittstelle an Olli, BA Auto Freigabe aktiv
- Touchmakro 51: serielle Schnittstelle an Olli, BA TeilAuto aktiv
- Touchmakro 52: serielle Schnittstelle an Olli, BA Hand aktiv

11.3.3 Quelltextauszug: Intro

Es wird darauf verzichtet, den gesamten Quelltext hier darzustellen. Es werden jedoch einige Beispiele dargestellt.

Den gesamten Quelltext des Projektes finden Sie auf der beiliegenden Dokumentations-CDR.

Dieses Bild ist ein Teil des Intros. Wie auch im Ablaufdiagramm der Makros zu erkennen, wird zuerst das Makro 0 zur Initialisierung des Displays durchlaufen.

(Abb. 11-10)

Zum Intro gehören ebenso eine Reihe Bilder, die in einem kurzen zeitlichen Abstand auf den Bildschirm geladen werden, um den Eindruck eines Filmes zu hinterlassen.

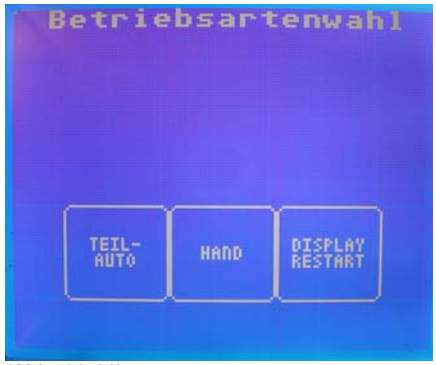
Auszug des Quelltextes:

Makro:1

```
#DL                                    ;Bildschirm löschen
#L 1,1                                 ;Textmodus setzen
#FT 3                                  ;Font Nummer3 einstellen
#QC 0                                  ;Cursor unsichtbar
#QZ 6                                  ;Blinktakt Cursor
#V 1                                   ;Grafikmodus setzen
#F 2,1,1                               ;Font 2, kein Zoom
#W 0,0                                 ;Cursor positionieren
#KS 255                                ;Adresse zuweisen
#YW 0,11111111                         ;alle Outputs rücksetzen
#TR                                    ;alle TouchFelder deaktivieren
#YI 1                                  ;Inputs invertieren
#DL
#ZZ 80,5,"by||||||JR||&||L. C. Kulf|||||||APRIL 2004"
#X 30                                  ;Wartezeit in 1/10s
#DL                                    ;Bildschirm löschen
```

11.3.4 Quelltextauszug: Betriebsartenauswahl

Die Betriebsartenauswahl unterscheidet sich in den Bildern. Hier kommt es darauf an, zu welchem Zeitpunkt die Oberfläche der Betriebsartenwahl aufgerufen wird.



(Abb. 11-11)

Im Beispiel dargestellt ist die Oberfläche der Betriebsartenauswahl, wie sie nach dem Start des Displays zu sehen ist.

Wird die Oberfläche zur Betriebsartenwahl nach einer Störung aufgerufen, wird die Oberfläche durch ein anderes Makro aufgerufen. Dort stehen dann andere Buttons und somit auch andere anzuwählende Funktionen zur Verfügung.

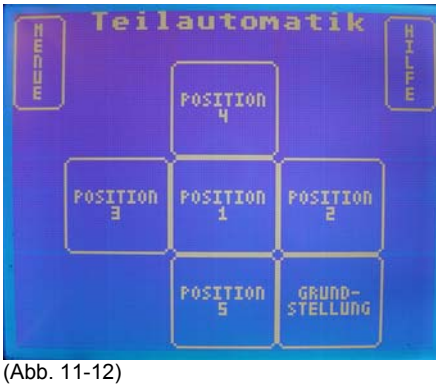
Auszug des Quelltextes:

Makro:3

```
#DL                                ;Bildschirm löschen
#TR                                ;alle TouchFelder deaktivieren
#F 4,1,1                           ;Font 4, kein Zoom
#X1                                ;Wartezeit
#ZZ 80,0 "Betriebsartenwahl"       ;Text ausgeben (Maße in Pixel !)
#F 1,1,1                           ;Font 1, kein Zoom
#TH 34,43,51,2,"TEIL-|AUTO"
#TH 36,45,52,2,"HAND"
#TH 38,47,29,2,"DISPLAY|RESTART"
```

11.3.5 Quelltextauszug: Oberfläche Teilautomatik

Die Oberfläche der Betriebsart Teilautomatik ist in Touchmakro 3 programmiert. Durch Berühren der aktiven Touchfelder wird intern ein Wert zurückgegeben, der Returnwert.



(Abb. 11-12)

Mit Hilfe dieses Returnwertes werden weitere Makros aufgerufen. Die Zahl des Rückgabewertes bestimmt die Nummer des aufzurufenden Makros.

Auszug des Quelltextes:

Touchmakro:3

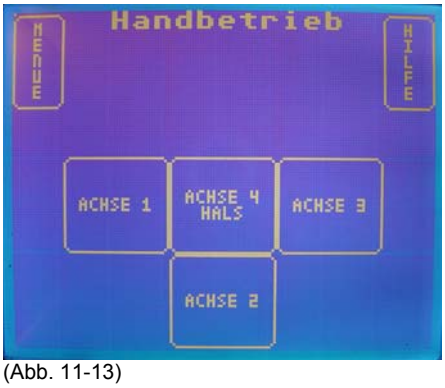
```
#DL                                ;Bildschirm löschen
#TR                                ;alle TouchFelder deaktivieren
#F 4,1,1                           ;Font 4, kein Zoom
#ZZ 80,0 "Teilautomatik"           ;Text ausgeben
#F 1,1,1                           ;Font 1, kein Zoom

#TH 12,21,5,2,"POSITION|4"         ;ret 5
#TH 28,37,6,2,"POSITION|1"         ;ret 6
#TH 44,53,7,2,"POSITION|5"         ;ret 7
#TH 26,35,8,2,"POSITION|3"         ;ret 8
#TH 30,39,9,2,"POSITION|2"         ;ret 9
```

```
#TH 46,55,10,2,"GRUND-|STELLUNG" ;ret 10
#TH 1,9,1,2,"M|E|N|U|E"          ;ret 1
#TH 8,16,17,2,"H|I|L|F|E"        ;ret 17
```

11.3.6 Quelltextauszug: Oberfläche Betriebsart Hand

Die im Beispiel gezeigte Oberfläche zur Betriebsart Hand stellt nicht die ganzen Bedienungsmöglichkeiten in dieser Betriebsart dar.



(Abb. 11-13)

Dieses Kapitel soll nicht die Oberflächen und ihre Funktionen erklären, sondern die Programmierung des LCD-Touchpanels dem Benutzer näher bringen.

Informationen zum Bedienen des Messautomaten entnehmen Sie bitte den entsprechenden Kapiteln.

Auszug des Quelltextes:

Touchmakro:4

```
#DL                              ;Bildschirm löschen
#TR                              ;alle TouchFelder deaktivieren
#F 4,1,1                         ;Font 4, kein Zoom
#ZZ 80,0 "Handbetrieb"           ;Text ausgeben
#F 1,1,1                         ;Font 1, kein Zoom
#TH 26,35,11,2,"ACHSE 1"         ;ret 11
#TH 44,53,12,2,"ACHSE 2"         ;ret 12
#TH 30,39,13,2,"ACHSE 3"         ;ret 13
#TH 28,37,14,2,"ACHSE 4|HALS"    ;ret 14
#TH 1,9,1,2,"M|E|N|U|E"          ;ret 1
#TH 8,16,18,2,"H|I|L|F|E"        ;ret 18
```

11.3.7 Quelltextauszug: Störung Achse 4

In der Übersicht zur Verwendung der programmierten Makros in Kapitel 11.3.2, finden Sie eine sehr große Anzahl an Makros für Meldungen zur Bedienerführung.

Makros zur Visualisierung von diagnostizierten Störungen sind ebenso in großer Anzahl vorhanden.

In diesem Kapitel werden nicht sämtliche Makros zu Meldungen oder Störungen aufgeführt, sondern nur ein kleiner Teil zur Erläuterung der Programmierung.

(Abb. 11-14)

In diesem Makro ist die Verwendung eines in das EEprom des Displays geladenen Bildes im Dateiformat *.bmp aufzuzeigen.

Auszug des Quelltextes:

```
;Grundmaske 2 Störbild
Makro:11
#TR                                 ;alle TouchFelder deaktivieren
#F 4,1,1                            ;Font 4, kein Zoom
#ZZ 80,0, "Stoerung"                ;Text ausgeben
#UE 5,5,14                          ;Stopschild laden
#F 3,1,1
#TH 43,54,30,2,"Achse|freifahren"   ;ret 30


;Störung Endlage - angefahren, Achse 4
Makro:31
#DL
#TR
#F3,1,1
#ZZ 80,40,"Achse 4|Endschalter Endlage -|angefahren"
#MN 11
```

Als Besonderheit ist in diesem Beispiel hervorzuheben, daß zum Zweck der Schonung der Speicherreserven im EEprom des Displays, zwei Bilder übereinander geladen werden.

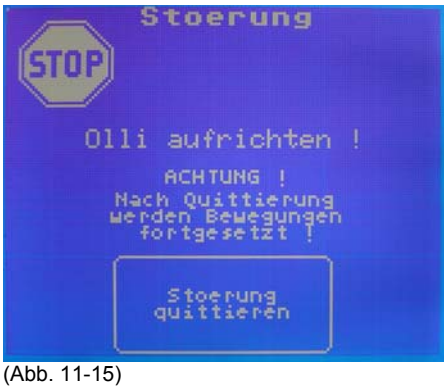
Diese Technik wurde bei sämtlichen Visualisierungen von Störungen angewendet. Es wird jeweils ein Grundbild als Hintergrund geladen. Die eigentliche Störung, der Störungstext, steht im Vordergrund.

Der Vorteil ist, daß nicht für jedes Meldungs- oder Störungsbild die Oberfläche mit allen enthaltenen Schriften, Grafiken oder Buttons programmiert werden muß.

Die compilierte Projektdatei *projekt.eep* wird somit wesentlich kleiner und benötigt deshalb im Speicher des Displays weniger Resourcen.

11.3.8 Quelltextauszug: Meldung Olli aufrichten

Dieses Bild ist ebenso wie das Bild in Kapitel 11.3.7 ein Beispiel für die Überlagerung mehrerer Makros.



(Abb. 11-15)

Ist diesem Bild soll zusätzlich die Verwendung der seriellen Schnittstelle aufgezeigt werden. Über das Betätigen des Buttons „Störung quittieren“ wird ein Makro aufgerufen, das über die serielle Schnittstelle einen definierten Befehl zum Mikrocontroller sendet.

Dieser Befehl setzt sich aus 4 Byte zusammen. Weitere Informationen zu diesen Befehlen, die über die serielle Schnittstelle versendet und empfangen werden, können Sie den Kapiteln zur Kommunikation entnehmen.

Auszug des Quelltextes:

```
;Störung Olli aufrichten
Makro:44
#DL
#TR


#F2,1,1
#TH 43,54,27,2,"Stoerung|quittieren"
#F3,1,1
#ZZ 80,45,"Olli aufrichten !"
#F2,1,1
#ZZ 80,60,"ACHTUNG !"
#ZZ 80,68,"Nach Quittierung|werden Bewegungen|fortgesetzt !"
#F 4,1,1
#ZZ 80,0, "Stoerung"
#UE 5,5,14

;Störung quittieren
Touchmakro:27
#S4,"A24x"
#MN 4
```

11.4 Compilieren und Laden des Projekts

Um das Projekt in das EEprom des LCD-Displays laden zu können, muß dieses compiliert werden.

Dazu steht das Tool *kitcompiler.exe* zur Verfügung.

Nach Installation des Tools *kitcompiler.exe* müssen Bilder, die mit in das EEprom des Displays übertragen werden sollen, in das gleiche Verzeichnis wie die im ASCII-Editor erstellte *Projekt.txt* Datei, kopiert werden.

Bevorzugt wurde das root-Verzeichnis einer Partition verwendet, in der das Tool *kitcompiler.exe* installiert wurde.

Das LCD-Display wird mit der seriellen Schnittstelle eines kompatiblen PCs über eine RS 232 – Leitung verbunden.

Die Schnittstelle des Displays ist voreingestellt mit einer Datenübertragungsrate von 9600 Baud und dem Datenformat 8, N, 1.

Informationen, die zu Änderungen der Schnittstellenparameter benötigt werden, können Sie Kapitel 6.2.5 entnehmen.

Das Tool *kitcompiler.exe* wird durch einen Doppelklick auf das Icon gestartet. Geöffnet wird eine dos-Box in der nun der komplette Verzeichnispfad der Projektdateien angegeben wird.

Dateiendungen müssen angegeben werden. Wird darauf verzichtet, bricht der Compiler mit einer Fehlermeldung den Vorgang ab.

Das Projekt wird compiliert und eine *project.eep* automatisch erstellt. Ist die Verbindung zwischen Display und PC nicht fehlerhaft, wird die project.eep in das EEprom des Displays übertragen.

Nach erfolgreicher Übertragung der compilierten Datei, wird das Makro 0 des Displays automatisch gestartet und das Display initialisiert.

Die Spannungsversorgung des Displays muß während des gesamten Vorgangs gewährleistet sein.

Weitere Informationen zur Compilierung und Übertragung eines Projekts zum Display erhalten Sie auf der Dokumentations-CDR.

11.4 Testen der erstellten Bilder

Leider wird von der Fa. Electronic Assembly kein Tool zum Testen der Bilder nach dem download in das EEprom des Displays angeboten.

Da die Bilder ausschließlich textbasierend programmiert werden, kann somit ein Testen der Bilder nicht erfolgen.

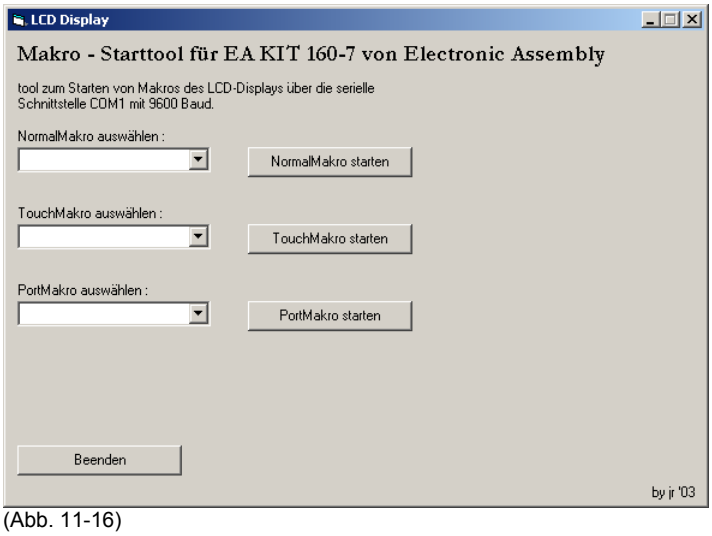
Um diesen Missstand zu beheben, wurde das *STARTTOOL* entwickelt. Es bietet die Möglichkeit, einzelne Makros im LCD-Display über die serielle Schnittstelle aufzurufen.

So können die programmierten Bilder visualisiert und getestet werden. Fehler in den Grafiken werden somit schon im Vorfeld entdeckt und können beseitigt werden.

Ein weiterer großer Vorteil ist die Möglichkeit, Oberflächen zu entwickeln, ohne eine funktionierende Ansteuerung durch einen Mikrocontroller oder ein ähnliches Steuerungssystem zu haben.

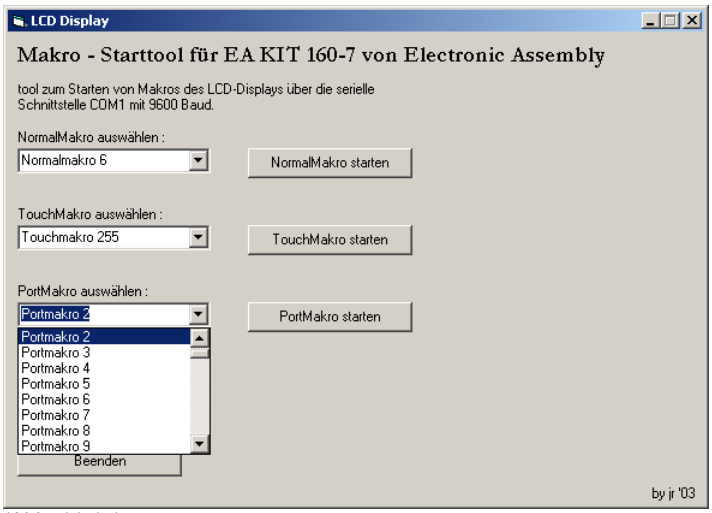
In den folgenden Abschnitten wird kurz über das *STARTTOOL* informiert.

Das *STARTTOOL* muß nicht installiert werden. Über einen Doppelklick auf *starttool.exe* wird das Programm gestartet.



(Abb. 11-16)

Über die drei Pulldown-Menüs kann das anzuzeigende Makro ausgewählt werden. Selbst Touchmakros und Portmakros können aufgerufen werden.



(Abb. 11-17)

Über einen Klick auf den entsprechenden Button, wird der Befehl zur Anforderung des ausgewählten Makros über die serielle Schnittstelle gesendet.

Das Bild wird durch den Aufruf des Makros visualisiert und kann getestet und nach Bedarf abgeändert werden.

Über den Button „Beenden“ wird das Tool geschlossen.

Das *STARTTOOL* erhalten Sie auf der Dokumentations-CDR.

12 Bedienung des Messautomaten

12.1 Allgemein

Der Messautomat kann auf verschiedene Art und Weise bedient werden. Die Standartbedienung wird das Matlabprogramm Hasheraccelerator sein. Weiter kann der Messautomat ersatzweise durch das Tool OLLICONTROL bedient werden.

Zu Servicezwecken und in Sonderfällen kann der Messautomat ebenso durch die HMI bedient werden.

Eine Bedienung über die HMI ist im „normalen“ Messablauf nicht notwendig. Hierzu wird die HMI nur als Visualisierungsmöglichkeit zur Bedienerführung genutzt.

12.2 Bedienung über die HMI

Zur Bedienung über die HMI des Messautomaten stehen auf dem Display verschiedene Oberflächen zur Verfügung.

Diese Oberflächen werden in diesem Kapitel erklärt.

12.2.1 Betriebsartenwahl

(Abb. 12-1)

Diese Oberfläche wird nach dem Hochlauf des Systems angezeigt. Drei Softkeys stehen zur Auswahl :

- TEILAUTOMATIK
- HAND
- DISPLAY RESTART

Softkey TEILAUTOMATIK:

- Über den Softkey TEILAUTOMATIK kann in die Betriebsart Teilautomatik gewechselt werden.
- In die Betriebsart Teilautomatik kann nur gewechselt werden, wenn sich der Messautomat in Grundstellung befindet.

- Wird versucht, in die Betriebsart Teilautomatik zu wechseln, ohne daß der Messautomat sich in Grundstellung befindet, wird eine Meldung zur Bedienerführung angezeigt.

Softkey HAND:

- Über den Softkey HAND kann in die Betriebsart Hand gewechselt werden.
- In die Betriebsart Hand kann jederzeit gewechselt werden.

Softkey DISPLAYRESTART:

- Über den Softkey DISPLAYRESTART kann das LCD-Display neu gestartet und initialisiert werden.

12.2.2 Betriebsartenwahl nach Störung

(Abb. 12-2)

Diese Oberfläche wird nach dem erfolgreichen quittieren einer quittierpflichtigen Störung angezeigt. Fünf Softkeys stehen zur Auswahl :

- TEILAUTOMATIK
- HAND
- FREIGABE AUTO
- STÖRUNG QUITTIEREN
- DISPLAY RESTART

Softkey TEILAUTOMATIK:

- Gleiche Funktion wie in Kapitel 12.2.1

Softkey HAND:

- Gleiche Funktion wie in Kapitel 12.2.1

Softkey DISPLAY RESTART:

- Gleiche Funktion wie in Kapitel 12.2.1

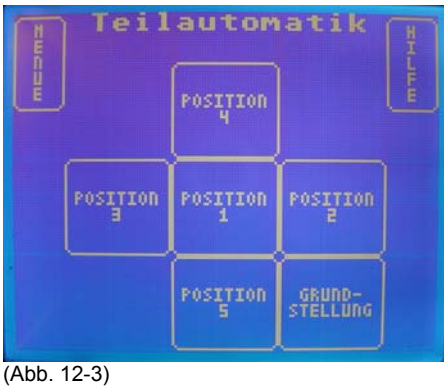
Softkey FREIGABE AUTO:

- Mit diesem Softkey kann die Freigabe zum Wechsel in Betriebsart Automatik erteilt werden.
- Ein Wechsel in die Betriebsart Automatik ist nur zulässig, wenn sich der Messautomat in Grundstellung befindet.
- Wird versucht in die Betriebsart Teilautomatik zu wechseln, ohne daß der Messautomat sich in Grundstellung befindet, wird eine Meldung zur Bedienerführung angezeigt.

Softkey STÖRUNG QUITTIEREN:

- Mit diesem Softkey kann eine noch anstehende Störung quittiert werden.
- Es wird nochmals die Anforderung zur Quittierung einer Störung über die serielle Schnittstelle an den Mikrocontroller gesendet.

12.2.3 Betriebsart Teilautomatik



(Abb. 12-3)

Über die Oberfläche zur Betriebsart Teilautomatik ist es möglich, den Messautomat auf die einzelnen Messpositionen zu fahren.

Eine bestimmte Reihenfolge zum Anfahren der Positionen muß nicht eingehalten werden. Die zum Anfahren ausgewählte Position wird grundsätzlich über die Grundstellung angefahren.

Zu beachten ist, daß die Nummerierung der Positionen den mechanischen Positionen des Messautomaten entspricht. Die Nummerierung der Messpositionen wird auf den in Fahrtrichtung rechten Sitzen in vertikaler Linie gespiegelt.

Die Grundstellung bleibt selbstverständlich die selbe.

Auf dieser Oberfläche stehen acht Softkeys zur Auswahl, sechs davon zur Auswahl einzelner Messpositionen, ein Softkey zur Navigation zwischen den Oberflächen und ein Softkey zum Aufruf der Hilfeseiten zur Oberfläche.

Die Funktionen der Softkeys:

- MENÜ
- HILFE
- POSITION 1
- POSITION 2
- POSITION 3

- POSITION 4
- POSITION 5
- GRUNDSTELLUNG

Softkey MENÜ:

- Mit diesem Softkey kann wieder zurück zur Oberfläche der Betriebsartenauswahl gesprungen werden. Er dient zur Navigation zwischen den Oberflächen.

Softkey HILFE:

- Mit diesem Softkey werden die Hilfeseiten zur Oberfläche Teilautomatik aufgerufen. Hier werden Hinweise zu den einzelnen Messpositionen gegeben. (siehe Abbildung 12-9,10).

Softkey POSITION 1:

- Mit diesem Softkey wird der Messautomat zur mechanischen Position 1 bewegt.

Softkey POSITION 2:

- Mit diesem Softkey wird der Messautomat zur mechanischen Position 2 bewegt.

Softkey POSITION 3:

- Mit diesem Softkey wird der Messautomat zur mechanischen Position 3 bewegt.

Softkey POSITION 4:

- Mit diesem Softkey wird der Messautomat zur mechanischen Position 4 bewegt.

Softkey POSITION 5:

- Mit diesem Softkey wird der Messautomat zur mechanischen Position 5 bewegt.

Softkey GRUNDSTELLUNG:

- Mit diesem Softkey wird der Messautomat zur Grundstellung bewegt. Grundstellung bedeutet, daß alle vier Schlitten der Antriebsachsen die Referenzpunktschalter betätigt haben.
- Die Grundstellung in der Betriebsart Teilautomatik unterscheidet sich nicht von den Grundstellungen in anderen Betriebsarten.

12.2.4 Betriebsart Hand

Für die Betriebsart Hand stehen mehrere Oberflächen zur Verfügung. Diese Oberflächen werden benötigt, um den Messautomaten einrichten zu können.

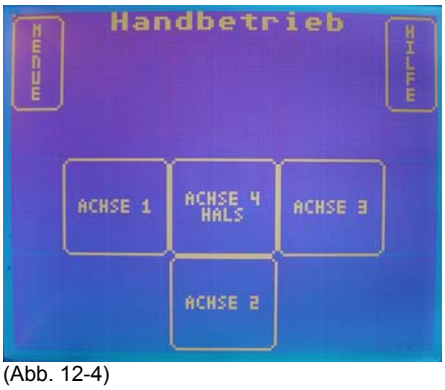
In dieser Betriebsart sind die mächtigsten Eingriffe in das System möglich.

Der Messautomat sollte nur von geschultem Fachpersonal in dieser Betriebsart bedient werden !

In den nächsten Abschnitten erhalten Sie Informationen zur Bedienung des Messautomaten in der Betriebsart Hand.

In dieser Betriebsart ist es möglich, die Schlitten der Antriebseinheiten auf die Endlagenschalter der Achsen zu fahren. In diesem Fall wird die Bedienungsgewalt dem Benutzer entzogen und der Messautomat schaltet selbsttätig intern auf die Betriebsart Teilautomatik um.

In der Betriebsart Teilautomatik erfolgt nach Bestätigung durch den Benutzer ein Freifahren der Antriebsachsen von den Endlagen.



(Abb. 12-4)

In dieser Oberfläche zur Betriebsart Hand stehen sechs Softkeys zur Bedienung zur Auswahl, ein Softkey davon zur Navigation zwischen den Oberflächen, ein Softkey zum Aufruf der Hilfeseiten zur Oberfläche.

Über die restlichen vier Softkeys können die Achsen angewählt werden, die im Handbetrieb verfahren werden sollen.

Sechs Softkeys stehen zur Auswahl:

- MENÜ
- HILFE
- ACHSE 1
- ACHSE 2
- ACHSE 3
- ACHSE 4 HALS

Softkey MENÜ:

- Mit diesem Softkey kann wieder zurück zur Oberfläche der Betriebsartenauswahl gesprungen werden. Er dient zur Navigation zwischen den Oberflächen.

Softkey HILFE:

- Mit diesem Softkey werden die Hilfeseiten zur Oberfläche Hand aufgerufen. Hier werden Hinweise zu den einzelnen Achsen gegeben. (siehe Abbildung 12-).

Softkey ACHSE 1:

- Mit diesem Softkey wird die Achse 1 zum Verfahren im Handbetrieb angewählt. Die Achse führt durch Betätigen dieses Softkeys noch keine Bewegung aus.

Softkey ACHSE 2:

- Mit diesem Softkey wird die Achse 2 zum Verfahren im Handbetrieb angewählt. Die Achse führt durch Betätigen dieses Softkeys noch keine Bewegung aus.

Softkey ACHSE 3:

- Mit diesem Softkey wird die Achse 3 zum Verfahren im Handbetrieb angewählt. Die Achse führt durch Betätigen dieses Softkeys noch keine Bewegung aus.

Softkey ACHSE 4 HALS:

- Mit diesem Softkey wird die Achse 4 zum Verfahren im Handbetrieb angewählt. Die Achse führt durch Betätigen dieses Softkeys noch keine Bewegung aus.

Durch das Betätigen der Softkeys ACHSE 1-4 werden weitere Oberflächen zur Bedienung des Messautomaten im Handbetrieb aufgerufen. In den weiteren Abschnitten werden nicht alle Oberflächen dargestellt.

(Abb. 12-5)

In Abbildung 12-5 wird ersatzweise für alle Achsen die weiterführende Oberfläche zur Bedienung der Achsen dargestellt.

Diese Oberfläche steht für alle vier Achsen zur Verfügung. Sie unterscheiden sich nur im Schriftzug „Achse 1“. Die Oberfläche zur Bedienung von Achse 2 enthält den Schriftzug „Achse 2“, usw.

Auf dieser Oberfläche stehen drei Softkeys zur Verfügung:

- MENÜ
- RICHTUNG +
- RICHTUNG –

Softkey MENÜ:

- Mit diesem Softkey kann wieder zurück zur letzten Oberfläche gesprungen werden. Er dient zur Navigation zwischen den Oberflächen.

Softkey RICHTUNG +:

- Durch das Betätigen dieses Softkeys wird die angewählte Achse um zwei Spindelumdrehungen in die Verfahrrichtung (+) bewegt. Um die Anzahl der zu verfahrenden Spindelumdrehungen zu ändern, lesen Sie Kapitel 9.4.15, Details zur Programmierung der Steuerung.

- Zwei Spindelumdrehungen entsprechen einem linearen Verfahrweg von zwei Millimetern.

Softkey RICHTUNG -:

- Durch das Betätigen dieses Softkeys wird die angewählte Achse um zwei Spindelumdrehungen in die Verfahrrichtung (-) bewegt. Um die Anzahl der zu verfahrenden Spindelumdrehungen zu ändern, lesen Sie Kapitel 9.4.15, Details zur Programmierung der Steuerung.

- Zwei Spindelumdrehungen entsprechen einem linearen Verfahrweg von zwei Millimetern.

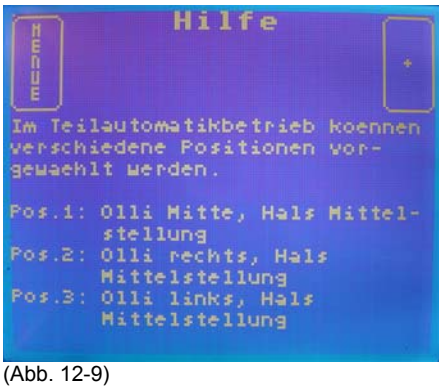
Zur besseren Übersicht werden nachfolgend die einzelnen Oberflächen zur Bedienung der vier Achsen im Handbetrieb dargestellt. Auf Erklärungen wird jedoch verzichtet.

(Abb. 12-6,7,8)

Zu Erklärungen dieser Oberflächen nehmen Sie bitte die ersten Abschnitte dieses Kapitels zur Hilfe. Diese Hinweise zu den Oberflächen sind neutral gehalten und können somit auf diese Abbildungen übertragen werden.

12.2.5 Online-Hilfe zur Betriebsart Teilautomatik

In den Betriebsarten Teilautomatik und Hand steht Ihnen eine Online-Hilfe zur Verfügung. Diese wird jeweils in den Oberflächen zur Betriebsart Teilautomatik und Betriebsart Hand aufgerufen. Dazu steht der Softkey HILFE zur Verfügung.



(Abb. 12-9)

In Abbildung 12-9 ist der erste Teil der Online-Hilfe zur Betriebsart Teilautomatik dargestellt. In dieser Oberfläche können Informationen zu den einzelnen Positionen eingeholt werden.

In dieser Oberfläche können zwei Softkeys zur Navigation verwendet werden.

- MENÜ

- PLUS

Softkey MENÜ:

- Mit diesem Softkey kann wieder zurück zur letzten Oberfläche gesprungen werden. Er dient zur Navigation zwischen den Oberflächen.

Softkey PLUS:

- Mit diesem Softkey kann auf die zweite Seite der Online-Hilfe zur Betriebsart Teilautomatik gesprungen werden. Er dient zur Navigation zwischen den Oberflächen.

(Abb. 12-10)

In Abbildung 12-10 ist der zweite Teil der Online-Hilfe zur Betriebsart Teilautomatik dargestellt.

Ein Softkey steht zur Bedienung zur Verfügung:

- MENÜ

Softkey MENÜ:

- Mit diesem Softkey kann wieder zurück zur letzten Oberfläche gesprungen werden. Er dient zur Navigation zwischen den Oberflächen.

12.2.6 Online-Hilfe zur Betriebsart Hand

Wie auch zur Betriebsart Teilautomatik steht auch in der Betriebsart Hand eine Online-Hilfe zur Verfügung.

(Abb. 12-11)

Ein Softkey steht zur Bedienung zur Verfügung:

- MENÜ

Softkey MENÜ:

- Mit diesem Softkey kann wieder zurück zur letzten Oberfläche gesprungen werden. Er dient zur Navigation zwischen den Oberflächen.

12.2.7 Bedienerführung: Störungen

Störungen durch angefahrene Endlagen

Zur Visualisierung von Störungen steht eine Vielzahl von Oberflächen und Bildern zur Verfügung. Ein kleiner Auszug hiervon wird in den nächsten Abschnitten dargestellt.

(Abb. 12-12)

In Abbildung 12-12 ist eine Störung der Achse 4 dargestellt. Es wurde die Endlagenüberwachung in Verfahrrichtung (-) betätigt.

Diese Diagnose steht für alle vier Achsen zur Verfügung. Selbstverständlich werden nicht nur die Endlagen der negativen Verfahrrichtungen überwacht, sondern auch die Endlagen der positiven Verfahrrichtungen.

Es ergeben sich hieraus acht Oberflächen zur Visualisierung dieser Art von Störungen. Es steht je Oberfläche ein Softkey zur Bedienung zur Verfügung.

- ACHSE FREIFAHREN

Softkey ACHSE FREIFAHREN:

- Durch Betätigen des Softkeys ACHSE FREIFAHREN wird die Achse automatisch von der entsprechenden Endlage gefahren.
- Der Bediener hat keinen Einfluss auf die Verfahrrichtung der Achse.
- Die Verfahrrichtung ist entgegengesetzt der Endlagenfahrt und wird vom System selbstständig diagnostiziert.
- Die Störung ist quittierpflichtig.

Störungen durch Paarfehler

Zur Endlagenüberwachung stehen weitere Oberflächen zur Verfügung. Die Antriebseinheiten werden auch auf Paarfehler überwacht.

Von Paarfehlern spricht man, wenn das unzulässige Ansprechen beider Endlagenschalter einer Achse diagnostiziert wird.

Auf eine Darstellung der vier Oberflächen zur Visualisierung der Störungen durch Paarfehler wird hier bewußt verzichtet.

Die Störungen durch Paarfehler führen zum sofortigen Stillstand der Antriebsachsen und sind quittierpflichtig.

Diese Art Störungen tritt in der Regel nicht auf und spielt deshalb hier eine untergeordnete Rolle.

Durch die Diagnose dieser Störungen werden defekte Endlagenschalter und eine eventuell beschädigte Verdrahtung der Endlagenschalter analysiert.

Die Endlagenschalter sind als Öffnerkontakte ausgeführt und deshalb als kabelbruchsicher zu bezeichnen.

Im Kapitel 6.2.4.2 erhalten Sie mehr Informationen zur Elektrik und zur Beschaltung der Sensorik einer Antriebseinheit.

Störungen durch Fehlbedienung

Eine fehlerfreie Bedienung kann auch durch noch so exakte Bedienerführung nicht ausgeschlossen werden.

Eine Zerstörung dieses Systems ist durch unbeabsichtigte Fehlbedienung nahezu ausgeschlossen.

Eine fehlerhafte Bedienung wird erkannt und ruft Meldungen zur Bedienerführung hervor.

(Abb. 12-13)

In Abbildung 12-13 wird die Visualisierung zur Störung „keine Grundstellung“ dargestellt. Diese Störung kann hervorgerufen werden, wenn zu unzulässigem Zeitpunkt ein Betriebsartenwechsel angefordert wurde, d. h. ein Betriebsartenwechsel wurde angefordert, obwohl der Messautomat sich nicht in Grundstellung befand.

Zwei Softkeys stehen in der dieser Oberfläche zur Bedienung zur Verfügung.

- GRUNDSTELLUNG FAHREN
- STOERUNG QUITTIEREN

Softkey STOERUNG QUITTIEREN:

- Die Störmeldung ist quittierpflichtig und kann mit diesem Softkey quittiert werden.
- Die Störung muß zur Quittierung beseitigt werden. Erfolgt keine Beseitigung der Störung, erscheint die Oberfläche erneut.

Softkey GRUNDSTELLUNG FAHREN

- Mit diesem Softkey wird der Messautomat intern auf die Betriebsart Teilautomatik umgeschalten.
- Der Messautomat führt selbstständig eine Grundstellungsfahrt durch.
- Das System zur Lageüberwachung wird initialisiert.
- Die Störung ist nun quittierbar.

Der Messautomat kann bei dieser Störung nicht in der Betriebsart Hand auf die Grundstellung gefahren werden.

Eine Grundstellungsfahrt ist immer automatisiert durchzuführen, da eine Initialisierung des Systems zur Lageüberwachung der Antriebsachsen zur korrekten Positionierung der Achsen notwendig ist.

Eine Initialisierung des Systems zur Lageüberwachung der Antriebsachsen wird ausschließlich durch eine automatisierte Grundstellungsfahrt durchgeführt.

Störung Deckelüberwachung

In Kapitel 6.2.4.3 wurde bereits auf diese Überwachung eingegangen. Nähere Informationen zur Elektrik und zum Service erhalten Sie in Kapitel 6 dieses Handbuches.

(Abb. 12-14)

Diese Störung tritt auf, wenn nach der Herstellung der Verbindung des Messautomaten mit der Netzversorgung, der Deckel des Gehäuses nicht abgenommen wurde.

Um eine Beschädigung des Messautomaten durch diese Art der fehlerhaften Inbetriebnahme durch den Bediener zu verhindern, werden als sofortige Reaktion alle Achsbewegungen stillgesetzt.

Beim Neustart des Systems wird die Referenzpunktfahrt erst ausgelöst, wenn der Gehäusedeckel des Messautomaten abgenommen wurde.

Diese Störung ist quittierpflichtig.

In dieser Oberfläche steht ein Softkey zur Bedienung zur Verfügung.

- STOERUNG QUITTIEREN

Softkey STOERUNG QUITTIEREN

- Die Störmeldung ist quittierpflichtig und kann mit diesem Softkey quittiert werden.
- Die Störung muß zur Quittierung beseitigt werden. Erfolgt keine Beseitigung der Störung, erscheint die Oberfläche erneut.
- Nach Beseitigung der Ursache und Quittierung der Störung wird die zuletzt aktive Bewegung des Messautomaten fortgesetzt.

Nach erfolgreicher Quittierung wird die zuletzt aktive Bewegung des Messautomaten fortgesetzt !

Störung Neigungssensor

Der Messautomat ist mit einem Neigungssensor versehen. Nähere Informationen zum Neigungssensor erhalten Sie im Kapitel zur Sensorik.

(Abb. 12-15)

Durch den Neigungssensor werden unzulässige Neigungswinkel des Messautomaten überwacht.

Beim Verlassen des durch diesen Neigungssensor festgelegten Arbeitsbereichs, wird diese Störung generiert.

Als Reaktion auf diese Störung werden alle Antriebsachsen mit sofortiger Wirkung gestoppt. Dadurch wird eine mögliche Beschädigung des Messautomaten verhindert.

Auf der Oberfläche zu dieser Störung steht ein Softkey zur Bedienung zur Verfügung.

- STOERUNG QUITTIEREN

Softkey STOERUNG QUITTIEREN:

- Die Störmeldung ist quittierpflichtig und kann mit diesem Softkey quittiert werden.
- Die Störung muß zur Quittierung beseitigt werden. Erfolgt keine Beseitigung der Störung, erscheint die Oberfläche erneut.
- Nach Beseitigung der Ursache und Quittierung der Störung wird die zuletzt aktive Bewegung des Messautomaten fortgesetzt.

Nach erfolgreicher Quittierung wird die zuletzt aktive Bewegung des Messautomaten fortgesetzt !

Störung Überlast

Ist eine oder mehrere Achsen des Messautomaten mechanisch schwergängig oder blockiert, wird dies von den Motorsteuer-ICs des Leistungsteils der Basisplatine diagnostiziert.

Eine Visualisierung der Störung findet durch die HMI statt.

Auf der Oberfläche zu dieser Störung steht ein Softkey zur Bedienung zur Verfügung.

- STOERUNG QUITTIEREN

Softkey STOERUNG QUITTIEREN:

- Die Störmeldung ist quittierpflichtig und kann mit diesem Softkey quittiert werden.
- Die Störung muß zur Quittierung beseitigt werden. Erfolgt keine Beseitigung der Störung, erscheint die Oberfläche erneut.

Das mehrfache Quittieren dieser Störung kann großen mechanischen Schaden hervorrufen ! Die Antriebseinheiten des Messautomaten sind unbedingt einer Sichtprüfung zu unterziehen !

Allgemeiner Hinweis :

Störungen, die hier nicht aufgeführt sind, können den Ablaufdiagrammen zur HMI in Kapitel 11.2 entnommen werden !

12.2.8 Bedienerführung: Meldungen

Meldungen zur Bedienerführung stehen in einer sehr großen Anzahl zur Verfügung. Es wird darauf verzichtet, alle Meldungen hier darzustellen.

Achsspezifische Meldungen stehen selbstverständlich zu allen Achsen zur Verfügung. Ersatzweise wird in diesem Kapitel jeweils die Meldung einer Achse dargestellt.

Meldungen sind nicht quittierpflichtig !
Meldungen werden auf Grund eines Vorganges aufgerufen und werden zeitabhängig angezeigt.

Meldungen beim Start des Systems

(Abb. 12-16,17)

Beim Neustart des Systems werden verschiedene Initialisierungsroutinen durchlaufen, während dieser Zeit informiert die HMI in einem Intro über den Titel des Projekts.

Meldung: Fahrt auf Position aktiv

(Abb. 12-18)

Wird in der Betriebsart Automatik oder Teilautomatik eine Positionierung des Messautomaten auf eine definierte Messposition vorgenommen, zeigt die HMI die aktive Fahrt auf die entsprechende Position an.

Diese Meldung steht natürlich nicht nur für die Fahrt auf Position 2 zur Verfügung, sondern für die folgende Positionen:

- Position 1
- Position 2
- Position 3
- Position 4

- Position 5

Die Fahrt auf die Grundstellung wird nicht angezeigt. Aus folgenden Gründen wurde darauf verzichtet, die Grundstellungsfahrt explizit anzuzeigen:

- Die Grundstellungsfahrt wird bewußt in Betriebsart Teilautomatik ausgewählt.
- Jedes Positionieren in der Betriebsart Automatik oder Teilautomatik hat immer eine Grundstellungsfahrt voraus.

Meldung: Position erreicht

Ist die Betriebsart Automatik oder die Betriebsart Teilautomatik angewählt, und wurde eine definierte Position automatisch oder halbautomatisch angefahren, wird dies über eine Meldung wie in Abbildung 12-19 visualisiert.

(Abb. 12-19)

Diese Meldung steht natürlich nicht nur für das Erreichen der Position 5 zur Verfügung, sondern für die folgende Positionen:

- Position 1
- Position 2
- Position 3
- Position 4
- Position 5

Meldung: Achse aktiv

Wird versucht, zu einem unzulässigen Zeitpunkt die Betriebsart zu wechseln, wird eine Meldung zur Bedienerführung ausgegeben.

Die Betriebsart darf nur gewechselt werden, wenn sich alle Achsen und somit der Messautomat in Grundstellung befinden.

In die Betriebsart Hand darf auch gewechselt werden, wenn der Messautomat nicht in Grundstellung ist. Allerdings darf zum Zeitpunkt der Anforderung der Betriebsart keine Achse in Bewegung sein.

Wird die Betriebsart Hand unter Missachtung dieser Punkte angefordert, wird die Meldung „Achse aktiv“ ausgegeben und die Anforderung abgewiesen.

Meldung: Messung starten

Wurde dem Messautomaten die Freigabe zur Betriebsart Automatik entzogen und in eine andere Betriebsart gewechselt, so muß bei Wiederanforderung der Betriebsart Automatik, die Freigabe bestätigt werden.

(Abb. 12-20)

Wurde die Freigabe zum Wechsel in die Betriebsart Automatik durch den Bediener bestätigt, wird dies durch diese Meldung visualisiert.

Eine Bedienung des Messautomaten durch das Tool OLLICONTROL oder das Matlab-Programm Hasheraccelerator kann nun erfolgen.

Allgemeiner Hinweis :

Meldungen, die hier nicht aufgeführt sind, können den Ablaufdiagrammen zur HMI in Kapitel 11.2 entnommen werden !

12.3 Bedienung über OLLICONTROL

Das Tool OLLICONTROL wurde entwickelt, um den Messautomaten unabhängig von der eigentlichen Software, dem Matlab-Programm Hasheraccelerator, das die Ansteuerung des Automaten im Messaufbau durchführt, aufbauen und später auch zu Test- und Servicezwecken bedienen zu können.

Außerdem eignet sich das Tool OLLICONTROL, um die Leistungsfähigkeit des Messautomaten demonstrieren zu können.

In erster Linie wurde das Tool aber entwickelt, um eine Bedienung des Automaten in Unabhängigkeit eines Messaufbaus zu ermöglichen.

12.3.1 Installation

Das Tool OLLICONTROL muß auf der Festplatte eines kompatiblen PCs installiert werden. Durch einen Doppelklick auf die *setup.exe* wird der Install-Shield geöffnet. Folgen Sie den weiteren Anweisungen auf dem Bildschirm.

Das Tool OLLICONTROL ist auf der beiliegenden Dokumentations-CDR enthalten und kann von dieser CD aus installiert werden.

12.3.2 Vorbereitung

Als Vorbereitung zur Bedienung muß der Messautomat über die RS 232 – Schnittstelle der Adapterplatine mit der COM – Schnittstelle des PCs verbunden werden.

Optional ist hier ersatzweise eine RS 232 – Funkverbindung im System integriert. Weitere Informationen hierzu erhalten Sie im Kapitel zur Adapterplatine.

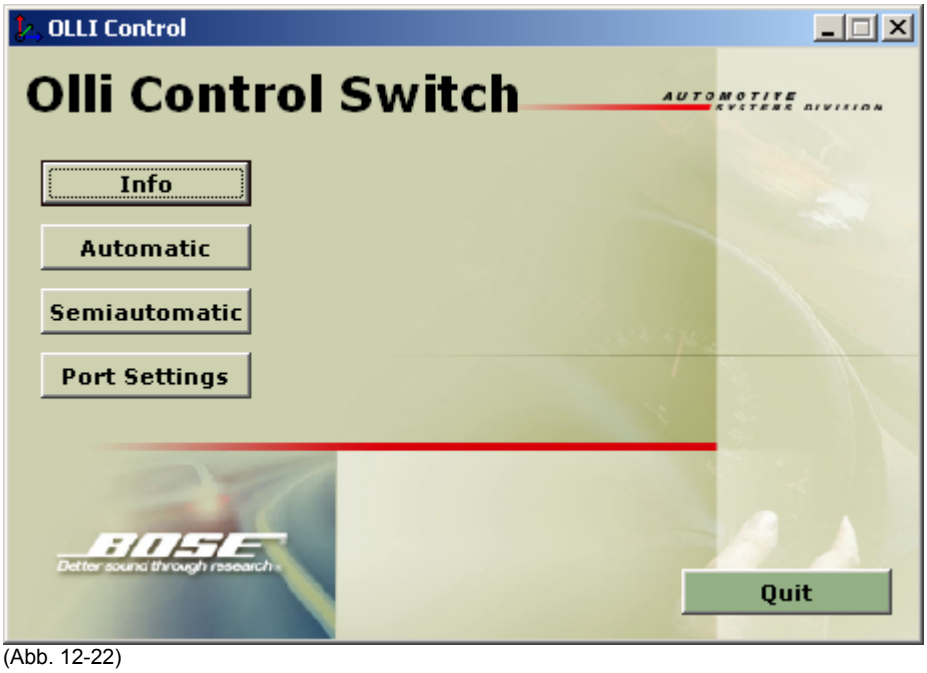
12.3.3 Bedienung OLLICONTROL

Das Tool OLLICONTROL wird durch einen Doppelklick auf *ollicontrol.exe* gestartet. Optional kann eine Verknüpfung auf den Desktop des Rechners generiert werden.

(Abb. 12-21)

Während die Anwendung im Hintergrund gestartet wird, ist ein Hinweis, der über die verwendete Version informiert, geöffnet.

Es erfolgt automatisch eine Weiterleitung zur eigentlichen Bedienungsoberfläche, von welcher aus der Messautomat kontrolliert werden kann.

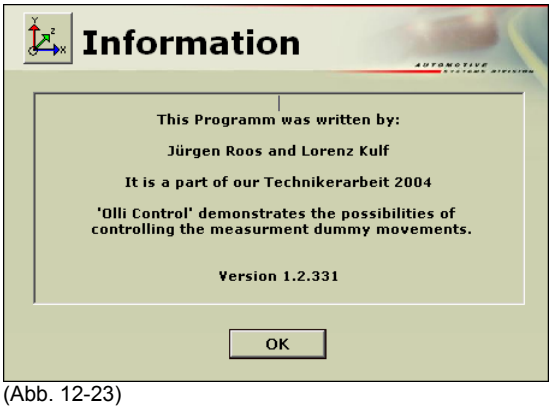


(Abb. 12-22)

Geöffnet wird das Grundbild der Anwendung. Dieses Bild stellt die Betriebsartenauswahl dar. Folgende fünf Buttons stehen zur Auswahl, die Bezeichnung der Buttons ist in englischer Sprache ausgeführt und wurde nach Möglichkeit zum besseren Verständnis übersetzt:

- Info
- Automatik
- Teilautomatik
- Port Settings
- Beenden

Über den Button „Info“ wird ein Informationsfenster geöffnet, das die üblichen Informationen wie Versionsnummer der Anwendung, usw. anzeigt. Das Fenster wird über den Button „OK“ wieder geschlossen.



(Abb. 12-23)

Über den Button „Port Settings“ wird ein Informationsfenster geöffnet, das die Einstellungen der seriellen Schnittstelle RS 232 anzeigt.

Ein Einstellen der Schnittstellenparameter ist bei dieser Version noch nicht möglich. Diese Option ist in Vorbereitung. Das Fenster wird über den Button „OK“ geschlossen.

(Abb. 12-24)

In den folgenden Abschnitten erhalten Sie Informationen zur Bedienung des Messautomaten durch die Software OLLICONTROL.

Hauptsächlich wird versucht, auf die Bedienung der Software einzugehen.

Alle Fehlermeldungen, die OLLICONTROL generieren kann, werden in den folgenden Abschnitten nicht dargestellt.

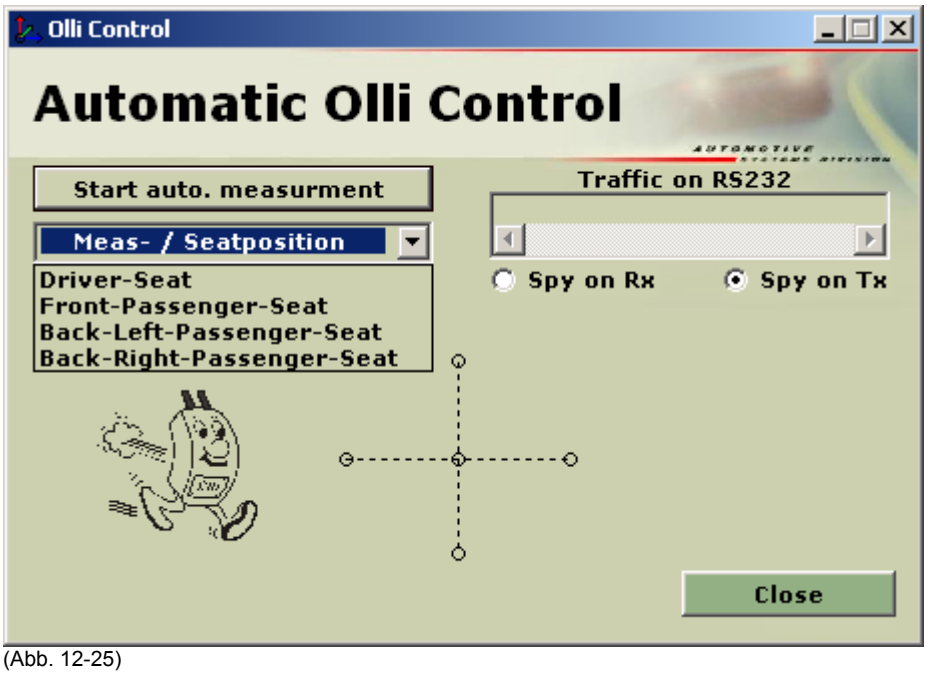
Die Fehlermeldungen oder Hinweise zur Bedienerführung sind in der Regel selbsterklärend und bedürfen keiner weiteren Kommentierung.

Sollten Sie aber dennoch Probleme mit dem Umgang der Software OLLICONTROL haben, wenden Sie sich bitte an Ihren BOSE - Ansprechpartner.

Betriebsart Automatik

Über den Button „Automatik“ im Hauptfenster wird die Oberfläche zur Bedienung des Messautomaten (Abbildung 12-25) in der Betriebsart Automatik aufgerufen.

Hier steht ein Button zum schließen dieses Fensters zur Verfügung.



(Abb. 12-25)

Über ein Pulldown-Menü muß zuerst die Position des Messautomaten im Kraftfahrzeug ausgewählt werden. Zur Verfügung stehen folgende Positionen:

- auf dem Fahrersitz
- auf dem Beifahrersitz
- auf dem Sitz hinter dem Fahrer
- auf dem Sitz hinter dem Beifahrer

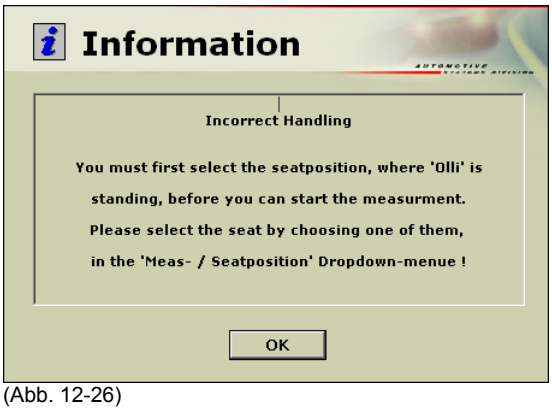
In Abhängigkeit von der gewählten Position im Kraftfahrzeug werden anschließend die verschiedenen zu messenden Positionen des Messautomaten angefahren.

In dieser Oberfläche ist ein kleines Fenster integriert, über das der Datenverkehr über die RS 232 – Schnittstelle beobachtet werden kann. Es kann ausgewählt werden, ob die empfangenen Daten visualisiert werden oder ob die gesendeten Daten angezeigt werden.

Das Fenster zum Anzeigen der Daten ist mit „Traffic on RS 232“ bezeichnet. Das Anzeigen der gesendeten Daten, „Spy on Tx“, ist die Voreinstellung.

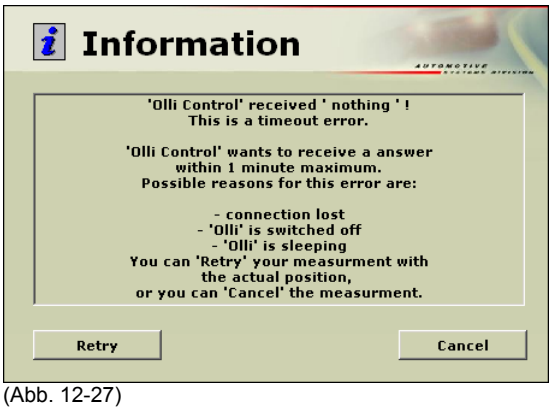
Zum Anzeigen der empfangenen Daten muß die checkbox „Spy on Rx“ angeklickt werden. Ein Umschalten ist jederzeit möglich.

Zum Start des Automatikablaufes muß der Button „start automatic measurement“ gedrückt werden – die automatische Messung wird gestartet.



(Abb. 12-26)

Wird der Button zum Start der Messung gedrückt, ohne daß eine Position im Kraftfahrzeug ausgewählt wurde, wird der Benutzer über seine Fehlbedienung informiert. Diese Meldung über unkorrekte Handhabung des Systems ist quittierpflichtig.



(Abb. 12-27)

Wird der Button zum Start der Messung gedrückt und die Anwendung OLLICONTROL erhält die Quittierung des Befehls nicht innerhalb einer Minute vom Messautomat zurück, wird ein Zeitüberschreitungsfehler ausgegeben.

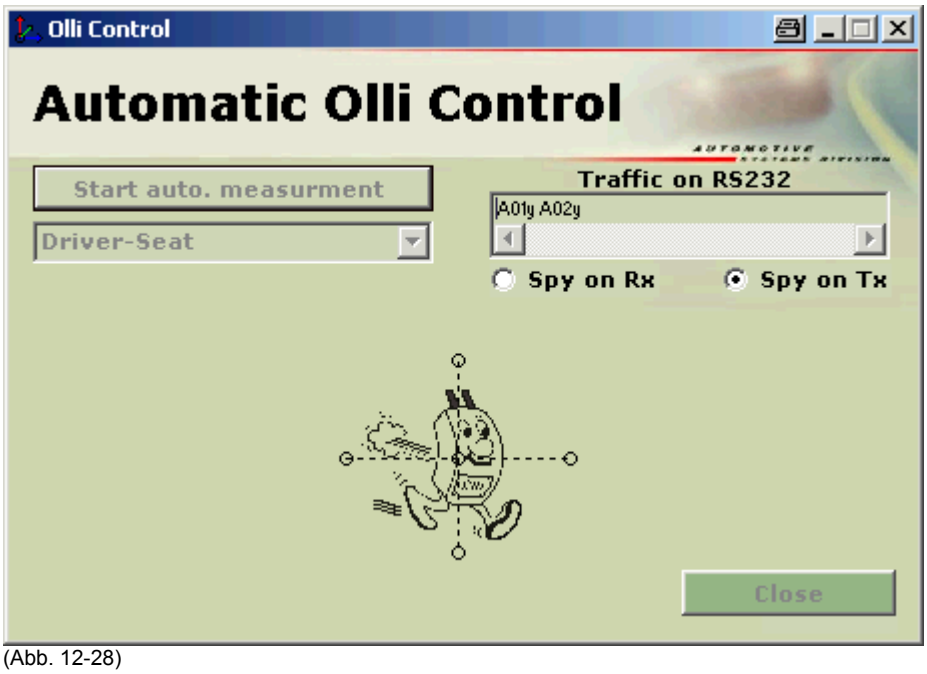
Dieser Fehler kann mehrere Ursachen haben. In erster Linie ist zu untersuchen, ob die Spannungsversorgung des Messautomaten gewährleistet ist. Anschließend ist zu untersuchen, ob eine korrekte Verbindung zwischen dem PC und dem Messautomaten besteht.

Dieser Fehler ist ebenso quittierpflichtig. Der Vorgang kann entweder über den Button „Cancel“ abgebrochen werden oder der Vorgang kann über den Button „Retry“ nochmals gestartet werden.

In Abbildung 12-25 ist im zentralen Bereich ein Fadenkreuz sichtbar. Dieses Fadenkreuz dient der Visualisierung der Istposition des Messautomaten. Ist eine Bewegung auf eine Messposition aktiv und wird diese Bewegung mit dem Erreichen der angeforderten Position erfolgreich beendet, so wird die Visualisierung aktualisiert.

Eine erfolgreiche Positionierung wird durch den Benutzer dadurch erkannt, daß die grafische Darstellung des Messautomaten sich auf die entsprechende Position im Fadenkreuz bewegt hat.

In den folgenden Abschnitten wird ein Automatikablauf dargestellt. Die über die serielle Schnittstelle gesendeten Befehle beziehen sich auf die Messungen auf dem Fahrersitz.

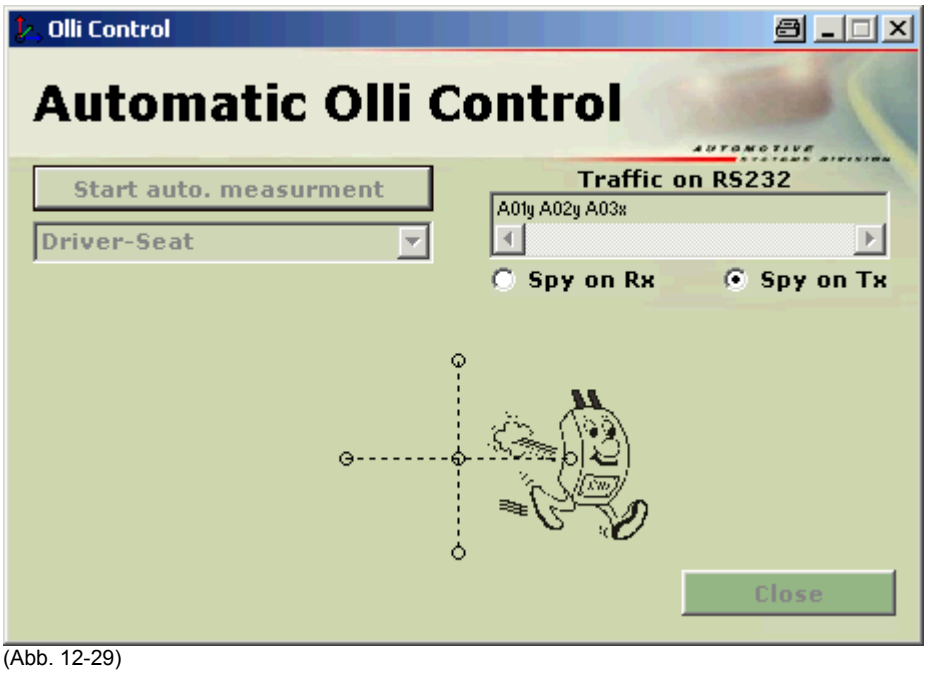


(Abb. 12-28)

In Abbildung 12-28 ist dargestellt, wie der Automatikablauf gestartet wurde und der Befehl A01y, der die Bewegung zur Messposition 1 auf dem Fahrersitz anfordert, bereits ausgeführt wurde.

Die grafische Darstellung des Messautomaten hat sich im Fadenkreuz auf die symbolische Messposition 1 bewegt. Dadurch wird symbolisiert, daß sich der Messautomat nun in der Messposition 1 befindet.

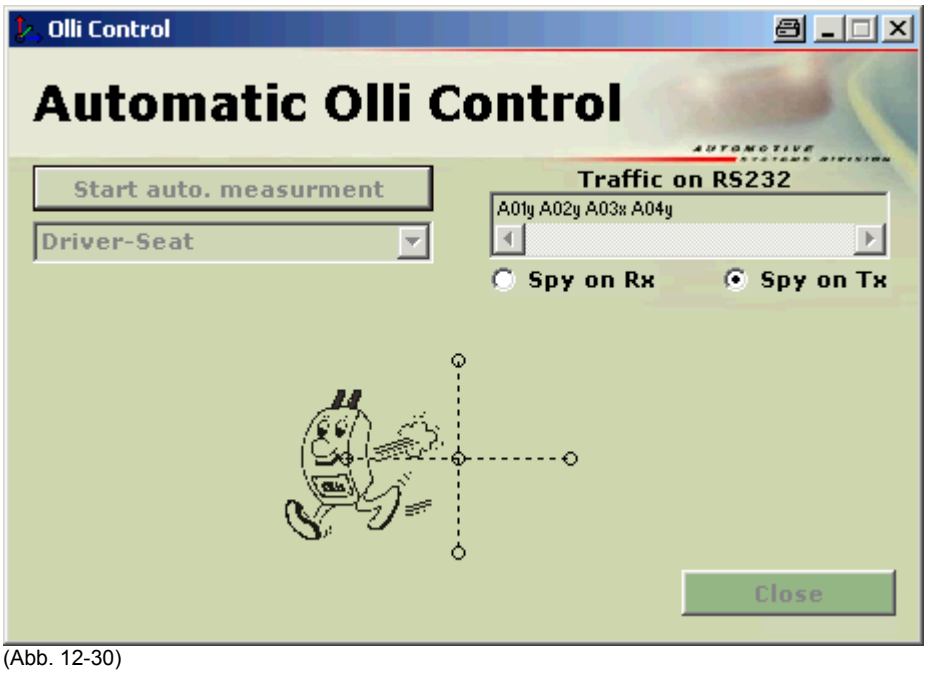
Der Befehl zur Ausführung der nächsten Bewegung wurde bereits über die serielle Schnittstelle gesendet.



(Abb. 12-29)

Das Erreichen der Messposition 2 wurde bestätigt und die symbolische Darstellung wurde aktualisiert.

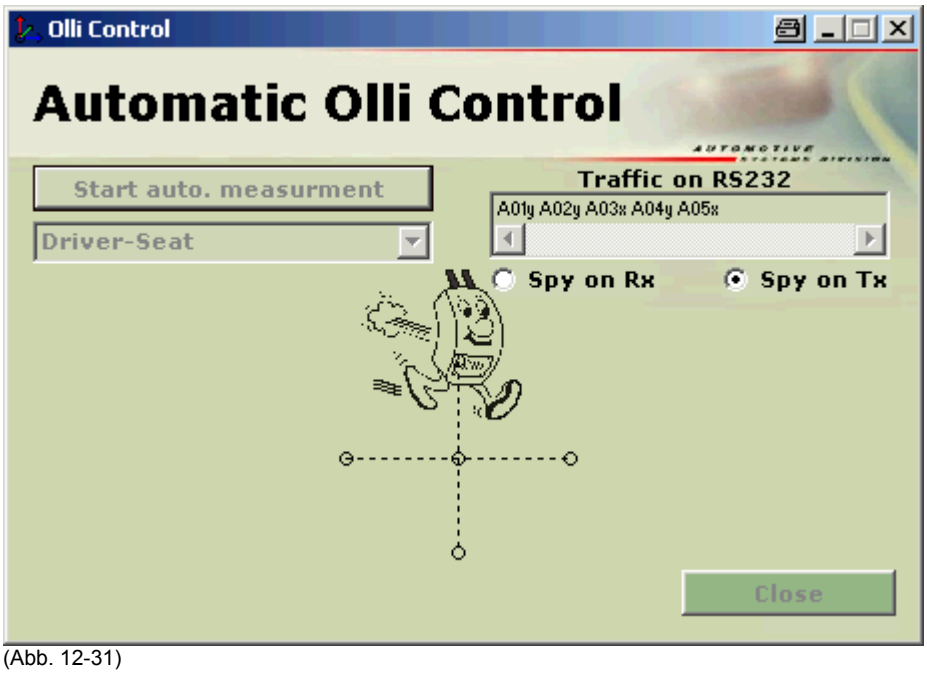
Der Befehl A03x als Anforderung der Bewegung zur Fahrt auf Messposition 3 wurde über die serielle Schnittstelle gesendet.



(Abb. 12-30)

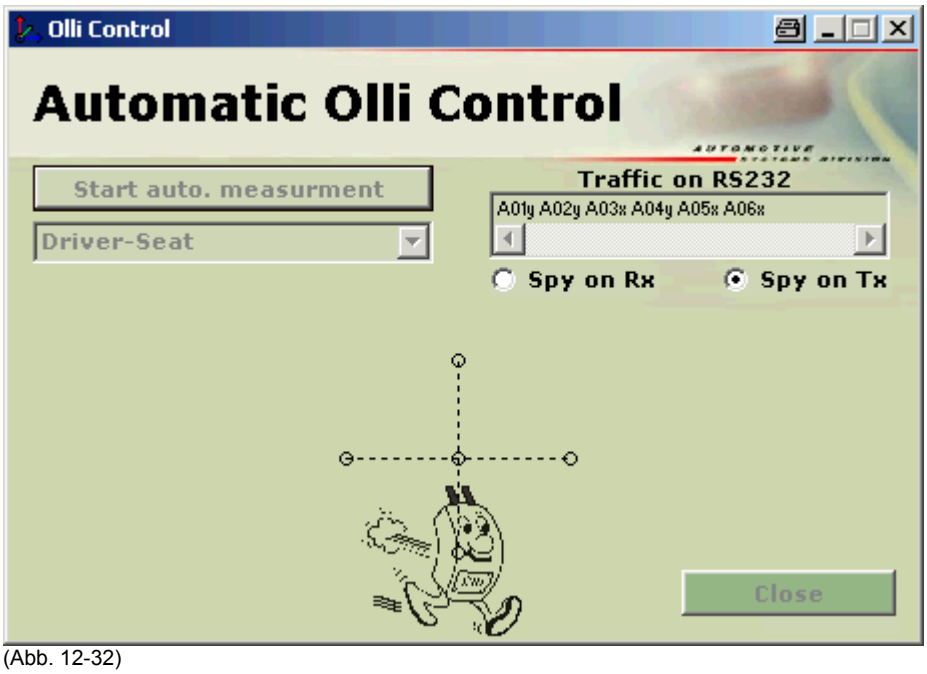
Die grafische Darstellung des Messautomaten hat sich auf die symbolische Position der 3. Messposition bewegt und zeigt damit das Erreichen dieser Position an.

Der Befehl zum Anfahren der Messposition 4 wurde gesendet.



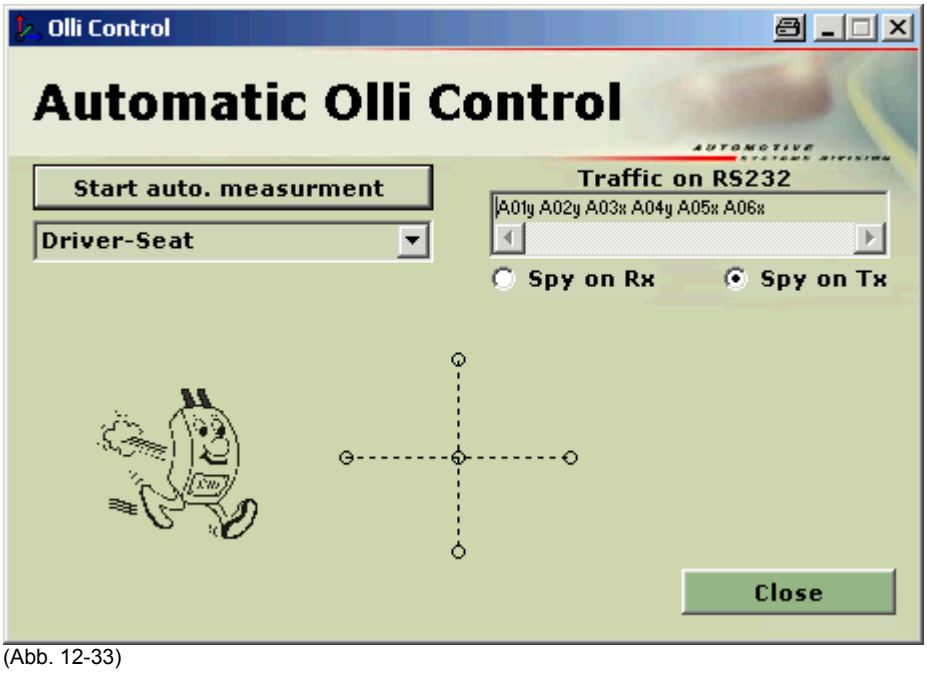
(Abb. 12-31)

OLLICONTROL zeigt an, daß der Automat die Messposition 4 erreicht hat und sendet die Anforderung zur Bewegung auf Messposition 5.



(Abb. 12-32)

Der Messautomat hat das Erreichen der Position 5 bestätigt. OLLICONTROL aktualisiert die Visualisierung und sendet die Anforderung zur Bewegung auf Position 6.



(Abb. 12-33)

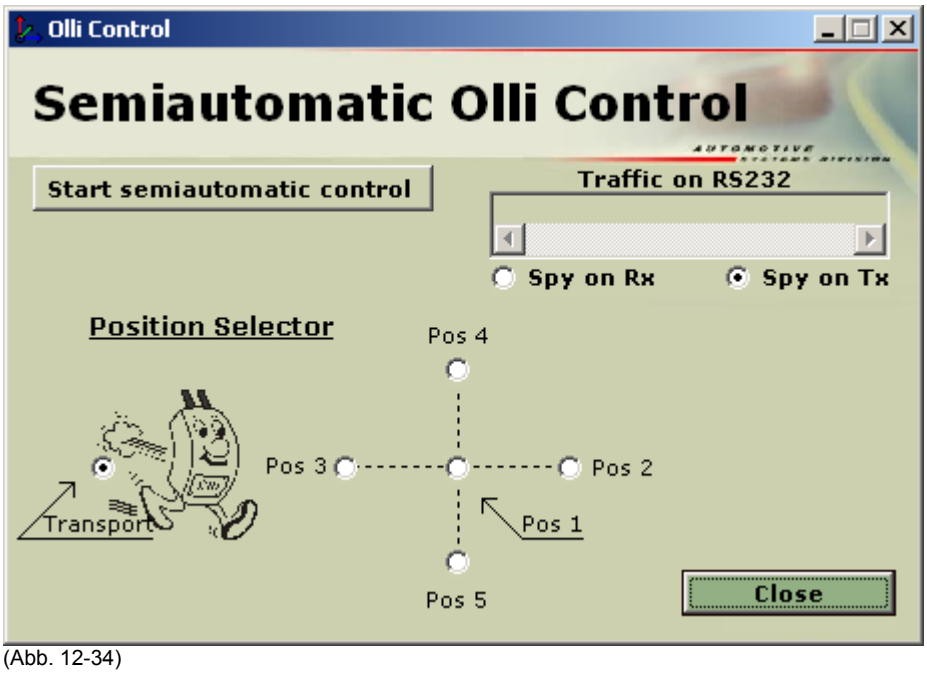
Position 6 entspricht der Grundstellung und wird durch eine Bewegung der grafischen Darstellung des Automaten auf die Startposition symbolisiert.

Im integrierten Fenster „Traffic on RS 232“ ist die Befehlsabfolge der gesendeten Kommandos zu erkennen.

Betriebsart Teilautomatik

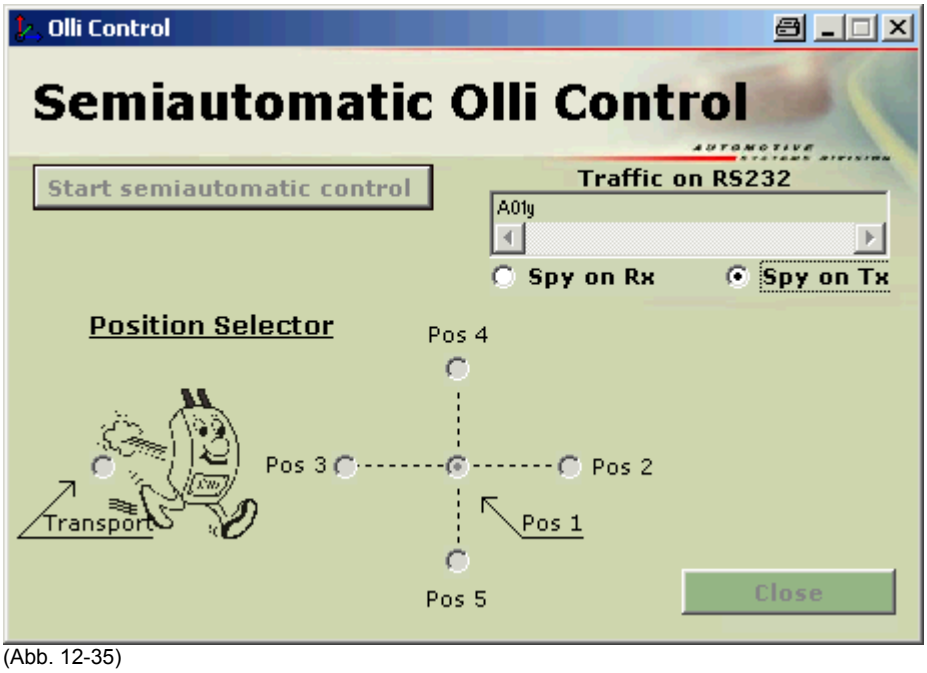
Wie bereits in den ersten Abschnitten dieses Kapitels erklärt wurde, kann in der Betriebsartenauswahl (Abbildung 12-22) über den Button Teilautomatik die Betriebsart Teilautomatik angefordert werden.

Folgende Oberfläche (Abbildung 12-34) steht dem Bediener zur Steuerung des Messautomaten in der Betriebsart Teilautomatik zur Verfügung.



(Abb. 12-34)

Eine beliebige anzufahrende Position kann durch anklicken der entsprechenden „checkbox“ ausgewählrt werden.

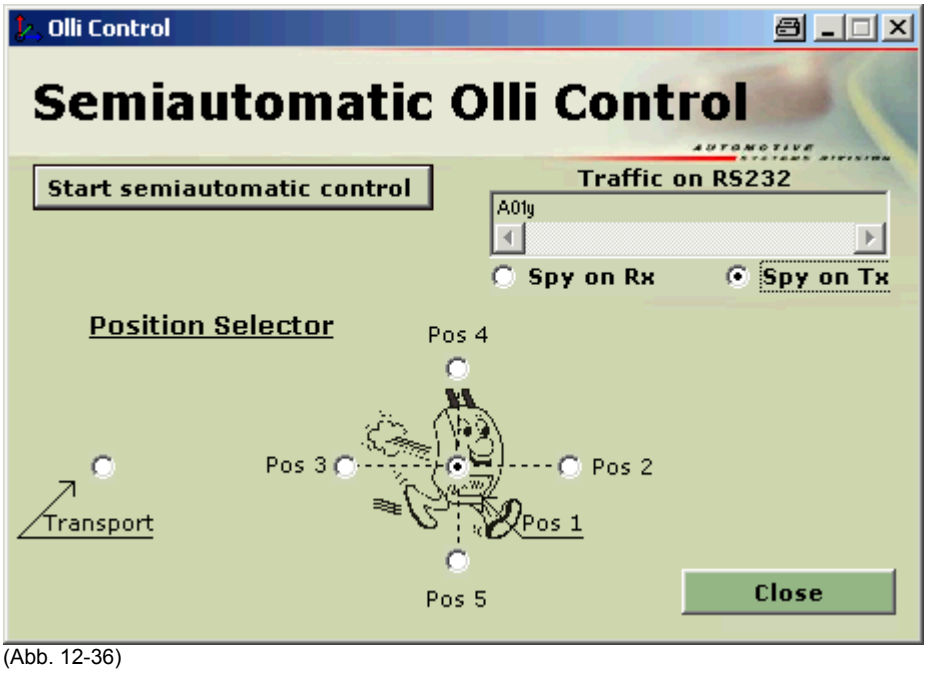


(Abb. 12-35)

Nach dem Auswählen einer anzufahrenden Position muß der Start-Button „Start semiautomatic control“ gedrückt werden.

Im integrierten Fenster „Traffic on RS 232“ kann beobachtet werden, welcher Befehl über die serielle Schnittstelle gesendet wurde.

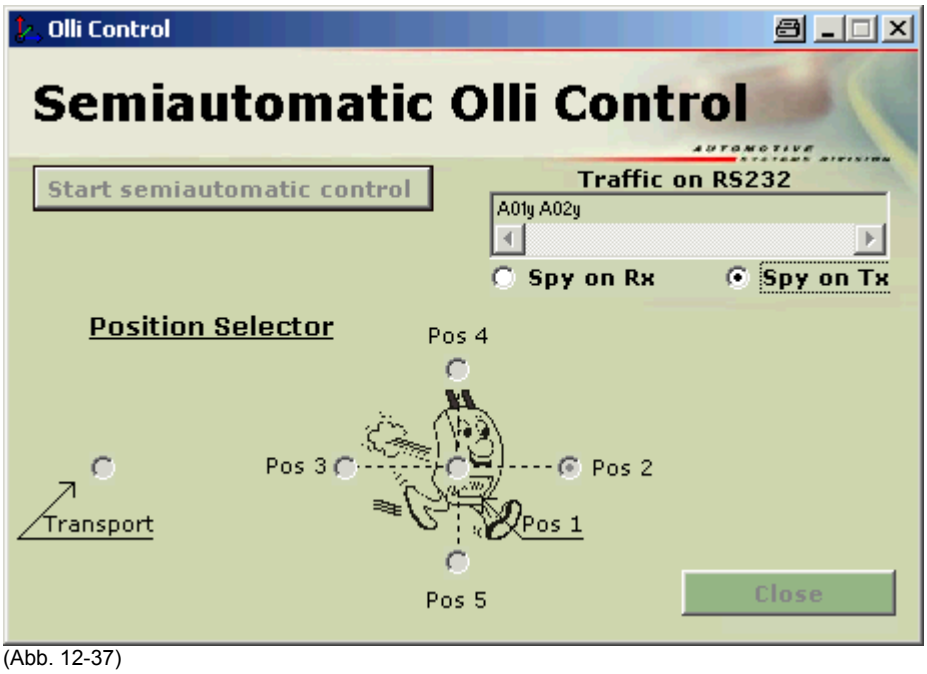
Während der Fahrt des Messautomaten auf die angeforderte Position werden die Felder im Fenster zur Betriebsart Teilautomatik grau hinterlegt und verhindern somit eine Fehlbedienung durch den Benutzer.



(Abb. 12-36)

Ist die angeforderte Position erreicht, sind die Auswahlfelder im Fenster zur Betriebsart Teilautomatik wieder beschreibbar.

Wie auch in der Betriebsart Automatik symbolisiert die grafische Darstellung des Messautomaten in der Betriebsart Teilautomatik durch ihre Positionierung im Fadenkreuz die tatsächlich angefahrene Position des Automaten.



(Abb. 12-37)

Die Aktualisierung der Oberfläche von OLLICONTROL wird durch die Bestätigung des Automaten zum Erreichen der angeforderten Position hervorgerufen.

Ist die angeforderte Position erreicht, kann die nächste anzufahrende Position angefordert werden.

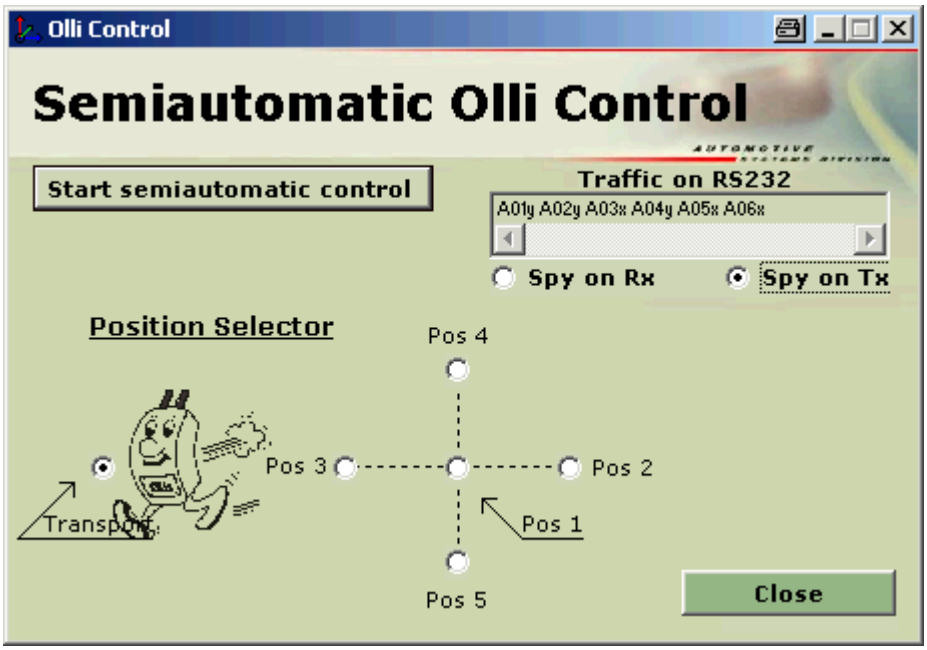
Im Beispiel in Abbildung 12-37 ist nach der Position 1 die Position 2 angefordert worden. Die Bewegung des Messautomaten auf die angeforderte Messposition 2 ist bereits aktiv.

Auch das Erreichen dieser Position wird der Automat bestätigen.

Dieses Verfahren gilt für alle auszuwählenden Messpositionen, die in der Betriebsart Teilautomatik angefordert werden können.

Zu beachten ist, daß zum erneuten Wechsel der Betriebsart, der Messautomat in seine Grundstellung gefahren werden muß.

Hierzu muß die „checkbox Transport“ angeklickt und die Bewegung über den Button „Start semiautomatic control“ angefordert werden.



(Abb. 12-38)

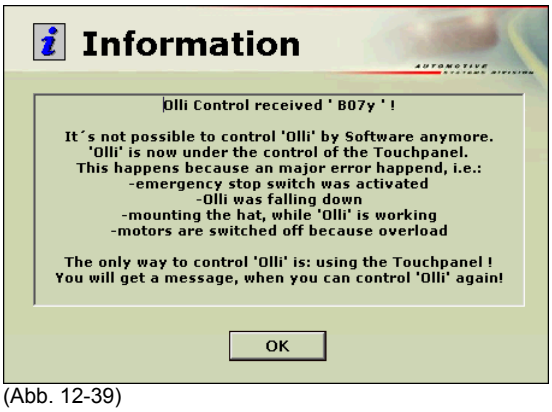
Wie auch bei den anderen Positionen wird das Erreichen der Grundstellung über die Bewegung der grafischen Darstellung des Messautomaten in OLLICONTROL symbolisiert.

Auch von der Bedienung her, hat die Grundstellung gegenüber den anderen Positionen keine Sonderstellung.

Ungeplante Ereignisse

Wie zu Anfang des Kapitels bereits erklärt, werden nicht alle Fehlermeldungen, die OLLICONTROL zu generieren in der Lage ist, hier dokumentiert. Folgende Informationen tragen zum Basiswissen über OLLICONTROL bei.

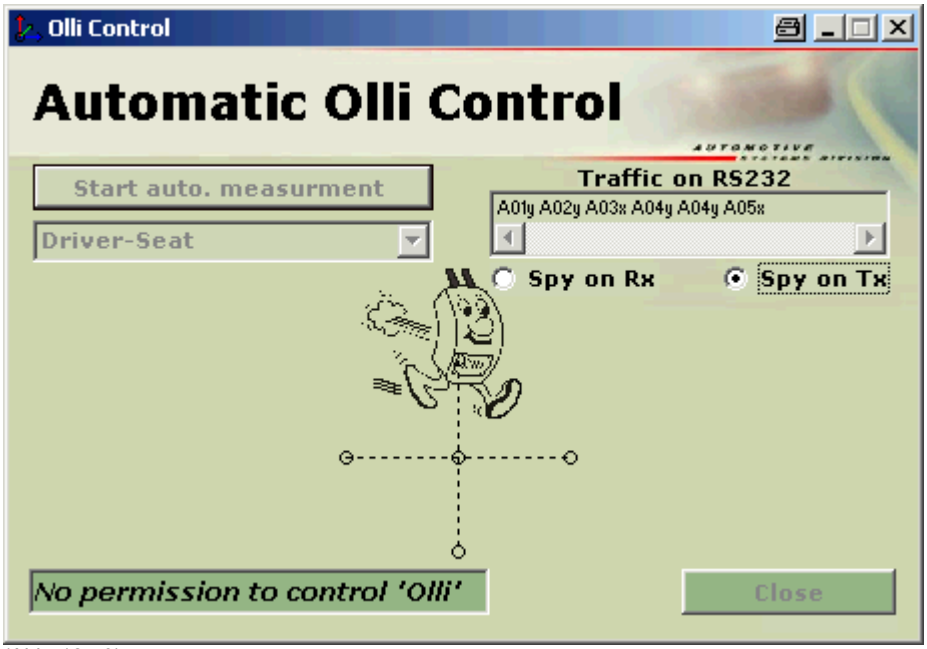
Zwischen OC und dem Messautomat findet ein Datenaustausch statt. Jedoch werden nicht nur Befehle zur Anforderung von Bewegungen gesendet oder das Erreichen einer angeforderten Position bestätigt, sondern es werden auch Daten gesendet, die rein zu Diagnosezwecken dienen.



(Abb. 12-39)

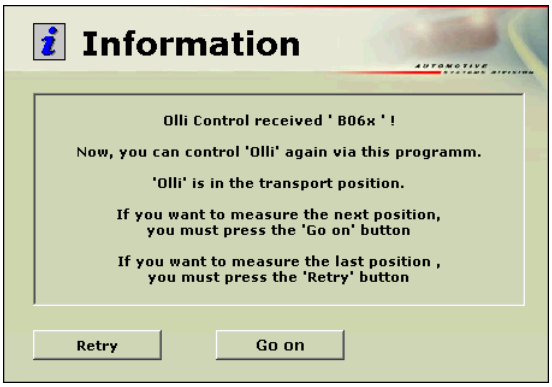
Die HMI ist gegenüber OC dominant, darüber wurde bereits im Kapitel zur HMI informiert. Das hat zur Folge, daß wenn der Messautomat über die HMI bedient wird, OC die Freigabe zur Bedienung des Automaten entzogen wird.

Gleichzeitige Bewegungsanforderungen von unterschiedlichen Orten wird somit verhindert. Sach- und Personenschaden wird durch unterbundene plötzliche Bewegungen des Automaten abgewendet.



(Abb. 12-40)

Der Bediener erhält eine quittierungspflichtige Meldung. Im Hauptfenster der jeweiligen Betriebsart wird eine Information zur Bedienerführung eingeblendet „No permission to control Olli“.



(Abb. 12-41)

Ist die Software OC wieder zur Bedienung des Messautomaten freigegeben, wird der Benutzer über eine quittierpflichtige Meldung darüber informiert.

Diese Meldung kann von dem dargestellten Informationsfenster in Abbildung 12-41 variieren, da dies zu unterschiedlichen Zeitpunkten geschehen kann und zusätzlich vom Betriebszustand des Messautomaten vor dem Entzug der Freigabe, abhängt.

Im Hauptfenster der entsprechenden Betriebsart ist die Information „No permission to control Olli“ wieder ausgeblendet. Die Software OC ist wieder freigegeben, den Messautomaten zu steuern.

Informationen zu weiteren Meldungsfenstern erhalten Sie im Kapitel Kommunikation. Dort sind die Codes, die zum Datenaustausch benutzt werden, hinterlegt.

Die Befehlscodes werden in den Meldungsfenstern mit angezeigt.

13 Stromlaufpläne

13.1 Allgemein

Die Stromlaufpläne wurden mit EPLAN 5.50.5 erstellt. Sie dienen in diesem Kapitel zur Übersicht über das Gesamtsystem.

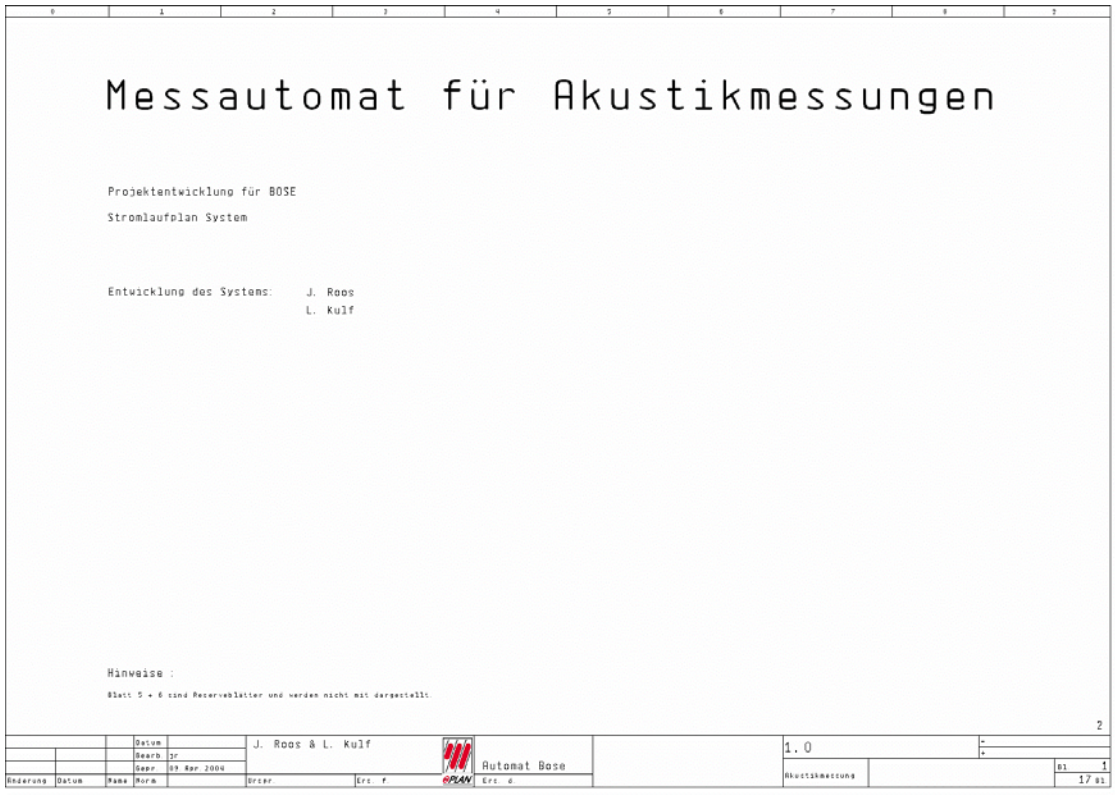
Auf der Dokumentations – CDR erhalten Sie die Stromlaufpläne zum Messautomat im Acrobat Reader pdf-Dateiformat und im Grafikdateiformat *.tif.

Eine Kopie der Stromlaufpläne im EPLAN-Format wurde auf der Dokumentations – CDR nicht abgelegt. Um diese Projektdateien in Originalformat ansehen zu können, müssten Sie EPLAN 5.50.5 installieren. Dies wird jedoch meist aus wirtschaftlichen Gründen abgelehnt.

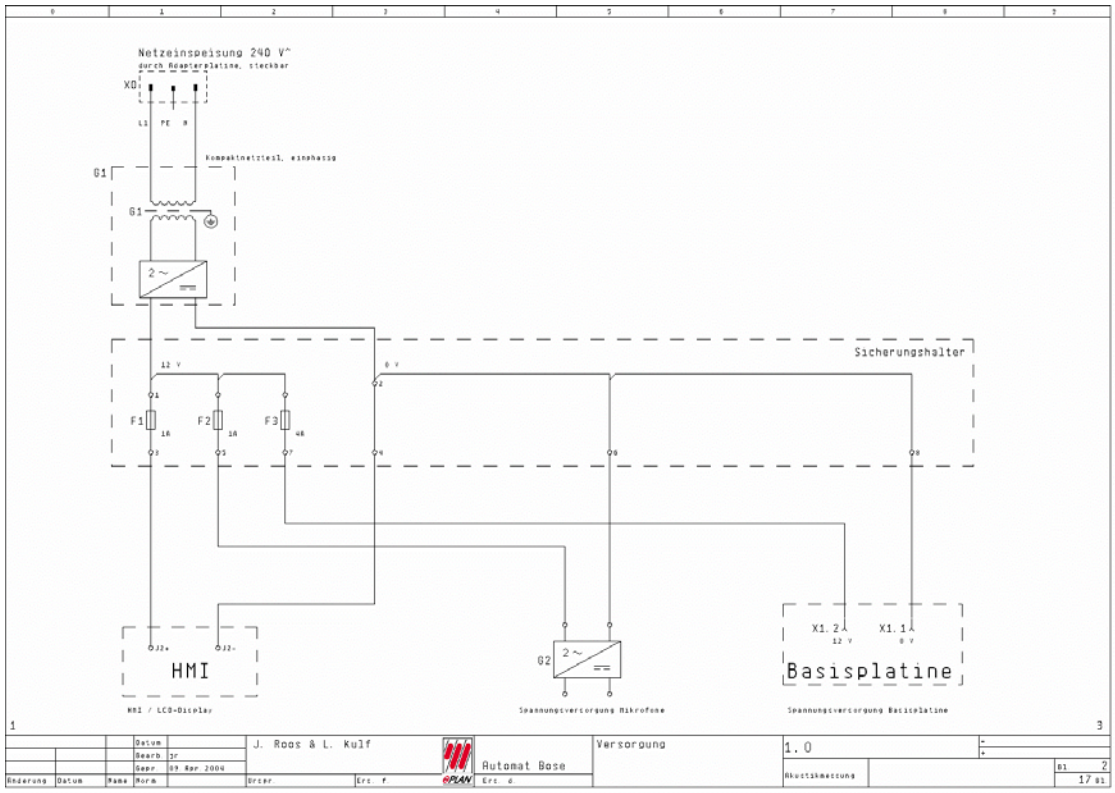
Um die Dateien dennoch im EPLAN-Format zu erhalten, wenden Sie sich bitte an Ihren BOSE®- Ansprechpartner.

Die Blätter 5 und 6 des Stromlaufplanes sind Reserveblätter, im Stromlaufplan sind diese Blätter mit Leerblatt gekennzeichnet. Es wird darauf verzichtet, diese Blätter hier darzustellen.

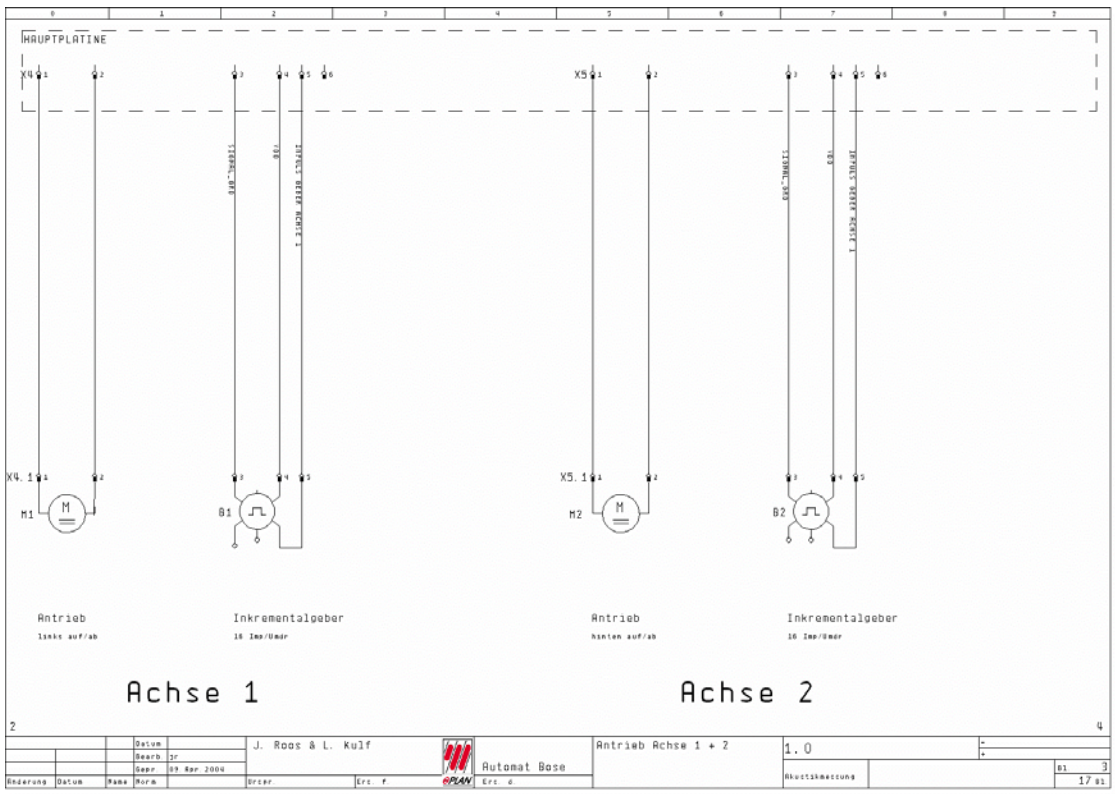
13.2 Stromlaufpläne Messautomat



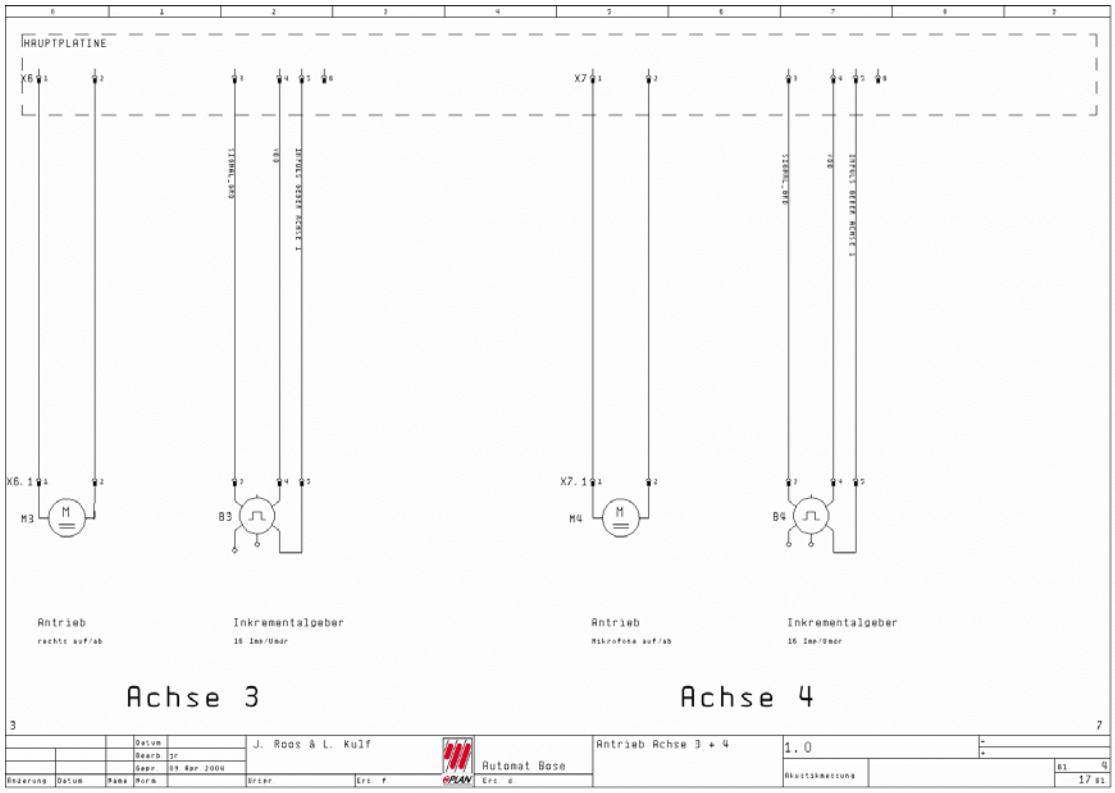
(Abb. 13-1)



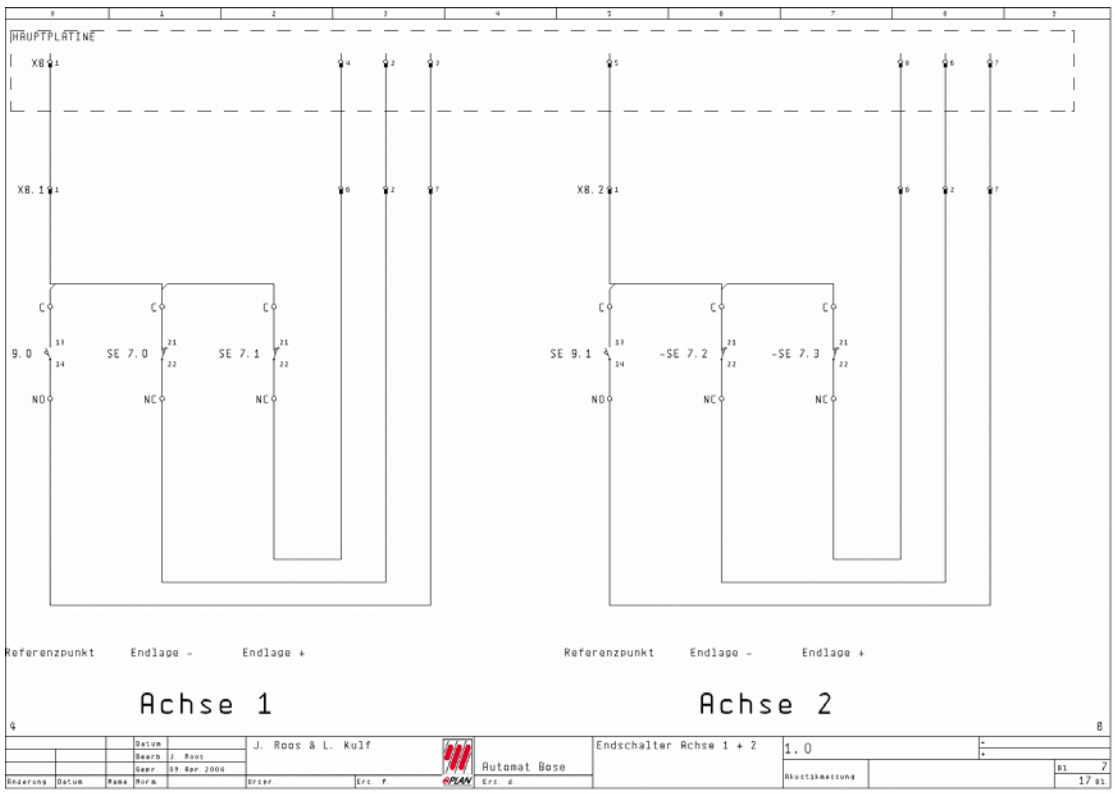
(Abb. 13-2)



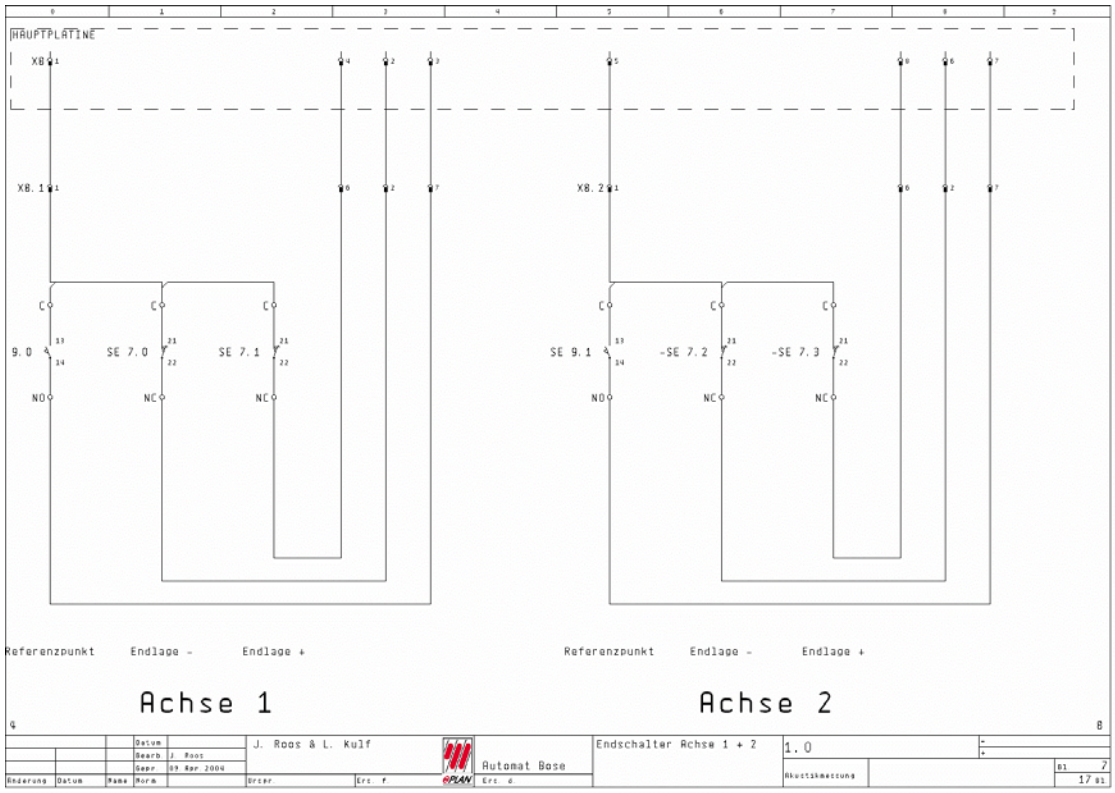
(Abb. 13-3)



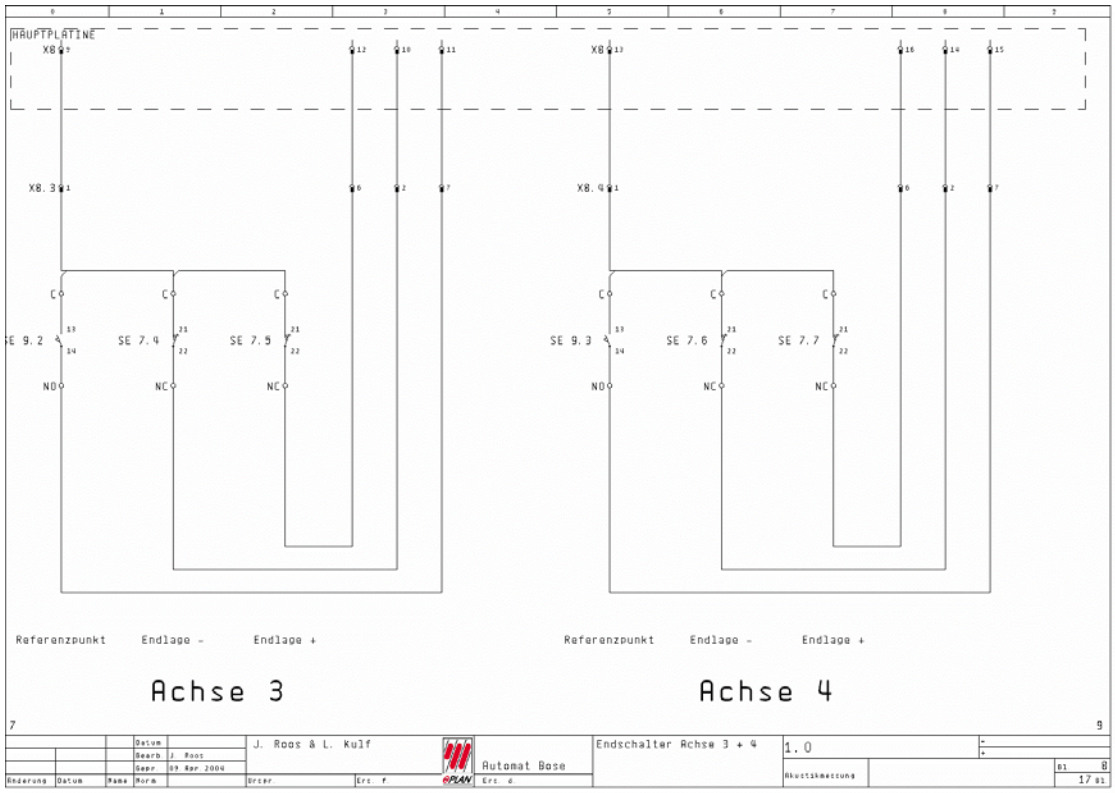
(Abb. 13-4)



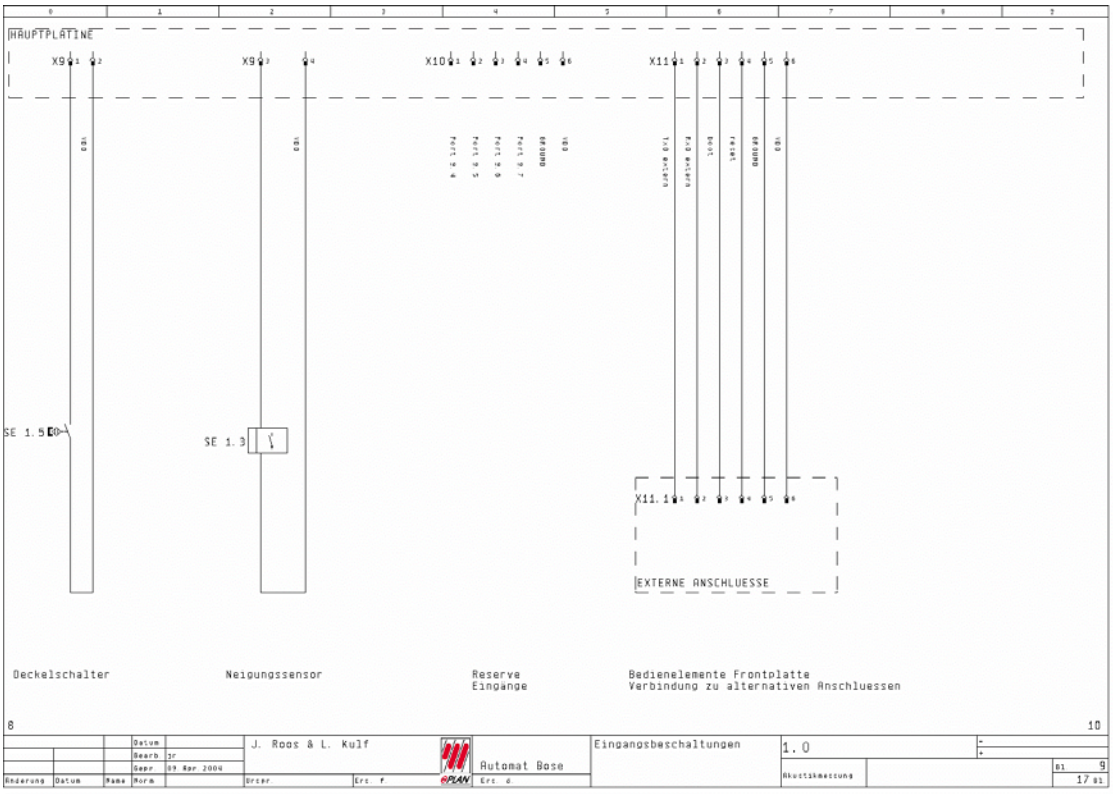
(Abb. 13-5)



(Abb. 13-6)



(Abb. 13-7)



(Abb. 13-8)

Die zugehörigen Klemmenpläne sind auf der beiliegenden Dokumentations – CDR enthalten. Die Pläne sind ebenso wie die Stromlaufpläne im Acrobat Reader pdf-Format und im Grafikdateiformat *.tif enthalten.

Auf eine Darstellung der Klemmenpläne wird in diesem Kapitel bewußt verzichtet.

14 Adapterplatine

14.1 Allgemein

Die Adapterplatine beinhaltet mehrere Funktionen. Diese Funktionen wurden bereits im Kapitel 6.2 angesprochen.

Die Adapterplatine ist im Gehäuse des Messautomaten installiert. Die Anschlüsse sind über eine Frontplatte nach außen geführt.

Das Kapitel 14 – Adapterplatine wurde der Dokumentation hinzugefügt, um über einige Details der Adapterplatine zu informieren.

14.2 Allgemeine Funktionen

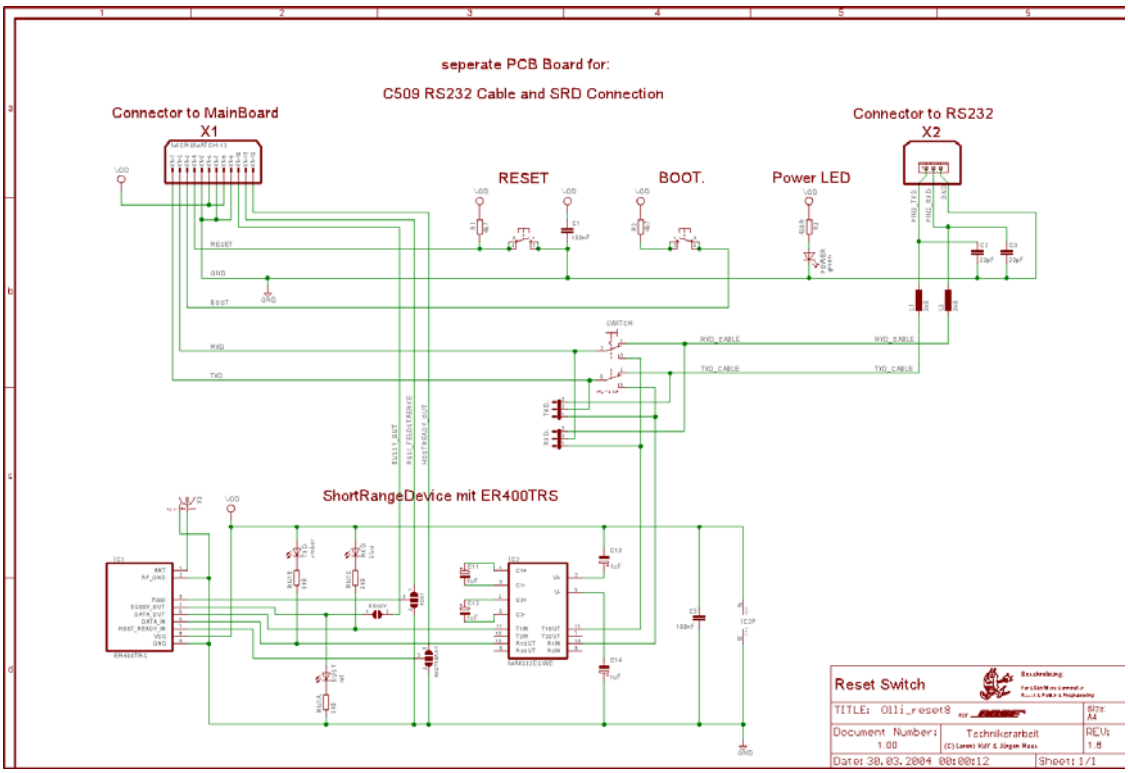
- Spannungsversorgung des Messautomaten ist nun in steckbarer Ausführung möglich.
- Die Taster „Boot“ und „Reset“ der Hauptplatine sind nach außen geführt. Um ein Softwareupdate durchzuführen, muß der Messautomat nicht mehr geöffnet werden.
- Die serielle Schnittstelle S0 ist über die Adapterplatine von außen erreichbar.
- Die Adapterplatine ist mit einem Funkmodul zur Datenübertragung ausgerüstet. Diese Funkverbindung kann alternativ zur konventionellen RS-232 Verbindung genutzt werden.

14.3 Frontplatte

(Abb. 14-1)

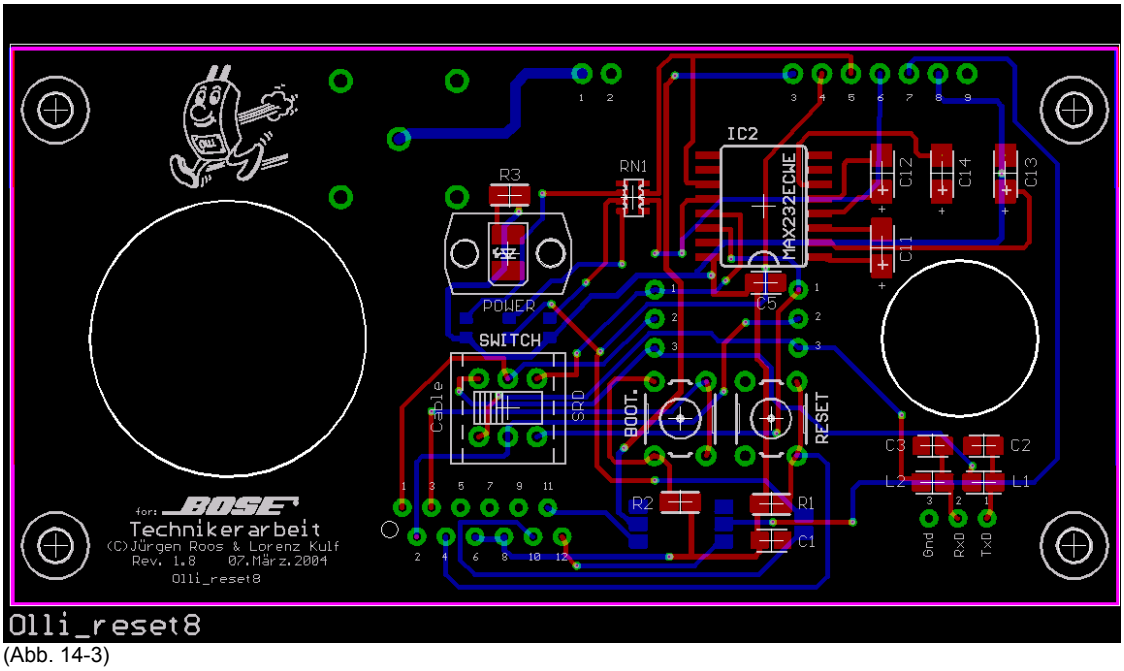
- Die Frontplatte ist in Aluminiumausführung. Die Steckverbindungen wurden vertauschsicher eingearbeitet.
- Die Beschriftungen sind dauerhaft graviert und gedruckt.
- Die Taster sind gegen versehentliches Betätigen gesichert.

14.4 Schaltplan der Adapterplatine



(Abb. 14-2)

14.5 Layout der Adapterplatine



(Abb. 14-3)

Anhang

Im Anhang finden Sie Datenblätter zu den wichtigsten Komponenten des Messautomaten. Es wird jedoch davon abgesehen, die Datenblätter aller verbauten Komponenten an dieser Stelle abzudrucken.

Zusätzliche Informationen erhalten Sie auf der beiliegenden Dokumentations – CDR. Auf dieser CD-ROM finden Sie zu allen Komponenten dieses Messautomaten Datenblätter im Adobe Acrobat Reader pdf-Format.

Anhang A : Auszug Datenblätter - Komponenten der Mechanik

Anhang B : Auszug Datenblätter - Komponenten der Elektrik

Anhang C : Auszug Datenblätter - Komponenten der Elektronik

Anhang D : Stücklisten

Anhang E : Notizen

Anhang F : Dokumentations – CDROM

Die Seitennummerierungen in den Dokumenten der Hersteller der Komponenten haben in diesem Anhang keine Bedeutung.

A 1 – Kugelgewindetrieb

THK

Präzisions-Kugelgewindetrieb BNK mit bearbeiteten Wellenenden

Der Präzisions-Kugelgewindetrieb mit bearbeiteten Wellenenden Typ BNK ist ein kompakter und einbaufertiger Kugelgewindetrieb mit standardisierten Abmessungen. Für ein einfaches Handling sind die Wellenenden und die Lagereinheiten werkseitig aufeinander abgestimmt.

Die vorgesehene Lagerung ist bei den kleineren Kugelgewindetrieben (BNK0401 und BNK0601) fest-frei und bei den größeren fest-los. Dies und die fertige Endenbearbeitung ermöglichen einen direkten Anschluss des Motors.

Endenbearbeitung und -lagerung sowie Muttergehäuse für den Präzisions-Kugelgewindetrieb BNK

Baugröße		BNK 0401			0601			0801			0802			1002			1004			1010			1202		
Toleranzklasse		C3, C5, C7			C3, C5, C7			C3, C5, C7			C3, C5, C7			C3, C5, C7			C3, C5, C7			C5, C7			C3, C5, C7		
Axialspiel		G0	GT	G2	G0	GT	G2	G0	GT	G2	G0	GT	G2	G0	GT	G2	G0	GT	G2	G0	GT	G2	G0	GT	G2
Hublänge (mm)	20	●																							
	40	●			●			●			●														
	50													●			●						●		
	70	●			●			●			●														
	100				●			●			●			●			●			●			●		
	150							●			●			●			●			●			●		
	200													●			●			●			●		
	250																●			●			●		
	300																			●					
	350																								
	400																								
	450																								
	500																								
	550																								
	600																								
	700																								
	800																								
	900																								
	1000																								
	1100																								
	1200																								
	1400																								
	1600																								
Festlager Blockausführung		EK4			EK5			EK6			EK6			EK8			EK10			EK10			EK10		
Festlager Flanschausführung		FK4			FK5			FK6			FK6			FK8			FK10			FK10			FK10		
Loslager Blockausführung		-			-			EF6			EF6			EF8			EF10			EF10			EF10		
Loslager Flanschausführung		-			-			FF6			FF6			FF6			FF10			FF10			FF10		
Muttergehäuse		-			-			-			-			-			MC1004			MC1004			-		

Axialspiel G0: mit Vorspannung
GT: max. 0,005 mm
G2: max. 0,02 mm

Zur Endenlagerung siehe S. 348 ff; zum Muttergehäuse siehe S. 363 ff.

Staubschutz und Schmierung

Die Muttern sind bereits mit einer geeigneten Fettfüllung versehen. Als Staubschutz sind in den Muttern ab BNK0802 Labyrinth-Dichtungen integriert. Die Muttern der Baugrößen BNK1510, 1520, 1616, 2020 und 2520 sind mit Endkappen ausgestattet, die aber auch die Funktion einer Labyrinth-Dichtung übernehmen.

Bei bestimmten Betriebsbedingungen, unter denen das Eindringen von Staub und Schmutz in die Mutter wahrscheinlich ist, muss der Kugelgewindetrieb vollständig abgedeckt sein. Dies geschieht üblicherweise mit einem Faltenbalg oder einer Blechabdeckung.

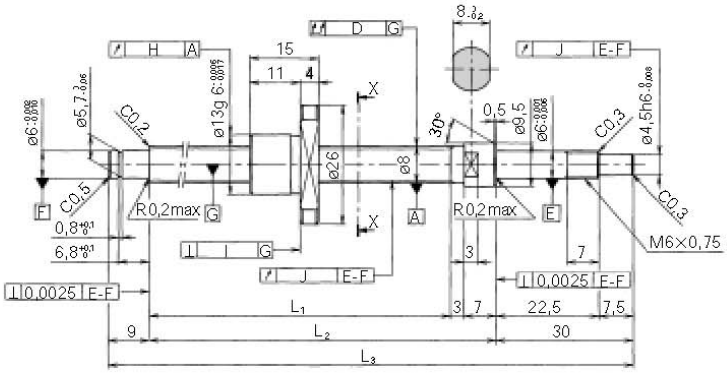
C

BNK									
1205	1402	1404	1408	1510	1520	1616	2010	2020	2520
C3, C5, C7	C3, C5, C7	C3, C5, C7	C5, C7	C5, C7	C5, C7	C5, C7	C5, C7	C5, C7	C5, C7
G0 GT G2	G0 GT G2	G0 GT G2	G0 GT G2	G0 GT G2	G0 GT G2	G0 GT G2	G0 GT G2	G0 GT G2	G0 GT G2
●	●								
●	●	●							
●	●	●	●	●	●	●			
●	●	●	●	●	●	●			
●			●	●	●	●			
	●	●	●	●	●	●	●	●	
			●	●	●	●			
		●	●	●	●	●	●	●	
			●	●	●	●			
			●	●	●	●	●	●	●
			●	●	●	●			
			●	●	●	●	●	●	●
			●	●	●	●	●	●	
				●	●	●	●	●	●
							●	●	
							●	●	●
							●	●	
									●
									●
									●
EK10	EK12	EK12	EK12	EK12	EK12	EK12	EK15	EK15	EK20
FK10	FK12	FK12	FK12	FK12	FK12	FK12	FK15	FK15	FK20
EF10	EF12	EF12	EF12	EF12	EF12	EF12	EF15	EF15	EF20
FF10	FF12	FF12	FF12	FF12	FF12	FF12	FF15	FF15	FF20
MC1205	-	-	MC1408	MC1408	MC1408	MC1408	MC2010	MC2020	-

THK

Präzisions-Kugelgewindetrieb BNK 0801-3

Durchmesser: 8 mm; Steigung: 1 mm



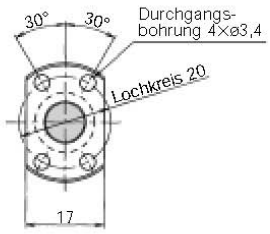
Baugröße [1) 2) 3)]	max. Hub	Lange Gewindespindel L_1	L_2	L_3
BNK0801-3G0+115LC3Y	40	66	76	115
BNK0801-3G0+115LC5Y				
BNK0801-3G2+115LC7Y				
BNK0801-3G0+145LC3Y	70	96	106	145
BNK0801-3G0+145LC5Y				
BNK0801-3G2+145LC7Y				
BNK0801-3G0+175LC3Y	100	126	136	175
BNK0801-3G0+175LC5Y				
BNK0801-3G2+175LC7Y				
BNK0801-3G0+225LC3Y	150	176	186	225
BNK0801-3G0+225LC5Y				
BNK0801-3G2+225LC7Y				

1) Die Baugröße BNK0801 ist auch in korrosionsbeständiger Ausführung erhältlich. Bei Anfragen oder Bestellungen fügen Sie bitte das Kennzeichen M an die Bestellbezeichnung an.

Beispiel:
BNK0801-3G0 + 115LC3YM
└── Kennzeichen für korrosionsbeständige Ausführung

2) Bei den Toleranzklassen C3 und C5 ist das Axialspiel GT standardmäßig eingestellt.

3) Zur Zusammensetzung der Bestellbezeichnung siehe S. 106.



Ansicht X-X

Daten Kugelgewindetrieb			
Steigung [mm]	1		
Kugelmittenkreis [mm]	8,2		
Kerndurchmesser [mm]	7,3		
Steigungsrichtung	rechts		
Anzahl Reihen × Umlauf	3 × 1		
Kennzeichen für Axialspiel	G0	GT	G2
Axialspiel [mm]	0	0,005 max.	0,02 max.
dynamische Tragzahl C_a [kN]	0,64		
statische Tragzahl C_{0a} [kN]	1,4		
Vorspannmoment [Nm]	~ $1,8 \times 10^{-2}$	-	
Abstandskugeln	keine		

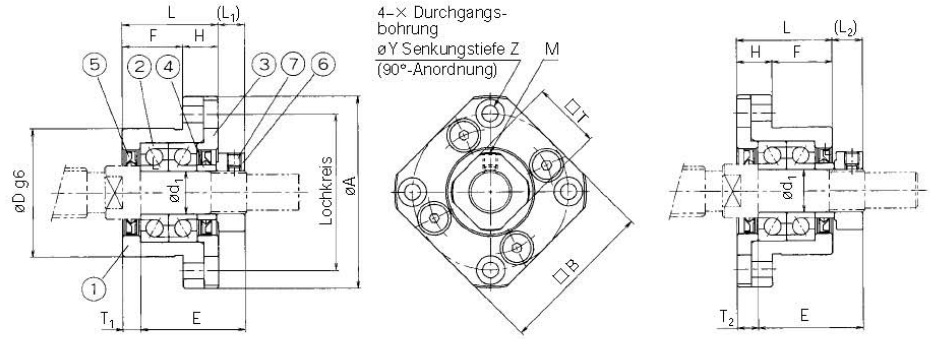
Einheit: mm

Spindel-gesamtrundlauf D	Muttern-rundlauf H	Flansch-rechtwinkligkeit I	Rundlauf Spindel-Lagersitz J	Wegabweichung und Wegschwankung mittlere Wegabweichung	Variation
0,025	0,009	0,008	0,008	±0,008	0,008
0,025	0,012	0,010	0,010	±0,018	0,018
0,035	0,020	0,014	0,014	mittlere Wegabweichung ±0,05/300	
0,030	0,009	0,008	0,008	±0,008	0,008
0,035	0,012	0,010	0,010	±0,018	0,018
0,050	0,020	0,014	0,014	mittlere Wegabweichung ±0,05/300	
0,030	0,009	0,008	0,008	±0,010	0,008
0,035	0,012	0,010	0,010	±0,020	0,018
0,050	0,020	0,014	0,014	mittlere Wegabweichung ±0,05/300	
0,035	0,009	0,008	0,008	±0,010	0,008
0,050	0,012	0,010	0,010	±0,020	0,018
0,065	0,020	0,014	0,014	mittlere Wegabweichung ±0,05/300	

A 2 – THK Lager

THK

Festlagereinheit in Flanschausführung Typ FK



Montagemethode A

Montagemethode B

FK10~30

Baugröße	Durch-messer Lager-zapfen d_1	L	H	F	E	D	A	Lochkreis
FK10	10	27	10	17	29,5	$34^{-0.009}_{-0.025}$	52	42
FK12	12	27	10	17	29,5	$36^{-0.009}_{-0.025}$	54	44
FK15	15	32	15	17	36	$40^{-0.009}_{-0.025}$	63	50
FK20	20	52	22	30	50	$57^{-0.010}_{-0.029}$	85	70
FK25	25	57	27	30	60	$63^{-0.010}_{-0.029}$	98	80
FK30	30	62	30	32	61	$75^{-0.010}_{-0.029}$	117	95

Einheit: mm

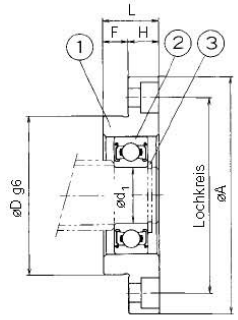
B	Montagemethode A		Montagemethode B		X	Y	Z	M	T	eingebautes Kugellager
	L_1	T_1	L_2	T_2						
42	7,5	5	8,5	6	4,5	8	4	M3	16	7000DFGMP5
44	7,5	5	8,5	6	4,5	8	4	M3	19	7001DFGMP5
52	10	6	12	8	5,5	9,5	6	M3	22	7002DFGMP5
68	8	10	12	14	6,6	11	10	M4	30	7204DFGMP5
79	13	10	20	17	9	15	13	M5	35	7205DFGMP5
93	11	12	17	18	11	17,5	15	M6	40	7206DFGMP5

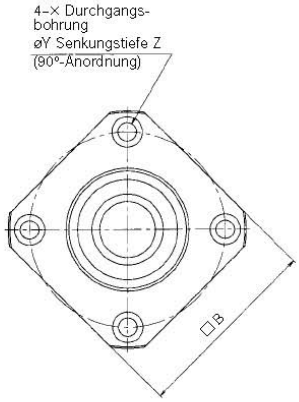
J

Inhalt des Lager-Sets FK10 ~ 30

lfd. Nummer	Teilebezeichnung	Anzahl
1	Blockgehäuse	1
2	Lagersatz	1
3	Gehäusedeckel	1
4	Distanzring	2
5	Dichtung	2
6	Sicherungsmutter	1
7	Innensechskantschraube (mit Druckstück)	1

Loslagereinheit in Blockausführung Typ FF





Baugröße	Durch-messer Lager-zapfen d_1	L	H	F	D	A
FF6	6	10	6	4	$22^{-0.007}_{-0.020}$	36
FF10	8	12	7	5	$28^{-0.007}_{-0.020}$	43
FF12	10	15	7	8	$34^{-0.009}_{-0.025}$	52
FF15	15	17	9	8	$40^{-0.009}_{-0.025}$	63
FF20	20	20	11	9	$57^{-0.010}_{-0.029}$	85
FF25	25	24	14	10	$63^{-0.010}_{-0.029}$	98
FF30	30	27	18	9	$75^{-0.010}_{-0.029}$	117

Einheit: mm

Lochkreis	B	X	Y	Z	eingebautes Kugellager	Wellen-sicherungsring
28	28	3,4	6,5	4	606ZZ	C 6
35	35	3,4	6,5	4	608ZZ	C 8
42	42	4,5	8	4	6000ZZ	C10
50	52	5,5	9,5	5,5	6002ZZ	C15
70	68	6,6	11	6,5	6204ZZ	C20
80	79	9	14	8,5	6205ZZ	C25
95	93	11	17,5	11	6206ZZ	C30

J

Inhalt des Lager-Sets FF6 ~ 30

lfd. Nummer	Teilebezeichnung	Anzahl
1	Blockgehäuse	1
2	Lager	1
3	Wellensicherungsring	1

A 3 – Federstegkupplung

MWC MWS

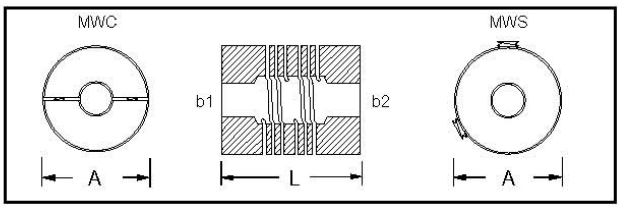
RULAND FLEXBEAM™-2 KUPPLUNGEN

EDELSTAHL

FLEXIBLE KUPPLUNG

MWC

MWS



- Torsionssteifer, höhere Drehmomente übertragbar
- Zwei Sätze von 2 Spiralschnitten für eine hohe Torsionsteife und geringen Aufwickeleffekt
- Kleinere Rückstellkräfte im Vergleich zur Edelstahlserie MF (Seite 9)
- Korrosionsbeständig
- Kein Drehspiel
- Ausgleich radialer, winkliger, axialer und kombinierter Verlagerung
- Klemmausführung dynamisch ausgeglichen

TEILENUMMER				ABMESSUNGEN					LEISTUNGSDATEN				
											VERLAGERUNG		
KLEMM-AUSF.	AUSF. MIT STELL-SCHRAUBE	∅ b1	∅ b2	∅ A (mm)	L (MWC)	L (MWS)	DIN 912[3] (MWC)	DIN 916[3] (MWS)	Mt stat[2] (Nm)	TORSIONS-STEIFE[2] (Nm/rad)	ANGULAR (Deg)	PARALLEL (mm)	AXIAL (mm)
MWC15	MWS15	3 3,18 4 4,76 5	3 3,18 4 4,76 5	15	22	20	M2	M3	1,30 1,30 1,30 1,20 1,20	25,7 25,7 25,7 22,7 22,7	3°	0,20	0,12
MWC20	MWS20	4 4,76 5 6	4 4,76 5 6	20	28	20	M3	M3	2,00 2,00 2,00 1,70	58,5 58,5 58,5 44,4	3°	0,20	0,12
MWC25	MWS25	6 6,35 8 9,53 10	6 6,35 8 9,53 10	25	30	24	M3	M4	5,10 5,10 5,10 4,60 4,60	98,8 98,8 98,8 69,1 69,1	3°	0,38	0,25
MWC30	MWS30	8 9,53 10 12 12,70	8 9,53 10 12 12,70	30	38	30	M4	M5	10,40 10,40 10,40 10,00 10,00	173,6 173,6 173,6 124,6 124,6	3°	0,38	0,25

Anmerkung 1 Alle Werte sind in mm angegeben, sofern nicht speziell andere Maßeinheiten in den Tabellen angegeben sind.

Anmerkung 2 Mt stat gilt bei maximaler Verlagerung
Mt dyn= 0,5 Mt stat
Mt dyn rev (im Reversierbetrieb)= 0,25 Mt stat.

Anmerkung 3 Nähere Angaben zu den verwendeten Schrauben siehe Seite 12.

Anmerkung 4 Führen die Wellen in das Innere der Kupplung, muß ein Abstand zwischen diesen vorgesehen werden.

Erhältliche Bohrung ***b1*** und Bohrung ***b2*** auswählen. Auswahl der Bestellnummer mit Millimeterangaben ergibt sich wie folgt:

7

RULAND MANUFACTURING CO., INC.
http://www.ruland.com

GESELLSCHAFT FÜR ANTRIEBSTECHNIK MBH orbit
Telefon: +49 05331/904096
Telefax: +49 05331/904097

RULAND FLEXBEAM™-3 KUPPLUNGEN

ALUMINIUM

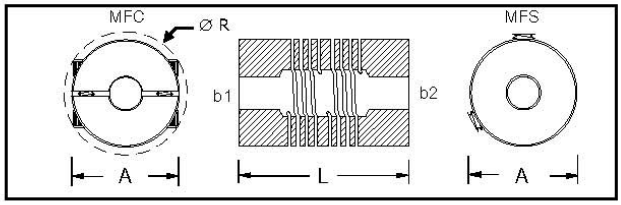
MFC MFS

FLEXIBLE KUPPLUNG

- Durch Drehsteifigkeit und Drehspielfreiheit ideal für Servoantriebe etc.
- Zwei Sätze von 3 Spiralschnitten für hohe Torsionssteife und hohe Drehmomente
- Größere Torsionssteife und für höhere Drehmomente als Aluminiumserie MW
- Kein Drehspiel
- Ausgleich radialer, winkliger, axialer und kombinierter Verlagerung
- Klemmausführung dynamisch ausgeglichen

MFC

MFS



TEILENUMMER				ABMESSUNGEN					LEISTUNGSDATEN				
KLEMM-AUSF.	AUSF. MIT STELL-SCHRAUBE	Ø b1	Ø b2	Ø A (mm)	Ø R (MFC)	L	DIN 912[3] (MFC)	DIN 916[3] (MFS)	Mt stat[2] (Nm)	TORSIONS-STEIFE[2] (Nm/rad)	VERLAGERUNG ANGULAR (Deg)	VERLAGERUNG PARALLEL (mm)	VERLAGERUNG AXIAL (mm)
MFC20	MFS20	5 6 6,35 8	5 6 6,35 8	20	22,8	30	M3	M4	2,90 2,90 2,70 2,30	43,1 43,1 34,3 28,1	3°	0,20	0,12
MFC25	MFS25	6 6,35 8 9,53 10 12 12,70	6 6,35 8 9,53 10 12 12,70	25	30,2	40	M4	M5	4,00 4,00 3,70 3,70 3,70 2,80 2,80	104,2 104,2 66,6 66,6 66,6 36,5 36,5	3°	0,38	0,25
MFC30	MFS30	8 9,53 10 12 12,70 14	8 9,53 10 12 12,70 14	30	34,9	45	M5	M6	7,30 6,30 6,30 5,10 4,70 4,70	168,5 133,3 133,3 83,1 74,4 74,4	3°	0,38	0,25
MFC40	MFS40	10 12 12,70 14 16	10 12 12,70 14 16	40	45,6	55	M6	M6	12,40 12,40 10,70 10,70 10,70	238,7 238,7 146,9 146,9 146,9	3°	0,76	0,38

Anmerkung 1 Alle Werte sind in mm angegeben, sofern nicht speziell andere Maßeinheiten in den Tabellen angegeben sind.

Anmerkung 2 Mt stat gilt bei maximaler Verlagerung
Mt dyn= 0,5 Mt stat
Mt dyn rev (im Reversierbetrieb)= 0,25 Mt stat.

Anmerkung 3 Nähere Angaben zu den verwendeten Schrauben siehe Seite 12.

Anmerkung 4 Führen die Wellen in das Innere der Kupplung, muß ein Abstand zwischen diesen vorgesehen werden.

Erhältliche Bohrung ***b1*** und Bohrung ***b2*** auswählen. Auswahl der Bestellnummer mit Millimeterangaben ergibt sich wie folgt:

25mm Ø 6mm Bohrung
Klemmausf. → **MFC25 - 8 - 6 - A**
8mm Bohrung Aluminium

RULAND MANUFACTURING CO., INC.
http://www.ruland.com

GESELLSCHAFT FÜR ANTRIEBSTECHNIK MBH orbit
Telefon: +49 05331/904096
Telefax: +49 05331/904097

8

MFC MFS

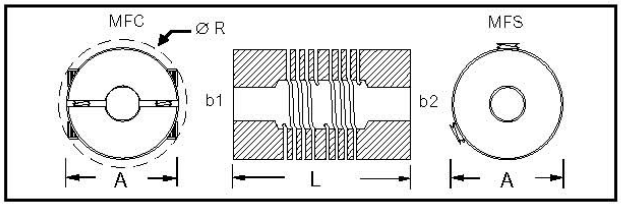
RULAND FLEXBEAM™-3 KUPPLUNGEN

EDELSTAHL

FLEXIBLE KUPPLUNG

MFC

MFS



- Durch Drehsteifigkeit und Drehspielfreiheit ideal für Servoantriebe etc.
- Zwei Sätze von 3 Spiralschnitten und hochfestes Material sorgen für höchste Torsionssteife u. Drehmomentübertragung
- Größere Torsionssteife und für höhere Drehmomente als Aluminiumkupplungen und Edelstahlserie MW
- Korrosionsbeständig
- Kein Drehspiel
- Ausgleich radialer, winkliger, axialer und kombinierter Verlagerung
- Klemmausführung dynamisch ausgeglichen

TEILENUMMER				ABMESSUNGEN					LEISTUNGSDATEN				
KLEMM-AUSF.	AUSF. MIT STELL-SCHRAUBE	Ø b1	Ø b2	Ø A (mm)	Ø R (MFC)	L	DIN 912[3] (MFC)	DIN 916[3] (MFS)	Mt stat[2] (Nm)	TORSIONS-STEIFE[2] (Nm/rad)	VERLAGERUNG ANGULAR (Deg)	VERLAGERUNG PARALLEL (mm)	VERLAGERUNG AXIAL (mm)
MFC20	MFS20	5 6 6,35 8	5 6 6,35 8	20	22,8	30	M3	M4	4,60 4,60 3,60 3,60	83,1 83,1 81,9 81,9	3°	0,20	0,12
MFC25	MFS25	6 6,35 8 9,53 10 12 12,70	6 6,35 8 9,53 10 12 12,70	25	30,2	40	M4	M5	6,10 6,10 5,60 5,60 5,60 3,90 3,90	197,6 197,6 136,4 136,4 136,4 66,6 66,6	3°	0,38	0,25
MFC30	MFS30	8 9,53 10 12 12,70 14	8 9,53 10 12 12,70 14	30	34,9	45	M5	M6	15,50 15,50 15,50 13,50 10,90 10,90	286,5 286,5 286,5 220,4 130,2 130,2	3°	0,38	0,25
MFC40	MFS40	10 12 12,70 14 16	10 12 12,70 14 16	40	45,6	55	M6	M6	23,56 23,56 23,00 23,00 23,00	337,1 337,1 212,2 212,2 212,2	3°	0,76	0,38

Anmerkung 1 Alle Werte sind in mm angegeben, sofern nicht speziell andere Maßeinheiten in den Tabellen angegeben sind.

Anmerkung 2 Mt stat gilt bei maximaler Verlagerung
Mt dyn= 0,5 Mt stat
Mt dyn rev (im Reversierbetrieb)= 0,25 Mt stat.

Anmerkung 3 Nähere Angaben zu den verwendeten Schrauben siehe Seite 12.

Anmerkung 4 Führen die Wellen in das Innere der Kupplung, muß ein Abstand zwischen diesen vorgesehen werden.

Erhältliche Bohrung ***b1*** und Bohrung ***b2*** auswählen. Auswahl der Bestellnummer mit Millimeterangaben ergibt sich wie folgt:

9

RULAND MANUFACTURING CO., INC.
http://www.ruland.com

GESELLSCHAFT FÜR ANTRIEBSTECHNIK MBH orbit
Telefon: +49 05331/904096
Telefax: +49 05331/904097

A 4 – Getriebe

FAULHABER

Planetengetriebe 0,5 Nm

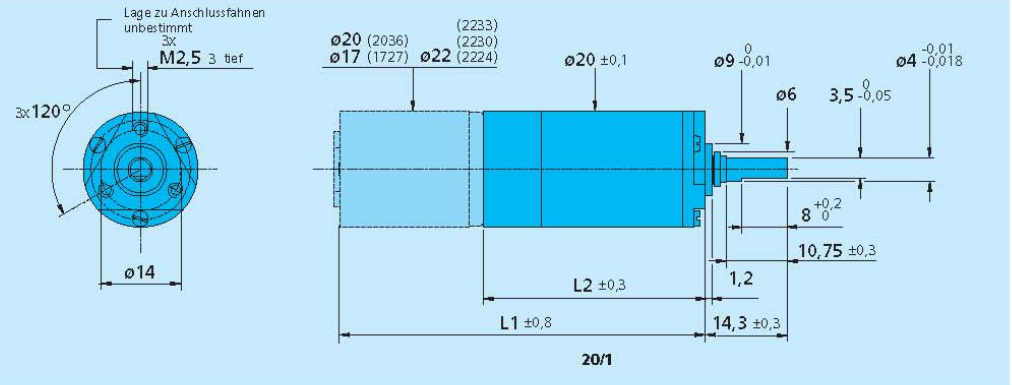
Kombinierbar mit
DC-Kleinstmotoren:
1727, 2224, 2230, 2233
Bürstenlosen DC-Servomotoren:
2036
DC-Motor-Tacho Kombinationen:
2251

Serie 20/1

	20/1
Gehäusewerkstoff	Stahl
Zahnräderwerkstoff	Metall
Max. empfohlene Eingangsdrehzahl für:	
– Dauerbetrieb	5000 rpm
Getriebespiel, unbelastet	≤ 1°
Abtriebswellenlager	Kugellager, vorgespannt
Maximal zulässige Wellenbelastung:	
– radial (8,5 mm vom Befestigungsflansch)	≤ 75 N
– axial	≤ 20 N
Maximale Aufpresskraft	≤ 35 N
Lagerspiel (gemessen am Lager):	
– radial	≤ 0,02 mm
– axial	= 0 mm
Betriebstemperaturbereich	– 30 … + 100 °C

Technische Daten

Untersetzungs-verhältnis (nominal)	Gewicht ohne Motor	Länge ohne Motor L2	Länge mit Motor					Drehmoment		Drehsinn der Welle	Wirkungs-grad
			1727 U L1	2036 U L1	2224 U L1	2230 U L1	2233 U L1	Dauer-betrieb	Kurzzeit-betrieb		
	g	mm	mm	mm	mm	mm	mm	M max. mNm	M max. mNm		%
3 ,71 :1	28	18,35	45,55	54,35	42,55	48,35	50,95	500	700	=	88
9 ,7 :1	38	23,45	50,65	59,45	47,65	53,45	56,05	500	700	=	80
14 :1	38	23,45	50,65	59,45	47,65	53,45	56,05	500	700	=	80
23 :1	38	23,45	50,65	59,45	47,65	53,45	56,05	500	700	=	80
43 :1	48	28,55	55,75	64,55	52,75	58,55	61,15	500	700	=	70
66 :1	48	28,55	55,75	64,55	52,75	58,55	61,15	500	700	=	70
86 :1	48	28,55	55,75	64,55	52,75	58,55	61,15	500	700	=	70
134 :1	58	33,65	60,85	69,65	57,85	63,65	66,25	500	700	=	60
159 :1	58	33,65	60,85	69,65	57,85	63,65	66,25	500	700	=	60
246 :1	58	33,65	60,85	69,65	57,85	63,65	66,25	500	700	=	60
415 :1	68	38,75	65,95	74,75	62,95	68,75	71,35	500	700	=	55
592 :1	68	38,75	65,95	74,75	62,95	68,75	71,35	500	700	=	55
989 :1	68	38,75	65,95	74,75	62,95	68,75	71,35	500	700	=	55
1 526 :1	68	38,75	65,95	74,75	62,95	68,75	71,35	500	700	=	55



Angaben zu Gewährleistung und Lebensdauer sowie weitere technische Erläuterungen siehe „Technische Informationen".

Änderungen vorbehalten.

B 1 – Antriebsmotor

FAULHABER

DC-Kleinstmotoren

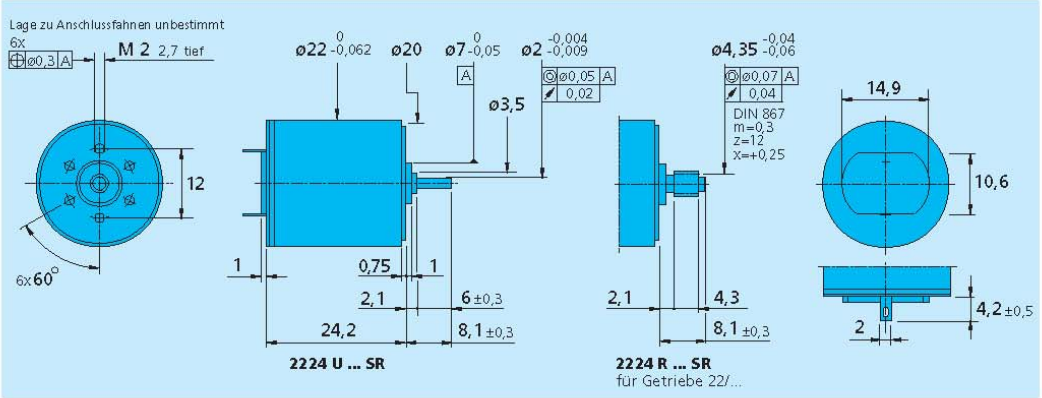
Edelmetallkommutierung

5 mNm

Kombinierbar mit
Getriebe:
20/1, 22E, 22/2, 22/5, 22/6, 23/1, 38/3
Impulsgeber:
IE2

Serie 2224 ... SR

		2224 U	003 SR	006 SR	012 SR	018 SR	024 SR	036 SR	
1	Nennspannung	U_N	3	6	12	18	24	36	Volt
2	Anschlusswiderstand	R	0,56	1,94	8,71	17,50	36,30	91,40	Ω
3	Abgabeleistung	$P_{2\,max.}$	3,92	4,55	4,05	4,54	3,88	3,46	W
4	Wirkungsgrad	$\eta_{max.}$	80	82	82	82	81	80	%
5	Leerlaufdrehzahl	n_0	8 100	8 200	7 800	8 100	7 800	7 800	rpm
6	Leerlaufstrom (bei Wellen ø 2,0 mm)	I_0	0,066	0,029	0,014	0,010	0,007	0,005	A
7	Anhaltemoment	M_H	18,5	21,2	19,8	21,4	19,0	16,9	mNm
8	Reibungsdrehmoment	M_R	0,23	0,2	0,2	0,21	0,2	0,22	mNm
9	Drehzahlkonstante	k_n	2 730	1 380	657	454	328	219	rpm/V
10	Generator-Spannungskonstante	k_E	0,366	0,725	1,520	2,200	3,040	4,560	mV/rpm
11	Drehmomentkonstante	k_M	3,49	6,92	14,50	21,00	29,10	43,50	mNm/A
12	Stromkonstante	k_I	0,286	0,144	0,069	0,048	0,034	0,023	A/mNm
13	Steigung der n-M-Kennlinie	Δn/ΔM	438	387	394	379	411	462	rpm/mNm
14	Anschlussinduktivität	L	11	45	200	450	800	1 800	µH
15	Mechanische Anlaufzeitkonstante	τ_m	11	11	11	11	11	11	ms
16	Rotorträgheitsmoment	J	2,4	2,7	2,7	2,8	2,6	2,3	gcm²
17	Winkelbeschleunigung	$\alpha_{max.}$	77	78	74	77	74	74	$\cdot 10^3$rad/s²
18	Wärmewiderstände	$R_{th\,1}$ / $R_{th\,2}$	5 / 20						K/W
19	Thermische Zeitkonstante	τ_{w1} / τ_{w2}	6,8 / 440						s
20	Betriebstemperaturbereich:								
	– Motor		– 30 ... + 85 (Sonderausführung – 55 ... + 125)						°C
	– Rotor, max. zulässig		+ 125						°C
21	Wellenlagerung		Sinterlager		Kugellager		Kugellager, vorgespannt		
22	Wellenbelastung, max. zulässig:		(standard)		(Sonderausführung)		(Sonderausführung)		
	– für Wellendurchmesser		2,0		2,0		2,0		mm
	– radial bei 3000 rpm (3 mm vom Lager)		1,5		8		8		N
	– axial bei 3000 rpm		0,2		0,8		0,8		N
	– axial im Stillstand		20		10		10		N
23	Wellenspiel:								
	– radial	≤	0,03		0,015		0,015		mm
	– axial	≤	0,2		0,2		0		mm
24	Gehäusematerial		Stahl, schwarz beschichtet						
25	Gewicht		46						g
26	Drehrichtung		rechtsdrehend auf Abtriebswelle gesehen						
Empfohlene Werte									
27	Drehzahl bis	$n_{e\,max.}$	8 000	8 000	8 000	8 000	8 000	8 000	rpm
28	Dauerdrehmoment bis	$M_{e\,max.}$	5	5	5	5	5	5	mNm
29	Thermisch zulässiger Dauerstrom	$I_{e\,max.}$	2,200	1,200	0,570	0,400	0,280	0,180	A



Angaben zu Gewährleistung und Lebensdauer sowie weitere technische Erläuterungen siehe „Technische Informationen".

Sonderausführungen für DC-Kleinstmotoren sind auf Seite 62 ersichtlich.
Änderungen vorbehalten.
www.faulhaber.com

B 2 – Impulsgeber

FAULHABER

Impulsgeber

Magnetische Impulsgeber

Besonderheiten:
16 Impulse pro Umdrehung
2 Ausgänge
Digitalausgang

Serie IE2 – 16

		IE2 – 16	
Impulse pro Umdrehung	N	16	
Ausgangssignal, rechteckig		2	Ausgänge
Betriebsspannung	V_{DD}	4 ... 18	V DC
Nennstromaufnahme, Mittelwert (V_{DD} = 12 V DC)	I_{DD}	typ. 6, max. 12	mA
Ausgangsstrom, max. zulässig	I_{OUT}	15	mA
Pulsbreite[2]	P	180 ± 45	°e
Signal-Phasenverschiebung, Kanal A zu B[2]	Φ	90 ± 45	°e
Signal-Anstiegs-/Abfallzeit, max.(C_{LOAD} = 100 pF)	tr/tf	2,5 / 0,3	µs
Frequenzbereich [1], bis	f	7	kHz
Trägheitsmoment der Impulsscheibe	J	0,11	gcm²
Betriebstemperaturbereich		– 25 ... +85	°C

[1] Drehzahl (rpm) = f (Hz) x 60/N
[2] bei 2 kHz geprüft

Bestellhinweise

Impulsgeber	Ausgänge	Impulse pro Umdrehung	kombiniert mit DC-Kleinstmotoren
IE2 – 16	2	16	**Serie 1516 ... SR**
IE2 – 16	2	16	**Serie 1524 ... SR**
IE2 – 16	2	16	**Serie 1717 ... SR**
IE2 – 16	2	16	**Serie 1724 ... SR**
IE2 – 16	2	16	**Serie 2224 ... SR**

Besonderheiten

Diese inkrementalen Impulsgeber, in Verbindung mit den FAULHABER DC-Kleinstmotoren, eignen sich für die Überwachung und Regelung von Drehzahl und Drehrichtung sowie für die Positionierung der Antriebswelle.

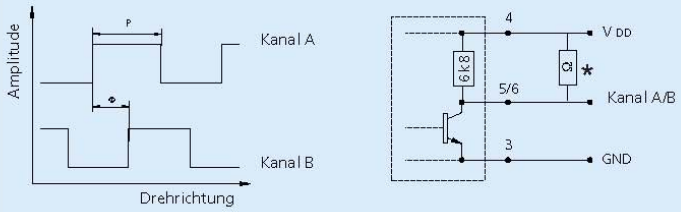
Der Impulsgeber ist im DC-Kleinstmotor der Serie SR integriert und verlängert diesen um lediglich 1,4 mm!

Durch die Verwendung von Hallsensoren und einem mehrteiligen Magnetring ergeben sich zwei um 90° phasenverschobene Kanäle.

Die Versorgungsspannung für den Impulsgeber und den DC-Kleinstmotor sowie die Ausgangssignale werden über ein Flachbandkabel mit Stecker angeschlossen.

Die Daten der DC-Kleinstmotoren und die dazu passenden Getriebe sind aus den entsprechenden Datenblättern zu entnehmen.

Ausgangssignale / Schaltdiagramm / Steckerinformation

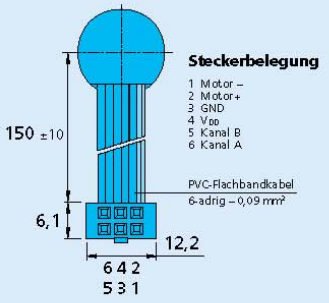


Zulässige Abweichung der Phasenverschiebung:

$$\Delta\Phi = \left| 90° - \frac{\Phi}{P} * 180° \right| \leq 45°$$

Ausgangssignale
bei Rechtslauf auf Abtrieb gesehen

Schaltdiagramm
* Ein zusätzlicher externer Pull-up Widerstand kann zur Erhöhung der Flankensteilheit zugeschaltet werden. Achtung: I_{OUT} max. 15 mA darf dadurch nicht überschritten werden!



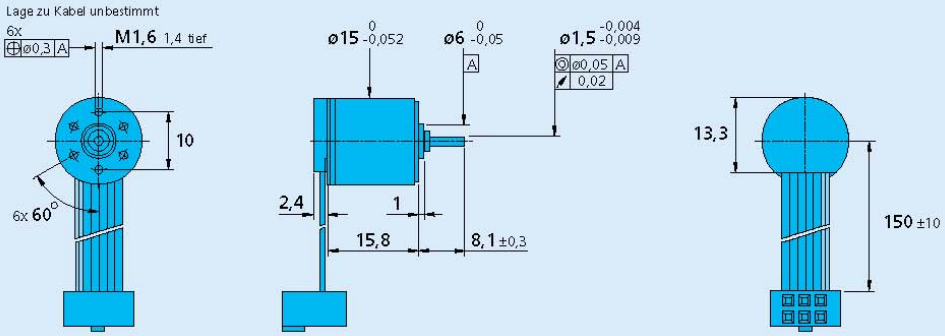
Anschlussstecker
DIN-41651
Rastermaß 2,54 mm

Angaben zu Gewährleistung und Lebensdauer sowie weitere technische Erläuterungen siehe „Technische Informationen".

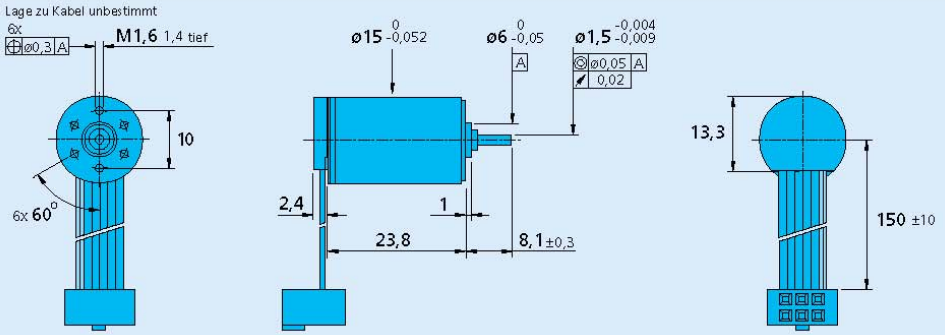
Änderungen vorbehalten.

FAULHABER

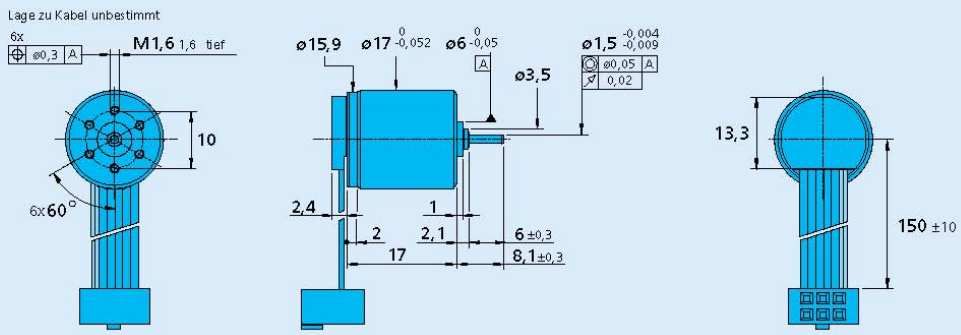
DC-Kleinstmotor 1516 T ... SR mit Impulsgeber IE2 – 16 ... 512



DC-Kleinstmotor 1524 T ... SR mit Impulsgeber IE2 – 16 ... 512



DC-Kleinstmotor 1717 T ... SR mit Impulsgeber IE2 – 16 ... 512

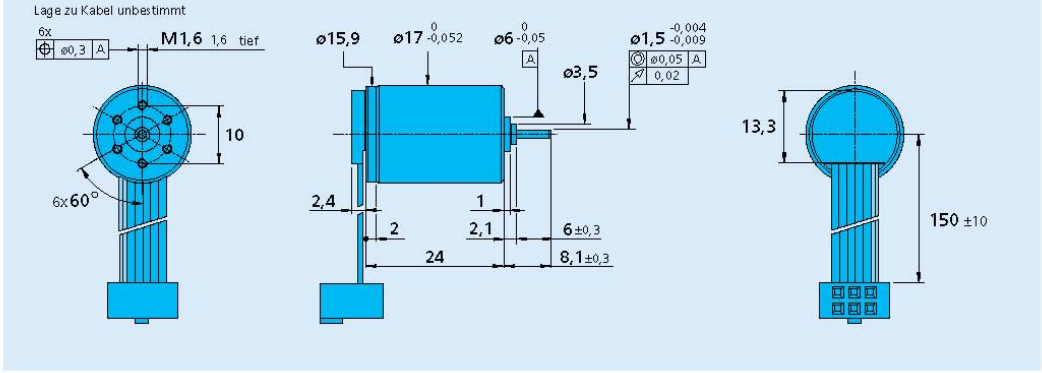


Angaben zu Gewährleistung und Lebensdauer sowie weitere technische Erläuterungen siehe „Technische Informationen".

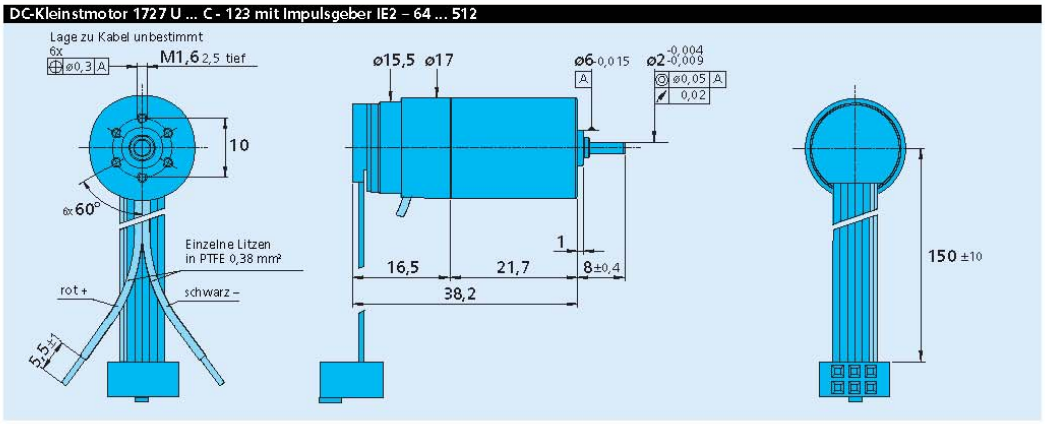
Änderungen vorbehalten.

FAULHABER

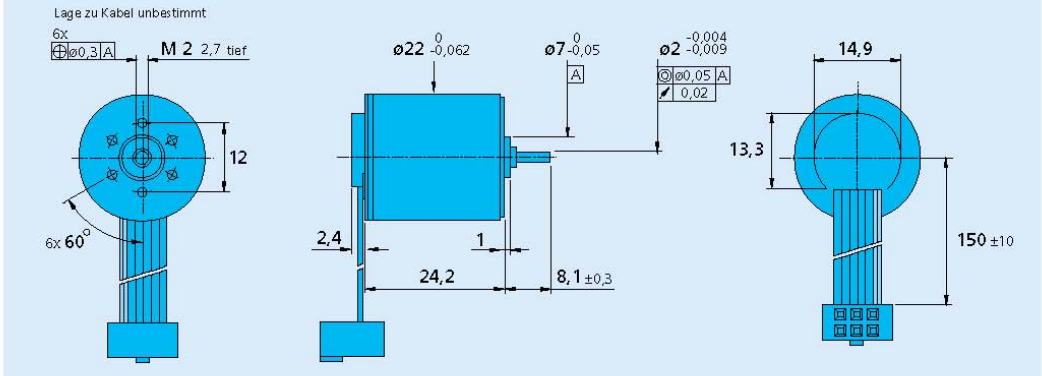
DC-Kleinstmotor 1724 T ... SR mit Impulsgeber IE2 – 16 ... 512



DC-Kleinstmotor 1727 U ... C - 123 mit Impulsgeber IE2 – 64 ... 512



DC-Kleinstmotor 2224 U ... SR mit Impulsgeber IE2 – 16 ... 512



Angaben zu Gewährleistung und Lebensdauer sowie weitere technische Erläuterungen siehe „Technische Informationen".

Änderungen vorbehalten.

B 3 – HMI

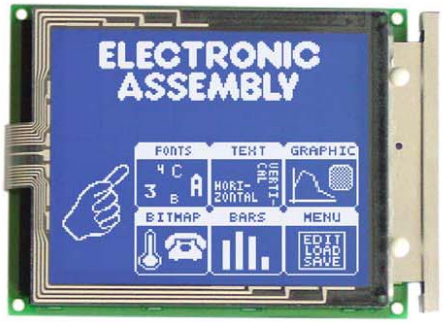
20.03.2002

EA KIT160-7

BEDIENEINHEIT MIT FONTS, GRAFIKBEFEHLEN UND MAKROS

5,1"

Touch Panel integriert



EA KIT160-7LWTP
Abmessungen 140x102mm

TECHNISCHE DATEN

* 160x128 PIXEL MIT CFL-BELEUCHTUNG BLAU NEGATIV
* AUCH MIT LANGLEBIGER LED-BELEUCHTUNG WEISS/BLAU
* INTEGRIERTES TOUCH PANEL MIT 8x7 FELDERN (ENTSPIEGELT, KRATZFEST)
* FONT ZOOM VON ca. 3mm ÜBER ca. 5mm BIS ZU ca. 50mm
* VERSORGUNGSSPANNUNG 5V/500mA (CFL)/300mA (LW) ODER 9..35V OPTIONAL
* RS-232 ODER OPTIONAL RS-422 MIT BAUDRATEN 1200..115200 BD
* **PIXELGENAUE** POSITIONIERUNG BEI ALLEN FUNKTIONEN
* PROGRAMMIERUNG ÜBER HOCHSPRACHENÄHNLICHE BEFEHLE:
* GERADE, PUNKT, BEREICH, UND/ODER/EXOR, BARGRAPH...
* BIS ZU 256 MAKROS PROGRAMMIERBAR (EEPROM ONBOARD)
* TEXT UND GRAFIK MISCHEN
* 4 CLIPBOARD FUNKTIONEN, PULL-DOWN MENÜS
* 8 DIGITALE EINGÄNGE UND 8 DIGTALE AUSGÄNGE
* BELEUCHTUNG PER SOFTWARE SCHALTBAR

OPTIONEN / ZUBEHÖR

VERSORGUNG +9..35V= STATT +5V=	**EA OPT-9/35V**
RS-422 SCHNITTSTELLE STATT RS-232	**EA OPT-RS4224**
OPTOKOPPLER ONBOARD FÜR 8 EIN- UND 8 AUSGÄNGE	**EA OPT-OPTO16**
ALUMINIUM EINBAUBLENDE, MATT-SCHWARZ ELOXIERT	**EA 0FP160-7SW**
ALUMINIUM EINBAUBLENDE, BLAU ELOXIERT	**EA 0FP160-7BL**
KABEL (1,5m) FÜR ANSCHLUSS AN 9-POL. SUB-D (RS-232 FEMALE)	**EA KV24-9B**
DISKETTE FÜR MAKROPROGRAMMIERUNG (PC-DOS/WIN)	**EA DISK240**

BESTELLBEZEICHNUNG

160x128 DOTS MIT CFL-BELEUCHTUNG, BLAU NEGATIV, TOUCH PANEL	**EA KIT160-7CTP**
WIE OBEN, JEDOCH OHNE TOUCH PANEL	**EA KIT160-7C**
160x128 DOTS MIT **WEISSER LED-BEL.**, BLAU NEGATIV, TOUCH PANEL	**EA KIT160-7LWTP**
WIE OBEN, JEDOCH OHNE TOUCH PANEL	**EA KIT160-7LW**

ELECTRONIC ASSEMBLY GMBH
LOCHHAMER SCHLAG 17 · D-82166 GRÄFELFING
TEL 089/8541991 · FAX 089/8541721 · http://www.lcd-module.de

EA KIT160-7

ELECTRONIC ASSEMBLY

ALLGEMEINES

EA KIT160 ist eine komplett aufgebaute Steuer- und Bedieneinheit mit diversen eingebauten Funktionen. Das kompakt aufgebaute Display bietet zusammen mit dem sehr guten Supertwistkontrast eine sofort einsetzbare Einheit. Die Ansteuerung erfolgt über die Standard Schnittstellen RS-232 oder RS-422. Die Bedieneinheit enthält neben kompletten Grafikroutinen zur Displayausgabe auch verschiendenste Schriften. Die Programmierung erfolgt über hochsprachenähnliche Grafikbefehle; die zeitraubende Programmierung von Zeichensätzen und Grafikroutinen entfällt hier völlig. Die simple Verwendung von Makros und die Eingabemöglichkeit über Touchpanel machen es zu einem richtigen Power Display.

DISPLAYVARIANTEN

CFL-Beleuchtung EA KIT160-7CTP: blauer Hintergrund mit weiss leuchtender Schrift. Extrem hell und kontraststark. Lebensdauer der Beleuchtung 10.000-20.000 Stunden. Ersatzbeleuchtung unter EA CFL160-7 lieferbar. Stromverbrauch typ. 450 mA.

LED-Beleuchtung EA KIT160-7LWTP: blauer Hintergrund mit weiss leuchtender Schrift. Guter Kontrast, Lebensdauer 100.000 Stunden. Stromverbrauch typ. 250 mA.

HARDWARE

Die Bedieneinheit ist für +5V Betriebsspannung ausgelegt. Optional ist eine Versorgung mit 9..35V möglich. Die Datenübertragung erfolgt seriell im RS-232 oder RS-422 Format. Das Übertragungsformat ist fest auf 8 Datenbits, 1 Stopbit, no Parity eingestellt. Die Baudrate kann über DIP-Schalter von 1.200 Baud bis zu 115.200 Baud ausgewählt werden. Handshakeleitungen RTS und CTS stehen zur Verfügung.

Datenformat:: 

TOUCH PANEL

Die Versionen EA KIT160-7CTP und -7LWTP sind mit einem integrierten Touch Panel ausgerüstet. Durch Berühren des Displays können hier Eingaben gemacht und Einstellungen per Menü getätigt werden. Die Beschriftung der "Tasten" ist flexibel und auch während der Laufzeit änderbar (verschiedene Sprachen, Icons). Das Zeichnen der einzelnen "Tasten", sowie das Beschriften oder Zusammenfassen mehrerer Felder wird von der eingebauten Software komplett übernommen.

SOFTWARE

Die Programmierung der Bedieneinheit erfolgt über Befehle wie z.B. *Zeichne ein Rechteck von (0,0) nach (64,15)*. Es ist keine zusätzliche Software oder Treiber erforderlich. Zeichenketten lassen sich **pixelgenau** plazieren. Das Mischen von Text und Grafik ist jederzeit möglich. Es können bis zu 16 verschiedene Zeichensätze verwendet werden. Jeder Zeichensatz kann wiederum 2- bis 8-fach gezoomt werden.

ZUBEHÖR

Frontpanel zur Montage

Als Zubehör ist ein Frontpanel aus eloxiertem Aluminium erhältlich. Damit läßt sich die Bedieneinheit ohne sichtbare Schrauben montieren. Das Frontpanel EA 0FP160-7 ist in den Farben schwarz (SW) und blau (BL) lieferbar.

Diskette zur Makroerstellung

Zur Makroprogrammierung ist eine Diskette EA DISK240 erforderlich*). Diese übersetzt die in eine Textdatei eingegebenen Befehle in einen für die Bedieneinheit lesbaren Code und brennt diesen dauerhaft ins EEPROM.

Kabel für PC

Für die einfache Anbindung an PC´s (Makroprogrammierung) liefern wir ein ca. 1,5m langes Kabel mit 9-pol. SUB-D Stecker (female) EA KV24-9B. Einfach an die COM 1 oder COM 2 anstecken und loslegen. Hinweis: Das Kabel ist nicht für die RS-422 Version EA OPT-RS4224 geeignet.

*) *auch im Internet unter http://www.lcd-module.de/deu/disk/disk240.zip*

EA KIT160-7

ELECTRONIC ASSEMBLY

EXTERNE TASTATUR

Am Steckanschluss J8 kann eine Tastatur (einzelne Tasten bis zur 8x7 Matrix-Tastatur) angeschlossen werden. Die angeschlossenen Tasten werden dabei per Software entprellt. Bitte beachten Sie, daß der Anschluß einer externen Tastatur nur bei den Versionen ohne integriertem Touch Panel möglich ist.

Jede Taste wird zwischen einen Ausgang und einen Eingang geschaltet. Jeder Eingang ist mit einem ca. 100kΩ Pullup abgeschlossen.

Um Doppeltastendrücke zu erkennen, müssen die Ausgänge voneinander entkoppelt werden. Dies geht am besten mit Schottky-Dioden (z.B. BAT 43).

Senden der Tastendrücke

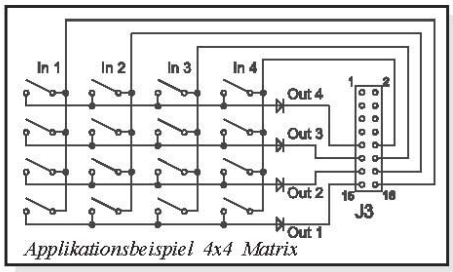
Bei jedem Druck einer Taste wird die dazugehörende Tastennummer (1..56) über die serielle Schnittstelle gesendet oder es wird ein internes Touch Makro, falls definiert, mit der Tastennummer (1..56) gestartet. Das Loslassen der Taste wird nicht gesendet. Soll auch das Loslassen gesendet werden, so kann das über die Definition des Touch Makros Nr.0 realisiert werden. Der automatische Tastaturscan läßt sich über den Befehl "ESC T A 0" deaktivieren. Falls die Handshakleleitung CTS das Senden nicht erlaubt, können Tastendrücke verloren gehen. Die Tastennummer kann folgendermaßen bestimmt werden:

Tastenummer = (Ausgang -1) * 8 + Eingang

(Ausgang: Zahl zwischen 1 und 7, Eingang: zahl zwischen 1 und 8).

Matrix - Tastaturanschluß J8

Pin	Symbol	Funktion	Pin	Symbol	Funktion
1	-	nc	2	IN 8	Eingang Spalte 8
3	OUT 7	Ausgang Zeile 7	4	IN 7	Eingang Spalte 7
5	OUT 6	Ausgang Zeile 6	6	IN 6	Eingang Spalte 6
7	OUT 5	Ausgang Zeile 5	8	IN 5	Eingang Spalte 5
9	OUT 4	Ausgang Zeile 4	10	IN 4	Eingang Spalte 4
11	OUT 3	Ausgang Zeile 3	12	IN 3	Eingang Spalte 3
13	OUT 2	Ausgang Zeile 2	14	IN 2	Eingang Spalte 2
15	OUT 1	Ausgang Zeile 1	16	IN 1	Eingang Spalte 1



Applikationsbeispiel 4x4 Matrix

TOUCH PANEL (NUR EA KIT160-7xxTP)

Die Versionen EA KIT160-7CTP und -7LWTP werden mit einem integrierten Touch Panel mit 56 Feldern geliefert. Die Bedieneinheit unterstützt dieses Touch Panel mit komfortablen Befehlen. So können z.B. mehrere Touch-Felder zu einer großen Gesamt-Taste zusammengefasst, die Taste gezeichnet und eine Beschriftung der Taste erfolgen. Ebenso kann dieser eben definierten Taste ein Return-Code (1..255) zugewiesen werden. Wird der Return-Code 0 zugewiesen, so ist die Taste deaktiviert und wird bei Betätigung nicht gemeldet.

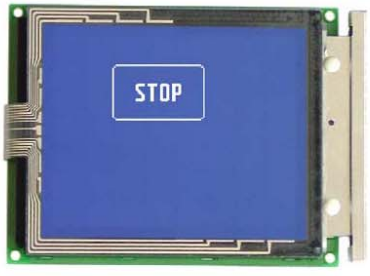
Beim Berühren der Touch-Tasten können diese automatisch invertiert werden und ein Summer signalisiert die Berührung. Der definierte Return-Code der Taste wird über die serielle Schnittstelle gesendet oder es wird ein internes Touch Makro mit der Nummer des Return-Codes gestartet.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56

Beispiel:

Definieren einer Taste von Feld 11 bis 21, mit dem Return-Code 65=´A´ und dem Text "STOP". Anmerkung: Vor der Definition einzelner Tasten sollten alle Felder durch "ESC T R" deaktiviert sein.

Beispiel	Auszugebende Codes												Bemerkung
für Compiler	#TH 11, 21, ´A´, 2, "STOP"												Die Endekennung 0 wird hier nicht angegeben !
als ASCII	ESC	T	H	.	.	A	.	S	T	O	P	.	die Punkte ´.´ stehen für nicht darzustellende ASCII-Zeichen
in Hex	$1B	$54	$48	$0B	$15	$41	$02	$53	$54	$4F	$50	$00	
in Dezimal	27	84	72	11	21	65	2	83	84	79	80	0	
	Befehlskennung	Einleitung Touch-Befehl	horizontale Beschriftung	linke oberes Touchfeld	rechtes untere Touchfeld	Return Code	Taste zeichnen mit Rahmen					Text Ende Kennung	



EA KIT160-7

ELECTRONIC ASSEMBLY

BAUDRATEN

Die Baudrate läßt sich über die linken 3 DIP Schalter einstellen. Im Auslieferungszustand sind 9.600 Baud eingestellt (DIP 3 ON). Bitte beachten Sie, daß der interne Datenpuffer lediglich 24 Byte umfaßt. Deshalb sollte unbedingt die Handshakeleitung RTS abgefragt werden (+10V Pegel: Daten können angenommen werden; -10V Pegel: Display ist Busy). Das Datenformat ist fest eingestellt auf 8 Datenbits, 1 Stopbit, keine Parität.

Baudraten

DIP Schalter 1	2	3	Datenformat 8,N,1
ON	ON	ON	1200
OFF	ON	ON	2400
ON	OFF	ON	4800
OFF	OFF	ON	9600
ON	ON	OFF	19200
OFF	ON	OFF	38400
ON	OFF	OFF	57600
OFF	OFF	OFF	115200

SCHREIBSCHUTZ FÜR MAKROPROGRAMMIERUNG

Über DIP Schalter 6 läßt sich ein versehentliches Überschreiben der Makros, Bilder und Fonts verhindern.

Schreibschutz

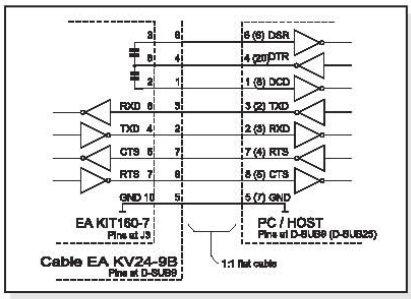
DIP 6	Schreibschutz für EEPROM
ON	Ein keine Makroprog. mögl.
OFF	Aus Makroprog. möglich

RS-232/RS-422 ANSCHLUSS

Standardmäßig wird die Bedieneinheit mit einer RS-232 Schnittstelle ausgeliefert. Die Stiftleiste J3 hat dann die Pinbelegung wie in der Tabelle links abgebildet. J3 ist im Raster 2,54mm ausgeführt. Wird die Bedieneinheit zusammen mit der Option EA OPT-RS4224 bestellt, sind spezielle RS-422 Treiber bestückt. Damit ist die Pinbelegung in der Tabelle rechts gültig.
An der Lötaugenleiste J5 stehen übrigens die gleichen seriellen Daten mit 5V Pegeln und TTL-Logik zur Verfügung. Diese Pegel sind für den direkten Anschluß an einen µC geeignet. Bei Verwendung dieser Signale müssen allerdings die Lötbrücken LB 5 + LB 6 geöffnet werden!

RS-232 Anschluß J3

Pin	Symbol	In/Out	Funktion
1	VDD	-	+ 5V Versorgung
2	DCD	-	Brücke nach DTR
3	DSR	-	Brücke nach DTR
4	TxD	Out	Transmit Data
5	CTS	In	Clear To Send
6	RxD	In	Receive Data
7	RTS	Out	Request To Send
8	DTR	-	siehe Pin 2, Pin 3
9	-	-	NC
10	GND	-	0V Masse



RS-422 Anschluß J3

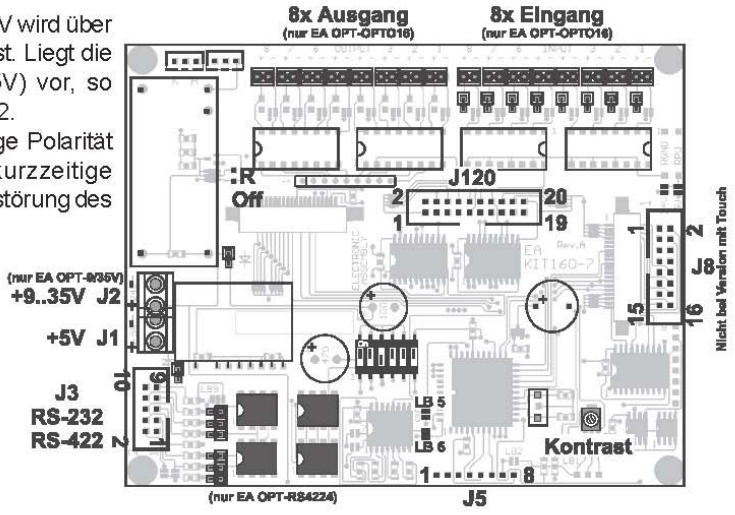
Pin	Symbol	Funktion
1	VDD	+ 5V Versorgung
2	Data In-	Receive Data
3	Data In+	Receive Data
4	Data Out-	Transmit Data
5	Data Out+	Transmit Data
6	HS In-	Handshake
7	HS In+	Handshake
8	HS Out-	Handshake
9	HS Out+	Handshake
10	GND	0V Masse

VERSORGUNGSSPANNUNG / EA OPT-9/35V

Die Versorgungsspannung von +5V wird über die Schraubklemme J1 eingespeist. Liegt die Version für 9..35V (EA OPT-9/35V) vor, so erfolgt die Stromversorgung über J2.
Achtung: Unbedingt auf die richtige Polarität achten! Eine auch noch so kurzzeitige Verpolung kann zur sofortigen Zerstörung des gesamten Displays führen.

Erweiterung J5

Pin	Symbol	In/Out	Funktion
1	VU	-	9..35V Versorgung
2	VDD	-	+ 5V Versorgung
3	GND	-	0V, Masse
4	TxD5	Out	Transmit Data
5	RxD5	In	Receive Data
6	RTS5	Out	Request To Send
7	CTS5	In	Clear To Send
8	RESET	In	H: Reset



EA KIT160-7

ELECTRONIC ASSEMBLY

EIN-UND AUSGÄNGE

Alle EA KIT160-7 werden mit 8 digitalen Ein- und 8 Ausgängen (5V CMOS Pegel, nicht potentialfrei) geliefert.

8 Ausgänge

Jede Leitung kann per Befehl "ESC Y W" individuell angesteuert werden. Pro Leitung kann ein Strom von max. 6mA geschaltet werden. Es ist somit möglich, mit einem Ausgang direkt eine LED (low current) zu schalten. Größere Ströme können mittels externen Transistors verstärkt werden.

8 Eingänge

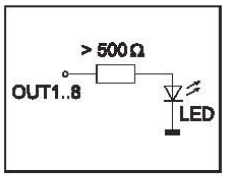
Die Eingänge können direkt über die serielle Schnittstelle abgefragt und ausgewertet werden ("ESC Y R"). Zusätzlich ist es möglich, bei Änderungen an den Eingängen ein Portmakro automatisch aufzurufen, durch die binäre Kombination von 8 Eingängen sind bis zu 256 Portmakros ansprechbar. Die automatische Portabfrage läßt sich mit dem Befehl "ESC Y A 0" deaktivieren.

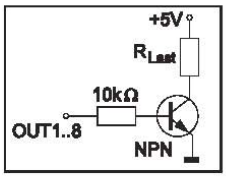
Anmerkung:

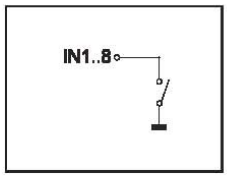
Die Logik ist für langsame Vorgänge ausgelegt; d.h. mehr als 3 Änderungen pro Sekunde können nicht mehr sinnvoll ausgeführt werden. Falls ein Eingang offen ist, so ist dieser High (interner 100 kOhm PullUp).

Ein- und Ausgänge J120

Pi	Symb	Funktion		Pin	Symb	Funktion
1	VDD	+5V Vers.		2	GND	0V, Masse
3	OUT 1	Ausgang 1		4	IN 1	Eingang 1
5	OUT 2	Ausgang 2		6	IN 2	Eingang 2
7	OUT 3	Ausgang 3		8	IN 3	Eingang 3
9	OUT 4	Ausgang 4		10	IN 4	Eingang 4
11	OUT 5	Ausgang 5		12	IN 5	Eingang 5
13	OUT 6	Ausgang 6		14	IN 6	Eingang 6
15	OUT 7	Ausgang 7		16	IN 7	Eingang 7
17	OUT 8	Ausgang 8		18	IN 8	Eingang 8
19	GND	0V, Masse		20	VDD	+5V Vers.

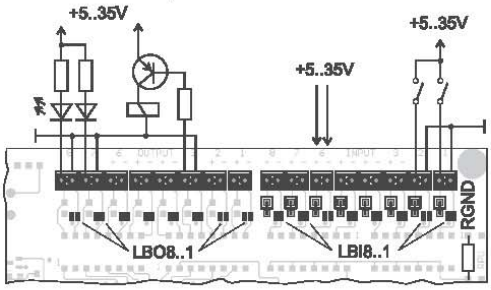






EIN- UND AUSGÄNGE ÜBER OPTOKOPPLER (EA OPT-OPTO16)

Die Ein- und Ausgänge können optional mit Optokopplern ausgestattet werden (EA OPT-OPTO16). Die Ein- und Ausgänge sind dann sowohl von der restlichen Elektronik, als auch untereinander isoliert. Der Anschluß erfolgt über 16 einzelne Schraublemmen.



An allen 8 Eingängen können direkt Spannungen von 5..35V angelegt werden. Pegel über 4V werden als H-Pegel erkannt, Pegel unter 2V gelten als L-Pegel. Spannungen zwischen 2 und 4V sind undefiniert.

Als Ausgang ist jeweils der Kollektor (+) und Emitter (-) eines NPN-Transistors an den Schraubklemmen herausgeführt. Jeder Ausgang kann 10mA schalten.

Hinweis: Der Minuspol jeder Schraubklemme kann durch Schließen der Lötbrücken LBI1..8 bzw. LBO1..8 zusammengeschalten werden. Zusätzlich können diese Lötbrücken auf die Systemmasse GND gelegt werden (0Ω Brücke RGND einlöten).

Anmerkung: Die Optokoppler invertieren die Eingangslogik (alle Eingänge offen: Portmakro 255). Hier empfielt es sich (z.B. im Power-On-Makro) mit dem Befehl "ESC Y I 1" die Eingänge invertiert auszuwerten (d.h. alle Eingänge offen: Portmakro 0).

GRUNDEINSTELLUNGEN

Nach dem Einschalten bzw. nach einem manuell ausgelösten Reset werden die nebenstehenden Register auf einen bestimmten Wert voreingestellt.

Beachten Sie bitte, daß alle Einstellungen durch Erstellen eines Power-On-Makros (Normal-Makro Nr.0) überschrieben werden können.

Grundeinstellungen

Register	Befehl	nach Power-On / Reset
Text-Modus	ESC L	setzen,schwarz
Terminal Font	ESC FT	Font 3, kein Zoom
Cursor	ESC QC	ein
Blinkzeit	ESC QZ	0,6 sek.
Selbst definierte Zch.	ESC E	undefiniert
Grafik-Modus	ESC V	setzen
Grafik Font	ESC F	Font 3, kein Zoom
Last xy	ESC W	(0;0)
Bargraph 1..16	ESC B	undefiniert
Clipboard	ESC C	leer
Selekt/Deselekt	ESC K	selektiert
Ausgänge OUT1..8	ESC Y	L-Pegel

EA KIT160-7

ELECTRONIC ASSEMBLY

MAKRO PROGRAMMIERUNG

Einzelne oder mehrere Befehlsfolgen können als sog. Makros zusammengefasst und im EEPROM fest abgespeichert werden. Diese können dann mit den Befehlen *Makro ausführen* gestartet werden. Es gibt 3 verschiedene Makrotypen:

Touch Makro (1..255)

Start bei Berührung eines Touchfeldes (nur bei Versionen mit Touch Panel TP) oder bei Betätigung einer ext. angeschlossenen Taste/Matrixtastatur oder per Befehl 'ESC MT nr'. Das Touch Makro Nr.0 hat eine Sonderstellung: Beim Loslassen einer x-beliebigen Taste wird das Touch Makro Nr.0 gestartet.

Port Makro (0..255)

Start bei Anlegen/Ändernung einer Spannung an den Eingängen IN 1..8 oder per Befehl 'ESC MP nr'.

Normal Makro (1..255)

Start per Befehl 'ESC MN nr' über die serielle Schnittstelle oder von einem anderen Makro aus. Es können auch mehrere hintereinander liegende Makros automatisch zyklisch aufgerufen werden (Movie, sich drehende Sanduhr, mehrseitiger Hilfetext).

Power-On-Makro

Das Normal Makro Nr.0 hat eine Sonderstellung: es wird automatisch nach dem Einschalten ausgeführt. Hier kann man zB. den Cursor abschalten und einen Startbildschirm definieren.

Achtung: Wird im Power-On-Makro eine Endlosschleife programmiert, ist das Display nicht mehr ansprechbar. In diesen Fall hilft nur noch (ab REV. B): DIP Schalter 5 auf ON, Power off, Power on und dann DIP 5 wieder auf off. Jetzt müssen die Fonts und Makros wieder neu eingespielt werden.

256 BILDER FEST ABGELEGT

Um Übertragungszeiten der seriellen Schnittstelle zu verkürzen, oder auch um Speicherplatz im Prozessorsystem zu sparen, können bis zu 256 Bilder im internen EEPROM abgelegt werden. Der Aufruf erfolgt über den Befehl "ESC U E" über die serielle Schnittstelle oder aus einem Touch-/Port-/Normal-Makro heraus. Verwendet werden können alle Bilder im Windows BMP Format. Die Erstellung und Bearbeitung erfolgt über Standardsoftware wie z.B. Windows Paint oder Photoshop.

ERSTELLEN INDIVIDUELLER MAKROS

Um nun Ihre speziellen Makros erstellen zu können, benötigen Sie folgende Hilfsmittel:

- die Diskette EA DISK240*); sie enthält einen Compiler, Beispiele und Fonts
- einen PC mit serieller Schnittstelle COM1 oder COM2, mit ca. 500kB Platz auf der Festplatte
- einen Texteditor wie z.B. WordPad, Norton Editor o.ä.

Um eine Befehlsfolge als Makro zu definieren, werden alle Befehle auf dem PC in eine Datei z.B. DEMO.KMC geschrieben. Hier bestimmen Sie welche Zeichensätze eingebunden werden und in welchen Makros welche Befehlsfolgen stehen sollen. Sind die Makros definiert, startet man das Programm C:>KITCOMP DEMO.KMC. Dieses erzeugt eine EEPROM-Datei DEMO.EEP, welche dann automatisch mit der eingetragenen Baudrate in das Display-EEPROM gebrannt wird. Dieser Vorgang dauert nur wenige Sekunden und sofort danach können die selbstdefinierten Makros genutzt werden. Eine ausführliche Beschreibung zur Programmierung der Makros finden Sie zusammen mit Beispielen auf der Diskette EA DISK240*) unter dem Namen DOKU.DOC (für WORD) bzw. DOKU.TXT (DOS).

```
;Makro Demo
KIT160-7                    ; KIT festlegen
COM2: 115200                ; KIT ist an COM2 angeschlossen,
                            ; Übertragung mit 115.200 Baud
;-------------------------------------------------------------
;Konstanten definieren
AUS =  0
EIN =  1
FONT4X6 = 1
FONT5X6 = 2
FONT6X8 = 3
FONT8X8 = 4
FONT8X16= 5
;-------------------------------------------------------------
;Fonts einbinden
Font: FONT4X6, 32, 95 INTERN4X6
Font: FONT5X6, 32,158 INTERN5X6
Font: FONT6X8, 32,158 INTERN6X8
Font: FONT8X8, 32,158 INTERN8X8
Font: FONT8X16, 32,158 INTERN8X16
;-------------------------------------------------------------
Makro: 0                    ; Power-On/Reset Makro
      #QC EIN               ; Cursor sichtbar
      #FT FONT8X16          ; Terminalfont einstellen
      #UL 0,20,<EA2.BMP>    ; ELECTRONIC ASSEMBLY Logo
```

*) *auch im Internet unter http://www.lcd-module.de/disk/disk240.zip*

EA KIT160-7

ELECTRONIC ASSEMBLY

INTEGRIERTE FONTS

In jeder Grafikeinheit sind standardmäßig 5 Zeichensätze integriert. Jeder Zeichensatz kann in 1- bis 8-facher Höhe verwendet werden. Unabhängig davon läßt sich auch die Breite verdoppeln bis verachtfachen.

Upper \ + Lower	$0 (0)	$1 (1)	$2 (2)	$3 (3)	$4 (4)	$5 (5)	$6 (6)	$7 (7)	$8 (8)	$9 (9)	$A (10)	$B (11)	$C (12)	$D (13)	$E (14)	$F (15)
$20 (dez: 32)		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_

Font 1: 4x6

Upper \ + Lower	$0 (0)	$1 (1)	$2 (2)	$3 (3)	$4 (4)	$5 (5)	$6 (6)	$7 (7)	$8 (8)	$9 (9)	$A (10)	$B (11)	$C (12)	$D (13)	$E (14)	$F (15)
$20 (dez: 32)		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	J	k	l	m	n	o
$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	(	¦	)	~	⌂
$80 (dez: 128)	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	ß	

Font 3: 6x8

Nr.	Zeichen höhe	Zeilen x Zeichen	Größe in Pixel	ASCII-Bereich	Frei def. ASCII-Codes	Bemerkung
1	2,2 mm	21 x 60	4 x 6	32 - 95	1..21	Microschrift
2	2,2 mm	21 x 48	5 x 6	32 - 158	1..21	Minischrift
3	3,1 mm	16 x 40	6 x 8	32 - 158	1..16	Normalschrift
4	3,1 mm	16 x 30	8 x 8	32 - 158	1..16	Fettschrift
5	6,3 mm	8 x 30	8 x 16	32 - 158	1..8	Großschrift

Zusätzlich können, je nach Font, bis zu 21 eigene Zeichen definiert werden die solange erhalten bleiben, bis die Versorgungsspannung abgeschaltet wird. (Siehe Befehl ESC E).
Jedes Zeichen kann **pixelgenau** plaziert werden. Text und Grafik kann beliebig gemischt dargestellt werden. Auch mehrere verschiedene Schriftgrößen lassen sich gemeinsam darstellen.

Jeder Text läßt sich linksbündig, rechtsbündig und zentriert ausgeben. Auch eine 90° Drehung (vertikaler Einbau des Displays) ist möglich.
Die Makroprogrammierung erlaubt die Einbindung von weiteren 11 Fonts, sowie die komplette Umgestaltung der einzelnen Zeichen. Durch einen Fonteditor auf der Diskette EA DISKFONT6963 können alle nur erdenklichen Schriften mit bis zu 16x16 Pixeln Größe erstellt und einprogrammiert werden.

Upper \ + Lower	$0 (0)	$1 (1)	$2 (2)	$3 (3)	$4 (4)	$5 (5)	$6 (6)	$7 (7)	$8 (8)	$9 (9)	$A (10)	$B (11)	$C (12)	$D (13)	$E (14)	$F (15)
$20 (dez: 32)		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	⌂
$80 (dez: 128)	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	ß	

Font 5: 8x16

TIP: SCHRIFTEFFEKTE

Mit dem Befehl ESC L TEXT-Modus (Verknüpfung, Muster) können bei grossen Schriften interessante Effekte durch Überlagerung (mehrmaliges versetztes Schreiben eines Wortes) erzielt werden.

*Orginalschrift 8x16 mit ZOOM 3 an **Position 0,0** mit Muster Schwarz*

*Durch Überlagerung **(EXOR)** an **Pos.1,1** enstandene "Outline Schrift"*

*Nochmalige Überlagerung **(EXOR)** der "Outline Schrift" an **Pos.2,2.** führt zu einer "Outline Schrift mit Füllung"*

*Überlagerung (**ODER**) mit Muster 50% Grau der "Outline Schrift" an **Pos.0,0.** führt zu einer "Schrift mit Musterfüllung"*

EA KIT160-7

ELECTRONIC ASSEMBLY

ALLE BEFEHLE AUF EINEN BLICK

Befehlstabelle für EA KIT160-7

Befehl	Codes									Anmerkung
Befehle für den Terminal Betrieb										
Formfeed FF (dez:12)	^L									Bildschirm wird gelöscht und der Cursor nach Pos. (1,1) gesetzt
Carriage Return CR(13)	^M									Cursor ganz nach links zum Zeilenanfang
Linefeed LF (dez:10)	^J									Cursor 1 Zeile tiefer, falls Cursor in letzter Zeile dann auf 1. Zeile setzen
Cursor On / Off	ESC	Q	C	n1						n1=0: Cursor ist unsichtbar; n1=1: Cursor blinkt (invers 6/10s);
Cursor positionieren	ESC	O	n1	n2						n1=Spalte; n2=Zeile; Ursprung links oben ist (1,1)
Terminal Font einstellen	ESC	F	T	n1						n1=1: Font Nr. n1 (1..16) für Terminal Betrieb einstellen
Befehle zur Textausgabe										
Text-Modus	ESC	L	n1	mst						Modus n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace; mst: Muster Nr. 0..7 verwenden;
Font einstellen	ESC	F	n1	n2	n3					Font mit der Nummer n1 (1..16) einstellen; n2=X- n3=Y-Zommfaktor (1x..8x);
Zeichenkette horizontal ausgeben	ESC	Z	L Z R	x1	y1	Text ...	NUL			Eine Zeichenkette (...) an x1,y1 ausgeben. ´NUL´ ($00)=Zeichenkettenende; Mehrere Zeilen werde durch das Zeichen '|' ($7C, dez: 124) getrennt; 'L':= Linkbündig an x1; 'Z':= Zentriert an x1; 'R':= Rechtsbündig an x1; y1 ist immer die Oberkannte der Zeichenkette
Zeichenkette 90° gedreht (vertikal) ausgeben	ESC	Z	O M U	x1	y1	Text ...	NUL			Eine Zeichenkette (...) um 90° gedreht an x1,y1 ausgeben; ´NUL´ ($00)=Ende; Mehrere Zeilen werde durch das Zeichen '|' ($7C, dez: 124) getrennt; 'O':= Oben-Bündig an y1; 'M':= Mittig an y1; 'U':= Unten-Bündig an y1; x1 ist immer die Rechte Kannte der Zeichenkette
Zeichen definieren	ESC	E	n1	daten ...						n1=Zeichen Nr.; daten=Anzahl Bytes je nach akt. Font
Befehle zum Zeichnen										
Grafik-Modus	ESC	V	n1							Zeichenmodus einstellen für die Befehle: 'Punkt setzen', 'Gerade zeichnen', 'Rechteck', 'Rundeck' und 'Bereich mit Füllmuster' n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace;
Punkt setzen	ESC	P	x1	y1						Ein Pixel an die Koordinaten x1, y1 setzen
Gerade zeichnen	ESC	G	x1	y1	x2	y2				Eine Gerade von x1,y1 nach x2,y2 zeichnen
Gerade weiter zeichnen	ESC	W	x1	y1						Eine Gerade vom letzten Endpunkt bis x1, y1 zeichnen
Rechteck Befehle										
Rechteck zeichnen	ESC	R	R	x1	y1	x2	y2			Ein Rechteck (Rahmen) von x1,y1 nach x2,y2 zeichnen
Rundeck zeichnen			N	x1	y1	x2	y2			Ein Rechteck mit runden Ecken von x1,y1 nach x2,y2 zeichnen
Bereich löschen			L	x1	y1	x2	y2			Einen Bereich von x1,y1 nach x2,y2 löschen (alle Pixel aus)
Bereich invertieren			I	x1	y1	x2	y2			Einen Bereich von x1,y1 nach x2,y2 invertieren (alle Pixel umkehren)
Bereich füllen			S	x1	y1	x2	y2			Einen Bereich von x1,y1 nach x2,y2 füllen (alle Pixel ein)
Bereich m. Füllmuster			M	x1	y1	x2	y2	mst		Einen Bereich von x1,y1 nach x2,y2 mit Muster mst (0..7) zeichnen
Box zeichnen			O	x1	y1	x2	y2	mst		Ein Rechteck mit Füllmuster mst (0..7) zeichnen; (immer Replace)
Rundbox zeichnen			J	x1	y1	x2	y2	mst		Ein Rundeck mit Füllmuster mst (0..7) zeichnen; (immer Replace)
Bitmap Bilder Befehle										
Bild aus EEPROM	ESC	U	E	x1	y1	nr				internes Bild mit der nr (0..255) aus dem EEPROM nach x1,y1 laden
Bild laden			L	x1	y1	daten ...				Ein Bild nach x1,y1 laden; daten des Bildes siehe Bildaufbau
Hardcopy senden		H	x1	y1	x2	y2				Es wird ein Bild angefordert. Zuerst werden die Breite und Höhe in Pixel und dann die eigentlichen Bilddaten über RS232 gesendet.
Display-Befehle (Wirkung auf das gesamte Display)										
Display löschen	ESC	D	L							Displayinhalt löschen (alle Pixel aus)
Display invertieren			I							Displayinhalt invertieren (alle Pixel umkehren)
Display füllen			S							Displayinhalt füllen (alle Pixel ein)
Display ausschalten			A							Displayinhalt wird unsichtbar bleibt aber erhalten, Befehle weiterhin möglich
Display einschalten			E							Displayinhalt wird wieder sichtbar
Display Clipboard			C							Inhalt des Clipboards wird dargestellt. Displayausgaben sind nicht mehr sichtbar
Disp. Normaldarstellung			N							Aktuelles Bild wird dargestellt (Normalbetrieb). Alle Ausgaben wieder sichtbar
Display Reset			R							Der Displaykontroller wird per Befehl rückgesetzt und neu initialisiert
Makro Befehle										
Makro ausführen	ESC	M	N	n1						Das (Normal-)Makro mit der Nummer n1 aufrufen (max. 7 Ebenen)
Touch Makro ausführen			T	n1						Das Touch-Makro mit der Nummer n1 aufrufen (max. 7 Ebenen)
Port Makro ausführen			P	n1						Das Port-Makro mit der Nummer n1 aufrufen (max. 7 Ebenen)
autom. Makro zyklisch			A	n1	n2	n3				Makros n1..n2 automatisch zyklisch abarbeiten; n3=Pause in 1/10s
autom. Makro pingpong			J	n1	n2	n3				Makros autom. von n1..n2..n1 (PingPong) abarbeiten; n3=Pause in 1/10s

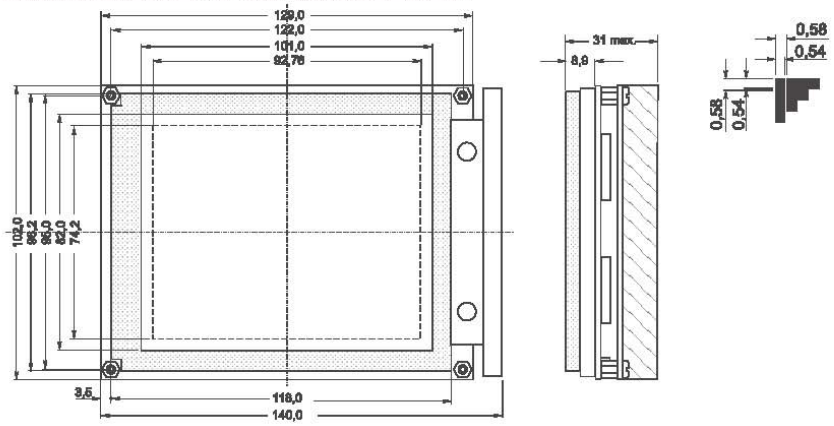
EA KIT160-7

ELECTRONIC ASSEMBLY

Bargraph Befehle												
Bargraph definieren	ESC	B	R L O U	nr	x1	y1	x2	y2	aw	ew	mst	Bargraph nach L(inks), R(echts), O(ben), U(nten) mit der ´nr´ (1..16) definieren. x1,y1,x2,y2 sind das umschließende Rechteck des Bargraphs. aw,ew sind die Werte für 0% und 100%. mst=Muster (0..7)
Bargraph zeichnen			nr	wer								Den Bargraph mit der Nummer nr (1..16) auf den neuen Benutzer-´wert´ setzen
Clipboard Befehle (Zwischenspeicher für Bildbereiche)												
Displayinhalt sichern	ESC	C	B									Der gesamte Displayinhalt wird als Bildbereich ins Clipboard kopiert
Bereich sichern			S	x1	y1	x2	y2					Der Bildbereich von x1, y1 bis nach x2, y2 wird ins Clipboard kopiert
Display restaurieren			R									Der Bildbereich im Clipboard wird wieder ins Display zurückkopiert
Bereich kopieren			K	x1	y1							Der Bildbereich im Clipboard wird ins Display nach x1, y1 kopiert
Tastatur / Touch-Panel Befehle												
Touch-Taste mit horizontaler Beschriftung definieren	ESC	T	H	f1	f2	Ret. Code	Form	Text ...	NUL			Die Touch-Felder f1 bis f2 (gegenüberliegenden Eckfelder), werden zu einer Touch-Taste mit dem Rückgabewert ´Ret. Code´ (=1..255) zusammengefasst (Ret.Code=0 Touch-Taste nicht aktiv). ´Form´:Touch-Taste (=0 nichts; =1 löschen; =2 mit Rahmen) zeichnen ´Text´: es folgt eine Zeichenkette die zentriert mit dem akt. Font in der Touch-Taste plaziert wird, mehrzeilige Texte werden mit dem Zeichen '|' ($7C, dez: 124) getrennt; Zeichen NUL ($00) = Zeichenkettenende
Touch-Taste mit vertikaler (90° gedreht) Beschriftung definieren			V									
Touch-Tasten (P)Reset			P									Alle Touch-Tasten werden aufsteigend aktiviert (Felder mit Code 1..60)
			R									Alle Touch-Tasten werden deaktiviert (alle Felder mit Code 0)
Touch-Tasten Reaktion			I	n1								n1=0: kein invertieren beim Berühren der Touch-Taste n1=1: Touch-Taste wird beim Berühren automatisch invertiert
			S	n1								n1=0: kein Summer beim Berühren einer (Touch-)Taste n1=1: Summer piepst kurz beim Berühren einer (Touch-)Taste
Touch-Taste Invertieren			M	n1								Die Touch-Taste mit dem zugeordnetem Return-Code n1 wird manuell Invertiert
Taste manuell abfragen			W									Die momentan gedrückte (Touch-)Taste wird auf der RS-232/RS-422 gesendet
Tasten-Abfrage Ein/Aus			A	n1								Tastaturabfrage wird n1=0:deaktiviert; n1=1:aktiviert, Tastendrücke werden automatisch gesendet; n1=2:aktiviert, Tastendrücke werden nicht gesendet (mit ESC T W abfragen)
Menü / Popup Befehle												
Menü mit horizoltalen Einträgen definieren	ESC	N	H	x1	y1	nr	Text ...	NUL				Ein Menü wird ab der Ecke x1,y1 (Horizotales Menü = linke obere Ecke; Vertikales menü = rechte obere Ecke) mit dem akt. Font gezeichnet. nr:= aktuell invertierter Eintrag (z.B: 1 = 1. Eintrag) Text:= Zeichenkette mit den Menüeinträgen. Die einzelnen Einträge sind durch Zeichen '|' ($7C,dez:124) getrennt z.B. "Eintrag1|Eintrag2|Eintrag3" Der Hintergrund des Menüs wird automatisch ins Clipboard gesichert. Ist bereits ein Menü definiert, wird dieses automatisch abgebrochen+entfernt.
Menü mit vertikalen (90° gedrehten) Einträgen definieren			V									
Menübox invertieren			I									Die gesamte Menübox wird invertiert. Sinnvoll für negative Darstellung
nächster Eintrag			N									Der nächste Eintrag wird invertiert oder bleibt am Ende stehen
vorheriger Eintrag			P									Der vorherige Eintrag wird invertiert oder bleibt am Anfang stehen
Menüende / Senden			S									Das Menü wird vom Display entfernt und durch den Clipboardinhalt ersetzt der aktuelle Eintrag wird als Nummer (1..n) gesendet (0=kein Menü dargestellt)
Menüende / Makro			M	nr								Das Menü wird vom Display entfernt und durch den Clipboardinhalt ersetzt Für Eintrag 1 wird Makro 'nr' aufgerufen; für Eintrag 2 Makro nr+1 usw.
Menüende / Abbrechen			A									Das Menü wird vom Display entfernt und durch den Clipboardinhalt ersetzt
Kontroll- / Definitions-Befehle												
Automatisch blinkender Bereich (Cursor-Funktion)	ESC	Q	D	x1	y1	x2	y2					Definiert einen Blinkbereich x1,y1 bis x2,y2; Blinkfunktion aktivieren
			Z	n1								Einstellen der Blinkzeit n1= 1..15 in 1/10s; 0=Blinkfunktion deaktivieren
			M	I								Invers-Modus (Blinkbereich wird invertiert); Blinkfunktion aktivieren
				mst								Clipboard-Modus mst=Muster (0..7) des Blockcursors; Blinken aktivieren
			C	n1								Automatisch blinkender Bereich als Cursor für den Terminal Betrieb n1=0: Blinkfunktion deaktivieren; n1=1: Blinkfunktion aktivieren (Invers, 6/10s)
Selekt / Deselekt	ESC	K	S	adr								Kit mit Adresse n1 aktivieren (n1=255: alle)
			D	adr								Kit mit Adresse n1 deaktivieren (n1=255: alle)
			A	adr								Neue Adresse adr zuweisen (z.B. im Power-On Makro)
Warten (Pause)	ESC	X	n1									n1 Zehntel-Sekunden abwarten bevor der nächste Befehl ausgeführt wird.
Summer Ein / Aus	ESC	J	n1									n1=0:Summer Aus; n1=1:Summer Ein; n1=2..255:für n1 1/10s lang Ein
Bytes senden	ESC	S	anz	daten ...								Es werden anz (1..255; 0=256) Bytes auf der RS-232/RS-422 gesendet daten ... = anz Bytes (z.B Ansteuerung eines externen seriellen Druckers)
I2C-Bus lesen	ESC	I	R	adr	anz							Von dem Baustein am I2C-Bus mit der Device Adresse adr werden anz (1..255; 0=256) Bytes angefordert und über die RS-232/RS-422 gesendet.
I2C-Bus schreiben	ESC	I	W	adr	anz	daten ...						Auf dem I2C-Bus für den Baustein mit der Device Adresse adr werden anz (1..255; 0=256) Bytes gesendet. daten ... = anz Bytes
Port-Befehle												
Output-Port schreiben	ESC	Y	W	n1	n2							n1=0: Alle 8 Ausgabe-Ports entsprechend n2 (=8-Bit Binärwert) einstellen n1=1..8: Ausgabe-Port n1 rücksetzen (n2=0); setzen (n2=1); invertieren (n2=2)
Eingabe-Port lesen			R	n1								n1=0: Alle 8 Eingabe-Ports als 8-Bit Binärwert einlesen n1=1..8: Eingabe-Port <n1> einlesen (1=H-Pegel=5V, 0=L-Pegel=0V)
Port Scan Ein/Aus			A	n1								Der automatiche Scan des Eingabe-Port wird n1=0: deaktiviert; n1=1: aktiviert
Eingabe-Port invers			I	n1								Der Eingabe-Port wird n1=0: normal; n1=1: invertiert ausgewertet
Beleuchtung Ein/Aus			L	n1								CFL/LED-Beleuchtung n1=0: AUS; n1=1: EIN; n1=2: INVERTIEREN; n1=3..255: Beleuchtung für n1 Zehntel Sek.. lang einschalten

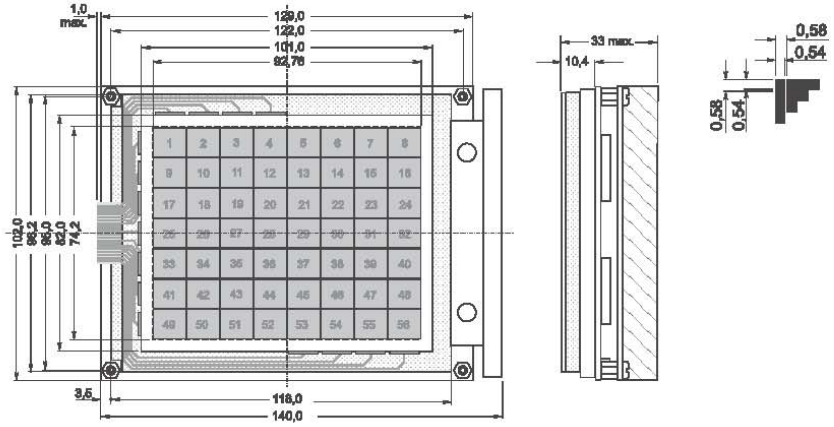
EA KIT160-7

ABMESSUNGEN OHNE TOUCH PANEL



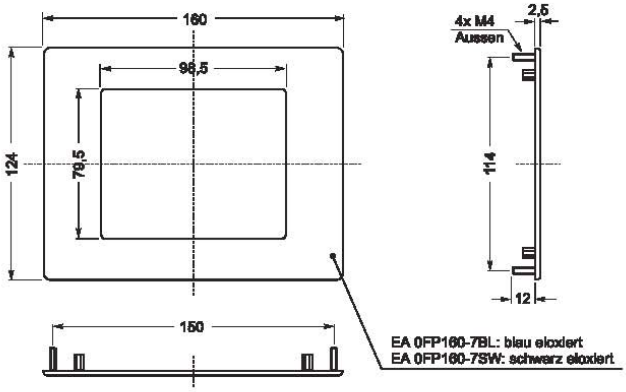
alle Maße in mm

ABMESSUNGEN MIT TOUCH PANEL

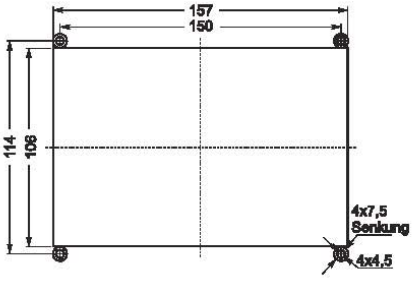


alle Maße in mm

FRONTPANEL EA 0FP160-7



PANEL CUT OUT



alle Maße in mm

LOCHHAMER SCHLAG 17 · D-82166 GRÄFELFING
TEL 089/8541991 · FAX 089/8541721 · http://www.lcd-module.de

ELECTRONIC ASSEMBLY GMBH

B 4 – Netzteil

EOS Z-Series Power-Z

EXTERNAL POWER ADAPTERS

Single Output 60W, 65W, and 70W universal AC-DC power supplies

Big power, little package.

The EOS "Power-Z" AC-DC universal power adapter gives you up to 70 watts of continuous power in a package so small it could only come from EOS. Delivering nearly 8 watts per cubic inch, the "Power-Z" has over twice the density of any other high wattage power adapter on the market. That puts the "Power-Z" in a class by itself. And it means competitive advantage and more value for you and your customers.

Measuring just 5.24" x 2.28" x 1.15", the "Power-Z" is slightly larger than a deck of cards and weighs just 9 ounces—making it far lighter and much smaller than any competitive adapter available. In fact, the Power-Z has set the industry standard for a 70 watt external power supply.

With full worldwide safety standards and a CE mark, the "Power-Z" series is the power adapter of choice for demanding applications like mobile and laptop computers, point of sale terminals, thermal printers, data networking, telecommunications, medical, bar code readers, Internet servers and a host of industrial products. Millions of EOS "Power-Z" AC-DC adapters are in use worldwide, powering the industry's leading electronics products.

EOS

Highest Density
Smallest Size
Most Efficient

FEATURES:

- Very High Efficiency: above 90%
- High Power Density: nearly 8 watts/inch^3
- Ultra Miniature Size: 5.24" x 2.28" x 1.15"
- Lightweight: only 9 ounces
- Power Factor Correction in HD Models
- Universal input: 90-264 VAC
- FCC & CISPR Level "B" EMI filter
- No Minimum Load Required
- Fully Regulated Output
- Over Current Protection (OCP)
- 100% Burn-In
- High MTBF
- Meets Worldwide Safety Standards
- CE marked

APPLICATIONS:

- Mobile & Laptop Computers
- Point of Sale terminals and computers
- Thermal printers
- Bar Code Readers
- Data Networking products
- Telecommunications products
- Internet servers
- Industrial equipment
- Medical equipment

Contact us at: 1.800.293.9998 or www.eoscorp.com

EOS Z-Series Power-Z

SPECIFICATIONS

INPUT

AC Input	90 to 135 VAC and 170 to 264 VAC
Efficiency	86-90% typical, measured at Full load on main and 230 VAC input voltage
Input Frequency	47–63 Hz

OUTPUT

Output Power	60W, 65W, 70W
DC Output	12V, 15V, 19V, 24V
Hold-Up Time	10msec at full rated load, 130VAC/230VAC
Line Regulation	1%, over operating range
Load Regulation	3%, with minimum to full load, with standard cord
Overload Protection	Yes
Ripple and Noise	500mV peak-peak maximum
Power-Up Initialization Period	4 seconds Max at 120VAC

ENVIRONMENTAL

Operating Temperature	0 to 40°C
Storage Temperature	–40°C to +85°C
Cooling	Convection

EMI AND SAFETY

CE Mark	Full compliance with LVD and EMC directives
Safety Standards	Meets Worldwide Safety Standards; IEC950, EN60950, UL1950 Class 2, SELV
AGENCY APPROVALS EMI/RFI	UL, c-UL, VDE, NORDICS, MITI EN55022 Class B, FCC Part 2 &15 Class B, CISPR22 Class B
MTBF	Over 200,000 hours @ 25°C ambient condition

MECHANICAL

Dimensions	See table below
Weight	9 ounces, includes DC cord
Ac Input Connector	Dual pin, IEC 320 approved C8 type, mates with interchangeable cords for world wide operations.
DC Output Connector	Right angle, hollow coaxial plug, 0.10" (2.54 mm) inside diameter, 0.21" (5.49 mm) outside diameter. Center positive polarity.
DC Cord Length	40 inches

OUTPUT VOLTAGE/CURRENT RATING CHART

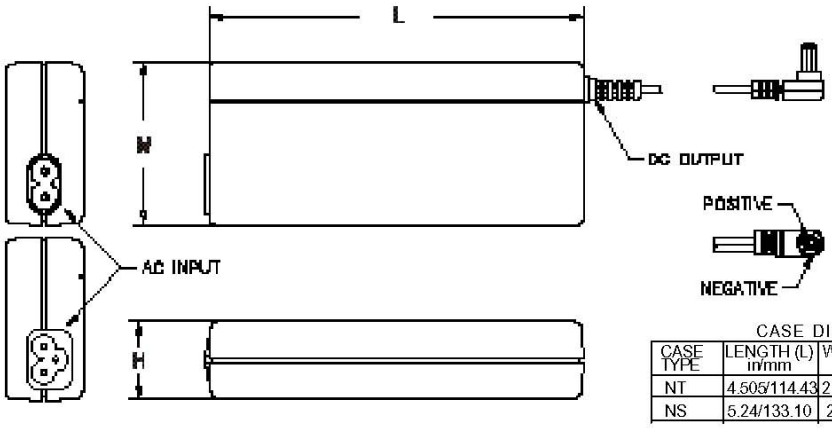
Model Number	Output Voltage	Current max (amp)	Power max peak	
ZVC60NT12E	12.0 VDC	5.00	60W	65W
ZVC60NT15E	15.0 VDC	4.00	60W	65W
ZVC65NT19E	19.0 VDC	3.40	65W	70W
ZVC65NT24E	24.0 VDC	2.70	65W	70W
ZVC70NS19E	19.0 VDC	3.50	65W	80W
ZVC70NS24E	24.0 VDC	2.90	70W	80W

EOS

Highest Density
Smallest Size
Most Efficient

EOS Corporation
1070 Flynn Road
Camarillo, California 93012
Tel: 805.484.9998
Toll free: 1.800.293.9998
Fax: 805.484.5854
E-mail: info@eoscorp.com
Web: www.eoscorp.com

Consult your local representative below or the factory for custom requirements or modifications to standard products. Specifications are subject to change without notice.



CASE DIMENSIONS

CASE TYPE	LENGTH (L) in/mm	WIDTH (W) in/mm	HEIGHT (H) in/mm
NT	4.505/114.43	2.384/60.55	1.15/29.21
NS	5.24/133.10	2.28/57.10	1.15/29.21

Contact us at: 1.800.293.9998 or www.eoscorp.com 06/00 R02

C – Elektronik

Auf den Druck der Datenblätter der Komponenten zur Elektronik des Messautomaten wird komplett verzichtet.

Alle Datenblätter der verbauten Komponenten erhalten Sie auf der beiliegenden Dokumentations – CDR.

Die Datenblätter sind im Acrobat Reader pdf-Format auf der CD-ROM hinterlegt.

D 1 – Stückliste Basisplatine

Basisplatine Rev. 1.6

Platinendatei main_final 6
letzte Änderung: 05.03.2004

Menge	Wert	Bauteil Beschreibung	Bauteilnr	Lieferant	Bestellnummer	pdf-Datei
1		AMP MINI MATE-N-Lock 2pol 2pol AMP Mini Mate-N-Lock Leiterplatten Buchse 90 Grad 5A	X1	Farnell	361-5182	X_AMP_Buchse
1		IC Festspannungsregler MC78A800CDTPACK 7805 5V/1A SMD Regulator mit 2% Tol 0° - +125°	IC1	R&S Comp.	444-9291	IC_MC7800-D
1		IC Festspannungsregler LM1085IS-5.0 DPACK 7805 5V/1A SMD Regulator	IC1	R&S Comp.	263-9795	
1		LED SMD SMD LED 5mich 0805 in rot	LED0	R&S Comp.	238-2219	LED_Osram
1		LED SMD SMD LED 5mich 0805 in blau	LED0	R&S Comp.	444-4864	
8		LED SMD SMD LED 5mich 0805 in blau	LED1..7	Farnell	322-5112	LED_SMD_0603
2		Taster SMD von Phytec 4pol Um Tasten anzusteuern	BOOT, RESET	Phytec/Mentor/Conrad	KT007 & KT002	SW_Mentor
1		Leiterplattenbuchse Micro-Match 4pol SMD 4pol Leiterplatten Buchse von AMP MicroMatch in SMD	X9	R&S Comp.	366-1721	X_Micro-Match_SMD_Buchse
6		Leiterplattenbuchse Micro-Match 6pol SMD 6pol Leiterplatten Buchse von AMP MicroMatch in SMD	X4, X5, X6, X7, X10, X11	R&S Comp.	366-1737	X_Micro-Match_SMD_Buchse
1		Leiterplattenbuchse Micro-Match 10pol SMD 10pol Leiterplatten Buchse von AMP MicroMatch in SMD	X8	R&S Comp.	366-1787	X_Micro-Match_SMD_Buchse
1		SUB D 9pol geschirmt 9pol Sub D Buchse gesch rmte Version	X2	Reichelt	EMV-Buchse 09W	X_SUB-D_9pol
1		SUB D 9pol geschirmt 9pol Sub D Buchse gesch rmte Version	X3	Reichelt	EMV-Stift 09W	X_SUB-D_9pol
1	0.250A	Sicherungshalter mit Sicherung SMD Sicherungsfolien mit Sicherung in SMD	F3	R&S Comp.	219-6068	Fuse_SMD
1	0.500A	Sicherungshalter mit Sicherung SMD Sicherungsfolien mit Sicherung in SMD	F1	R&S Comp.	219-6074	Fuse_SMD
1	5A	Sicherungshalter mit Sicherung SMD Sicherungsfolien mit Sicherung in SMD	F2	R&S Comp.	219-6131	Fuse_SMD
1	0R	Widerstand SMD 0603	R60	R&S Comp.	213-1982	R_SMD_0603
1	0R	Widerstand SMD 0805	R3	R&S Comp.	223-0146	R_SMD_0805
1	1A	Diode SMD MRA_4003_T3_SMB Diode in SMD 1A 200V ähnlich 4003 in SMB Gehäuse	D1	R&S Comp.	419-1368	D_MRA4003SMD
4	2K2	Widerstand SMD 0603	R21, R22, R31, R32	R&S Comp.	213-2317	R_SMD_0603
4	2u2	Induktivität SMD 3225M 3,3X2,5 0,8 Ohm 425mA 2,2uH Drahtl	L1, L2, L3, L4	Reichelt	LQH3N 2,2u	L_SMD_3225
6	2u2	Induktivität SMD 0805 0,90hm 50mA 2,2uH in 0805 gesucht	L10, L11, L12, L13, L14, L15	R&S Comp.	308-8586	L_WÜRTH_0603_0805_1206
2	4,7uF	Kondensator SMD 1206 gelb	C17, C18	R&S Comp.	405-9488	C_SMD_1206_TAJ
2	4,7uF	Kondensator SMD schwarz	C17, C18	R&S Comp.	405-9933	C_SMD_THJ
1	4k7	Widerstand SMD 0805	R1, R4	R&S Comp.	223-0528	R_SMD_0805
4	5,6K> max 5A	Widerstand SMD 0603 Gemäß auf 10k, bei 11k abfallen bei Schutzumfang	R23, R24, R33, R34	R&S Comp.	5K6- 213-2373 15K- 213-2430	R_SMD_0603
1	6A	Diode SMD High Current	D2	BOSE	-	
4	10k	Widerstand SMD 0603	R25, R26, R35, R36	R&S Comp.	213-2418	R_SMD_0603
2	10nF	Kondensator SMD 0603	C20, C30	R&S Comp	264-4595	C_SMD_0603_0805
2	10uF	Kondensator SMD 1210 16V polo	C11, C14	R&S Comp	405-9855 schwarz oder 405-9919 gelb	C_SMD_1210_TPS
2	15V	Diode Zener BZG03C_DO214 SMD Zener Diode SMD 5Watt 15V im DO214 Gehäuse	DZ1, DZ2	R&S Comp	447-2724	DZ_SMD
2	15V	Diode Zener BZG03C_DO214 SMD Zener Diode SMD 5Watt 15V im DO214 Gehäuse	DZ1, DZ2	R&S Comp	245-6253	DZ_SMD
4	22pF	Kondensator SMD 0805	C42, C43, C52, C53	R&S Comp	237-6804	C_SMD_MuRata
6	74 LS 14M	IC74 LS 14M SMD SO14	IC5, IC6, IC7, IC11, IC12, IC13	Farnell	527-361	IC_DM74LS14
3	74 ALS 32M	IC74 ALS 32M SMD SO14 hweres Schmitt Trigger Eingänge	DM74 LS 14 M Farnell#	Farnell	632-510	IC_DM74ALS32
1	74 ALS 153M	IC74 ALS 153M SMD SO16 Gatter	DM74 ALS 32 M Farnell#	Bürklin	42 S 8412	IC_DM74ALS153
2	100R	Widerstand SMD 0603 Multiplexer Dual 4-Kanal nicht invertierend mit enable Eingang	SN 74 ALS 153 M	R&S Comp.	213-2143	R_SMD_0603
2	100nF	Kondensator SMD 0603	C22, C32	R&S Comp	220-7922	C_SMD_MuRata
18	100nF	Kondensator SMD 0805	C4, C5, C6, C7, C8, C9, C10, C12, C15, C16, C18, C19, CA1, CB1, CB7, CB8, CB4	R&S Comp	237-7100	C_SMD_MuRata
2	100uF	Kondensator SMD 0810	C23, C33	R&S Comp	367-9615	C_SMD_FC_Panasonic
2	220nF	Kondensator SMD 0603 16V polo	C21, C31	R&S Comp	220-7950	C_SMD_MuRata
1	470R	Widerstand SMD 0805 16V	R2	R&S Comp.	223-0376	R_SMD_0805
1	10K	Widerstand SMD 0603	R35, R36, R25, R26	R&S Comp.	213-2418	R_SMD_0603
1	680R	Widerstand SMD CAY16	RN1, RN2	R&S Comp.	241-9664	R_CatCay
1	2k3	680 Ohm 4 Stück R s in einem Bauteil 1206 Widerstand SMD CAY16	RN3, RN4, RN5, RN8, RN9	R&S Comp.	241-9692	R_CatCay
1	10k	2k3 Ohm 4 Stück R s in einem Bauteil 1206 Widerstand SMD CAY16	RN6, RN7	R&S Comp.	241-9743	R_CatCay
2	BAV99	10k Ohm 4 Stück R s in einem Bauteil 1206 Diode SMD BAV/99 Doppeldiode	D20, D30	R&S Comp	287-263	D_BAV99_SMD
2	L6206 PD	50V/2,8 200mA IC Motorsteuer L6206 PD SMD	IC2, IC3	Bürklin	41 S 1076	IC_L6206_ST
1	MM_C509_2	3 f Motorsteuerung L6206PD (Power-SO36) Buchsenleiste für Phytec C509 Infineon	Sockelleiste fürs Standart 2x12 & 2x32	Phytec	VS019 & VS013	nicht notwendig
1	C509	Micro Controller C509 auf Platine		Phytec	MM-103-C509	IC_C509_Infineon,IC_MiniModul_C509_Phytec

D 2 – Stückliste Adapterplatine

Adapterplatine Rev. 1.5, Rev. 1.6

letzte Änderung : 07.01.2004

Menge	Wert	Bauteil Beschreibung	Bauteilnummer	Lieferant	Bestellnummer	pdf-Datei
1		LED SMD	LED0	R&S Comp.	238-2219	LED_Osram
1		LED SMD	LED 0603	R&S Comp.	454-4762	nicht verfügbar
1		LED SMD	LED 0603	Farnell	322-5148	LED_SMD_0603_amber
1		LED SMD	LED 0603	Farnell	322-9531	LED_SMD_0603_rot
1		LED SMD	LED 0603	Farnell	322-5112	LED_SMD_0603_blau
1		Stiftleiste einreihig	JP1, JP2	R&S Comp.	173-2809	X_Stiftleiste_einreihig
1		Stiftleiste zweireihig	JP4	R&S Comp.	173-2764	X_Stiftleiste_zweireihig
2		Taster 9mm hoch	BOOT, RESET	R&S Comp.	378-6482	SW_2
1		Leiterplattenbuchse Reichelt 3pol	X2	Reichelt	PS25/3G br oder ws	X_PS25
1		Leiterplattenbuchse Micro-Match	X1	Farnell	148-635,148-799,149-445,149-093,378-4496	
1	4k7	Widerstand SMD 0805	R1, R2	R&S Comp.	223-0528	R_SMD_0805
1	470	Widerstand SMD 0805	R3	R&S Comp.	223-0376	R_SMD_0805
6	2u2	Induktivität SMD 0805	L1, L2	R&S Comp.	308-8586	L_Würth_0603_0805_1206
4	1uF	Kondensator SMD 1206	C11, C12, C13, C14	R&S Comp.	405-9848	C_SMD_1206_TAJ
2	100nF	Kondensator SMD 0805	C1, C4, C5, C6	R&S Comp	237-7100	C_SMD_MuRata
2	22pF	Kondensator SMD 0805	C2, C3	R&S Comp	237-6804	C_SMD_MuRata
1	680 R	Widerstand SMD CAY16	RN1	R&S Comp.	241-9664	R_Cat_Cay
1	ER400TRS	SRD Chip mit uC	IC1			IC_ER400TRS
1	MAX232ECWE	RS232 Treiber in SMD	IC2	Farnell	305-0075	IC_MAX232_SMD
1		LWL Leiter flexibel 80mm	Power	Farnell	324-5305	LED_Flex_LWL
1		BNC Buchse gerade	X3	Farnell	365-0558	X_BNC_gerade
1		BNC Buchse 90 Grad gewinkelt 50 Ohm	X3	R&S Comp.	304-5501	
1		BNC Buchse gerade Frontplatteneinbau	X3	Farnell	365-0510	X_BNC_front
1		BNC Buchse 90 Grad Frontplatteneinbau	X3	Farnell	251-094	X_BNC_90front
1		2 Fach Umschalter gerade	SW1	R&S Comp.	334-246	SW_1
1		Jumper Brücken rot		R&S Comp.	251-8698	
1		Jumper Brücken schwarz lang		R&S Comp.	251-8575	
1		Jumper Brücken schwarz kurz		R&S Comp.	251-8682	

D 3 – Stückliste Sonstiges

Menge	Bauteil Beschreibung	Bauteilnummer	Lieferant	Bestellnummer	pdf-Datei
1	Schaltnetzteil extern 12V/5A		R&S Comp.	377-7141	Netzteil
3	Neigungsschalter		R&S Comp.	361-5043	Neigungsschalter
1	Omron Typ Schalterinsatz 40Grad im Kunststoffgehäuse AMP MINI MATE-N-Lock 2pol Stecker		R&S Comp.	302-0215	AMP_Stecker
1	2pol AMP Mini Mate-N-Lock Steckergehäuse 5A AMP MINI MATE-N-Lock 2pol Stecker		Farnell	133-620	AMP_Stecker
1	2pol AMP Mini Mate-N-Lock Steckergehäuse 5A AMP MINI MATE-N-Lock Crimpkontakte		R&S Comp.	302-0467	AMP_Steckerpins
1	2pol AMP Mini Mate-N-Lock Crimpkontakte 5A AMP MINI MATE-N-Lock Crimpkontakte	22-18 AWG	Farnell	133-796	AMP_Steckerpins
1	2pol AMP Mini Mate-N-Lock Crimpkontakte Buchse 5A Micro Match Kabel 16 pol	250mm: ...-4551; 150mm: ...-4540	Farnell	378-4651	
1	Micro Match Kabel 4 pol	250mm: ...-4370; 150mm: ...-4368	Farnell	378-4370	
4	Micro Match Kabel 6 pol	250mm: ...-4400; 150mm: ...-4393	Farnell	378-4400	
4	MicroMatch Flachbandkabel Stecker 4 pol		Farnell	149-032	MicroMatch_Kabelstecker
4	MicroMatch Flachbandkabel Stecker 6 pol		Farnell	149-068	MicroMatch_Kabelstecker
4	MicroMatch Flachbandkabel Stecker 16 pol		Farnell	149-147	MicroMatch_Kabelstecker
5	Sub-D 9pol Flachbandkabelstecker		R&S Comp.	409-3208	AMP_SubD_Flachbandkabel_Stecker
5	Sub-D 9pol Flachbandkabelbuchse		R&S Comp.	409-3242	AMP_SubD_Flachbandkabel_Buchse
5	Flachbandkabelstecker zur Festmontage		Farnell	525-091	3M_Flachbandkabel
5	1 Opal grau zum Schrauben Flachbandkabelbuchse zum Schneidklemmen ohne Zugentlastung		Farnell	468-010	3M_Flachbandkabel
10	1 Opal grau Verriegelungen für Flachbandkabelstecker		Farnell	413-9136	3M_Flachbandkabel
10	1 Opal grau Bolzen für Flachbandkabelstecker Verriegelungen		Farnell		3M_Flachbandkabel
4	1 Opal grau Motor DC mit Getriebe + Drehzahlgeber	DC Kleinstmotor 2224...SR, Getriebe 20/1...-1, Impulsgeber IE2-16	Faulhaber		Motor2224-012SR Impulsgeber IE2-16
1	Faulhaber DC Kleinstmotor 2224...SR Motor DC mit Getriebe + Drehzahlgeber	DC Kleinstmotor 2224...SR, Getriebe 20/1...-1, Impulsgeber IE2-16	Faulhaber		Getriebe20_1
12	Faulhaber DC Kleinstmotor 2224...SR Micro Schalter für Achsenmechanik		Reichelt	MAR 1006.0701	MAR1006_Mar
1	Wechsler Touchpanel		Electronic Assembly	8-A KIT LMT LPTP,DPT 5-15V,DPT-DPTDIS,KV3-56	LCDkit160-7
1	Neutrik 220V Einbaubuchse		R&S Comp.	246-8284	Neutrik_Connector
1	Neutrik 220V Stecker		R&S Comp.	246-8313	Neutrik_Connector
1	Zugentlastung Lemo rund 10pol 2B	Stecker + Buchse für RS232 / DAMS Kabel	R&S Comp.	178-8969	Lemo_Instructions
1	Einbaubuchse Lemo rund 10pol 2B	Stecker + Buchse für RS232 / DAMS Kabel	R&S Comp.	178-9029	Lemo_Buchse
1	Stecker Lemo rund 10pol 2B	Stecker + Buchse für RS232 / DAMS Kabel	R&S Comp.	178-8896	Lemo_Stecker
1	9 pol SubD Einbaubuchse	RS232 Kabel Verbindung Olli <> PC	R&S Comp.	472-843	SubD_9pol
1	vergoldet Gehäuse für 9 pol SubD Einbaubuchse	RS232 Kabel Verbindung Olli <> PC	R&S Comp.	160-0768	Kappe_SubD
1	zum Schrumpfen Multicore Kabel grün	Microphonkabel 4x2x0.14 se Inflex bei mehreren Meter	Farnell	135-240	Cable_green
1	Multicore Kabel blau	Microphonkabel 4x2x0.14 se Inflex bei mehreren Meter	Farnell	798-459	Cable_blue
2	Mini XLR Kabel-Stecker		R&S Comp.	453-662	Switchcraft_Connector
1	Deckelkontaktschalter				
1	Festlager	loderes Lager(am Motor/Welle) für die Spindel	THK	FK6	THK_Lager
1	Loslager	festes Lager für die Spindel	THK	FF6	THK_Lager
2	Präzisionswelle	8mm Durchmesser Präzisionswelle 400mm lang	R&S Comp.	341-1532	SKF_Welle_LJM
2	Linearlager / Wellenlager	SKF LJM 8/400 nach Ref Lager das auf der Präzisionswelle läuft	SKF	LBBR 8-2LS/HV6	SKF_LBBR
2	Linearlager / Wellenlager	Lager in dem der Kalt läuft	R&S Comp.	294-8100 oder LBBR 8-2LS/HV6	SKF_LBBR
1	Spindel mit Spindelmutter 145mm lang	Kugelgewindetrieb	THK	BNK0801-3G2+145LC7Y	THK_Kugelgewindtrieb
1	Spindel mit Spindelmutter 175mm lang	Kugelgewindetrieb	THK	BNK0801-3G2+175LC7Y	THK_Kugelgewindtrieb
1	Kupplung 4mm><4mm	Verbindung zwischen Getriebe & Gewindetrieb	R&S Comp	814-562	Kupplung_Flexbeam
1	Würfelverbinder für Rahmensystem	zur Verbindung von 3 Profilen	R&S Comp	390-1928	Rexroth_Verbinder
1	Strebenprofil mit 8erNut Bosch Rexroth	30x30mm 2000mm lang	R&S Comp	389-9780	Rexroth_Profile2
1	Abdeckprofil für Strebenprofil		R&S Comp	418-0841	Rexroth_Profile
1	Gehäuse für Main_Board				
1	Flachbandkabelhalter		R&S Comp	609-506	Flachbandkabelhalter
2	Butterfly	Flay 105/5	Inter Technik I.T		
1	Griff	KGSM 150	Inter Technik I.T		
1	Distanzbolzen für Sub-D Buchse lang	M3 <> M3	R&S Comp.	105-8173	nicht notwendig
1	Distanzbolzen für Sub-D Buchse kurz	M3 <> 4/40	R&S Comp.	259-5544	nicht notwendig
1	Reed Sensor	M8 28mm lang	R&S Comp.	289-7957	Reed Sensor
1	Magnet für Reed Sensor	6mm Durchmesser; 20mm lang	R&S Comp.	289-7856	Reed Sensor
	2,5mm Einbaubuchse	für Easy Zweitgehäuse	R&S Comp.	286-8741	
	Sicherungshalter 5Fach		R&S Comp.	250-6314	
	Madenschraube M6 mit Kugel		R&S Comp.	257-8145	

E – Notizen

F – Dokumentation auf CDROM

Die beiliegende CDROM in ein CDROM – oder ein DVD-Laufwerk eines kompatiblen PCs einlegen. Die CDROM startet von selbst und muß nicht installiert werden.