Die approbierte Originalversion dieser Dissertation ist an der Hauptbibliothek der Technischen Universität Wien aufgestellt (http://www.ub.tuwien.ac.at).

The approved original version of this thesis is available at the main library of the Vienna University of Technology (http://www.ub.tuwien.ac.at/englweb/).

TU

Technische Universität Wien

Dissertation

Eine Anwendung von fehlertoleranten vernetzten Rechnersystemen

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften/Naturwissenschaften unter der Leitung von

o. Univ. Prof. Dr. Dr.h.c.mult. Peter Kopacek
E325/A6
Institut für Mechanik und Mechatronik
Intelligente Handhabungs- & Robotertechnik IHRT

eingereicht an der technischen Universität Wien
Fakultät für Maschinenwesen und Betriebswissenschaften

von

Dipl.-Ing. Johann Adlmaninger
Matr. Nr. 7926091
Schloßteichweg 45, 9020 Klagenfurt

Wien, im Oktober 2005

Unterschrift

Diese Dissertation haben begutachtet:

.....

Kurzfassung

Ziel dieser Arbeit war, die Grundlage für ein Rechensystem mit hoher Störungstoleranz und automatischer Ausfallsbehebung im Bereich von Transportsystemen zu erarbeiten.
Für die Transportsysteme wurde eine Steuerung entwickelt, implementiert und erprobt, welche auf Basis von verteilten Rechnersystemen arbeitet. Der Ausfall einer oder mehrerer Systemkomponenten wird automatisch erkannt und das System durch geeignete Maßnahmen wieder in einen konsistenten Zustand gebracht, so dass es für die gesamten funktionellen Abläufe keine nennenswerten Unterbrechungen gibt.

Im Rahmen dieser Arbeit wurden verschiedene Ansätze für die Realisierung dieser Wiederanlauftechnik diskutiert. Für das vorliegende Fallbeispiel wurde das Prinzip der Suchstrategie unter Einbeziehung aller verfügbarer Vorinformationen gewählt.

Der Einsatz erfolgte in einer Mehrlinien Rohrpostanlage. Das Projekt umfasst die Projektierung und die gesamte Hard- und Softwareentwicklung. Das Beispiel, der in dieser Arbeit ausgewählten pneumatischen Transportanlage, ist ein typischer Vertreter solcher Systeme.
Alle wesentlichen Aspekte werden im Detail diskutiert; dies sind insbesondere der Systementwurf, die Hardwareentwicklung, das Datenbankdesign, die Softwareentwicklung für die einzelnen Hardwarekomponenten, sowie das automatische Anrufsystem. Der Detaillierungsgrad ist so gewählt, dass das vorgestellte Material als Referenz für vergleichbare Projekte im Bereich fehlertoleranter verteilter Rechnersysteme dienen kann.

Die Steuerungssoftware ist frei konfigurierbar und damit auf andere Anlagenarchitekturen übertragbar. Die Auslegung der Stationsrechner und Kommunikationsrechner ist universell. Die Standardisierung dieser Komponenten kann bei Folgeprojekten ein beträchtlicher Kostenvorteil sein.

Abstract

It is the main objective of this doctoral thesis to develop the basic working principles of a computing system with high fault tolerance and automatic failure elimination in the field of transporting systems.

For such transport systems, a controller which works by means of distributed computing systems has been developed, installed and tested.

Any failure of one or more system components is immediately automatically identified, and through appropriate measures the system is then brought back into its consistent original state. This means that no major interruptions can paralyse the functional operations of the whole system.

Within the framework of this thesis, different approaches towards the realisation of this special re-start technology are evaluated. For this, the sub-strategy principle, inclusive of all available sources of information, has been selected.

The new technology was then installed in a multi-line pneumatic tube system. The project thus comprised the fields of project planning and project management, as well as all aspects of hardware and software development. The pneumatic tube system chosen for this project is a transport system that is typical of its kind. Some of the detailed tasks that had to be completed for the project are as follows : system planning and development, hardware development, database design, software development for the individual hardware components, and the automatic calling system. The degree of detailed illustration has been selected in such a way that the materials and documents introduced may serve as references for any further comparable projects in the field of fault-tolerant distributed computing systems.

The control software is freely configurable and may thus be transferred to other network architectures. The combination and design of the station computers and the communication computers are universal. This standardisation of necessary components may lead to a considerable cost benefit for future projects.

As the findings from in-company tests show, all specifications concerning fault tolerance have been completely fulfilled. The final overall solution has been met with the highest acceptance by both the users and the service personnel.

0 AUFGABENSTELLUNG 3

1 GRUNDLAGEN VERTEILTER SYSTEME 5

1.1 ÜBERBLICK 5
1.1.1 Geschichte 5
1.1.2 Definition 14
1.1.3 Probleme in verteilten Systemen 14
1.1.4 Architektur von verteilten Systemen 15
1.1.5 Multicomputer 16
1.1.6 Anwendungen von verteilten Systemen 17

2 STAND DER TECHNIK 18

2.1 ENTWICKLUNG PNEUMATISCHER TRANSPORTSYSTEME 18
2.2 HERAUSFORDERUNGEN UND NEUE CHANCEN FÜR PNEUMATISCHE TRANSPORTSYSTEME 19

3 FEHLERTOLERANTE VERTEILTE RECHNERSYSTEME 21

3.1 EINLEITUNG 21
3.1.1 Allgemeine Überlegung 22
3.2 ERKENNUNG UND BEHEBUNG VON FEHLFUNKTIONEN EINZELNER KOMPONENTEN 23
3.2.1 Erkennen eines Rechnerausfalls 23
3.2.2 Wiederaufsetzpunke festlegen 26
3.2.3 Anwendungsbeispiel 27
3.3 HERSTELLUNG EINES KONSISTENTEN ZUSTANDS IM VERTEILTEN SYSTEM 28
3.3.1 Lösungsweg 1: Vollständiger Systemreset 28
3.3.2 Lösungsweg 2: Minimierter Systemreset 29
3.3.3 Lösungsweg 3: Die Suchstrategie mit maximaler Vorinformation 30
3.4 LÖSUNGSANSÄTZE FÜR VERTEILTE SYSTEM IN VOLLSTÄNDIG STÖRUNGSBEHAFTETER UMGEBUNG 32
3.4.1 Redundanzlösung 32
3.4.2 Verteilte Intelligenz 33
3.5 EIN NEUES STEUERUNGSKONZEPT 34
3.5.1 Steuerungsübersicht 34
3.5.2 Softwarestruktur 35
3.5.3 Integration der Steuerung in die bestehende Telefonanlage 40
3.5.4 Leitrechner 40
3.5.5 Kommunikationsrechner 40
3.5.6 Stationsrechner 41
3.5.7 Kommunikation zwischen den Komponenten 43

4 EIN INDUSTRIELLES ANWENDUNGSBEISPIEL EINER PNEUMATISCHEN TRANSPORTANLAGE 44

4.1 TECHNISCHE ZIELE 44

4.2 MECHANISCHE KOMPONENTEN UND MECHANISCHER AUFBAU EINER PNEUMATISCHEN TRANSPORTANLAGE 47
4.2.1 Mechanische Komponenten 47
4.2.2 Mechanische Details der Stationen 48
4.2.3 Aufbau einer Anlage 50
4.2.4 Wartungsmaßnahmen 53
4.3 REALISIERUNG 53
4.3.1 Leitrechner 54
4.3.2 Kommunikationsrechner 81
4.3.3 Stationsrechner 90
4.4 BETRIEBSERFAHRUNGEN 106

5 ZUSAMMENFASSUNG UND AUSBLICK 109

6 ANHANG 110

6.1 NETZWERKGRUNDLAGEN 110
6.1.1 Controller Area Network – CAN 110
6.1.2 Universal Serial Bus – USB 115
6.1.3 LAN 121

7 LITERATUR 133

8 REGISTER 138

8.1 ABBILDUNGEN 138
8.2 TABELLEN 139

0 Aufgabenstellung

Transportsysteme sind eine interessante Klasse von Systemen. Sie bestehen in der Regel aus

- mechanischen/elektromechanischen Komponenten
- einem elektronischer Steuerungssystem (teilweise mit Sensoren)
- Mensch-Maschine-Schnittstellen

Es liegt in der Natur von Transportsystemen, dass die elektronische Steuerung als Hierarchie von lokalen Steuereinheiten und zentralen Steuereinheiten aufgebaut sein muss. Durch diese Randbedingungen sind die Steuerungssysteme solcher Systeme teilweise unvermeidbaren Störungen ausgesetzt, die zu Fehlfunktionen einzelner Komponenten des Systems führen. Solche Störungen lassen sich in vielen Fallen aus grundsätzlichen konstruktiven Gründen als auch aus Kostengründen nicht vollständig vermeiden. Ebenso erniedrigen Redundanzen einzelner oder aller Steuerungskomponenten die Störungswahrscheinlichkeit nur, können diese aber nicht vollständig ausschließen.

Ziel dieser Arbeit ist es daher, die Grundlagen für verteilte Steuerungssysteme *mit hoher Störungstoleranz* durch automatische Ausfallsbehebung im Bereich der Transporttechnik zu erarbeiten. Diese Eigenschaft wird in Folge Wiederanlauftechnik genannt und umfasst folgende Aspekte:

- Der Fehlfunktion eines oder mehrerer Komponenten innerhalb des verteilten Systems muss automatisch erkannt und behoben werden.
- Das System muss durch geeignete Maßnahmen wieder in einen konsistenten Zustand gebracht werden, so dass es für die gesamten funktionellen Abläufe keine nennenswerten Unterbrechungen gibt.

Das hier vorbestellte neue Konzept für eine Wiederanlauftechnik wird an einem konkreten Fallbeispiel implementiert und hinsichtlich ihrer Ergebnisse, Anwendbarkeit und Übertragbarkeit auf andere System diskutiert.
Die einzelnen Komponenten werden mit technischen Details vorgestellt und beschrieben.

Bei dem Fallbeispiel handelt es sich um ein pneumatisches Transportsystem, das in diesem Fall an einem Krankenhaus in Lahti zum Einsatz gekommen ist. . In allen Phasen des Projekts war der Autor als Projektleiter überwachend und in wesentlichen Teilbereichen gestaltend in das Projekt eingebunden.

Rohrpost-Transportsysteme sind in den letzten Jahren in vielen ihrer ursprünglichen Aufgabengebieten durch Kommunikationssysteme verdrängt worden. Heute wird ihr Nutzen jedoch neu entdeckt und sie gewinnen zusehends an Bedeutung. Primäre Aufgabe der Transportsysteme in früheren Zeiten war der Transport von Dokumenten. Heute liegt ihre Bedeutung hauptsächlich im Transport von Proben(Krankenhausbereich) oder Geld(Banken). Wurden diese Transportsysteme früher meistens aus Stahl gebaut, so ist man heute dazu übergegangen, sie aus Kunststoff zu bauen. Dies hat neben einigen Vorteilen aber den Nachteil, dass durch die Reibung der feststehenden mit den beweglichen Kunststoffteilen und /oder der Luft sehr hohe elektrische Ladungen auftreten. Diese Ladungen wirken sich auch auf die Elektronik (Rechnersysteme) aus und verursachen massive Störungen.

Abschließend wird ein Erfahrungsbericht über die Installation der ersten Anlage vorgenommen. Die positiven und negativen Erfahrungen im Zusammenhang mit dem Projekt werden ebenfalls angeführt.

1 Grundlagen verteilter Systeme

1.1 Überblick

1.1.1 Geschichte

Der PC: Von der Urgeschichte in die Neuzeit

Schon 30000 v. Chr. verwendeten die Menschen einfache Zahlenzeichen wie Kerben, Knoten oder Striche zum Zählen. Zwischen 3000 und 2000 v. Chr. wurde in Mesopotamien ein abstraktes Zahlensystem in Form eines Sexagesimalsystems verwendet.

200 n. Chr. entstand der erste „Handrechner" – der Abakus, der auch heute noch manchmal verwendet wird.

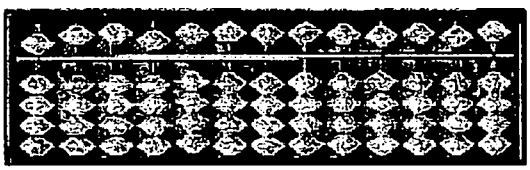


Abb.: 1.1 Asiatischer Abakus

500 n. Chr. wurde in Arabien das dezimale Zahlensystem entwickelt. 1600 n. Chr. wurden Logarithmen entdeckt. Damit wurde es möglich eine Multiplikation mit Hilfe der Addition durchzuführen. Bald darauf entwickelte der Schotte J. Napier Rechenstäbe, mit deren Hilfe die Multiplikation auf die Addition von Teilprodukten zurückgeführt wurde. Diese Rechenstäbe wurden später in einer Rechenmaschine bereits eingesetzt.

Um 1700 erschuf der Mathematiker Leibnitz das binäre Zahlensystem und er entwickelte die Gesetzte der binären Arithmetik (wurden das erste Mal in der Z1 von Zuse umgesetzt), die heute noch die Grundlagen der elektronischen Datenverarbeitung bilden.

Ab der Mitte des 20. Jahrhunderts hat sich dann die Informationstechnologie rasant weiterentwickelt. Neben den bereits erwähnten „Vorarbeiten" wurden viele Voraussetzungen dafür auch durch Forschungen von Wissenschaftlern im 2. Weltkrieg geschaffen. So konnten beispielsweise die Briten mit Hilfe von Colossus den deutschen Code ENIGMA knacken.

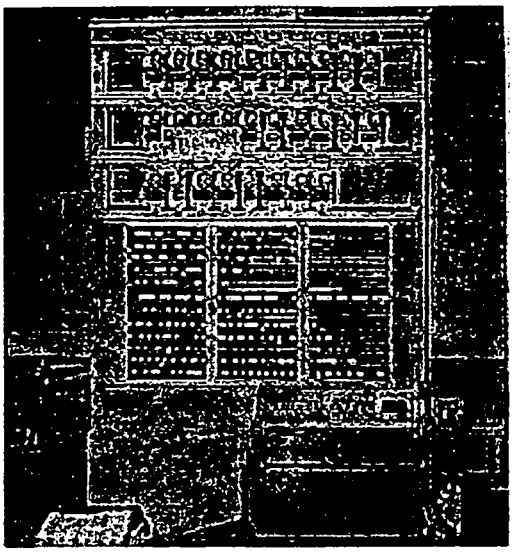


Abb.: 1.2 Colossus

Heutzutage ist ein Leben ohne Informationstechnologie nicht mehr wegzudenken. Sie hat in allen Bereichen – von der chipgesteuerten Kaffeemaschine bis zur Raumfahrt, Einzug gehalten.

Die 50er Jahre

Die fünfziger Jahre stellen den eigentlichen Beginn der Informationstechnologie dar. Die „Rechenmaschinen" funktionierten mit Hilfe von elektronischen Schaltkreisen und Relais und hatten nicht selten die Ausmaße von größeren Räumen. Auch Betriebssysteme gab es zu der Zeit noch nicht.

Da es noch keine Datenträger gab, über die Programme geladen werden konnten, mussten sie in der Maschine fest verdrahtet werden. Dementsprechend aufwändig und unflexibel war das Ausbessern von Fehlern.

Am 9.9. 1945 findet Grace Murray Hopper bei der Arbeit an einem Prototypen des Mark II den ersten Computer- „bug". Eine Motte hatte sich in ein Computerrelais eingeschlichen und einen Fehler verursacht. Genau dieses Exemplar kann heute im National Museum of American History in Washington, DC betrachtet werden.

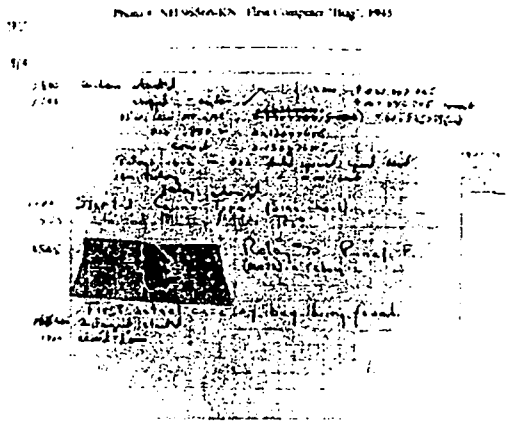


Abb.: 1.3 Photo des ersten „Bugs" in einem Computer

Der Begriff Bug wird heute aber generell für das Fehlverhalten von Computersystemen eingesetzt, unabhängig davon, ob es sich um einen Hardware- oder Softwarefehler handelt.
In diesen Zeitraum fällt auch die Entwicklung von Transistoren.

Einen wesentlichen Grundstein zur Entwicklung der heutigen Computersysteme leistete Prof. Heinz Zemanek. Er wurde am 01.01. 1920 in Wien geboren. Zwischen 1955-1958 baute er den ersten österreichischen Großrechner auf Transistorbasis ("Mailüfterl"), ab 1961 war Prof. H. Zemanek bei IBM und 1976-85 IBM-Fellow und Direktor des IBM-Laboratoriums in Wien.
Prof. H. Zemanek lehrte ab 1964 an der Technischen Universität in Wien. Er entwickelte die "Vienna Definition Language" als logische Fachsprache in der Computeranwendung.

Die 60er Jahre

In den 60er Jahren wurden die ersten kommerziellen Systeme zum Zwecke der Abrechnung gebaut. Der wahrscheinlich erste kommerziell vermarktete Computer wird UNIVAC genannt und kostet die damals astronomische Summe von 1 Million $.

Für die sehr großen und komplexen Anlagen mussten eigene Personen angestellt werden, die die Betreuung übernahmen – der Beruf des Systemadministrators war damit geboren.
Die Überlegung, dass auf diesen teuren und komplexen Systemen mehrere Programme gleichzeitig ausgeführt werden könnten um sie wirtschaftlicher zu nützen, führte zur Entwicklung von Betriebssystemen. Diese mussten die vorhandenen Ressourcen (CPU, Speicher, Kastenleser, etc) so verwalten, dass mehrere Programme quasi gleichzeitig ausgeführt werden konnten, ohne sich jedoch gegenseitig zu behindern.

Als Speichermedien wurden Lochkarten verwendet, auf denen Programme mittels Lochkartenstanzmaschinen abgebildet wurden. Eingelesen wurden die Lochkarten durch eigens entwickelte Lochkartenleser.

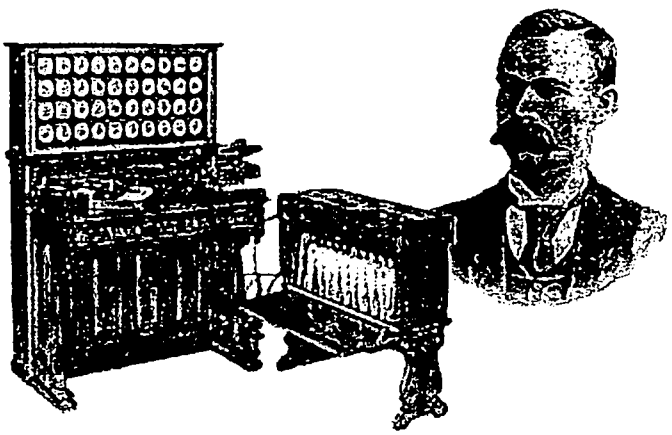


Abb.: 1.4 Hollerith mit Lochkartenmaschine

Da die Lochkarten als Stapel eingelesen wurden, wurden die Betriebssysteme Stapelverarbeitungsbetriebssystembe bzw. Batchbetriebssysteme genannt. Bei diesen Betriebssystemen musste der Startzeitpunkt eines Programms durch den Systemadministrator geplant werden, da sie natürlich für die Durchführung am Rechner sehr viel Zeit benötigten. Programme liefen oft in mehreren Steps ab, wobei ein Step als Input den Output des vorhergehenden Steps benötigte. Der Systemadministrator musste dafür sorgen, dass die Ausgabe eines Steps auf einen Kartenschreiber erfolgte, um als Eingabe für den nächsten Step zu dienen.

Die Entwicklung des ersten Compilers fällt ebenfalls in diesen Zeitraum. Die Rechenmaschinen werden mit Transistoren betrieben und als 2. Generation bezeichnet. Gegen Ende der 60-er Jahre wird das Patent auf integrierte Schaltkreise (IC) angemeldet und damit der Grundstein für eine neue Rechnergeneration gelegt.

Das ARPANET

Der wichtigste Vorläufer für das Internet in seiner heutigen Form ist das in den 60er Jahren in den Vereinigten Staaten entstandene **ARPANET** *(Advanced Research Projekts Agency.)* Das ARPANET entstand aber nicht an einer eigenen Forschungseinrichtung, sondern es wurde mit universitären und industriellen Vertragspartnern kooperiert.

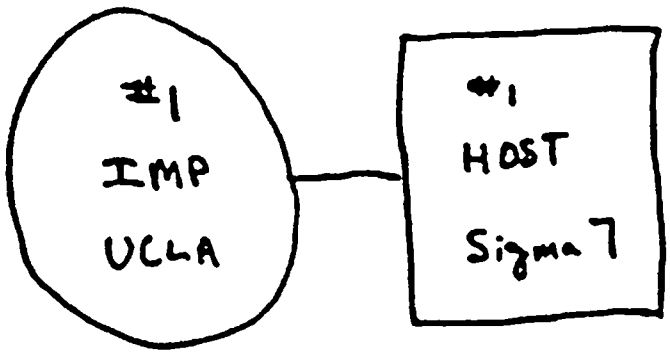


Abb.: 1.5 Erstes Diagramm für die Verbindung

Das erste ARPA-Net wurde mit Hilfe von 4 Rechnern mit jeweils 12 KB Speicher realisiert. AT&T stellte 50 kbps-Leitungen zur Verfügung.

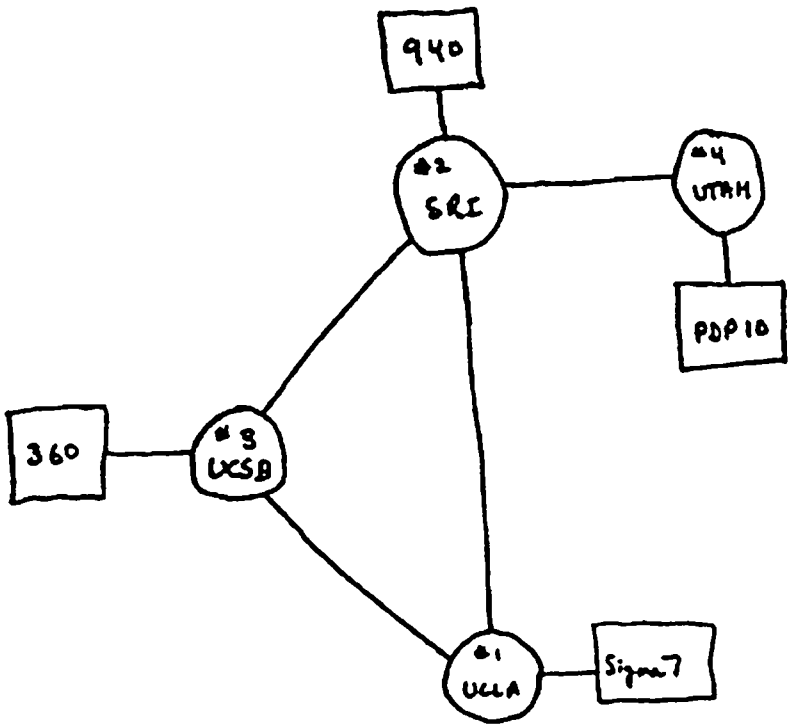


Abb.: 1.6 Diagramm für die 4-Knoten Verbindung

Im Oktober 1969 versuchte ein Wissenschaftler das erste Datenpaket von UCLA nach SRI zu schicken. Der Systemabsturz erfolgte bei Eingabe des Buchstabens G im Wort LOGIN.

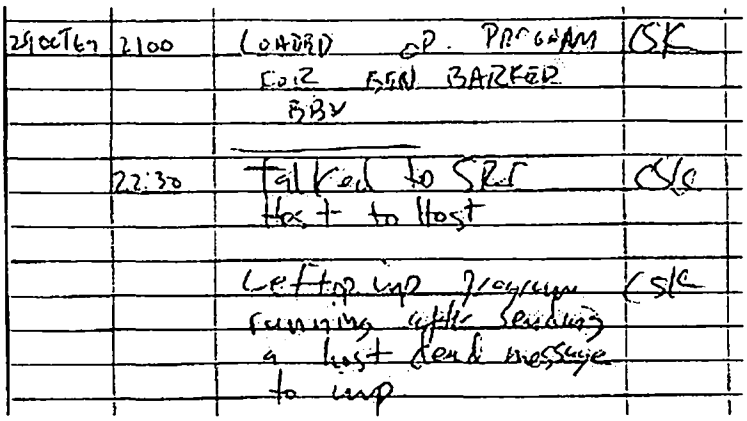


Abb.: 1.7 Handschriftliches Protokoll vom 29.10.1969

Durch die Entwicklung von Time-Sharing-Betriebssystemen war es für die Benutzer von Computern möglich geworden, interaktiv und gleichzeitig mit anderen Benutzern einen Teil der Rechenzeit eines zentralen Prozessors in Anspruch zu nehmen. Viele Terminals waren an einem zentralen Hostrechner angeschlossen, wobei die Terminals die Form eines Sterns mit dem Host in der Mitte bildeten. Der gravierende Nachteil war, dass bei Ausfall des Host das gesamte Netz zusammenbrach.

Paul Baran von der RAND-Corporation schlug, um der Sorge der US Air Force um die Überlebensfähigkeit ihrer Kommandostruktur im Falle eines sowjetischen Nuklearangriffs Rechnung zu tragen, an Stelle des konventionellen Sterns ein "distributed network" vor, bei dem spinnenförmig eine **Vielzahl von Verbindungen zwischen den in das Netzwerk eingebundenen Rechnern** vorgesehen war. Außerdem hatte *Baran* die Idee des "packet switching", einem Aufspalten der Nachricht in viele kleine Pakete, um das Datenaufkommen im Netzwerk besser zu verteilen. 1969 wurde schließlich mit der Vernetzung der ARPA-Forschungseinrichtungen begonnen.

Die 70er Jahre

Die Forschung im Bereich der Computertechnik schritt rasch voran.

Der erste Minicomputer mit Bildschirm und Tastatur (PDP1) kam auf den Markt und eine wesentliche Erneuerung waren entfernbare Disketten. Am Stanford Research Institut wurde die Maus entwickelt.

Durch die neuen Schaltkreise (IC) wurden auch die Ausmaße der Rechner beträchtlich reduziert.

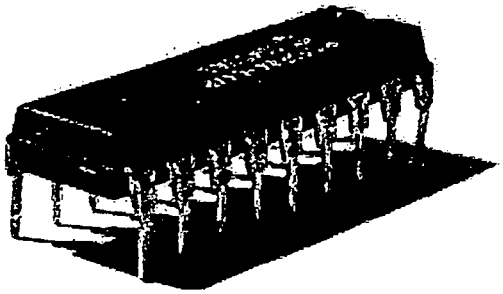


Abb.: 1.8 I(ntegrated)C(ircuit)

Damit war die dritte Computer-Generation geboren. Eine wesentliche Erleichterung im Zusammenhang mit den Neuerungen war das „interaktive Arbeiten". D.h. der Benutzer musste nicht mehr auf einen Papierausdruck warten, sondern bekam Rückmeldungen vom System direkt auf sein Terminal

.

Voraussetzung zum Betreiben solcher Computersysteme war aber immer noch ein großer Rechner und damit das Vorhandensein eines Rechenzentrums.

Auch war es nun, bedingt durch den Einsatz von Datenleitungen möglich, Terminals an weiter entfernten Orten aufzustellen. Besonders im Bankbereich („Online-Buchungen") wurde dadurch der Einsatz von Computersystemen stark forciert.

Im Bereich der Peripherie gab es Neuerungen durch die Einführung des Standards RS-232-C für serielle Schnittstellen und die Erfindung des Laserdruckers.

Kenneth Thomson und Dennis Ritchie entwickelten in den Bell-Laboratories das Unix-Betriebssystem.

Die 80er Jahre

Die vierte Computer-Generation kam auf den Markt. IBM brachte das IBM-System S/370 mit monolithischen Logikschaltkreisen heraus und bald darauf kam das OS/VS1 – Betriebssystem auf den Markt, in dem das erste Mal das „Virtual-Storage-Konzept" zum Tragen kam. Außerdem entwickelte IBM die erste Festplatte für Großrechner und gleichzeitig den Tintenstrahldrucker.

In dieser Dekade wurde durch das verstärkte Aufkommen von PCs (Personal Computer) eine wahre Revolution ausgelöst. Mit dem Apple II begann das PC – Zeitalter. Denn nun war der Einsatz von Informationsverarbeitung nicht mehr nur Firmen mit Rechenzentren möglich, sondern auch Privatpersonen konnten Computer zu Hause einsetzen. Die Rechner waren anfänglich zwar noch sehr leistungsschwach (16 kB Ram, 160 kB Floppy-Laufwert, 4,77MHz CPU) aber im Vergleich zu Hosts kosteten sie nur einen Bruchteil. Für die PCs erschienen die ersten Anwendungsprogramme (Wordstar – eine Textverarbeitung und VisiCalc – ein Tabellenkalkulationsprogramm).

Intel stellte den ersten 8-bit Prozessor vor, den 8008. Im Bereich der Programmiersprachen entwickelte Dennies Ritchie C. Gegen Ende des Jahrzehnts gab es bereits neue Prozessoren mit 16 Bit.

Trotzdem waren die großen, leistungsfähigen Rechner immer noch notwendig, um Betriebe entsprechend zu versorgen. Ende der 80er Jahre brachte IBM das MVS/ESA – Betriebssystem für Hosts (auch „Mainframes" genannt) auf den Markt, das bereits sehr großen Komfort bot (31-Bit Adressierung, erweiterter Speicher).

Die 90er Jahre

Motorola und Intel entwickelten die ersten 32-Bit Prozessoren und Microsoft brachte das PC-Betriebssystem MS-Dos 1 hervor. Die Programmiersprache C++ wurde entwickelt und für die Textverarbeitung gab es die erste Version des noch heute im Einsatz befindlichen MS-Word.

Gegen Ende der 80er Jahre kamen die ersten Festplatten für PCs zum Einsatz. Intel stellte den 486er vor mit 1.2 Mio Transistoren.

Die Weiterentwicklung an neuen Hardwarekomponenten, Systemprogrammen und Anwendungsprogrammen ging von da an geradezu rakentenhaft vor sich.

Die stand-alone PCs wurden Schritt für Schritt in lokale Netze eingebunden, da diese nun bereits leistungsfähig und preiswert genug waren. Gleichzeitig konnten damit Ressourcen gespart werden, da periphere Geräte wie Drucker gemeinsam genutzt werden konnten.

Die 2000er Jahre

Der Jahreswechsel von 1999 auf 2000 verursachte bei den IT- Verantwortlichen gewisse Unsicherheit. Es stand die Frage im Raum, welche Programme funktionieren nach dem Jahreswechsel noch und welche nicht. Der Grund dafür waren die unterschiedlichen Datumsformate die in den Programmen zum Einsatz kommen.

Die Entwicklung der Hardware ging rasant voran. CPUs werden ständig schneller und leistungsfähiger und Personalcomputer verfügen über zunehmend mehr Hauptspeicher und größere Festplatten.

Bei den Betriebssystemen gab es permanente Neuerungen und Verbesserungen und es folgte eine Version der nächsten (Umstellung von Win-NT auf Windwos2000 und Windows-XP).

Es gab auch eine wesentliche Steigerung der Übertragungsraten in den verschiedenen Netzwerken. So wurden in lokalen Netzwerken Übertragungsrate von 1 Gb/s möglich.

In der Automatisierungstechnik kam ein weiteres Bussystem hinzu, das so genannte Realtime Ethernet.

Durch die flächendeckende Vernetzung in allen Bereich kamen natürlich auf die Unternehmen wieder neue Chancen und Anwendungsgebiete dazu. Gleichzeitig wuchs aber auch die Anforderung im Bereich der Verfügbarkeit und der Sicherheit von Systemen.

Zusammenfassung

Die Vielfältigkeit der IT-Landschaften ist im Laufe der letzen Zeit sprunghaft angewachsen. Nahezu jährlich löst eine Systemgeneration die nächste ab. Die Abhängigkeit von Anwendungsprogrammen an Systeme ist jedoch nicht zu unterschätzen. Daher ist eine Umstellung von einer Generation auf die nächste meistens nur mit erheblichen finanziellen Aufwänden möglich.

Aus diesem Grund wird es immer wichtiger, dass Anwendungsprogramme nach Möglichkeit von den Systemen und ihrer Architektur unabhängig sind, damit bei Einsatz neuer Hardware nicht auch die Software ständig neu erstellt werden muss.

1.1.2 Definition

Verteilte Systeme sind Systeme die sich auf mehrere Prozessoren, Speicher und unterschiedliche Peripherie beziehen. Die Hardware- und Softwareressourcen werden untereinander aufgeteilt.

Definition eines verteilten Systems nach Andrew Tanenbaum: [1][4]

> *„Ein verteiltes System ist eine Kollektion von unabhängigen Computern, die dem Benutzer als Einzelcomputer erscheinen."*

Diese Definition impliziert, dass die Computer miteinander vernetzt sind und die Ressourcen wie Hardware, Software und Daten gemeinsam benutzt werden. Das gesamte System wird als eine Einheit gesehen.

1.1.3 Probleme in verteilten Systemen

Die viel zitierte nicht ganz ernst gemeinte Definition von Leslie Lamport [7] weist auf einige Kernprobleme von verteilten Systemen hin.

> *„Ein verteiltes System ist ein System, mit dem ich nicht arbeiten kann, weil irgendein Rechner abgestürzt ist, von dem ich nicht einmal weiß, dass es ihn überhaupt gibt."*

Diese Aspekte sind:

- Durch die Vernetzung verschiedener Einzelsysteme zu einem Gesamtsystem steigt die Komplexität. Anwendungen die auf viele, miteinander kommunizierende Systeme angewiesen sind, sind wesentlich schwerer zu handhaben, als Anwendungen die auf Einzelsystemen laufen.

- Zu der inhärenten Systemkomplexität kommen noch Fehlermöglichkeiten durch das Netzwerk.

- Häufige Fehler bei der Kommunikation der Systemkomponenten ergeben sich aus den implementierten Synchronisationsmechanismen. Bei asynchroner Abarbeitung kann jeder Rechner unabhängig von allen anderen den Befehl verarbeiten. Bei synchroner Abarbeitung müssen sich die Rechner untereinander jedoch abstimmen. Das bedeutet, dass die Rechner untereinander verstärkt kommunizieren müssen. Bei dieser Form der

Bearbeitung spielt auch die Zeit eine wesentliche Komponente. Wird bei solchen synchronen Systemen ein vorgegebenes Zeitintervall für einen Durchlauf eingehalten, so spricht man von Echtzeitsystemen (deterministische Systeme).

- Fehlfunktionen und Ausfälle einzelner Systemkomponenten müssen sofort erkannt werden. Die effiziente Reaktion auf solche Ereignisse ist ein zentraler Gegenstand dieser Arbeit.

1.1.4 Architektur von verteilten Systemen

Das bekannteste und einfachste Klassifikationsschema unterscheidet nach Einfachheit oder Vielfachheit von Befehls- und Datenströmen. Durch Kombination gelangt man zu vier Klassen.

Tabelle 1: Hardwarekonzept und die Taxonomie nach Flynn[1][4][7];

	SD (Single Data)	MD (Multiple Data)
SI (Single Instruction)	SISD	SIMD
MI (Multiple Instruction)	MSID	SIMD

- SISD

  Single Instruction Single Data Stream ist ein einfacher Einzelplatzrechner mit einem Prozessor, auf dem immer nur ein Programm zur gleichen Zeit laufen kann (konventionelle von Neumann Rechner).

- SIMD

  Single Instruction Multiple Data Stream ist ein Vektorrechner mit mehreren Prozessoren und einem Programm, das synchron auf allen Prozessoren läuft. Jeder Prozessor führt zur gleichen Zeit die selbe Operation aus, aber mit anderen Daten.

- MISD

  Zu Multiple Instruction Single Data gibt es keinen Rechnertyp. Pipeline – Rechner können in diese Klasse fallen.

- MIMD

  Zu Multiple Instruction Multiple Data gehören verteilte Rechner und Parallelrechner. Diese Systeme bestehen aus mehreren Prozessoren die mit eigenen Programmen und Daten arbeiten.

1.1.5 Multicomputer

Multicomputer sind nach Flynn MIMD – Strukturen. In einem verteilten System ist jeder Knoten ein Multicomputer, mit eigener CPU, Speicher und Ein- Ausgabeeinheit. Diese Multicomputer sind über ein Netzwerk miteinander verbunden und der Nachrichtenaustausch erfolgt über eine Rechner – Rechner Kommunikation. Die Übertragungsgeschwindigkeit des Netzwerks muss für das Datenaufkommen ausgelegt sein. Bei LAN – Systemen sind die gängigen Übertragungsraten bis zu 1GB/s. In der Automatisierungstechnik sind häufig Feldbussysteme im Einsatz mit Übertragungsraten bis zu 1MBit/s

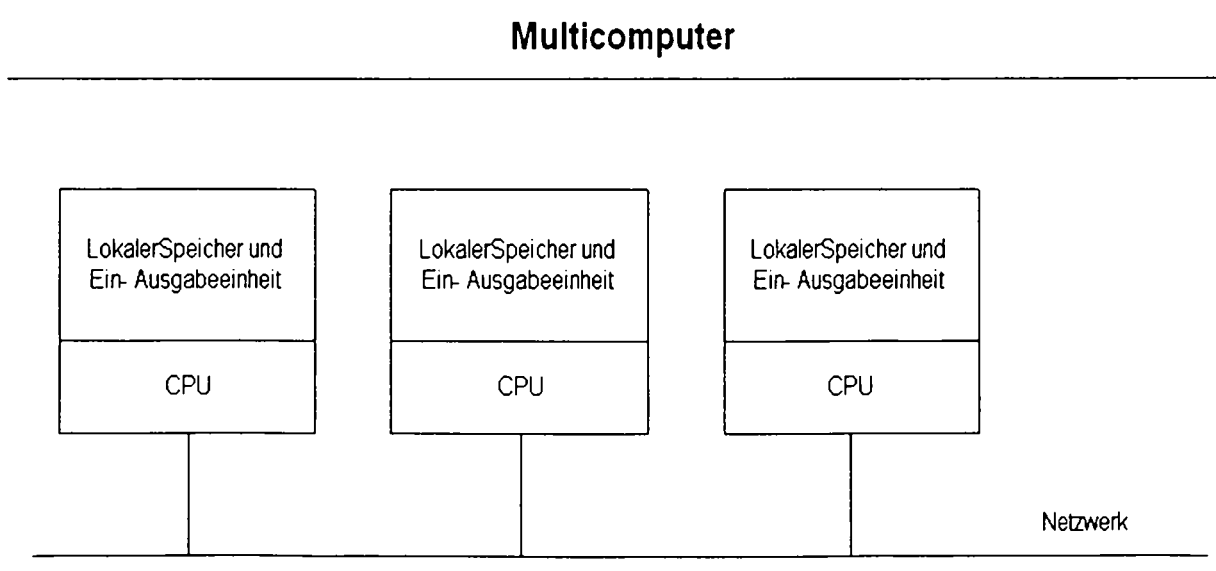


Abb.:1.9 Multicomputer

1.1.6 Anwendungen von verteilten Systemen

Ein verteiltes System ist eine Menge von Arbeitsplatzrechnern die über ein Netz (Ethernet) miteinander verbunden sind. Beispiele für solche Systeme sind:

1. Die Gesamtheit aller Internet – Knoten kann als verteiltes Netz betrachtet werden.
2. Das Paketradiosystem in der Amateurfunktechnik besteht aus einer Menge von Rechnern, die Europaweit verteilt sind und über Funk miteinander vernetzt sind.
3. Die durch einen Bus gekoppelten Prozessoren in einem Automobil (ABS, Motorsteuerung, automatisches Getriebe, Airbag, Navigationsrechner).
4. Die Vernetzung von Bargeldautomaten und die dazugehörigen Server.
5. Die über Busse und Netze verbundenen Rechner von Fertigungsstraßen und Transportsystemen.

Klassifikationsvorschlag für verteilte Systeme auf der Basis ihrer Anwendung:

- Systeme die nur zur Übertragung von Informationen dienen
- Systeme die Steuerungsaufgaben übernehmen
- Systeme die Steuerungsaufgaben und Regelungsaufgaben übernehmen

2 Stand der Technik

2.1 Entwicklung pneumatischer Transportsysteme

Sie haben eine entsprechend lange Entwicklungsgeschichte hinter sich. Bereits im Jahre 1857 hatten die beiden Ingenieure Thomas Webster Rammel und Latimer Clark die Idee zu einer Rohrpostanlage, die den Namen „Pneumatic Despatch Railway" bekam. Das Prinzip dieser ersten Rohrpostanlage ließen sie patentieren. Sie beschrieben dabei eine technische Einrichtung, mit deren Hilfe Blechbüchsen, die befüllt werden konnten, mit hoher Luftverdichtung oder- Verdünnung durch Rohre zu einem nahe gelegenen Ort verschickt werden konnten. Sechs Jahre später wurde diese Idee in die Realität umgesetzt und die erste unterirdische Rohrpostanlage der Welt wurde in London in Betrieb genommen. Diese Rohrpostanlage war Vorbild für viele andere wie beispielsweise das Rohrpostnetz in Berlin oder die Tower Tube in New York.

Die Voraussetzung für die „Pneumatic Despatch Railway" wurde aber bereits einige Jahre früher gelegt. Im Jahre 1810 konnte der dänische Ingenieur George Medhurst erstmalig erfolgreich atmosphärische Luft für industrielle und technische Zwecke einsetzen. Er hatte die Idee, die Luft aus den eisernen Röhren abzupumpen und so durch den erreichten Druckunterschied eine verwertbare Triebkraft zu bekommen. Er ist damit als Vater der pneumatischen Rohrpost anzusehen.

1818 verfolgte der englische Ingenieur Vallance die Idee weiter und versuchte Güter und Personen in großen tunnelartigen Gusseisenröhren von London nach Brighton zu befördern. Dieser Versuch misslang allerdings. Nach diesem Fehlschlag beschränkte man sich also auf den Transport von kleineren Poststücken mittels Luftdruck über Röhren mit kleinem Durchmesser und hatte damit die ersten Erfolge zu verzeichnen.

1852 wurde dann durch einen Franzosen eine Versuchsstrecke eingerichtet, die letztendlich ausschlaggebend für das Patent von Rammel und Clark waren.

Die ersten Rohrpostanlagen im heutigen Sinne hatten zunächst manuell zu stellende Weichen, später brachten Elektromagnete eine gewisse Bedienungserleicherung.

Mit dem Fortschritt der Steuerungstechnik – mit der Einführung des Relais - entstand die traditionellen pneumatischen Transportsysteme mit zentraler elektrischer Steuerung. Dies hatte zur Folge, dass der Verkabelungsaufwand enorm hoch war und die Bedienteile wenig komfortabel und daher komplex zu bedienen waren. Darüber hinaus war es mit diesen

Steuerungen bei weitem nicht möglich alle gewünschten bzw. notwendigen Funktionen zu realisieren.
Solche pneumatische Transportsysteme mit zentralen Steuerungen waren bis ca. 1975 weit verbreitet. Viele größere Firmen, Krankenhäuser, Kaufhäuser oder Banken hatten solche Anlagen im Einsatz. Ihre primäre Aufgabe bestand darin Lieferscheine, Rechnungen, Aufträge und dergleichen weiter zu leiten. Die Verfügbarkeit der Mikroprozessortechnik beeinflusste die Weiterentwicklung der pneumatischen Transportsysteme in zweierlei Hinsicht.

2.2 Herausforderungen und neue Chancen für pneumatische Transportsysteme

Mit der Weiterentwicklung der Netzwerktechnik und der damit verbundenen Verbreitung am Markt, verloren die alten Rohrpostanlagen einen Teil ihres Aufgabengebietes. Die ursprünglichen Transportaufgaben verschoben sich zunehmend zu EDV-Systemen, da Schriftstücke und Akten wesentlich leichter durch so genannte Workflow- Systeme verwaltet und weitergeleitet werden können. Das Einsatzgebiet von pneumatischen Transportsystemen hat sich dadurch geändert. Rohrpostanlagen werden heute überwiegend in Gebäuden eingesetzt um beispielsweise im Handel Geld oder Bücher zu transportieren, in Krankenhäusern zum internen Versand von Blutproben, Patientenproben, Röntgenbildern oder anderen Analysematerialien. Rohrpostanlagen in Warenhäusern dienen dazu, die einzelnen Kassen mit der Hauptkasse zu verbinden, um eingenommenes Bargeld abzuliefern oder Geld zu wechseln. In Industriebetrieben können damit auch gefährliche Materialen wie Chemikalien oder Öle aber auch Metalle befördert werden.

Außerdem wurden für den Transport von kleinen, handlichen Gegenständen verstärkt Boten eingesetzt. Auf die Dauer entwickelte sich bei den Betrieben jedoch ein Bewusstsein für die hohen Kosten dieser Lösung und so gibt es heute wieder ein reges Interesse für kostenoptimierte und zuverlässige Rohrpostanlagen.

Der Schlüssel für wettbewerbsfähige Rohrpostanlagen ist dabei die konsequente Weiterentwicklung der Steuerungstechnik unter Ausnützung der neuen Standards für Netzwerk- und Bussysteme.

Die Ablösung veralteter Steuerungen, die sehr komplex, umständlich und aufwändig waren, gestaltete sich nicht einfach. Beim Ersatz durch neuere Steuerungen zeigte sich, dass die ersten Konzepte wenig ausgereift, waren und sich in der realen Betriebssituation als sehr fehleranfällig erwiesen, da die Herausforderung durch externe Störungen nicht oder nur unzureichend berücksichtigt wurden.
Durch die Entstehung von kleinen, leistungsfähigen Mikroprozessoren konnten neue Konzepte für dezentrale Steuerungen erstellt und umgesetzt werden. Wegen fehlender Standards für die Kommunikation zwischen diesen trat ein Wildwuchs von nicht genormten Schnittstellen auf.
Nebenbei gab es durch die großen zu überbrückenden Entfernungen zwischen den einzelnen Steuerungen weitere Probleme. Durch die Reibung zwischen bewegter Büchse und dem feststehenden Rohr kam es zu extrem hohen elektrischen Störungen. Ein klagloses Funktionieren solcher Anlagen war damit nicht möglich. Hohe Ausfallszeiten der Anlagen konnten jedoch weder aus wirtschaftlicher Sicht noch aus Sicht der Benutzerakzeptanz in Kauf genommen werden.

Durch die rasche Entwicklung von Feldbussystemen in der Automatisierungstechnik war ein neuer Weg geschaffen, um zu genormten Standards zu kommen und so lag es nahe auch in der Rohrposttechnik alte Schnittstellen durch neue, moderne Feldbussysteme zu ersetzen.

Die in dieser Arbeit vorgestellten Steuerungssysteme sind ein Vorschlag für ein solches fehlertolerantes System. Ein Ansatz wäre es, Schnittstellen zwischen alten und neuen Systemen zu bilden und so die alten Systemen in die neuen einzubinden um sie dann schrittweise abzulösen. Diese Arbeit beschäftigt sich aber ausschließlich mit der Entwicklung eines neuen Gesamtsystems. Anbindung an alte Systeme sind darin nicht vorgesehen.

3 Fehlertolerante verteilte Rechnersysteme

3.1 Einleitung

In dieser Arbeit werden Steuerungen basierend auf verteilte Rechnersystemen behandelt, bei denen einzelne Komponenten auf Grund von Störungen ausfallen können, das Gesamtsystem aber ausfallssicher arbeitet. In solchen Systemen ergibt sich eine Großteil der Komplexität aus den spezifischen Umwelteinflüsse, die zu Störungen führen. Die Störungen in solchen Steuerungen setzen sich aus der Summe einzelner Störungen zusammen und können dabei erheblich werden.
Das Beispiel, der in dieser Arbeit ausgewählten Rohrpostanlage, ist ein typischer Vertreter solcher Systeme:
Wird eine Büchse durch das Rohr bewegt, so schwebt diese auf einem Luftpolster. Durch die Reibung zwischen dem Rohr, der Luft und der Büchse entstehen hohe elektrische Aufladungen (bis zu 40 kV). An Stößen, Bögen, Weichen kann es dann zu Entladungen und daraus resultierenden Störungen der Subsysteme kommen. Diese Fehler können weder lokalisiert werden noch sind sie vollständig behebbar. Durch geeignete Maßnahmen bei der Montage der Anlagen kann der Einfluss der Störungen wesentlich verringert werden. Der verbleibende Rest, der zu einem Fehlverhalten oder Ausfall des jeweiligen Subsystems führt, sollte durch einen gezielten Wiederanlauf der einzelnen Stationen beseitigt werden, ohne dass sich die Störung auf die Funktionalität der Anlage auswirkt.

Durch die stetige Weiterentwicklung der Speicherkapazität moderner Steuerungen ist es heutzutage möglich, softwaretechnisch viele Problemlösungen zu realisieren. Um diese Realisierungen sinnvoll umsetzen zu können, muss aber auch die Mechanik und speziell die Konzeption der Anlagen dafür ausgelegt sein.

3.1.1 Allgemeine Überlegung

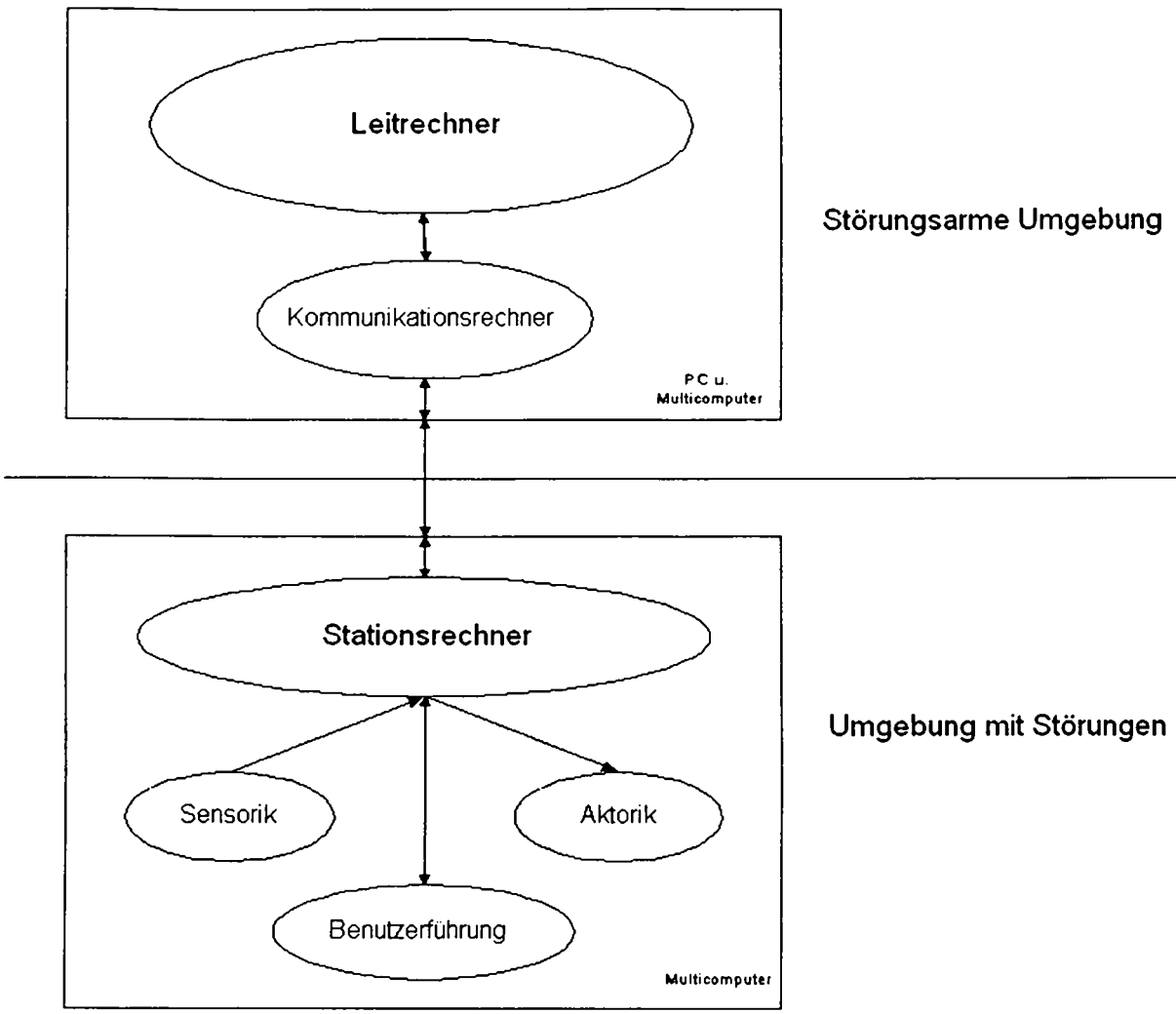


Abb.: 3.1 Abstrakte Anlagendarstellung

Abb. 3.1 zeigt die charakteristische Situation für das hier behandelte Steuerungssystem. Der Leitrechner befindet sich in der störungsarmen Umgebung. Für die Weiterbehandlung des Problems wird daher angenommen, dass der Leitrechner nicht ausfällt. Für Systeme für die diese Annahme nicht zulässig ist, müssen Konzepte mit verteilter Intelligenz in allen Subsystemen zur Anwendung kommen. Überlegungen zu dieser Situation sind im Abschnitt 3.3 beschrieben. Dies ist aber nicht Gegenstand dieser Arbeit.

Stationsrechner befinden sich in der störungsbehafteten Umgebung. Was zu spontanen Fehlfunktionen führen kann. Wie schon in der Problemstellung erwähnt, besteht die zentrale Aufgabenstellung nun darin, dass das verteilte System die als unvermeidlich angenommene Störung optimal bewältigt.

Diese Aufgabenstellung wird in Abb. 3.2 schematisch dargestellt.

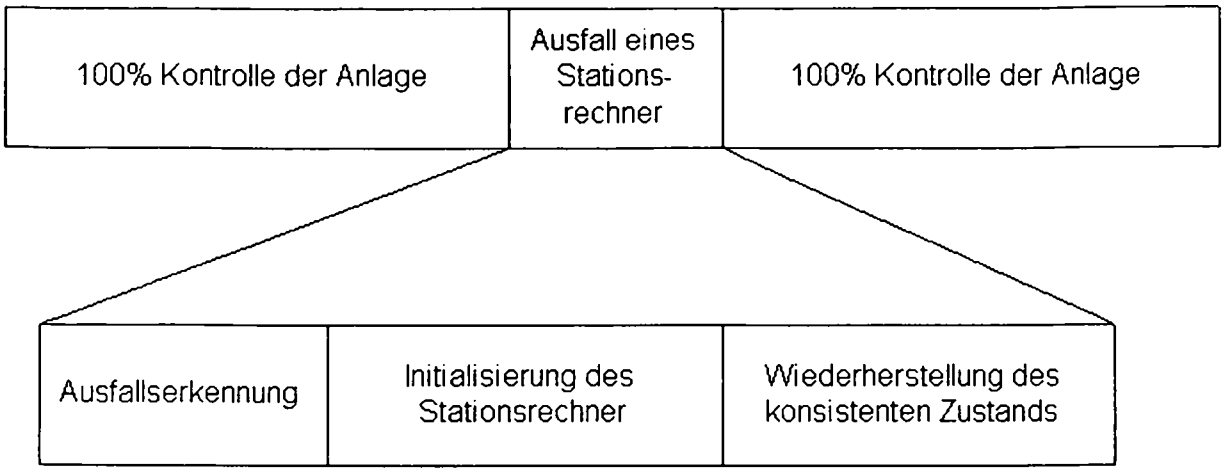


Abb.: 3.2 Bearbeitung von Fehlfunktionen

Im störungsfreien Betrieb (alle Systemkomponenten sind voll funktionsfähig) hat der Leitrechner bzw. die Steuerungssoftware die Systemfunktionen unter Kontrolle. Die Aktionen an allen Komponenten werden zentral mitgespeichert. Durch die Fehlfunktion eines Stationsrechners können unkontrollierte Zustände entstehen. Wichtigste Aufgabe des Systems ist daher, schnellstmöglich die Fehlfunktion zu erkennen. Ist eine solche erkannt durchläuft das System die beiden Schritte der Wiederanlauftechnik: (a) Initialisierung des Stationsrechners (Behebung der Fehlfunktion) und (b) die Widerherstellung eines konsistenten Zustands des gesamten Transportsystems. Ergebnis dieser Prozessschritte ist die Wiederherstellung des vollständig kontrollierten Zustands.

3.2 Erkennung und Behebung von Fehlfunktionen einzelner Komponenten

3.2.1 Erkennen eines Rechnerausfalls

Das automatische Erkennen eines Rechnerausfalls ist eine sehr komplexe Funktion. Eine Ausfallserkennung mit einem Watch-Dog, der an einer bestimmten Stelle im Programm angesprochen wird, ist auf jeden Fall nicht Ziel führend. Dieses Problem wurde von mir schon bei einem früheren Projekt (Hackschnitzelheizungssteuerung) festgestellt. Genauso wenig ist es Ziel führend wenn die Ausfallserkennung in Interrupt- Routinen passiert. Das eigentliche Programm befindet sich möglicherweise bereits in einem undefinierten Zustand, der Rechner kann aber bestimmte Interrupts immer noch funktionsgerecht abarbeiten. Um eine sichere

Ausfallserkennung zu implementieren, gäbe es die Möglichkeit die Watch-Dog Routine mit so genannten Meilensteinen zu verknüpfen.

3.2.1.1 WatchDog

Der Watch-Dog ist ein Hardwarezeitglied. Wird dieses Zeitglied aktiviert, so muss es in bestimmten Zeitintervallen synchronisiert (rückgesetzt) werden. Ist das nicht der Fall, läuft die Zeit ab und es löst am gegebenen Rechner einen Reset aus. Die meisten Mikroprozessoren haben einen integrierten Watch-Dog. Der integrierte Watch-Dog könnte eventuell den Nachteil haben, dass im Falle eines Prozessorausfalls auch der interne Watch-Dog ausfällt. Bei einem entsprechend EMV- gerechten Entwurf sollte das allerdings weitestgehend ausgeschlossen werden können. Eine weitere Möglichkeit besteht darin einen externen Watch-Dog zu verwenden. Dabei ist die Wahrscheinlichkeit wesentlich geringer, dass dieser ausfällt. Der Nachteil bei einem externen Watch-Dog ist der zusätzliche Port der benötigt wird, um den Watch-Dog zu synchronisieren.
Der Watch-Dog hat entweder eine fix vorgegebenes Zeitintervall oder er ist über eine Schnittstelle programmierbar (z.B. Interner Watch-Dog) wobei das Zeitintervall aus einem bestimmten Bereich wählbar ist. Innerhalb dieses Zeitintervalls müssen dann die Rücksetzimpulse erfolgen. Diese Impulse werden nur dann ausgegeben wenn der interne Programmablauf auch erfolgreich durchgeführt wurde. Dieser Programmablauf wird mit Hilfe von Meilensteinen überwacht.

3.2.1.2 Meilensteine

Die Meilensteine dienen zur Überwachung der einzelnen Programmabschnitte. Einem bestimmten Programmabschnitt wird ein Meilenstein zugeordnet. Wird dieser Programmabschnitt ausgeführt, so wird der laufende Meilenstein mit dem zugeordneten Meilenstein verglichen. Stimmen die beiden überein so kann mit sehr hoher Wahrscheinlichkeit angenommen werden, dass der Programmablauf in richtiger Reihenfolge und ohne Fehler erfolgt ist. Dieser Fehler tritt immer dann auf, wenn die CPU einen bestimmten Programmteil nicht ausführt, weil eine externe Störung aufgetreten ist. Diese Überwachung durch die Meilensteine ist auch in der Synchronisationsroutine für den Watch-Dog enthalten.

Beispiel:

```
#define MAXMS 16
#define WD_MS 0

// Deklaration einer globale Variablen
byte CurMs;
 byte ChPo;
void main () {
        // Deklaration
        ...
        //Initialisierung
        ...
        // laufenden Meilenstein mit der höchsten Nummer initialisieren
        CurMs = MAXMS ;
        ChPo = 0;
        ...
        // Watch –Dog synchronisieren
        // Laufenden Meilenstein um eins erhöhen und Modulo MAXMS rechnen
        // Durch die Modulo – Funktion nimmt der laufende Meilenstein (CurMs) die
        // 0 bis MAXMS – 1 an.
        CurMs = (CurMs + 1) mod MAXMS;
        // Stimmt der laufende Meilenstein mit dem erwarteten nicht überein,
        // so bewirkt die for – Schleife eine Endlosschleife und der Watch –Dog
        // wird nicht mehr synchronisiert.
        if (CurMs != WD_MS)  for (;;);
        // Watch –Dog synchronisieren
        WD = 1;
        if (ChPo != 0) {
                // Checkpointmeldung zum Leitrechner senden ! Alles OK !
                ....
                // Rücksetzen der Checkpointanforderung
                ChPo = 0;
        }
```

```
        //
        ...
} // Programmende main
```

In jedem zu prüfenden Programmabschnitt wird derselbe Algorithmus verwendet. Das Rücksetzen des Watch-Dog findet nur an bestimmten Stellen statt.
Die Programmüberwachung mit Watch-Dog und Meilensteinen muss im Kommunikationsrechner und in den Stationsrechnern erfolgen. Im Leitrechner kann auf den Einsatz von Meilensteinen verzichtet werden. Eine Watch-Dog-Karte ist auf jeden Fall sinnvoll.

3.2.2 Wiederaufsetzpunke festlegen

Für ein Wiederaufsetzen bei einem Rechnerausfall muss erstens der Ausfall erkannt werden und weiters müssen sämtliche Aktionen bekannt sein, die jetzt ausgeführt werden müssen um den Systemzustand und / oder Stationszustand zu erreichen, der ohne Rechnerausfall existiert hätte. Um den Ausfallszeitpunkt einzugrenzen, wurden so genannte Checkpoints eingeführt. In genau definierten Zeitintervallen wird zu jedem Stationsrechner eine Checkpoint- Meldung geschickt. Wir diese Meldung positiv quittiert, so heißt das, dass bei der Station alles in Ordnung ist. In der Station wird nach Empfang dieser Meldung eine Variable „ChPo“ gesetzt. Wird im Programm nun der Meilenstein 0 abgefragt, so wird geprüft ob diese Variable aktiviert ist. Wenn ja, wird die Checkpointmeldung an den Leitrechner gesendet. Im Leitrechner ist eine Zeitglied aktiviert, durch welches die Checkpointmeldung überwacht wird. Läuft dieses ab, so ist das ein sicheres Signal, dass mit dem Stationsrechner etwas nicht stimmt. Genauso ist die Checkpointmeldung mit den Meilensteinen in den Stationsrechner verknüpft. Die Quittierung der Checkpointmeldung erfolgt nur im Meilenstein 0.
Dieser Checkpoint ist eine unsigned Integer Variable mit 16 Bits. Der Wertebereich geht von 0 bis 65535. Das sind 65536 Checkpoints pro Station. Eine wichtige Größe ist das Zeitintervall zwischen den Checkpoints. Ist das Zeitintervall zu klein gewählt, so könnten die Checkpoints für eine Aktion nicht ausreichen. Bei viel zu großem Zeitintervall stellt sich anderseits die Frage, welche Aktionen müssen nun ausgeführt werden. In der Rohrposttechnik dauern Aktionen durchschnittlich 5 – 10 Sekunden. zum Beispiel das Stellen einer Weiche auf eine bestimmte Position. Nimmt man nun als Zeitintervall 5 Sekunden an, so kommt man mit 65536 Checkpoints 91 Stunden aus.

Jeder Stationsrechner sendet nach einem Reset automatisch eine Initialisierungsmeldung zum Leitrechner. Aufgrund dieser Meldung kann der Leitrechner auch einen Ausfall des Stationsrechners erkennen.
Alle Aktionen die nun ausgeführt werden, werden mit dieser Nummer verknüpft und in der Datenbank gespeichert. Aktionen können sich auf Stationen oder Linien beziehen. Beispielsweise werden bei einer Sendung alle Motorbewegungen einer Station oder Weiche in der Datenbank gespeichert. Fällt eine Station aus, wird über die Aktionsvariable die laufende Aktion ermittelt und über den letzten aktuellen Checkpoint kann der Ausfallszeitpunkt ermittelt werden. Alle Einstellungen die nach diesem Checkpoint vorgenommen wurden müssen neuerlich ausgeführt werden.

Tabelle 2: Speichermodell der Systemabläufe

Linienstatus	In welchen Zustand befindet sich die Line (Anlauf, Betrieb, Störung, Gesperrt, Wartung)
Gerätestatus	Ist ein Speicherbereich von 50 Byte der für jede Station eine etwas andere Struktur hat, in der die gesamten Geräteeinstellungen abgelegt sind.
Gerätenummer	Eindeutige Gerätenummer
Aktionsnummer	Für jede Einstellung die es an einer Station gibt, wird eine Nummer vergeben.
Displaytext	Gesamter Text der aktuell an der Station angezeigt wird
Checkpoint	Laufende Checkpoint
Aktion	Ist ein Speicherbereich von 50 Byte der für jede Station eine andere Struktur hat
SendId	Nummer der laufenden Sendung
Aktionszeit	Zeitpunkt des Eintrags

3.2.3 Anwendungsbeispiel

An einer Station soll eine Büchse versendet werden. Der Benutzer gibt an der Station die gewünschte Adresse ein. Anschließend wird die Büchse in den Versandspeicher gelegt. Dadurch wird der Speicherkontakt betätigt und eine Statusmeldung wird zum Leitrechner übermittelt, dass eine Büchse zur Versendung ansteht. Der Leitrechner ermittelt nun aus der eingegebenen Adresse alle benötigten Informationen (existiert die Adresse, ist diese Adresse frei, etc.). Wenn diese Informationen OK sind, so wird festgestellt ob der Weg dorthin auch in Ordnung ist und anschließend wird die Sendung in einer Tabelle der Datenbank gespeichert. Der Benutzer bekommt am Display das OK zur Sendung. Stürzt der Rechner durch eine Störung ab, so geht er durch einen Watch-Dog-Reset wieder in Betrieb und am Display wird

eine Standardmeldung angezeigt. Man weiß nun nicht was mit der Büchse, die im Versandspeicher steht, passieren soll. Die Wiederanlauftechnik muss nun erkennen, dass der Rechner ausgefallen ist und welche Einstellungen an dieser Station durchzuführen sind. In diesem Fall soll der gesamte Text für die letzte Sendung an die Station gesendet werden. Somit hat der Benutzer wieder alle Informationen. Der Benutzer bemerkt von den Systemaktionen im Hintergrund jedoch kaum etwas, außer ein kurzzeitig nervöses Blinken der Anzeige.

3.3 Herstellung eines konsistenten Zustands im verteilten System

Für die Wahl eines geeigneten Lösungsweges waren folgende Kriterien von Bedeutung

- Zeitaufwand bis zur Behebung der Fehlfunktion,
- Benutzerfreundlichkeit,
- Zuverlässigkeit sowie
- Realisierungsaufwand.

In den folgenden drei Abschnitten werden die im Rahmen des Projekts evaluierten Lösungswege in Reihenfolge ihrer steuerungstechnischen Komplexität vorgestellt und diskutiert.

3.3.1 Lösungsweg 1: Vollständiger Systemreset

Die Grundidee dieses Ansatzes ist, dass nach jeder erkannten Fehlfunktion das gesamte Transportsystem in einen Grundzustand gebracht wird, der demjenigen beim Einschalten entspricht. Wesentlich ist, dass die Büchsen nicht am gewünschten Zielort ausgeliefert werden, sondern an einer Notstation ausgeschleust werden.

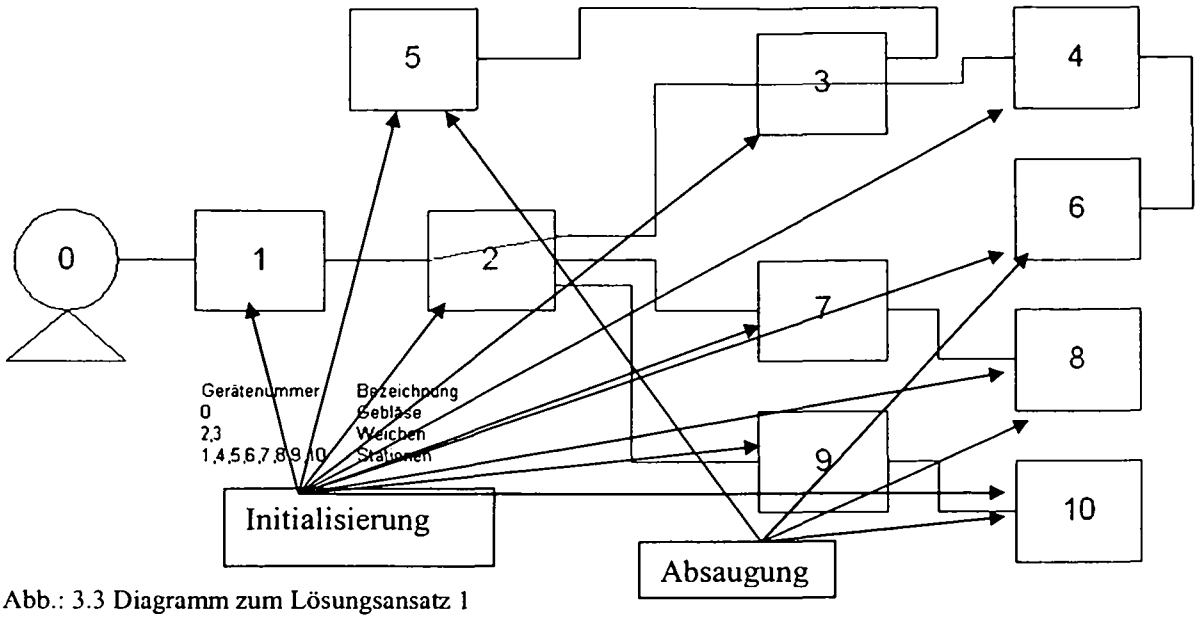


Abb.: 3.3 Diagramm zum Lösungsansatz 1

Abbildung 3.3 zeigt schematisch die Vorgänge beim Systemreset. Alle Stationen und Weichen werden neu initialisiert und befinden sich danach im Grundzustand. Das System hat keine Historie mehr. Danach beginnt ein standardisierter Suchvorgang: Ausgehend von jeweils einer Endstation (in Abbildung 3.3 zum Beispiel Werden die Teillinien mit den Endstation Nr. 6, Nr. 5, Nr. 8, Nr. 10 abgesaugt) werden nacheinander alle Linien in Richtung Gebläse (Gerät Nr. 0) angesaugt, bis eine eventuell in dieser Linie befindliche Büchse gefunden wird und zur Notstation transportiert wird.

Dieser Lösungsweg ist steuerungstechnisch sehr einfach, da er immer nach dem gleichen Ablauf verläuft. Der Zeitaufwand bis zur Behebung der Störung ist in allen Fällen relativ hoch. Der Normalbetrieb für die Anwender wird erheblich gestört; die Büchsen müssen erneut verschickt werden (der Zielort der Büchsen geht verloren).

3.3.2 Lösungsweg 2: Minimierter Systemreset

Die Grundidee dieses Ansatzes ist, dass nach jeder erkannten Fehlfunktion nur der betroffene Stationsrechner initialisiert wird. Das Transportsystem als Ganzes wird dagegen nicht in einen Grundzustand gebracht. Danach wird die Anlage abgesaugt und die Büchsen an einer Notstation ausgeschleust.

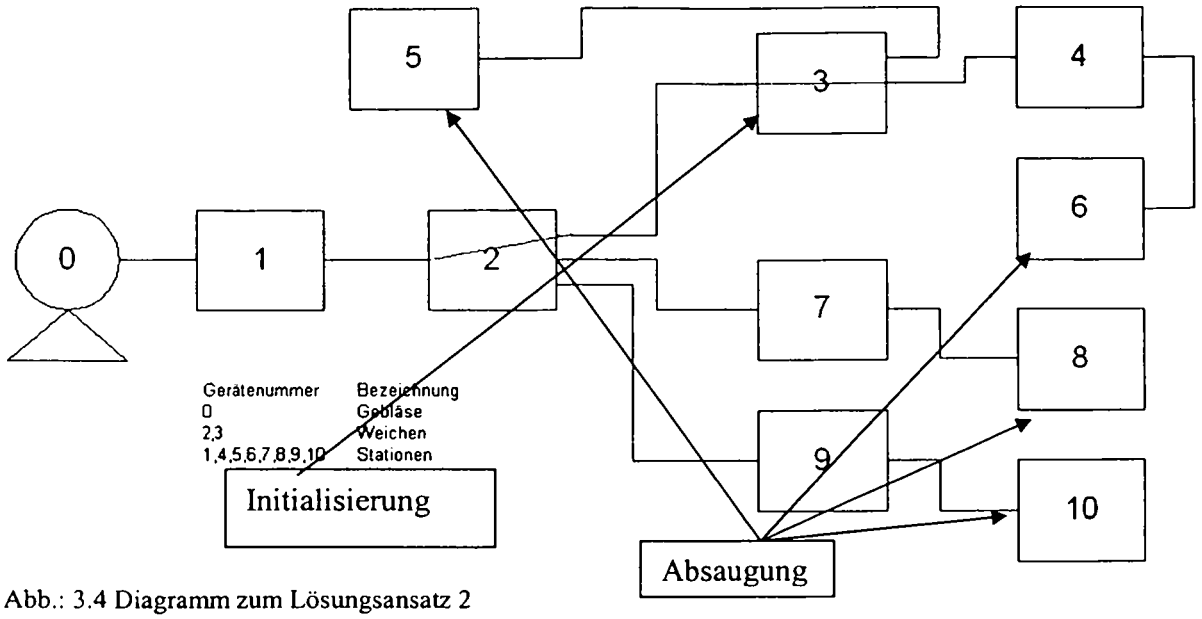


Abb.: 3.4 Diagramm zum Lösungsansatz 2

Abbildung 3.4 verdeutlicht die Abläufe bei diesem Verfahren. Der wesentliche Zeitvorteil besteht darin, dass nur ein Stationsrechner (Nr. 2) initialisiert werden muss. Ansonsten sind die Abläufe gleich. Wie in Abbildung 3.4 dargestellt werden die Teillinien mit den Endstationen (Nr. 5, Nr. 6, Nr. 8, Nr. 10) der Reihe nach abgesaugt.
Dieser Lösungsweg ist steuerungstechnisch ebenfalls einfach, aber gegenüber dem Lösungsweg 1 schon zeitoptimiert. Der Normalbetrieb für die Anwender wird gestört; die Büchsen müssen erneut verschickt werden (der Zielort der Büchsen geht verloren). Die Unterbrechung ist aber merklich kürzer.

3.3.3 Lösungsweg 3: Die Suchstrategie mit maximaler Vorinformation

Der Grundgedanke bei diesem Lösungsweg war, alle zuverlässigen Informationen über das System (alle Aktionen die bis zum letzten Checkpoint vor Ausfall einer Komponente mitgeschrieben wurden) für die Suchstrategie zu verwenden. Wie bei Lösungsweg 2 wird nach jeder erkannten Fehlfunktion nur der betroffene Stationsrechner initialisiert. Aufgrund der Berücksichtigung der Vorinformation wird die Sendung lokalisiert und an den Zielort ausgeliefert. Die konkreten Schritte zu Herstellung eines konsistenten Zustands werden für

jeden neuen Fall vom Leitrechner neu ermittelt.

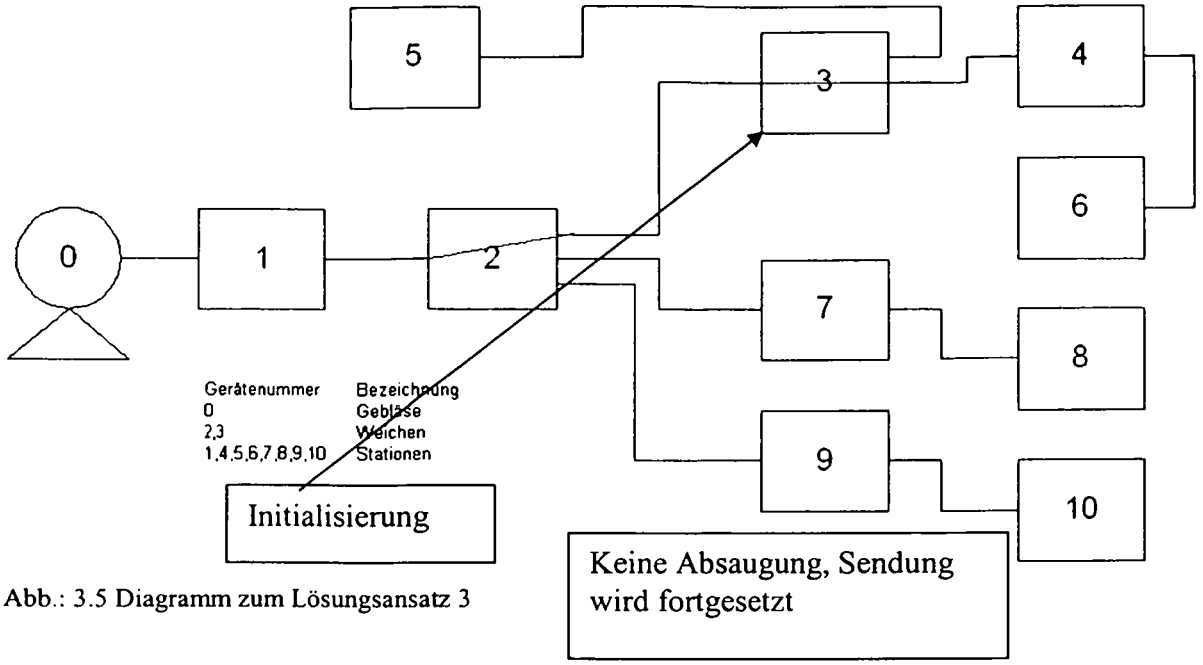


Abb.: 3.5 Diagramm zum Lösungsansatz 3

Abbildung 3.5 zeigt das konkrete Beispiel für den Ausfall des Geräts 3 bei einer Versendung von Gerät 5 nach Gerät 6. Durch die zentral verfügbare Systemhistorie lässt sich die Position der Büchse gut eingrenzen. Nach Neuinitialisierung des Geräts 3 kann die Sendung fortgesetzt werden und die Büchse direkt am Gerät 6 ausgeliefert werden. Es geht daher keine Büchse bei einer gestörten Sendung verloren.

Dieser Lösungsweg ist steuerungstechnisch deutlich komplexer und kann so nur realisiert werden, wenn sich die Fehlfunktion auf einzelne Komponenten beschränkt. Der Normalbetrieb für die Anwender wird nahezu nicht gestört; die Büchsen werden direkt zum Zielort zugestellt. Fehlfunktionen von Komponenten außerhalb des aktuellen Transportsweges sind unkritisch und werden parallel automatisch behoben. Mehrfach-Ausfälle von Komponenten entlang eines Transportweges sind äußerst selten; wenn sie auftreten, ist dies ein Hinweis auf ein systematisches Problem mit der Installation (typischerweise Stromversorgung), das prinzipiell behoben werden muss.

Lösungsweg 3 wurde in der an die Benutzer übergebenen Version implementiert.

3.4 Lösungsansätze für verteilte System in vollständig Störungsbehafteter Umgebung

3.4.1 Redundanzlösung

Im Bereich industrieller Leit-, Schutz-, Steuerungs- und Regelungssysteme geht man davon aus, dass beim Einsatz von Rechnern in Industrie, Verkehr oder Handel sowohl Verfügbarkeit (kein Ausfall der Strecke) als auch Sicherheit (kein Unfall) gegenüber herkömmlichen Lösungen nicht zurückstehen dürfen.

Verfügbarkeit und Sicherheit können zwar durch eine hohe Zuverlässigkeit der Elemente gesteigert werden; dem sind jedoch wirtschaftliche und technische Grenzen gesetzt. Dort, wo die Zuverlässigkeit nicht ausreicht, wird Redundanz eingesetzt, d. h. Betriebsmittel, die ohne Berücksichtigung von Ausfällen nicht notwendig wären [22].

Fehlertolerante Rechner besitzen Redundanz, um ihre eigenen Ausfälle überbrücken zu können oder um sich bei Ausfällen zumindest definiert zu verhalten. Diese Rechner sind heute noch wenig verbreitet. Dies liegt weniger an der technischen Machbarkeit als am Verhältnis der Kosten dieser Rechner zum erbrachten Gewinn an Verfügbarkeit oder Sicherheit.
Redundanz verursacht nicht nur höhere Anschaffungskosten, sondern auch höhere Unterhaltskosten – eine verdoppelte Steuerung fällt zunächst zweimal häufiger aus als eine einfache Steuerung. Die Redundanz lässt auch die Verlässlichkeit nicht ins Unendliche steigen, sie ist auch kein Ersatz für Qualität[23].

In Anwendungen, die eine hohe Verfügbarkeit verlangen, müssen sich fehlertolerante Rechner auszahlen durch die Produktionsausfallkosten, die sie ersparen. In Anwendungen, die eine hohe Sicherheit erfordern, werden fehlertolerante Rechner meist durch die Betriebsvorschriften aufgezwungen. Dort müssen sie sich bezahlt machen durch die Versicherungsprämien, die eingespart werden.

Eine Redundanzlösung wäre in diesem Konzept nach Abb. 3.1 nur in der Leitebene einsetzbar. Doch das Verhältnis der Kosten zum Nutzen ist nicht akzeptabel und daher ist dieses Lösungskonzept nicht weiter verfolgt worden.

3.4.2 Verteilte Intelligenz

Verteilte Intelligenz ist auch mit künstlicher Intelligenz zu vergleichen. In diesem Fall, müssten alle Rechner des Systems gleich aufgebaut sein. Konkret würde das heißen, dass es nach Abb. 3.1 nur eine Stationsrechnerebene gibt. Die einzelnen Funktionen werden auf irgendeinem Rechner im System ausgeführt und bei dem Auftreten einer Fehlfunktion auf einem Rechner übernimmt ein anderer Rechner des Systems dessen Funktionen.

Die Komplexität dieser Lösung steigt mit jedem weiteren Stationsrechner. Weiters sind die Kosten sehr hoch, da jeder Stationsrechner nahezu identisch mit der Ausstattung eines Leitrechners sein muss. Auch in diesem Fall ist eine Realisierung aus Kostengründen nicht möglich

3.5 *Ein neues Steuerungskonzept*

3.5.1 Steuerungsübersicht

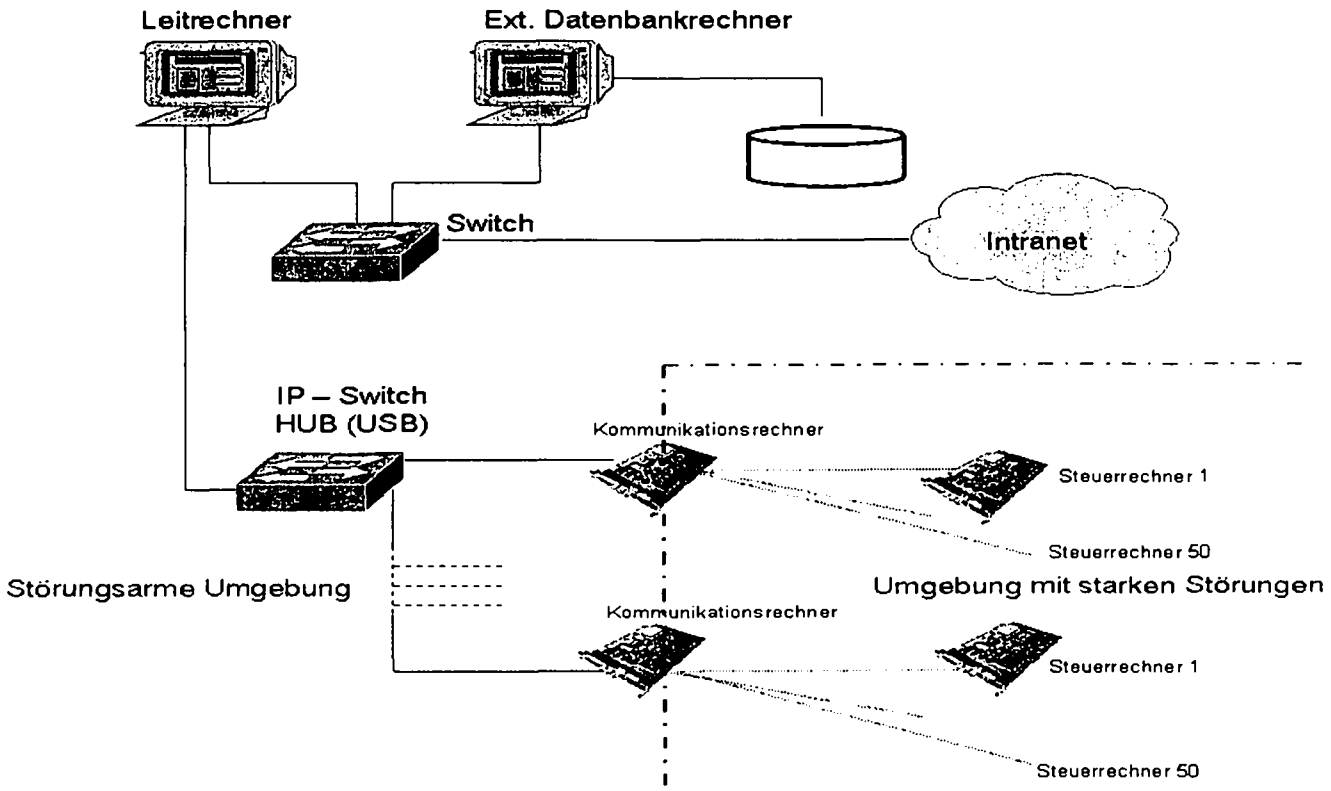


Abb.: 4.2 Steuerungsübersicht

Die strichpunktierte rote Linie in der Anlagenübersicht zeigt dass die Anlage in zwei Umgebungen getrennt ist. Die störungsarme Umgebung entspricht etwa einer Büroumgebung. In dieser Umgebung können handelsübliche Komponenten eingesetzt werden. Die restliche Umgebung sind beliebige Bereiche einer Fabrik. Die Komponenten für diese Umgebung sind speziell angepasste Entwicklungen. Die starken Störungen die auf die Komponenten einwirken, fordern schon bei der Entwicklung wesentlich höhere (bessere) EMV- Werte einzuhalten, als die Norm vorschreibt.

3.5.2 Softwarestruktur

Die gesamte Softwarestruktur kann man in sechs Blöcke zusammenfassen. Bestimmte Blöcke (z.B. Statusauswertung, im Normalzustand die Sendungsverarbeitung) werden immer durchlaufen, die restlichen Blöcke nur in Abhängigkeit des Anlagenzustands.

- Der Block „Initialisierung Leitrechner" wird nur beim Programmstart ausgeführt.
- Der Block „Initialisierung der Stationsrechner" wird nur beim Programmstart ausgeführt.
- Der Block „Knotenrechner aktivieren" wird nach einer Fehlfunktion eines Stationsrechner aufgerufen.
- Der Block „Anlaufprogramm" wird nur dann aufgerufen, wenn nach einer Fehlfunktion nicht sichergestellt ist, dass sich in der Anlage keine Büchsen mehr befinden.
- Der Block „Sendungen verarbeiten" ist jener Block der im ordnungsgemäßen Zustand der Anlage immer durchlaufen wird.

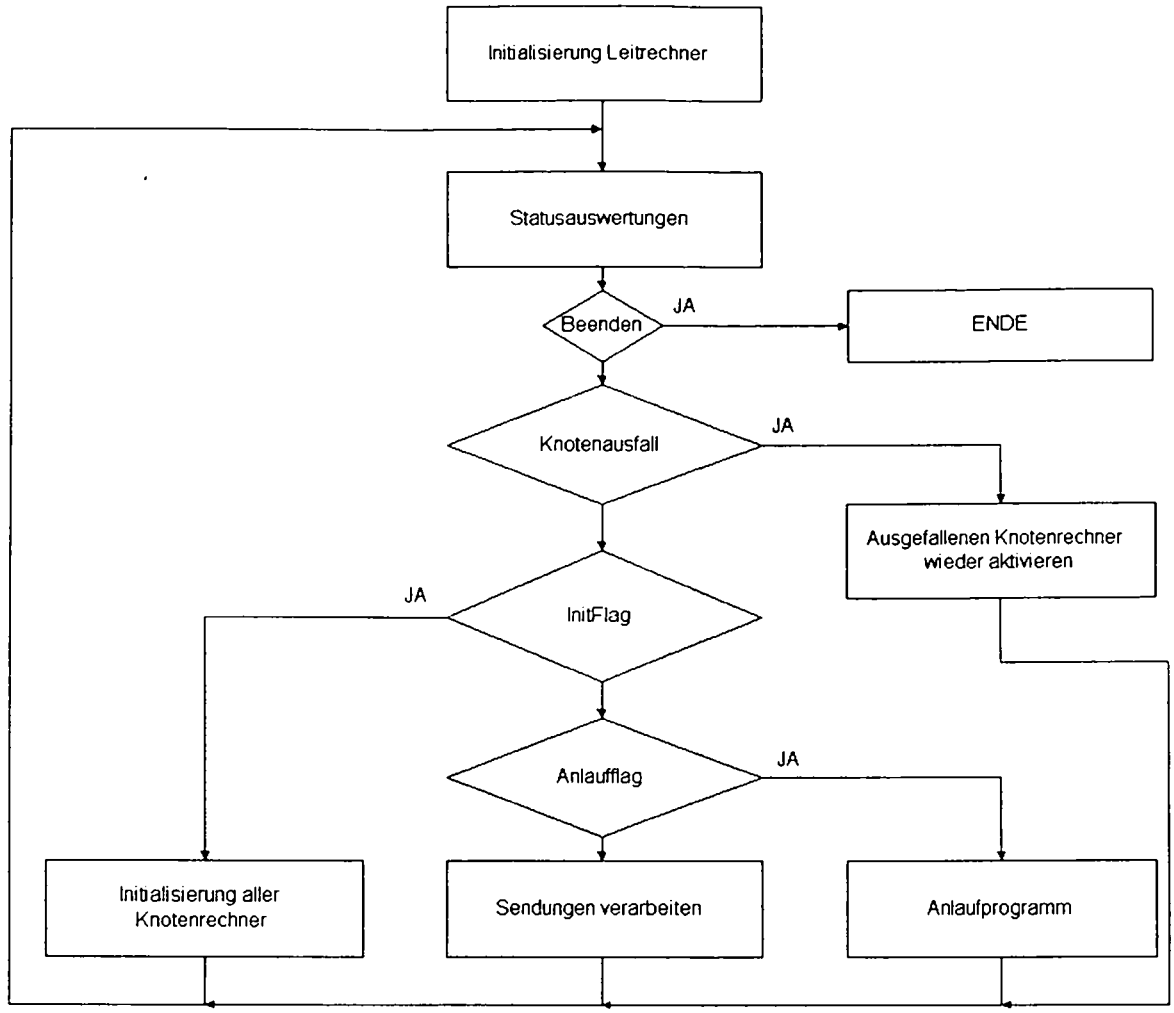


Abb.: 4.3 Flussdiagramm

3.5.2.1 Initialisierung des Leitrechners

Bei dem Vorgang „Initialisierung des Leitrechners" werden interne Variablen initialisiert, und die Verbindung zur Datenbank aufgebaut. Aus der Datenbank werden nun alle relevanten Daten vom Anlagenaufbau eingelesen.

Nun kann der Leitrechner die Verbindungen zu den Kommunikationsrechnern aufbauen und den Programmablauf fortführen.

3.5.2.2 Initialisierung aller Knotenrechner

Die Initialisierung der Anlage ist die erste Aktion die nach dem Anlagen- / Programmstart ausgeführt wird. Es erfolgt eine Überprüfung der Kommunikation mit allen Geräten. Nach erfolgreicher Überprüfung derselben stehen im Leitrechner dann bereits alle aktuellen Zustände der einzelnen Stations- und Weichenrechner zur Verfügung. Diese Zustände werden nun nach verschiedenen Gesichtspunkten ausgewertet und je nach Bedarf versucht sie

sofort in den Ausgangszustand zu bringen oder entsprechend koordinierte Maßnahmen zur Erreichung dieser durchzuführen.
Für jede Station ist eine bestimmte Grundstellung definiert in der sie sich befinden sollte oder in die sie gebracht werden muss. Eine Rohrschaltermeldung in einer Station könnte ein defekter Rohrschalter sein oder eine Büchse, die in der Station verklemmt ist. Wenn es eine verklemmte Büchse ist, so soll nach dem Anlaufprogramm dieser Fehler behoben sein. Ist es ein fehlerhafter Rohrschalter, muss dieser gewechselt werden.
Nach Abschluss der erfolgreichen Initialisierung ist sichergestellt, dass die Kommunikation zu den Stationen funktioniert und dass alle Aktoren in den Stationen ansprechbar und betriebsbereit sind. Die gesamte Linie, d. h. alle Stationen dieser Linie wurden in eine definierte Stellung gebracht und in den Anzeigen steht eine Standardmeldung mit aktuellem Datum und Uhrzeit. Man sagt, die Linie befindet sich in Grundstellung. Zu diesem Zeitpunkt sollte es in der Anlage keine gröberen Störungen geben, da die Aktionen, die durch die Initialisierung ausgelöst wurden, bereits abgeschlossen sind und da die Anlage keinen Betrieb zulässt ist dies der ideale Zeitpunkt zum Starten der Checkpointüberwachung. Der Leitrechner sendet zu jedem Stationsrechner eine Checkpointmeldung und dieser quittiert sie. Mit der Quittierung wird die Checkpointnummer mit dem Initialisierungsstatus jeder Station in der Datenbank gespeichert. Von nun an wird nach bestimmten Zeitintervall immer wieder ein neuer Checkpoint generiert und verwaltet. Am Ende der Initialisierungsroutine erfolgt der Start des Anlaufprogramms.

3.5.2.3 Anlaufprogramm

Mit dem Anlaufprogramm werden Büchsen, die sich im System befinden, lokalisiert und versucht, sie zu einer Notstation zu transportieren. Das Anlaufprogramm baut von jeder Endstation (Blatt) eine spezielle Sendung zu Notstation auf. Bei diesem Sendeablauf wird kein Einschleusvorgang vorgenommen. Zusätzlich wird für diese Sendung ein Zeitglied gesetzt. Spricht innerhalb dieser Zeit kein Rohrschalter an, wird die Sendung abgebrochen. Sonst wird diese Sendung zu Ende geführt und die Büchse an der Notstation ausgegeben. Für jede Linie muss eine Notstation definiert sein. An der Notstation muss die Möglichkeit bestehen, mehrere Büchsen auszuschleusen um einen Anlagenstillstand wegen Verklemmung zu vermeiden. Nach dem Anlaufprogramm geht der Zustand der Anlage in Betrieb über.

3.5.2.4 Status und Checkpoints bearbeiten

Statusmeldungen werden automatisch in bestimmten Zeitintervallen von den Stationsrechnern an den Leitrechner übertragen. Im Leitrechner wird der neue Status mit dem alten verglichen. Treten Statusänderungen auf, so werden die zugehörigen auszuführenden Aktionen in Tabellen in einer Datenbank gespeichert. Die Abarbeitung dieser Aktionen wird in einem anderen Programmbereich durchgeführt.
Die Statusmeldungen werden im Leitrechner für jede Station mit einem Zeitglied überwacht und bei einem Zeitgliedablauf fordert der Leitrechner den Status vom Knotenrechner an.
Die Generierung der Checkpoints erfolgt ebenfalls über ein Zeitglied. Nach Zeitablauf wird eine neue Checkpointnummer generiert und zu allen Stationen eine Checkpoint – Meldung übertragen. Nach der Checkpoint – Quittierung von der Station wird der Checkpoint für diese Station aktiviert.

3.5.2.5 Sendungen bearbeiten

Der Benutzer stellt an der Station über das Bedienpult die gewünscht Adresse ein und gibt anschließend die Büchse in der Versandspeicher der Station. Dieser Vorgang bewirkt in der Station eine Statusänderung (Speicherkontakt „SPK" spricht an), die zum Leitrechner übertragen wird und in diesem ausgewertet wird. Der Leitrechner prüft nun die Adresse auf Gültigkeit und alle für die Sendung benötigten Geräte (Linien, Stationen, Weichen, etc) auf Funktionalität. Die positive Erledigung dieser Anfragen bewirkt, dass die Sendung gespeichert wird und der Benutzer die entsprechenden Informationen am Display sieht. Der Transport der Sendung erfolgt zeitverzögert. Sollte der Benutzer sich in der Adresse geirrt haben, kann er die Büchse herausnehmen und der Sendevorgang wird automatisch abgebrochen.
Entfernt der Benutzer die Büchse wenn der Sendevorgang bereits begonnen hat, wird eine Leersendung durchgeführt, die im Anschluss ein Anlaufprogramm auslöst. Alle eingespeicherten Sendungen der anderen Stationen sind dadurch nicht betroffen. Die Sendungsausführung verzögert sich um die Dauer des Anlaufprogramms.

3.5.2.6 Sendung starten

Eine Sendung ist immer ein abgeschlossener Büchsentransport innerhalb einer Linie. Für die Ausführung der Sendung wird vom Zielgerät zum Gebläse ein Weg gesucht, ebenso wird ein

Weg vom Sendegerät zum Gebläse gesucht. Aus der Schnittmenge dieser beiden Wege ergeben sich folgende Abläufe.

- Zielgerät und Sendegerät sind gleich
  Dieser Fall kann bereits bei der Sendungseinstellung unterbunden werden. Bei manchen Stationen wird diese Sendung (Eigenwahl) für Wartungszwecke (z.B. Einstellen des Bremsrohrschalters) zugelassen.
- Zielgerät und Sendegerät befinden sich im gleichen Strang und das Zielgerät ist näher am Gebläse. Hier wird die Büchse zum Ziel gesaugt.
- Zielgerät und Sendegerät befinden sich im gleichen Strang und das Sendegerät ist näher am Gebläse. Hier wird die Büchse zum Ziel gedrückt.
- Zielgerät und Sendegerät sind in verschiedenen Strängen. Die Büchse muss zuerst in Richtung Gebläse gesaugt werden bis sie die Weiche erreicht hat, die diese beiden Stränge verbindet (Umsteuergerät). Nach dem Umsteuern der Weiche (Einstellung auf den neuen Strang) wird das Gebläse auf Druck geschaltet und die Büchse Richtung Zielgerät transportiert.

Nach abgeschlossener Wegesuche werden die Weichen in die richtige Position gebracht und die Büchse an der Sendestation ins System eingeschleust. Mit dem Start des Gebläses beginnt nun der Transport der Büchse im Rohrsystem. Dieser Transport wird mit Rohrschaltern überwacht und gesteuert. Nach Erreichen der Zielstation wird die Büchse aus dem System an der Zielstation ausgeschleust.
Die beiden Funktionen „Ein- und Ausschleusen einer Büchse" sind je nach Stationstyp unterschiedlich implementiert und müssen gemäß der Stationsbeschreibung der Hersteller gehandhabt werden.

3.5.2.7 Umleitung und Sperren einer Adresse

Alle Geräte haben eine eindeutige Gerätenummer. Diesen Gerätenummern sind – wie oben beschrieben – Adressen zugeordnet. Die Versendung von Büchsen erfolgt über Adressen. Wenn zu bestimmten Zeiten Stationen nicht besetzt sind, kann diese Adresse gesperrt werden oder zu einer anderen Adresse (Abteilung) umgeleitet werden. Die Aktivierung dieser Dienste kann direkt am Leitrechner oder an den Stationsrechnern erfolgen.
An den Stationsrechnern ist dafür zusätzlich ein Passwort erforderlich.

3.5.3 Integration der Steuerung in die bestehende Telefonanlage

Die Verfügbarkeit über Anlagen wird in unserer hektischen Zeit immer wichtiger. Auch mit der besten und tollsten Steuerung ist es nicht immer möglich alle Fehler zu beseitigen. Damit die Ausfallszeit bei einem Fehler möglichst kurz gehalten werden kann, wird vom Leitrechner eine Störungsnummer angerufen und dem Wartungspersonal die Fehlermeldung sprachlich übermittelt. Das Wartungspersonal hat somit sämtliche Informationen über dem Fehler und kann direkt zu der fehlerhaften Station gehen. Diese Sprachübermittlung wird auch für Betriebsmeldungen verwendet. Versorgt eine Station mehrere Abteilungen oder Büros, so wurde bis jetzt die Aufteilung der Büchsen mit Ankunftssignalen gelöst. Die aufwändige Montage von Ankunftssignalen wird durch den Eintrag einer Telefonnummer ersetzt. Der Leitrechner ruft diese Telefonnummer an und übermittelt nach dem Melden des Teilnehmers den Text. Meldet sich der Teilnehmer innerhalb einer bestimmten Zeit nicht so wird der Anruf beendet und nach einiger Zeit wiederholt. Die Anzahl der Wiederholungen kann eingestellt werden.

3.5.4 Leitrechner

Das gesamte System kann als Einheit betrachtet werden. Der Leitrechner steuert und kontrolliert das gesamte System. Wird ein Befehl abgesetzt, bleibt dem Benutzer eigentlich verborgen in welchen Teilen des Systems dieser Befehl Auswirkungen hat. Alle Aktionen die im System anfallen, werden im Leitrechner zentral erfasst, verarbeitet und protokolliert. Die Protokollierung erfolgt in einer Datenbank. Eine lückenlose Speicherung der gesamten Vorgänge wird immer wichtiger, um den Betreiber von Rohrpostanlagen effektiv immer mehr Kennzahlen über die Anlage liefern zu können und so einen entscheidenden Schritt in Richtung Qualitätssicherung und -überwachung zu implementieren. Typische Kennzahlen sind Sendungen pro Tag, Woche und Monat, Sendungen zu einer Station oder Sendungen von einer Station. Sehr wichtig sind auch Kennzahlen für Wartungen. In welchen Abständen verreibt eine Station oder muss ein Rohrschalter gewechselt werden. Alle diese Kennzahlen können über die Datenbank ermittelt werden. Der Betreiber hat auch die Möglichkeit sich eigene Auswertungen zu erstellen.

3.5.5 Kommunikationsrechner

Der Kommunikationsrechner hat eine Schlüsselfunktion in diesem System. Er dient als Trennstelle zwischen störungsloser und störungsbehafteter Umgebung. Dieser Rechner sollte

in der Anlage so positioniert werden, dass die wesentlichen Störeinflüsse nur mehr über die verschiedenen Verbindungskabeln kommen können. Das heißt, dass der Rechner in einem bestimmten Abstand vom Rohrsystem und weiteren Anlagenteilen montiert werden muss. Auch die Verkabelung sollte hier extra geführt werden um unnötige Störungen, z. B. durch Leitungsübersprechen, zu vermeiden. Der Kommunikationsrechner wird mit einem Watch-Dog auf Funktionalität überwacht, ist aber nicht in das Wiederanlaufprogramm eingebunden. Falls er ausfallen sollte, so wird über den Watch-Dog ein Reset ausgelöst. Der Rechner wird neu initialisiert. D. h. alle Daten die in den Puffern gespeichert sind gehen verloren. Die dabei auftretenden Fehler wirken sich nur auf Sendungen aus, die gerade eingespeichert werden. Alle anderen Funktionalitäten die vom Leitrechner aus gesteuert werden, arbeiten mit Quittierungen und werden wiederholt wenn die Quittierung fehlt.
Für jede Rohrpostlinie (Linie siehe Punkt 4.1.1) ist ein eigener Kommunikationsrechner vorgesehen und dieser übernimmt für die Linie die Routerfunktionen. Der Kommunikationsrechner übernimmt die Daten von der USB- oder Ethernetschnittstelle und verschickt diese über den CAN- Bus zu den einzelnen Geräten. Genauso werden sämtliche Daten die über den CAN- Bus empfangen werden in einem Puffer zwischengespeichert und anschließend blockweise zum Leitrechner übertragen.

3.5.6 Stationsrechner

In jedem Rohrpostgerät ist ein Stationsrechner positioniert. Der Stationsrechner ist so ausgelegt, dass er fast alle Funktionen der bekannten Rohrpostgeräte abdeckt. In Stationen ist ein Stationsrechner mit einer zusätzlichen Bedieneinheit (Tastatur und Display) montiert. Bei den meisten anderen Geräten (Weichen, Sondergeräte, ...) ist eine Bedieneinheit nicht erforderlich.
Die Bedieneinheit ist ein fertiger Zukaufsteil und wird über ein SPI- Interface angesteuert. Pro Station können maximal zwei Gleichstrommotore angesteuert werden. Die Motorspannung beträgt 24 Volt. Der maximale Motorstrom kann 3A sein, ein kleinerer Motorstrom kann aber eingestellt werden. Der momentane Spitzenstrom beim Einschalten oder ein höherer Motorstrom z.B. bei einer Verklemmung wird mit einer dynamischen Stromüberwachung begrenzt. Eine Komparatorschaltung vergleicht den eingestellten, maximal zulässigen Strom mit dem tatsächlich fließenden Strom und schaltet den Motor ab, wenn der eingestellte Wert überschritten wird. Mit Hilfe eines Taktgebers wird der Motor wieder eingeschaltet. Der Motor wir also in einem bestimmten Rhythmus getaktet. Der

gemittelte Motorstrom ist kleiner als der maximal eingestellte Strom und es sollte dadurch zu keiner Überhitzung oder Zerstörung der Motortreiber kommen.
Ingesamt gibt es pro Stationsrechner 11 Transistorausgänge. Bei drei Ausgängen beträgt der maximale Strom 1A, die restlichen Ausgänge können 0,5A schalten. Die 0,5A Ausgänge werden meistens zum Schalten von Signallampen verwendet. Um den Einschaltstrom bei den Lampen zu begrenzen ist bei jedem Ausgang ein 100 Ohm Widerstand vorgeschaltet.
Alle Eingänge sind über Optokoppler geführt und für eine exakte Signalaufbereitung ist ein Schmitttrigger nachgeschaltet. Die Eingangssignale haben folgende Funktionen:

- Speicherschalter: Wenn eine Büchse versendet wird, so wird der Speicherschalter betätigt.
- Rohrschalter – Durchfahrt: Dieser Schalter gibt ein Signal, wenn eine Büchse diese Stelle passiert.
- Rohrschalter – Empfang: Wenn sich die Büchse bereits im Gerät zum Ausschleusen befindet, wird dieser Schalter betätigt.
- Bremsschalter: Für Stationen die von oben die Büchsen empfangen, muss das Gebläse schon etwas früher abgeschaltet werden bevor die Büchse in der Station ist. Es wird der Luftpolster der sich im Rohr aufbaut zur Bremsung der Büchse verwendet und die Büchse sollte sanft in die Station hineingleiten.

Die restlichen Schalter können für die verschiedenen Motorpositionen genutzt werden. Für jede Motorposition wird ein eigener Eingang verwendet. Bei Stationen mit einem Motor gibt es drei bis vier unterschiedliche Positionen. Bei Stationen mit zwei Motore, hat der erste Motor zwei bis drei Positionen und der zweite Motor hat zwei Positionen.
Über einen CAN- Bus ist der Stationsrechner mit dem Kommunikationsrechner verbunden und erhält über diesen die Anweisungen des Leitrechners.
Eine zusätzliche serielle Schnittstelle am Stationsrechner ist für Zusatzkomponenten, z.B. für RFID- Geräte, um eine automatische Identifikation der Büchsen vorzunehmen. Weiters ist diese Schnittstelle in Testphasen oder für Wartungsarbeiten äußerst hilfreich.
Der Stationsrechner hat eine Stromversorgung von 24V / 5A und alle anderen benötigten Spannungen werden von diesen 24V abgeleitet. Die Spannungsversorgung sollte stabil sein, bei größeren Leitungslängen muss auch ein entsprechender Leitungsquerschnitt verwendet werden um größere Spannungsabfälle auf der Leitung zu vermeiden..

3.5.7 Kommunikation zwischen den Komponenten

Die Kommunikation erfolgt zwischen Leitrechner und Kommunikationsrechner wahlweise über USB oder über Ethernet. Dabei dient der Kommunikationsrechner immer als Server. Der Client ist der Leitrechner, also jener der aktiv wird. Der Leitrechner sendet Daten einer bestimmten Länge zum Kommunikationsrechner und empfängt Daten vom Kommunikationsrechner. Die Pufferlänge ist so ausgelegt, dass immer ganze CAN- Pakete übertragen werden. Die CAN- Pakete werden getunnelt über USB oder Ethernet übertragen. Wird ein USB – Bus verwendet so können theoretisch bis zu 127 Kommunikationsrechner eingesetzt werden. Die Anschaltung der Kommunikationsrechner erfolgt über USB- Hubs. Vorteilhaft sind Hub`s mit einer integrierten Stromversorgung. Über diese Hub`s können die Kommunikationsrechner versorgt werden, man spart sich pro Rechner ein Netzgerät.
Bei der Verwendung einer Ethernet- Schnittstelle kann ein gängiger Ethernetswitch verwendet werden. Dann ist für jeden Kommunikationsrechner ein zusätzliches Netzgerät erforderlich.
Die Datenübertragung zwischen Kommunikationsrechner und Stationsrechner erfolgt über den CAN- Bus. Alle Stationen sind gleichwertig und können jederzeit zum Senden beginnen. Da der CAN- Bus meldungsorientiert (siehe Kapitel 6.1.1) arbeitet, empfangen immer alle Stationen die Meldungen. Wenn nun mehrere Stationen gleichzeitig senden, so setzt sich jene Meldung durch, die von der Station mit der kleineren Kennummer stammt. Jede Station bekommt eine eindeutige Nummer die Teil der Meldungsnummer ist. Statistiken über Anlagengrößen von Rohrpostfirmen zeigen, dass bei Anlagengrößen über 25 Stationen bereits Mehrlinienanlagen eingesetzt werden. In Einzelfällen kommt es aber trotzdem vor, dass Einlinienanlagen bis zu 50 Stationen aufweisen. Ein wesentlicher Grund für diese Teilung ist eine Verringerung der Fahrtzeit beim Büchsentransport von einer Station zu einer anderen. In meinem Konzept wurden diese Einzelfälle berücksichtigt und daher wurde es für Anlagengrößen bis 50 Stationen ausgelegt. Die verwendete Meldungsnummer hat 11 Bits davon werden sechs Bits für die Identifizierung der Station verwendet. Mit sechs Bits können bis zu 63 Stationen verwaltet werden. Die restlichen fünf Bits ergeben 32 unterschiedliche Meldungen, was in diesem Fall ausreichend ist.
Durch das Meldungssystem ergibt sich ein kleiner Nachteil, weil die Meldung von der Station mit der niedrigsten Nummer immer bevorzugt wird. Dieser Nachteil hat aber nur Auswirkungen bei der Initialisierung des Systems, da hier ein ziemlich hoher Datenverkehr herrscht. Aber bereits in der Anlaufphase ist der Datenverkehr soweit reduziert, dass es kaum noch zu Kollisionen kommt.

4 Ein industrielles Anwendungsbeispiel einer pneumatischen Transportanlage

4.1 Technische Ziele

Das technische Ziel war die Entwicklung einer Steuerung, die aus einem verteilten Rechnersystem besteht und sich für den Einsatz von mechanischen und pneumatischen Transportsystemen eignet. Im konkreten Beispiel erfolgte der Einsatz in einer Mehrlinien-Rohrpostanlage (Transro). Das Projekt umfasst die Projektierung, Entwicklung, den Test und die Prototypenfertigung der gesamten Hardware sowie das komplette Softwareengineering für die Basishardware, die Treiber, die Kommunikationsschnittstellen, das User-Interface, die Sprachausgabe und die integrierte Datenbank.

Randbedingungen des Projekts

- Bestehende Rohrpostanlage (mechanische Komponenten ohne Steuerung)
- 5-Linien-Anlage mit ca. 50 Stationen sowie 10 Weichen
- Linienlängen bis zu 600 m
- 24V-Gleichstrom-Motoren an den Weichen und Stationen müssen direkt angesteuert werden
- optische Sensoren (einfache Fremdlicht-unempfindliche Reflexionslichtschranken) mit 24 V Versorgung und digitalem Signalausgang (npn)
- Positionsschalter für die Rohrstellungen einfache Reed-Kontakte
- Ansteuerung von Fremdkomponenten
- Kunde fordert hohe Systemverfügbarkeit (wegen schlechter Erfahrungen mit konventionellen Steuerungssystemen)
- Vollständige, detaillierte Systemhistorie mit der Möglichkeit zur Auswertung (z.B. für Wartungskontrolle)

Grundsätzliche Überlegungen zur Hardwarearchitektur/Systemarchitektur

Die Hardwarearchitektur ist weitgehend durch den Anlagenaufbau vorgegeben.

Eine zentrale Steuerung des System von einem Zentralrechner verbietet sich aufgrund der großen Zahl und der Länge der dann notwendigen Zuleitungen (insbesondere wegen der hohen Einschaltströme bzw. hoher Spannungsabfälle).
Auch aus Kostengründen sollte die Leitungslänge der signifikant Strom trangenden Leitungen minimiert werden. Die Leitungslänge des CAN-Bus (twisted pair Kabel) dagegen ist unkritisch.
Aus den obengenanten Gründen ist es optimal an jeder Weiche bzw. Station einen Stationsrechner zu installieren. Die Stationsrechner dienen gleichzeitig als Nutzerschnittstelle sowie für Wartungszwecke.
Die Stationsrechner einer Linie wurden auf jeweils einen Kommunikationsrechner zusammengefasst, die die Datenpakete an den Leitrechner weitergeben. Vorteile diese Lösung:

- Pufferung der Datenpakete aus einer Linie
- Kommunikationsrechner ermöglicht galvanische Trennung (Linien können teilweise andere Gebäude geführt sein, Gefahr durch unterschiedliche Erdpotentiale)
- Flexibilität bei Erweiterung des Systems auf mehr Linien bzw. bei der Ansteuerung von Fremdgeräten

Daher ergeben sich folgende Hardwarebaugruppen:

- Leitrechner
- Kommunikationsrechner
- Stationsrechner

Anforderungen an die Ausführung der Hardwarekomponenten:

- Die Hardwarebaugruppen müssen kompakt sein
- Unterstützung von verschiedenen Sprachen und Schriftarten (Latein, Kyrillisch, Griechisch,...).
- Die Schaltung und die Anordnung der Bauelemente müssen so ausgelegt sein, dass sie weitgehend unempfindlich gegenüber den auftretenden Störungen sind.
- Einsatz eines leistungsfähigen Rechners auf der Steuerplatine .

- Die Motorensteuerung muss mit einer dynamischen Stromüberwachung ausgestattet sein, damit bei einer Verklemmung oder bei einem Kurzschluss der Motor extrem schnell abschalten kann.
- Verwendung von normgerechten Standardschnittstellen und Bussystemen (TCP/IP, CAN,..).
- Einfache Anbindung an Fremdsysteme über genormte Schnittstellen.

Daraus ergab sich folgende konkrete Realisierung

- Leitrechner
  Als Leitrechner wird ein handels üblicher Personal Computer verwendet.
- Kommunikationsrechner
  Die Kommunikationsrechner sind Eigenanfertigung (Einplatinenrechner, alle in 19 Zoll Rack integriert, Mikrokontroller-basierend , optogekoppelter CAN-Bus Schnittstelle, USB-Schnittstelle, Versorgungsspannung über USB-Schnittstelle)
- Stationsrechner
  Die Stationsrechner sind Eigenanfertigung (Einplatinenrechner, alle in 19 Zoll Rack integriert, Mikrokontroller-basierend , optogekoppelter CAN-Bus Schnittstelle, Sensoreingänge, Stellglieder für Aktoren).

Anforderungen an die Software

- Realisierung einer automatischen Fehlerbehebung während des Betriebs (Wiederanlauftechnik).
- Visualisierung der gesamten Anlage
- Verwendung von Standardprotokoll (z.B. TCP/IP, USB, IEEE1394, ...)
- Abbildung der Anlagenstruktur und die Protokollierung aller Betriebvorgänge in einer Datenbank (Statusabbild, Dokumentation, Statistikauswertungen, Informationsquelle für Qualitätsmanagement und Evaluierung)
- Einfache Wartung und Bedienbarkeit der Anlage
- Geringer Installations- und Wartungsaufwand
- Möglichkeit der Fernwartung
- Einhaltung von Sicherheitsstandards
- Entwicklung eines automatischen Anrufsystems (AAS).

4.2 Mechanische Komponenten und mechanischer Aufbau einer pneumatischen Transportanlage

4.2.1 Mechanische Komponenten

Der folgende Abschnitt soll einen Überblick über die wichtigsten mechanischen Komponenten einer pneumatischen Transportanlage geben.

- Gebläse (Verdichter) und Reversierautomat
  Mit dem Gebläse wird in den Rohrleitungen ein Druck oder ein Vakuum erzeugt. Die Umschaltung zwischen Druck und Vakuum wird mit dem Reversierautomaten vorgenommen.

- Offene Stationen
  Ein offenes Rohrende wird als offene Station bezeichnet. Der Benutzer einer solchen Station ist für das Einschleusen der Büchse verantwortlich. Solche Stationen sind kostengünstig in der Anschaffung aber sehr störungsanfällig. Es können mehrere Büchsen ins System gebracht werden.

- Automatik- Stationen
  Rohrpoststationen an denen das Versenden und Empfangen der Büchsen automatisch abläuft werden Automatikstationen genannt. In diesen Stationen werden bei der Versendung von Büchsen, diese automatisch in die Anlage eingeschleust und beim Empfang einer Büchse wird diese bei der Stationseinfahrt gebremst und anschließend automatisch ausgeschleust.

- Bandstationen
  Bei Übergabestationen zwischen zwei oder mehreren Anlagen kommen solche Bandstationen zum Einsatz. Mit diesen Bandstationen können Büchsen die in einem Rohrspeicher aneinander gereiht werden, wieder vereinzelt werden.

- Weichen
  Weichen werden eingesetzt damit pneumatische Transportsysteme ähnlich einer Baumstruktur aufgebaut werden können. Eine Weiche verbindet immer nur einen Teil

einer Linie mit dem Gebläse und nur in dieser Teillinie ist ein Transport möglich.

- Verbindungsrohre
  Für die Verbindung der einzelnen Komponenten werden Kunststoffrohre eingesetzt. Der Durchmesser der Rohre ist zugleich die Größenangabe der Anlage (z.B. „110 Anlage" bedeutet, dass es sich um eine Anlage mit einem Rohrdurchmesser von 110mm handelt).

- Linie
  Ein gesamtes System mit Gebläse, Stationen, Weichen und Rohre wird als Linie bezeichnet.

- Verteiler oder Übergabestationen
  Das ist jene Einheit die mehrere Linien miteinander verbindet.

4.2.2 Mechanische Details der Stationen

Beim Anlagenaufbau muss eine Notstation vorhanden sein, in der jede Büchse ausgeschleust werden kann. Es kommt immer wieder vor, dass sich eine Büchse in der Anlage befindet und diese muss irgendwo hinbefördert werden. Solche "verlorenen" Büchsen bereiten auch bei normalen Sendungen immer wieder Probleme. Eine Sendung läuft ab, nach dem Einschleusen der Büchse befinden sich zwei Büchsen im System. Wenn nun dieser Fehler nicht erkannt wird, können folgende Fälle auftreten.

- Die verlorene Büchse wird an der Zielstation ausgeschleust und gleichzeitig bleibt die eingeschleuste Büchse im System liegen. Es hat zwar den Anschein, dass die gesamte Sendung ordnungsgemäß abgelaufen ist. Jedoch wurde eine falsche Büchse zur Zielstation zugestellt. Softwaretechnisch sind solche Fehler nur begrenzt zu lokalisieren. Möglich wird das durch Rohrschaltermeldungen, wenn sich diese beiden Büchsen örtlich verteilt im Rohrsystem befinden.

- Beide Büchsen befinden sich direkt hintereinander im Rohrsystem. Dies führt beim Ausschleusen in der Zielstation zu einer Verklemmung. Hier ist der Hersteller der Station gefordert, diese so zu bauen, dass jede Verklemmung die auftritt steuerungstechnisch zu beheben ist. Die Stationen haben meistens mehrere Positionen,

z.B. für Einschleusen, Ausschleusen und Durchfahrt. In diesem Fall wären es drei Positionen. Diese Positionen werden immer von einer bestimmten Richtung angefahren um eine Übereinstimmung zwischen Fahrrohr und Stationsrohr zu gewährleisten. Bei den beiden Endpositionen tritt hier kein Problem auf, da sie entweder im Uhrzeigersinn oder gegen den Uhrzeigersinn angefahren werden. Bei der Position in der Mitte kann beides eingestellt sein. Daher sollten Büchsen nicht über die Mittenposition ausgeschleust werden, da in dieser Position Verklemmungen nicht immer lösbar sind. Bei den beiden Endpositionen ist das einfacher, da diese nur von einer Richtung angefahren werden. Wenn nun die Verklemmung der Büchse behoben ist, besteht noch immer das Problem mit mehreren Büchsen im Rohr. Diese müssen nun vereinzelt werden und der Reihe nach an der Station ausgeschleust werden. Ohne größeren Aufwand läst sich das an Stationen verwirklichen, die von oben empfangen. Im Empfangsrohr in der Station wird ein Loch angebracht. Die Höhe dieses Lochs muss so gewählt werden, dass das Loch auch dann vollkommen frei ist, wenn sich die längste Büchse im Empfangsrohr befindet. Sind nun mehrere Büchsen im Empfangsrohr können alle bis auf eine ins Rohrsystem zurückgesaugt werden. Die verbleibende Büchse kann an der Station ordnungsgemäß ausgeschleust werden. Dieser Vorgang wird so oft wiederholt, bis sich keine Büchsen mehr im Rohr befinden. Steuerungen mit einer Wiederanlauftechnik einzusetzen hat also nur einen Sinn für Anlagen, in denen auch die gesamte Mechanik dafür ausgelegt ist. Das sollte nicht heißen, dass bei Anlagen mit einer beliebigen Mechanik die Steuerung nicht einsetzbar wäre. Die auftretenden Fehler können aber nicht automatisch behoben werden und es ist immer ein Eingriff vom Wartungspersonal erforderlich, was Zeit und Geld kostet.

4.2.3 Aufbau einer Anlage

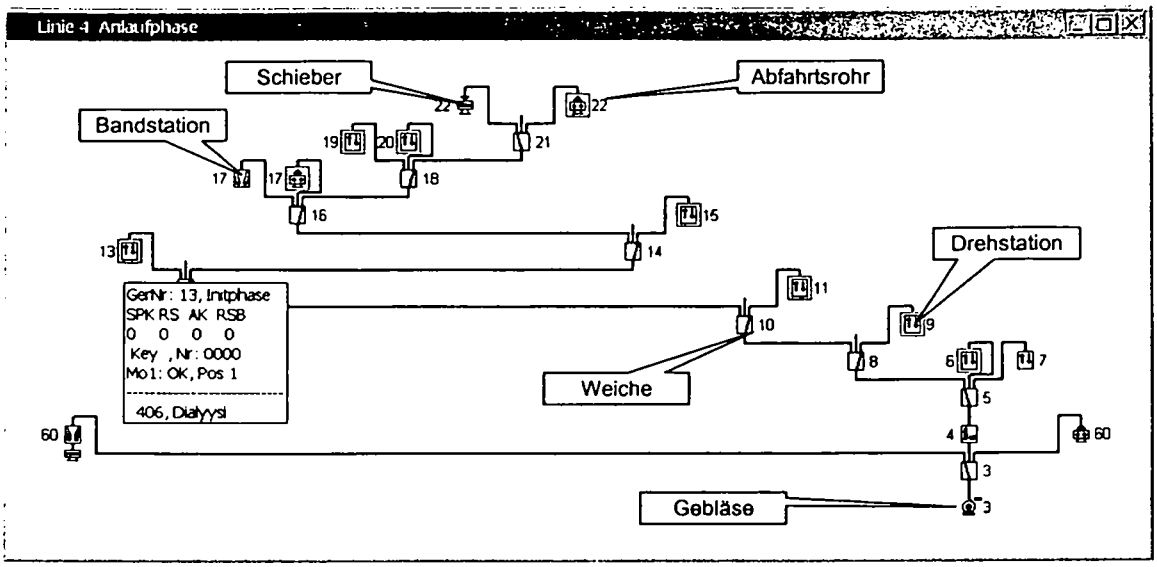


Abb.: 4.1 Anlagenschema

Ein Rohrpostsystem besteht aus einer oder mehreren Linien die über einen Verteiler miteinander verbunden sind. Eine Linie (siehe Bild) ist eine in sich abgeschlossene Einheit und besteht aus Weichen, Stationen und einem Gebläse und eventuell einem Hilfsgebläse. Diese Stationen und Weichen sind entsprechend einer Baumstruktur durch Rohre miteinander verbunden. Eine Baumstruktur besteht aus einer Wurzel, Knoten, Blättern und Kanten. In der Wurzel befindet sich das Gebläse. In einem Knoten kann eine Weiche oder eine Station sein. Befindet sich eine Weiche im Knoten so hat dieser Knoten mehr als zwei Kanten. Bei einer Station hat ein Knoten genau zwei Kanten. In den Blättern des Baumes dürfen nur Stationen vorkommen. Dort muss es möglich sein, eine Büchse aus dem System zu entfernen. Die Kanten repräsentieren das Rohrsystem der Anlage.
Baumstrukturen haben in der Informatik eine große Bedeutung und es gibt viele Theorien über Baumstrukturen. Um Baumstrukturen zu verwalten eignen sich verkettete Listen oder Arrays dessen Felder über einen Index miteinander verkettet sind. Die gesamte Anlage ist im Leitrechner vollständig gespeichert, wobei jede Linie in einem eigenen Array verwaltet wird. Jedes Gerät, egal ob Weiche, Station oder Gebläse, bekommt eine eindeutige Gerätenummer (vgl. auch CAN-Bus). Über diese Nummer sind alle Daten der Stationen im Leitrechner indiziert und abrufbar. Auch die Kommunikation mit dem Gerät erfolgt über diese Nummer.

- Gebläse

  Die Verdichter, die in diesen Anlagen zum Einsatz kommen, werden mit einem

asynchronen Drehstrommotor angetrieben. Die Drehrichtung des Motors ist immer gleich. Die Umschaltung der Luft wird mit einem Reversierautomaten durchgeführt. Dieser Automat wird im Wesentlichen durch einen Schieber der über einem Stellmotor positioniert wird, realisiert. Das Gebläse hat für die Luft zwei Anschlüsse, einer davon ist für Druckluft der andere Anschluss ist für das Saugen. Der Schieber verbindet entweder den Anschluss „Druck" oder „Saug" mit dem Rohr der Anlage, der andere Anschluss ist frei. Dadurch wird in der Anlage die Luft in Richtung Gebläse bewegt oder in Richtung Endgerät. Wird das Gebläse abgeschaltet so werden die beiden Anschlüsse des Gebläses miteinander verbunden und die Luft wird im Gebläse im Kreis bewegt und das Luftrohr der Anlage wird abgedichtet, sodass eine Weiterströmung der Luft in der Anlage verhindert wird.

Das Gebläse hat die drei Stati:

Stopp: Das Gebläse ist abgeschaltet

Saug: Das Gebläse ist eingeschaltet und der Luftstrom geht Richtung Wurzel.

Druck: Das Gebläse ist eingeschaltet und der Luftstrom geht Richtung Blätter

- Weichen

  Durch den Einsatz von Weichen kann eine Breitenwirkung erzielt werden ohne dass die Wege, die die Büchsen zurücklegen, wesentlich länger werden. Eine Weiche hat immer ein Rohr zur Wurzel und zwei oder mehrere Rohre zu den Blättern. Die Weichenstellung wird mit einem Gleichstrommotor vorgenommen. Die Rückmeldung der Position erfolgt über Reedkontakte. Ein optischer Sensor, der in der Weiche moniert ist (Rohrschalter), liefert beim Vorbeifahren einer Büchse ein Signal.

  Eine Weiche hat immer genau ein Vorgängergerät und mehrere Nachfolgegeräte. Je nach eingestellter Weichenposition wir ein bestimmter Strang (Teil einer Linie) mit dem Gebläse verbunden. Die restlichen Stränge der Linie stehen dann nicht für einen Transport zur Verfügung, Bei Verwendung geeigneter Stationen können aber durchaus während dieser Zeit durch den Benutzer der Anlage Büchsen ins System eingeschleust werden. Der Weitertransport kann aber erst nach Ablauf der letzten Sendung durchgeführt werden.

- Stationen

  Über Stationen werden Büchsen in das System ein- und ausgeschleust. Stationen können Durchfahrts- oder Endstationen sein. Eine Durchfahrtsstation hat immer eine Vorgängerstation (kann auch eine Weiche oder ein Gebläse sein) und sie kann eine Nachfolgestation haben. Eine Endstation hat nur eine Vorgängerstation.
  Jede Station hat – je nach Bauart – mindestens einen drehbaren Teil, der für die Aufnahme der Büchse und den Transport der Büchse innerhalb der Station zu den diversen Rohren (Ausschleusung, Fahrtrohr,...) Verwendung findet. Diese Stationseinstellungen werden mit Gleichstrommotoren vorgenommen. Die Rückmeldung der Motorpositionen erfolgt über Reedkontakte. Neben diesen primären Positionsmeldern werden etliche Kontaktschalter benötigt.

  - Der Speicherkontakt meldet, wenn eine Büchse zur Versendung ansteht.
  - Einen Rohrschalter für die Erkennung einer Büchsendurchfahrt und
  - einen Rohrschalter für den Büchsenempfang.

  Zusätzlich werden Elektromagnete für diverse Verriegelungen, die meist zur Erhöhung der Sicherheit des Bedieners dienen, verwendet.
  Die Bedienung der Station durch den Benutzer erfolgt über ein Display-Tastaturmodul.
  Für jede Station wird zusätzlich zur intern verwendeten Gerätenummer noch eine Liste von allgemein verständlichen Adressen verwaltet. Die Versendung der Büchsen erfolgt über diese für den Bediener verständlicheren Bezeichnungen. Hat eine Station mehrere Adressen, wird jeder Adresse ein Ankunftssignal zugeordnet, das

  - entweder in akustischer Form eines Signalgebers oder einer Klingel ausgeführt werden kann
  - oder in optischer Form durch Lichtsignale oder Anzeigetafeln.

  Eine weitere und in dieser Anlage erstmals verwendete Möglichkeit der Ankunftsverständigung der Benutzer sind eine oder mehrere im System für diese Station gespeicherten Telefonnummer und die dazugehörenden zu 'sprechenden' Meldungen. Nach der Büchsenankunft wählt - über einen entsprechenden Anschluss an die Haustelefonanlage - das automatische Anrufsystem diese Telefonnummer und gibt die Meldung durch. Dabei müssen natürlich auch etwaige Abwesenheiten oder Umleitungen (tageszeit- oder wochenabhängig) berücksichtigt werden.

4.2.4 Wartungsmaßnahmen

Führt die Rohrverlegung durch mehrere Räume in denen unterschiedliche Temperaturen herrschen (warm und kalt), so bildet sich in den Rohren Kondenswasser, wenn längere Zeit keine Sendung durchgeführt wird. Dieses Problem kann einfach behoben werden, indem in bestimmten Zeitabständen Leersendungen generiert und durchgeführt werden. Diese Leersendungen führen zu einem regelmäßigen Luftaustausch und unterdrücken eine Kondenswasserbildung

4.3 Realisierung

Mit der Hardwarerealisierung der Stationseinheiten wurde im Grunde drei mal begonnen. Beim ersten Versuch war die Grundidee, bestimmte Filterfunktionen durch Hardware zu realisieren. Daher wurde ein FPGA eingesetzt, in dem bestimmte Logikfunktionen programmiert wurden und zusätzlichen Funktionen während der Laufzeit einstellt wurden. Diese Stationseinheiten funktionierten auf dem Prüfstand und in einer Testanlage optimal. Erst bei einem Einsatz in einer Produktionsanlage traten plötzlich Fehler auf und diese Fehler waren willkürlich auf den diversen Stationseinheiten verteilt. Eine Fehlerbehebung war nur durch Aus- und Einschalten der Versorgungsspannung möglich. Dadurch wurde die Fehlersuche zu einem Alptraum. Nach dem Aus- und Einschalten funktionierte alles wieder problemlos sogar manchmal tagelang. Bei zusätzlichen Prüfungen in einem EMV- Labor war diese Situation nicht nachvollziehbar. Erst durch einen Hinweis, dass es sich um einen „Latchup" – Fehler handeln könnte, konnte die Fehlerursache mit Strommessungen festgestellt werden. Jene Stationseinheit die diesen Fehler aufwies nahm wesentlich mehr Strom auf als normal. Es war mir aber nicht möglich bei den Stationseinheiten diesen Fehler durch irgendwelche Maßnahmen zu beseitigen. Durch das sporadische Auftreten des Fehlers war es einfach unmöglich gezielte Messungen durchzuführen. Daher konnte die tatsächliche Störquelle nicht ermittelt werden. Bei dieser Stationseinheit zeigte sich auch, dass ein positiver EMV- Test bei Steuerungen für pneumatische Transportanlagen nicht immer zum Erfolg führen muss. Ein gravierender Nachteil war bei dieser Version das fix programmierte FPGA. Bei einer Programmänderung mussten immer alle Stationen aufgesucht werden und Vorort die FPGA`s mit einem Programmiergerät neu programmiert werden.
Bei dem nächsten Hardwareentwurf wurde auf diese FPGA`s verzichtet und soweit wie möglich alle Programmierfunktionen als „In System Programing" (ISP) durchgeführt. Es war einfach wichtig, dass bei Systemtest in Produktionsanlagen, Updates direkt vom Leitrechner aus durchgeführt werden können. Bei dieser Hardwareversion traten in den ersten Tests keine

fatalen Fehler auf. Alle Fehler die am Beginn in dieser Hardwareversion bei Stationsrechner vorkamen, konnten mit Hilfe von kleinen Nachbesserungen behoben werden. Nach und nach wurden die Hardwarebaugruppen immer stabil und es gab keine dubiosen Ausfälle mehr.
In der dritten Version wurden dann alle Erkenntnisse, die bei den vielen Test in den vorigen Versionen erarbeit wurden, eingearbeitet.

4.3.1 Leitrechner

4.3.1.1 Allgemein

Der Leitrechner übernimmt die gesamte Steuerungslogik des Systems. D. h. alle Stationsrechner senden die Sensorsignale zum Leitrechner und setzen die Befehle des Leitrechners um.

4.3.1.2 Hardware

Bei der gesamten Realisierung der Anlage wurde generell darauf Wert gelegt, wo es möglich war Standardkomponenten einzusetzen. Der Leitrechner ist normalerweise keinen erhöhten elektromagnetischen Störungen ausgesetzt, daher reicht ein normaler Personalcomputer aus. Dieser muss ev. mit einer speziellen „Watch-Dog" Karte versehen werden, um einen Absturz des Computers (blue screen) automatisch beheben zu können. Weiters muss ein System bei Fehlern, möglichst schnell wieder in einen definierten Zustand gebracht werden. Jeder handelsübliche Personal Computer hat heute USB- Schnittstellen, Ethernet- Anschluss, serielle und parallele Schnittstellen. Für das automatische Anrufsystem muss der Personalcomputer mit einem AVM- ISDN Adapter (FritzCard) erweitert werden. Das AAS – Modul wird speziell für diese Karte entwickelt und getestet.

4.3.1.3 Schnittstellen

Für die Datenübertragung zwischen Personalcomputer und Kommunikationsrechner der Anlage wurde der USB- Bus verwendet. Es wird das USB- Protokoll der Version 1.1 verwendet. Die Datenübertragung zwischen diese beiden Komponenten wird mit BULK-Transfer durchgeführt. Diese Transferart bietet sich besonders gut für dieses System an, da immer mehrere CAN- Pakete auf einmal übertragen werden. Die maximale

Übertragungslänge der USB- Blöcke beträgt 256 Byte. Die CAN- Pakete haben eine konstante Länge von 10 Byte.
Nach der Installation der ersten Anlage musste man sich die Frage stellen, ob es nicht besser gewesen wäre, man hätte einen Kommunikationsrechner mit einer Ethernet- Schnittstelle. In den Fabriken gibt es eine Infrastruktur für Ethernet und es ist daher nahe liegend diese zu verwenden.

4.3.1.4 Software

Bei der Entwicklung der Software stand zuerst die Frage im Mittelpunkt, soll ein kommerzielles Softwareprodukt (z. B. Windows von Microsoft) oder ein Open Source Produkt (z. B. Linux) verwendet werden. Nach dieser Entscheidung konnten die weitern Softwarewerkzeuge ausgewählt werden. Schon nach den ersten Tests mit den beiden Betriebssystemen war klar, dass nur ein kommerzielles System zum Einsatz kommen kann. Die gesamte Softwareunterstützung und die technische Dokumentation waren wesentlich besser und vor allem vollständiger als bei einem Linuxsystem. Für die Programmierung wurde Visual Studio 6.0 verwendet. Für diese Arbeit war es einfach ein wesentlicher Vorteil, dass es sehr viele und umfassende Testbeispiele in der Microsoft- Dokumentation MSDN gab. Die gesamte Software wurde objektorientiert implementiert. Es wurde besonderer Wert darauf gelegt, dass sämtliche Hardwareobjekte, wie Linien, Stationen, Weichen und Verteiler auch softwaremäßig als Objekt existieren. Jeder Programmierer der Systemkenntnisse besitzt, sollte sich einfach in das Programm einarbeiten können.

Aufbau der Software

Das Softwareprojekt wurde in sechs Module zerlegt, die über Schnittstellen miteinander kommunizieren. Jedes Modul ist eine abgeschlossene Einheit. Bei Softwareänderungen in einem Modul genügt es, dieses eine Modul zu testen. Die Module umfassen die folgenden Teilbereiche:

- Initialisierung
- Benutzerschnittstelle
- Datenverwaltungsmodul
- Betriebs- und Anlaufmodul
- Kommunikations- und Sicherheitsmodul
- Automatisches Anrufsystem

Ein spezieller Programmteil ist die Sprachenunterstützung die sich auf alle Module auswirkt. Eine wesentliche Voraussetzung bei der Implementierung war für mich, dass das Programm für beliebig viele Sprachen ausgelegt sein muss. Wenn das Programm in einem anderen Land installiert wird, so sollte es möglich sein, über eine Softwareeinstellung diese Sprache auszuwählen. Umgekehrt besteht der Vorteil bei Test- und Wartungsarbeiten, dass man auf die deutsche Sprache zurückschalten kann. In dieser Software werden alle Daten in einer Datenbank gespeichert, die Sprachunterstützung bildet dabei eine Ausnahme. Für jede Sprache muss ein Ressourcefile (z. B. Deutsch.rc muss immer zur Verfügung stehen) vorhanden sein. In diesem Ressourcefile sind alle Texte einer bestimmten Sprache gespeichert. Wenn die Software nach der Installation das erste Mal gestartet wird, ist die eingestellte Sprache Deutsch. Im folgenden Abschnitt ist ein kurzer Auszug aus den Ressourcefiles Deutsch und Englisch dargestellt.

Datei: Deutsch.rc

```
STRINGTABLE PRELOAD DISCARDABLE
BEGIN
IDR_RPCONTTYPE "Rohrpost - CONTROL"
IDR_LINE "Linie"
IDS_FAHRT_ANLAUF_F "Running:\tANLAUFPHASE Linie%-d: Strang leersaugen  von Gerät %d"
IDS_FAHRT_ANLAUF_W "Waiting:\tANLAUFPHASE Linie%-d: Strang leersaugen  von Gerät %d"
IDS_FAHRT_RUNNING_REFERENZ "Running:"
IDS_FAHRT_EINSCHLEUSEN "\tBüchse wird eingeschleust"
.
.
.
IDS_M10_ZEILE4 "¢£¢£¢£¢£"
IDS_M10_ZEILE6 "N Minus"
IDS_M10_ZEILE8 "C Clr     # Ende"
IDS_M_DUMMY "Dummy"
IDM_LAN_FIN "Umschalten auf FINNISCH\nFinnisch"
END
```

Datei: Englisch.rc

```
STRINGTABLE PRELOAD DISCARDABLE
BEGIN
IDR_RPCONTTYPE "CONTROL for pneumatic tube"
IDR_LINE "Line"
IDS_FAHRT_ANLAUF_F "Running:\tSTARTUPPHASE Line%-d: sucking tube from device %d"
IDS_FAHRT_ANLAUF_W "Waiting:\tSTARTUPPHASE Line%-d: sucking tube from device %d"
IDS_FAHRT_RUNNING_REFERENZ "Running:"
IDS_FAHRT_EINSCHLEUSEN "\tMoving Can into system"
.
```

```
.
.
IDS_M10_ZEILE4 "¢£¢£¢£¢£"
IDS_M10_ZEILE6 "N Minus"
IDS_M10_ZEILE8 "C Clr      # Exit"
IDS_M_DUMMY "Dummy"
IDM_LAN_FIN "Change to Finnish\nFinnish"
END
```

Um die Übersetzungen so einfach wie möglich zu gestalten und sie von speziellen Übersetzen dieser Sprache durchführen zu lassen, wurde ein Access – Programm entwickelt. Mit dem Access – Tool „LANGUAGE.MDB" kann die Sprachenübersetzung einfach durchgeführt werden. Die Texte werden in einer Tabelle gespeichert. Für jede Sprache ist in dieser Tabelle eine eigene Spalte angelegt. Zusätzlich gibt es noch die Spalte „ResId". ResId ist der Ressourceidentifier und dieser stellt den Zusammenhang zwischen Programm und Text her. Das Feld „ResId" kann und darf nicht verändert werden.

Abb.: 4.4 Benutzerschirm für die Sprachunterstützung

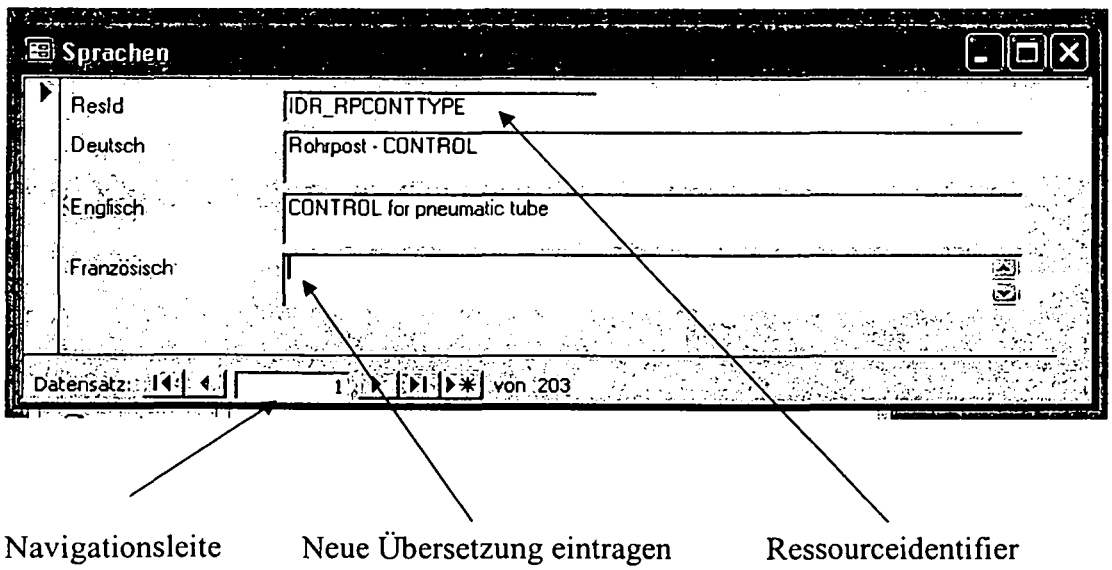


Tabelle 3: tab_sprachen
Generierung der Ressoucefiles für die einzelnen Sprachen

Feldname	Felddatentyp	Beschreibung
Id	Integer	Primärschlüssel
ResId	Char Width 25	Resource identifier
Deutsch	Char Width 50	Text in deutscher Sprache
Englisch	Char Width 50	Text in englischer Sprache
Französisch	Char Width 50	Text in französischer Sprache
. . .		

Nach der Übersetzung kann man direkt für die gespeicherten Sprachen die einzelnen Ressourcefiles erzeugen. Für die Einbindung dieser Ressourcefiles in das Programm „RPControl" muss ein Compilerlauf durchgeführt werden.

4.3.1.4.1 Initialisierungmodul

Nach dem Programmstart wird als erstes die Initialisierung des gesamten Systems durchgeführt. Die Initialisierung umfasst die folgenden Punkte.

- Zugriff auf die Datenbank
  Sämtliche Systeminformationen sind in der Datenbank gespeichert. Um weitere Schritte durchführen zu können, muss zuerst die Datenbankverbindung hergestellt werden und die Systeminformationen aus der Datenbank gelesen werden.

- Testen aller Kommunikationskanäle die in den Systeminformationen enthalten sind. Das sind die Datenübertragungen zwischen Leitrechner und Kommunikationsrechnern und zwischen Kommunikationsrechner und Stationsrechnern. Nur bei erfolgreichen Tests wird die Initialisierung der entsprechenden Komponenten weiter geführt .

- Nun wird jeder Stationsrechner initialisiert und getestet. Jeder Stationsrechner kann bis zu zwei Motore haben. Falls er Motore hat, so werden die verschiedenen Positionen angefahren und anschließend die Motorgrundstellung eingestellt. Die Rohrschalter, Bremsschalter und Speicherschalten können zwar nicht auf Funktionalität geprüft werden, aber es kann festgestellt werden, ob so ein Schalter permanent eingeschaltet ist.

- Initialisieren sämtlicher Anzeigen bei den Stationsrechnern.

- Initialisierung der Statuspunkt – Logik

- Aufruf des Anlaufprogramms

4.3.1.4.2 Benutzerschnittstelle

Bei der Benutzerschnittstelle wurde sehr großer Wert auf Bedienerfreundlichkeit gelegt. Sämtliche Anlagenteile und die gesamte Topologie einer Linie werden in einem Fenster grafisch dargestellt. Für jede Linie wird ein eigenes Fenster verwendet. Die Verbindung der einzelnen Linien untereinander wird in einem Verteilerfenster dargestellt. Weiter gibt es ein Betriebsfenster in dem alle Meldungen angezeigt werden und ein eigenes Errorfenster in dem nur Fehler angezeigt werden. Dieses Errorfenster ist vor allem für das Wartungspersonal von großer Bedeutung. In der Früh bei Dienstbeginn reicht ein Blick ob in der Nacht oder während des Wochenendes Fehler aufgetreten sind.

Über diese Benutzerschnittstelle kann immer der aktuelle Status von jeder Linie und jedem Gerät abgelesen werden. Der Status der Geräte wird in der Grafik mit Hilfe von farbigen Bildpunkten angezeigt oder kann mit Hilfe der rechten Maustaste textuell angezeigt werden.

Die Anlagenprogrammierung erfolgt über die graphische Benutzerschnittstelle mit einem Auswahlverfahren. In dieser Version können nur Stationen, Weichen und Sondergeräte verwendet werden, die bereits in der Datenbank gespeichert und im Programm „RPCONTROL“ vorgesehen sind.

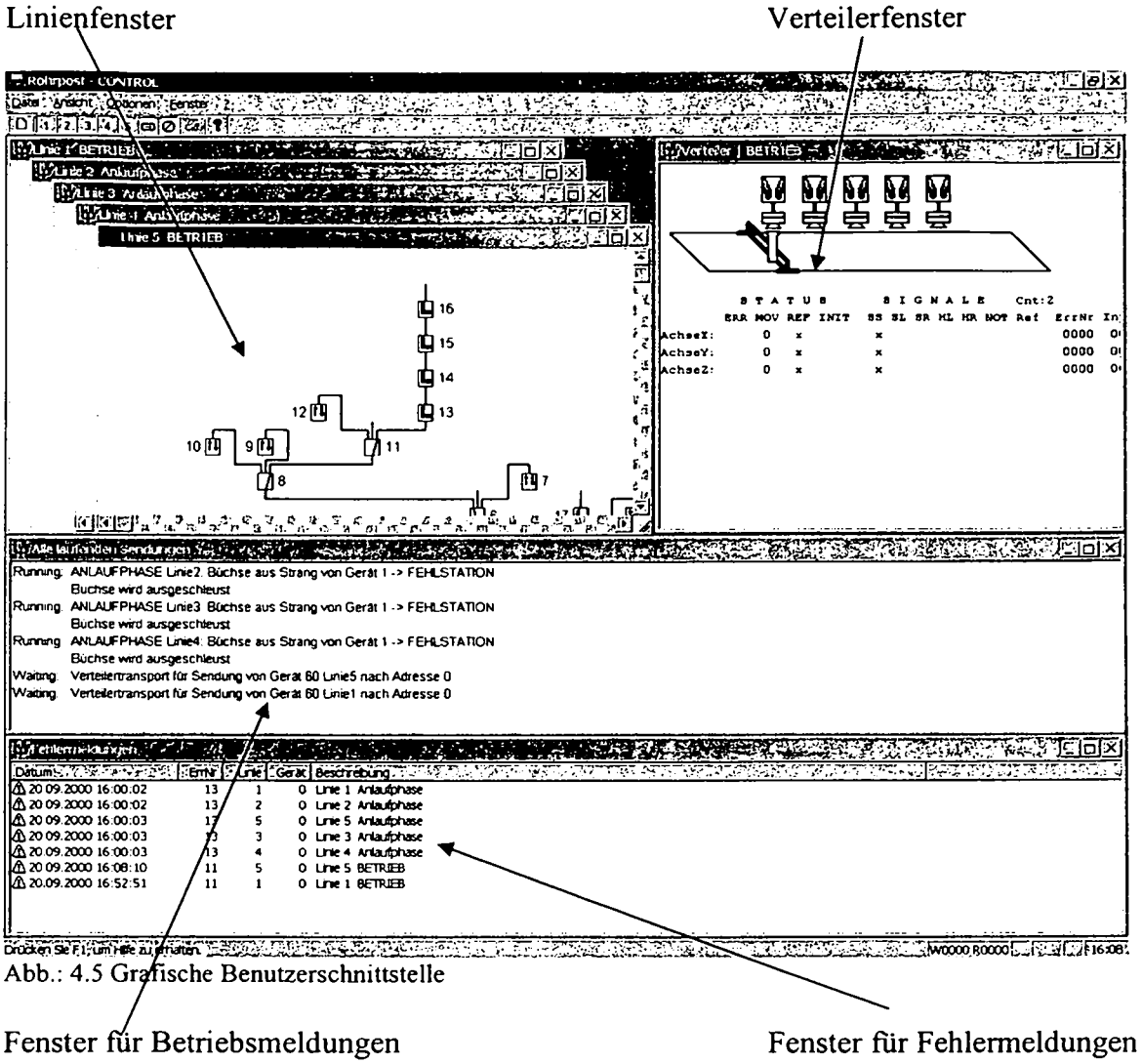


Abb.: 4.5 Grafische Benutzerschnittstelle

4.3.1.4.3 Betriebs- und Anlauf- Modul

Nach der Initialisierung oder nach einem Ablauffehler wird immer das Anlaufprogramm gestartet. Das Anlaufprogramm sollte dafür sorgen, dass sichergestellt wird dass sich keine Büchsen in der Linie befinden. Werden Büchsen in der Linie gefunden so müssen diese zu einer Notstation befördert und dort ausgeschleust werden. Die Auffindung der Büchsen geht folgendermaßen vor sich. Zuerst wird das Gebläse eine bestimmte Zeit auf Druck geschaltet. Wenn sich eine Büchse in der Wurzel der Linie befinden sollte, wird diese in irgendeinen Strang der Linie befördert. Anschließend wird von jeder Endstation eine Sendung zur Notstation ausgelöst. Zusätzlich wird die größte Fahrzeit zwischen zwei Stationen ermittelt. Zu dieser Zeit wird ein gewisser Sicherheitsfaktor addiert. Startet die Sendung, ohne dass nach Ablauf dieser Zeit ein Rohrschalter aktiv wird, kann angenommen werden, dass sich keine Büchse in diesem Strang befindet. Die Sendung wird abgebrochen, sonst wird die

Sendung wie eine normale Sendung fortgesetzt bis die Büchse an der Notstation angekommen ist. Nach Beendigung der Anlaufphase geht der Status der Linie in „Betrieb" über und von nun an sind alle Stationen dieser Linie ebenfalls betriebsbereit. Die Anlaufphase wird in allen Linien durchgeführt, tritt dabei ein Fehler auf, wird die betroffene Linie gesperrt. Der restliche Teil der Anlage ist in Betrieb.

Teilbereiche:

- Allgemeine Aufgaben
- Checkpoint Verarbeitung für Fehlererkennung
- Aufnahme und Verwaltung von Sendungen
- Ausführungen von Sendungen
- Erkennung von doppelten Büchsen
- Beseitigen von Verklemmungen

4.3.1.4.4 Datenverwaltungs-Modul

Dieses Modul dient zur Speicherung aller Konfigurationsdaten und Betriebsdaten. Für die Verwaltung der Daten wird eine Datenbank verwendet. Der Zugriff auf die Datenbank erfolgt über ODBC (Open Database Connectivity). Eine ODBC- Schnittstelle steht für fast alle Datenbanken und Datenbankmanagementsystemen (DBMS) zur Verfügung. Der Zugriff auf die Daten erfolgt über SQL. In dieser Arbeit wird eine MS- Access Datenbank eingesetzt. Der Grund dafür war einfach der doch sehr komfortable Assistent der in MS- Access zur Verfügung steht. Mit dem Assistenten können ohne großen Programmieraufwand, Formulare und Berichte für die diversen Auswertungen erstellt werden.

Tabelle 4: tab_Adressen

Adressenverwaltung der einzelnen Stationen

Feldname	Felddatentyp	Beschreibung
ID	Integer	Identifikationsnummer
Adresse	Integer	Adresse des gewünschten Gerätes
Line	Integer	Liniennummer in der das Gerät liegt zu dem die Adresse gehört
GeId	Integer	Gerätenummer, der die Adresse zugeordnet ist
Name	Char Width 25	Bezeichnung der Adresse (z.B. Stationsname)
Slow	Bit	Langsamfahrt der Büchse gewünscht
UmleitAdresse	Integer	Wenn diese Adresse aktiviert ist, wird die Büchse zu der zugehörigen Station umgeleitet
Gesperrt	Bit	Wenn aktiv, kann die Station nicht angefahren werden
AKNr	Integer	Ankunftsnummer ist ein Zusatzsignal das die Ankunft einer Büchse anzeigt
Blinken	Bit	Bei Aktivierung blinkt das Ankunftssignal, sonst ist es dauerhaft eingeschaltet
BuechsenCnt	Integer	Anzahl der leer verschickten Büchsen
UseCnt	Bit	Aktivierung der Leerbüchsenfunktion
UseAAS	Bit	Automatisches Anrufsystem aktivieren
AASNr	Char Width 50	Telefonnummer die angerufen wird
AASData	LongChar Attribute 32	Text oder Dateiname in dem der Text steht, der bei Anruf gesprochen wird
AASIsText	Bit	Kennzeichen ob im Feld AASData Text oder Dateiname steht

Tabelle 5: tab_Geraete

Konfiguration der Stationen

Feldname	Felddatentyp	Beschreibung
ID	Integer	Identifikationsnummer
Line	Short	Liniennumer
CanGeNr	Short	Kommunikationsnummer für CAN- Bus
GeId	Short	Gerätenummer
Typ	Short	Stationstyp
Vorne	Short	Gerätenummer des Vorgängergeräts
Hinten1	Short	Gerätenummer des Nachfolgegerätes, bei Weichen und Stationen
Hinten2	Short	Gerätenummer des Nachfolgegerätes, bei Weichen
Hinten3	Short	Gerätenummer des Nachfolgegerätes, bei Weichen
Laufzeit	Short	Fahrtzeit der Büchse zum Vorgängergerät
Priority	Short	Aktionen dieses Geräts haben höhere Priorität
Betrieb	Bit	Betriebsstatus (ein- ausgeschaltet)
LuftVonUnten	Bit	Rohr zum Gebläse von unten oder oben
Bremszeit	Short	Zeit im wie viel Gebläse vor Erreichen der Station ausgeschaltet werden muss
WartezeitEmpfang	Short	Zeit nach der Büchse nach dem Eintreffen in der Empfangsstation ausgeschleust werden kann
Grundstellung	Short	Motorposition der Station im Normalfall
Fehlstation	Bit	Station ist zugleich Notstation
Data1	Short	Zusätzliche Informationen für spezielle Stationen
Data2	Short	- „ -
X	Integer	Position des Icons am Schirm
Y	Integer	- „ -

Tabelle 6: tab_Linien

Konfiguration der Linien

Feldname	Felddatentyp	Beschreibung
LineNr	Integer	Liniennummer
Bez	Char Width 50	Linienbezeichnung
SerNr	Short	Nummer des USB- Anschlusses
TCP	Bit	TCP/IP Kommunikation Ja/Nein
IP	Integer	IP – Adresse
MASK	Integer	Subnetmask
MAC	Char Width 50	MAC des Kommunikationsrechners

Tabelle 7: tab_Verteiler

Konfiguration der Linienübergabe

Feldname	Felddatentyp	Beschreibung
Pos	Integer	
LineNr	Integer	
X_AB	Integer	
Y_AB	Integer	
X_AN	Integer	
Y_AN	Integer	
Z_UNTEN	Integer	
Z_OBEN	Integer	

Tabelle 8: tab_Sending_Restrictions

Zuteilung der Adressen zu den Stationen

Feldname	Felddatentyp	Beschreibung
GerId	Integer	Gerätenummer
AdrId	Integer	Adresse des Gerätes

Tabelle 9: tab_Errors

Fehlerprotokoll

Feldname	Felddatentyp	Beschreibung
ID	Integer	Identifikationsnummer
Warning	Bit	Art des Fehlers
Datum	Date	Datum und Uhrzeit des Fehlers
ErrNr	Integer	Fehlernummer des Systems
Line	Integer	Liniennummer
GerId	Integer	Gerätenummer
Descr	LongChar Attribute 32	Beschreibung des Fehlers

Tabelle 10: tab_Checkpoint

Checkpointverwaltung

Feldname	Felddatentyp	Beschreibung
Id	Integer	Primärschlüssel
LiStatus	Short	Linienstatus (Anlauf, Betrieb, Störung, Gesperrt, Wartung
GeStatus	Char Width 50	Ist ein Speicherbereich von 50 Byte der für jede Station eine etwas andere Struktur hat, in der die gesamten Geräteeinstellungen abgelegt sind.
GeNr	Short	Eindeutige Gerätenummer
Aktionsnummer	Integer	Für jede Einstellung die es an einer Station gibt, wird eine Nummer vergeben.
Displaytext	Char Width 50	Stationstext
Checkpoint	Integer	Checkpointnummer
SendId	Integer	Nummer der laufenden Sendung
Aktionszeit	Date	Timestamp

Tabelle 11: tab_Fahrt

Eingespeicherte Sendungen

Feldname	Felddatentyp	Beschreibung
SendID	Integer	Sendungsnummer
VonLine	Integer	Absenderlinie
VonGer	Integer	Sendegerät
NachAdr	Short	Empfangslinie
Umleitung	Short	Umleitegerät bei Umleitungen
AnkunftsAdr	Short	Adresse
StartDatum	Date	Datum und Uhrzeit der Sendungsaufgabe
EndeDatum	Date	Datum und Uhrzeit des Sendungsendes
Status	Short	Status der Sendung

Tabelle 12: tab_Fahrtdetail

Protokollierung der Sendungen

Feldname	Felddatentyp	Beschreibung
SendID	Integer	Sendungsnummer
Datum	Date	Datum und Uhrzeit der Sendung
Line	Integer	Liniennummer
GerId	Integer	Gerätenummer
Code	Integer	Aktionen, die während der Fahrt der Büchse von den Stationen rückgemeldet werden (Rohrschalter, Motorbewegung, ...)

Tabelle 13: tab_Stati

Namenstabelle für alle Stati

Feldname	Felddatentyp	Beschreibung
Name	Char Width 50	
ValueN	Integer	
ValueS	Char Width 50	
Description	LongChar Attribute 32	

Tabelle 14: tab_Language

Spracheneinstellung

Feldname	Felddatentyp	Beschreibung
LanguageID	Integer	Identifikation der Sprache
Language	Char Width 50	Name der Sprache

Tabelle 15: tab_Fahrtdetaildescription

Text für die Sendungsdetails

Feldname	Felddatentyp	Beschreibung
Code	Integer	Code aus Tabelle Fahrtdetail
Descr	Char Width 100	Beschreibung des Codes
Sprache	Integer	Sprachnummer

Tabelle 16: tab_Gerätearten

Beschreibung der Stationstypen

Feldname	Felddatentyp	Beschreibung
GerTyp	Integer	Gerätetyp
GerBezeichnung	Char Width 50	Name des Geräts

Tabelle 17: tab_Ankunftssignale

Beschreibung der Ankunftssignale

Feldname	Felddatentyp	Beschreibung
Nr	Integer	Nummer des Ankunftssignals
Signaltext	Char Width 50	Name des Ankunftssignals

4.3.1.4.5 Kommunikations- und Sicherheitsmodul

Diese Komponente ist zuständig für die Kommunikation zwischen den Rechensystemen. Für die Datenübertragung werden unterschiedliche Protokolle verwendet. Die Kommunikation zwischen Leitrechner und Kommunikationsprozessor erfolgt mit dem USB- Protokoll. Der Kommunikationsprozessor ist dabei der Server, der auf die Anforderungen des Clients wartet. Der Leitrechner ist der Client und damit jene Komponente die aktiv werden muss. Der Leitrechner kann Daten zum Kommunikationsrechner senden oder Daten anfordern.

Das CAN- Protokoll wird für die Datenübertragung zu allen Stationen verwendet. Für die Übertragung der Daten zwischen Leitrechner und Kommunikationsrechner werden mehrere CAN- Pakete zu einem Block zusammengefasst und dieser Block wird mit dem USB-Protokoll übertragen. Im Leitrechner erfolgt die Überprüfung dieser Blöcke auf Konsistenz und Vollständigkeit und die Zerlegung in die einzelnen CAN- Pakete. Die CAN- Pakete werden durch das USB- Protokoll getunnelt.

Auf der Serverseite müssen in diesem Modul eine erhöhte Sicherheit bezüglich des Datenverlustes durch Kommunikationsfehler/-ausfall des TCP-IP-Protokolls kompensiert werden. Zusammen mit den Notlaufeigenschaften der Endgeräte ist diese Schicht für eine Stabilisierung des Gesamtsystems von eminenter Bedeutung.

4.3.1.4.6 Automatisches Anrufmodul (AAS)

Die Abwicklung der Anrufe erfolgt über ein ISDN- oder analoges Modem. Für die Anwahl des Modems werden Standard TAPI (Telefon Applikation Programm Interface) Module verwendet, auf denen eine Sprach- Maschine aufsetzt. Mittels dieser Maschine erfolgen die Auswahl der Sprache sowie die Stimmart (Männer- oder Frauenstimme).

4.3.1.5 Verwendete Software

Für die Programmerstellung auf dem Personal Computer wurden ausschließlich Microsoft – Produkte verwendet.

- Für die Programmentwicklung wurde das Softwarepaket MS – Visual Studio 6.0 verwendet.
- Zusätzlich wurden von Microsoft die folgenden Produkte verwendet.
  - System Development Kit (SDK)
    Im SDK sind Erweiterungsroutinen für den Systemzugriff enthalten.
  - Driver Development Kit (DDK)
    Diese Erweiterung dient zum Zugriff auf Gerätetreiber der Firma Microsoft.
  - Speach System Development Kit (Speach SDK)
    Die Erweiterung wird für die Sprachausgabe verwendet.
- Für die Speicherung der Daten wurde das Datenbanksystem MS – Access verwendet.
- Für die Telefonanbindung über einen ISDN- Adapter wurden zusätzliche DLL von der Firma AVM benötigt.

4.3.1.6 Installation der Software auf einem PC

Die gesamte Realisierung der Leitrechner- Software wurde mit Hilfe von Microsoft Betriebssystemen und Tools durchgeführt.
Diese Benutzerdokumentation erläutert die einzelnen Funktionen der Komponenten der Transportsteuerung RPCONTROL. Diese Komponenten betreffen im einzelnen:

- **das Steuerungsprogramm RPCONTROL**
  enthält die gesamte Steuerungslogik und die Anzeige
- **die Datenbank**
  enthält die Struktur der Anlage sowie alle notwendigen Adressen und Bezeichnungen, sowie die Basisdaten für Statistiken
- **das optionale Programm RTA**
  realisiert sprachliche Ankunftsmeldungen über Modem

Einzelne Menüpunkte werden wie folgt dargestellt: **`Ansicht/Zoom`**. Dies bedeutet den Aufruf des Menüpunktes **`Ansicht`** in der Menüleiste und anschließend Anklicken des Unterpunktes **`Zoom...`**. Dies kann beliebig tief verschachtelt werden. Mit den gleichen Textmerkmalen werden die Menüpunkte des Speed-Menüs (rechte Maus) gekennzeichnet.

Das Programm RPCONTROL kann in verschiedenen Sprachen ausgeführt werden. (Umschaltung vergleiche Kapitel 0 **Die Menüleiste**). Diese Dokumentation verwendet ausschließlich die Begriffe für die Sprache Deutsch. Bei anderen Sprachen wird nur der Text ersetzt, die Funktionen bleiben gleich.

Ein generelles Verständnis für die Funktion von pneumatischen und mechanischen Transportsystemen wird vorausgesetzt.

Installationshinweise

Voraussetzungen

USB-BOARD !*! NICHT !*! anstecken

Das USB-Board UNBEDINGT erst NACH erfolgreicher Basisinstallation von Win2000 installieren. Die mitgelieferten Basis-Treiber für WIN-2000 sind NICHT für die Verwendung dieses Produktes geeignet!

Durchführung

Windows 2000(Professional oder Server) und Office 2000/Access 2000

Voraussetzung für das gesamte Programm ist eine Standardinstallation von Win2000 und optional Office 2000/Access 2000.

Die meisten Installationsvorgänge müssen als Administrator durchgeführt werden.

Es wird empfohlen einen weiteren User anzulegen (Vorgangsweise: rechter Mausklick auf das Symbol 'Arbeitsplatz' am Desktop und dann 'Verwalten/Lokale Benutzer und Gruppen/Benutzer'. Wiederum rechter Mausklick auf Benutzer und dann 'Neuer Benutzer ...'. Die erforderlichen Daten eingeben. Dann in den Eigenschaften des neu angelegten Benutzers (rechte Maus auf 'MyUser',.....) den Bereich 'Mitgliedschaft auswählen und die Rechte für 'Hauptbenutzer' hinzufügen. Weitere Hinweise finden sich in der Dokumentation von Win2000.)

HINWEIS:

Um die Anlage zu programmieren und Auswertungen mit der Datenbank erstellen zu können, ist es notwendig Microsoft Office 2000 bzw. Access 2000 zu installieren. Das Programm RP_Control selbst arbeitet aber OHNE Installation von Office 2000

Sollte ein Netzwerk vorhanden sein, und eine beliebige Workstation hat Office 2000 installiert, so kann von dort auf die Datenbank zugegriffen werden, und die notwendigen Programmierungen können somit von dieser Arbeitsstation vorgenommen werden.

Erstellen eines Arbeitsverzeichnisses und Kopieren der Dateien (Administrator)

Es wird ein Verzeichnis benötigt, in das anschließend alle wesentlichen Programmteile kopiert werden. (z.B.: C:\Transro 2000)

Die Datenbank 'DB_TRANSRO.mdb' darf NICHT umbenannt werden, sie enthält alle Eingabemasken, Programme und Auswertungen und kann daher bei Versionsänderungen leicht ausgetauscht werden. Die Datenbank für die eigentlichen Daten kann umbenannt werden, dann müssen aber UNBEDINGT ALLE EXTERN EINGEBUNDENEN TABELLEN NEU VERKNÜPFT WERDEN. (Vorgehensweise dazu: Access starten mit DB_TRANSRO, dann über Menü 'Extras/Datenbank-Dienstprogramme/Tabellenverknüpfungsmanager' die umbenannte Datei auswählen! Weitere Hinweise finden sich in der Dokumentation von ACCESS 2000)

Erstellen einer Verknüpfung am Desktop

Zur leichteren Handhabung ist es auch empfehlenswert am Desktop des USERS (!) eine Verknüpfung zu RPCONTROL und - wenn nötig - RTA herzustellen.

Angabe der ODBC-Verbindung (Administrator)

Aufruf von
'Start/Einstellungen/Systemsteuerung/Verwaltung/Datenquellen (ODBC) '.
Dort ist die Erstellung einer neuen System-DSN erforderlich. ➲

Hinzufügen 'Microsoft Access Treiber'. Datenquellenname = TRANSRO1 und dann die DB auswählen. (z.B.: C:\TRANSRO 2000\DB_TRANSRO.MDB).

USB-SUPPORT (Administrator)

Nach dem erstmaligen Anstecken eines neuen USB-Gerätes, das an diesem USB-Anschluss noch nicht verwendet wurde, muss der Treiber angegeben werden. Windows fordert dazu den Benutzer auf, den Speicherort des Treibers anzugeben, auch wenn er bereits bei einem anderen Port verwendet wurde! Es ist die Datei **Bulkusb.inf** aus dem Directory <**Install**>/**Win2000** zu verwenden

Kontrolle der USB-Geräte (als User Administrator)

Aufruf von
'Start/Einstellungen/Systemsteuerung/Verwaltung/Computerverwaltung'.
Öffnen der Liste für 'System/Geräteeinstellungen' in der rechten Fensterhälfte:
Sicherstellen, dass im Ordner USB-Controller KEINE Warnungen existieren.
Falls doch : ➲

Diese Einträge löschen, den Rechner herunterfahren, abschalten, und neu starten.
Dann sollten die USB-Geräte und die Treiber neu definiert werden.

AAS-SUPPORT

Installation der Fritzcard und der virtuellen Modems. Sicherstellen, dass in

'Start / Einstellungen / Systemsteuerung / Telefon- und Modemoptionen / Erweiterte Optionen'
der Eintrag
'AVM ISDN TAPI Services for CAPI' vorhanden ist.
(NUR für diesen ‚Telefonanbieter' ist das AAS möglich!)

HINWEIS: AAS.EXE muss gestartet sein, BEVOR RP_CONTROL gestartet wird! Dies erfolgt NICHT automatisch.

RPCONTROL

Genereller Aufbau des Bildschirms

Die folgende Abbildung zeigt einen typischen Bildschirm mit seinen einzelnen Komponenten (von oben nach unten):

- Menüleiste (enthält einige allgemeine Befehle)
- Symbolleiste (Schnellanzeige von wichtigen Fenstern)
- Fenster für die Linien (Visualisierung der Struktur der Linien, sowie der wichtigsten Aktivitäten)
- Fenster für einen Verteiler (wenn vorhanden)
- Fenster mit Hinweisen zu allen laufenden Fahrten
- Fenster mit den letzten Warn- bzw. Fehlerhinweisen.
- Statuszeile mit Kurzhinweisen auf ausgewählte Funktionen, Tastatureinstellungen und Uhrzeit

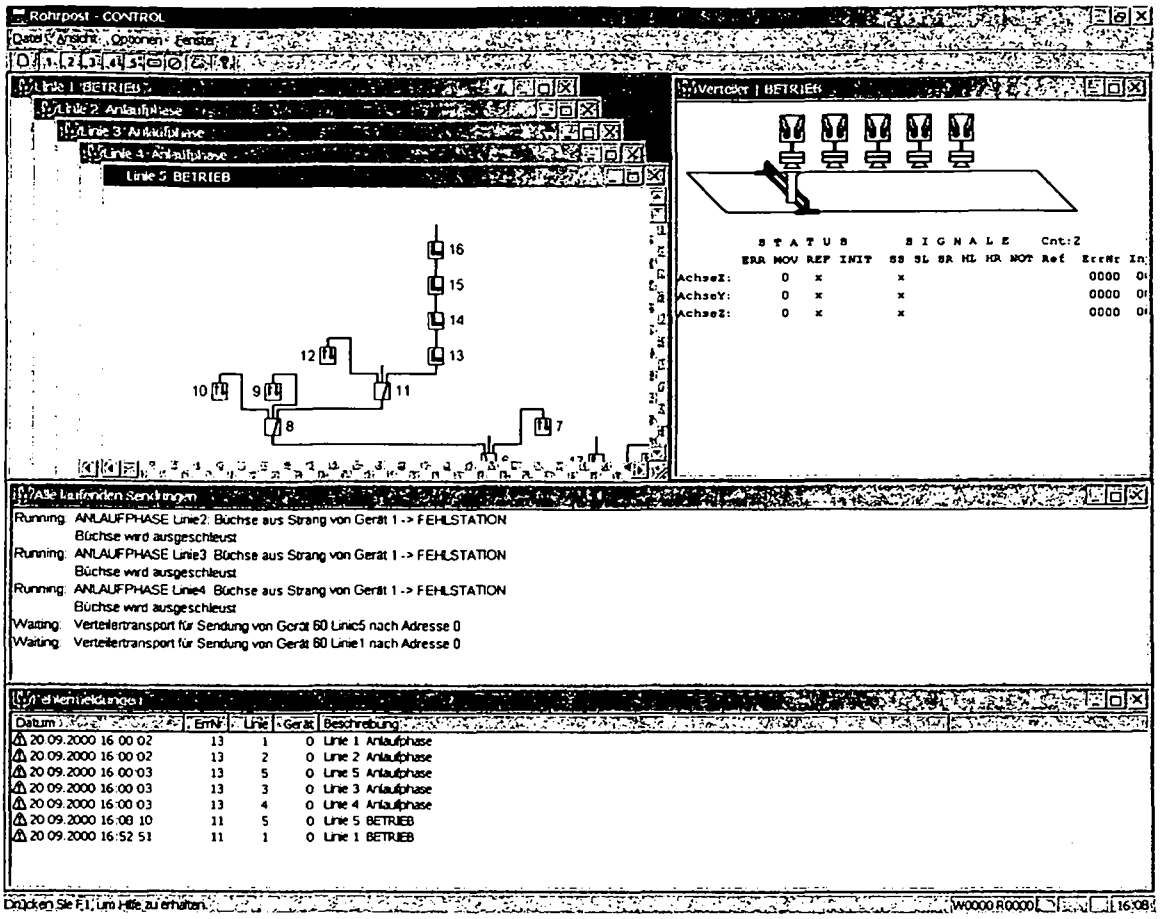


Abb.: 4.6 Grafische Benutzerschnittstelle

Die Menüleiste

Die Menüleiste besteht immer aus den 5 Bereichen **Datei, Ansicht, Optionen, Fenster** und **?**. Sie kann NICHT ausgeblendet oder verschoben werden. Im Folgenden werden nun die einzelnen Bereiche erläutert:

Datei:

Dieser Menüpunkt enthält die Umschaltung für die einzelnen Sprachen, sowie den üblichen **Beenden**-Eintrag. Dzt. stehen 3 Sprachen zur Verfügung: Deutsch, Englisch und Finnisch. Nach dem Auswählen einer Sprache wird der Benutzer in der neuen Sprache aufgefordert, das Programm neu zu starten.

Ansicht:

Hier können die Symbolleiste bzw. die Statusleiste ein- und ausgeblendet werden. Der letzte Eintrag **Zoom..** ist nur aktiv, wenn ein Anzeigefenster für Linien ausgewählt ist. Dann kann man verschiedene Zoomfaktoren für die Darstellung auswählen. Durch mehrmaliges klicken

können verschiedene Zoomstufen durchlaufen werden, nach der höchsten Vergrößerung beginnt man wieder mit der kleinsten Stufe.

Optionen:

Hier können dzt. zwei Dialogfelder für programmweite Parametereinstellungen geöffnet werden. Die zwei Einträge lauten:

`Parameter für Trocknung/Hausleitsystem`

In diesem Dialog wird zum einen festgelegt, ob eine Trocknung überwacht werden soll oder nicht. Falls die Trocknung aktiviert ist, wird von allen in der DB (vgl. Kap. Xxx) definierten Geräte mit den unten angeführten Parametern gesaugt. (in diesem Beispiel alle 20 min mit einer Dauer von 15 Sekunden)

Zum anderen wird hier festgelegt, ob bei Fehler oder Störung ein Signal an ein Hausleitsystem ausgegeben wird. Bei 0 wird KEIN Signal erzeugt. Der einzugebende Wert berechnet sich dabei aus der Formel:

$$Wert = LinienNummer * 100 + Gerätenummer$$

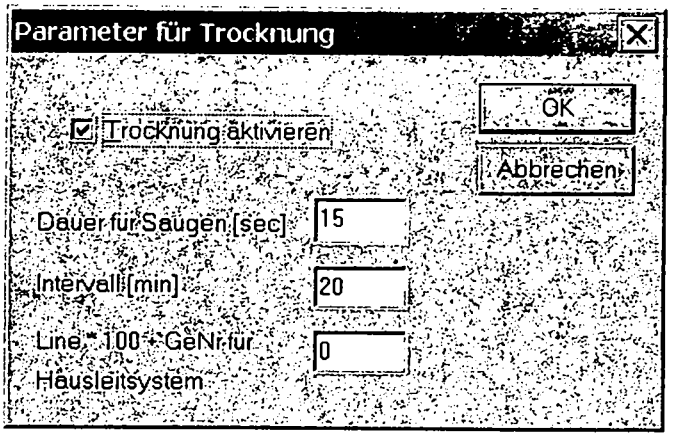


Abb.: 4.7 Parameter Trockensaugen

- **`Zeitintervall für Störung`**

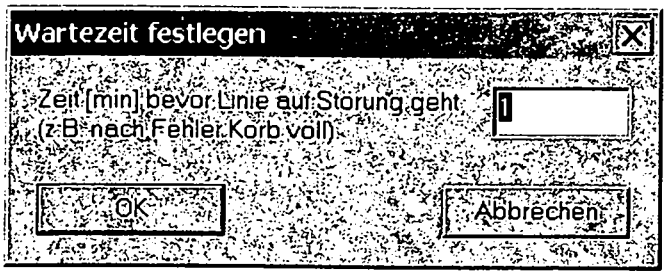


Abb.: 4.8 Störungsparameter

In diesem Dialog legt man die Zeitspanne fest, die das Programm bei einer fehlerhaften

Sendung wartet, bevor eine Störung ausgelöst wird (Es könnte diese ja in der Zwischenzeit behoben werden, speziell bei einem Korb-Voll-Fehler!).

Fenster:

Dieser Menüpunkt stellt die Windows- üblichen Funktionen zur Anwahl und Anordnung von Fenstern zur Verfügung.

?:

Dieser Menüpunkt stellt die Windows- üblichen Funktionen zur Anwahl des Hilfesystems und einer generellen Programminformation bereit.

Symbolleiste

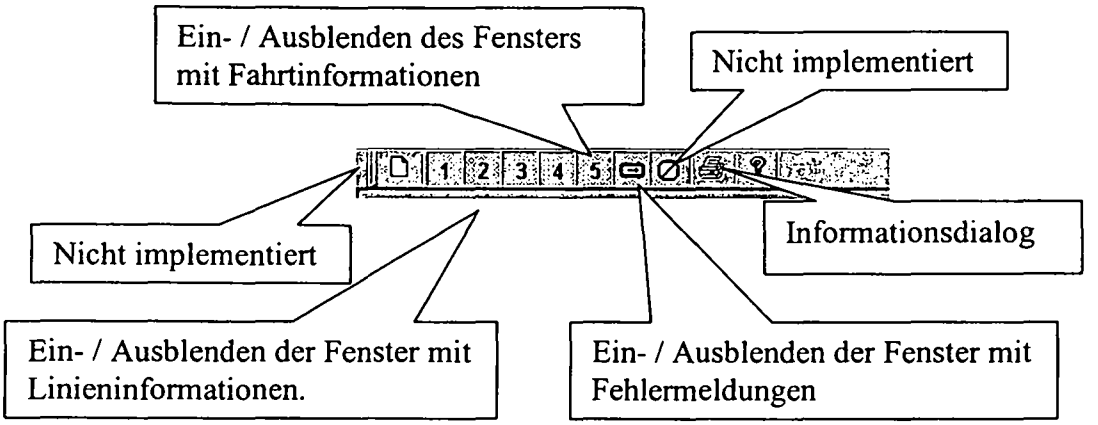


Abb.: 4.9 Symbolleiste

Linienfenster

Diese Fenster zeigen die logische Struktur einer Linie, die aktuellen Aktivitäten sowie in der Titelleiste des Fensters den momentanen `Status` der Linie.

Alle Geräte samt Verbindungen werden beim Start des Programms aus der Datenbank ausgelesen und anschließend von den einzelnen Linienfenstern angezeigt. Die verschiedenen Geräte werden durch unterschiedliche Symbole (die, die Funktionsweise etwas erklären sollen) dargestellt, der Zustand durch verschiedene Farben signalisiert(vgl. Abb.).

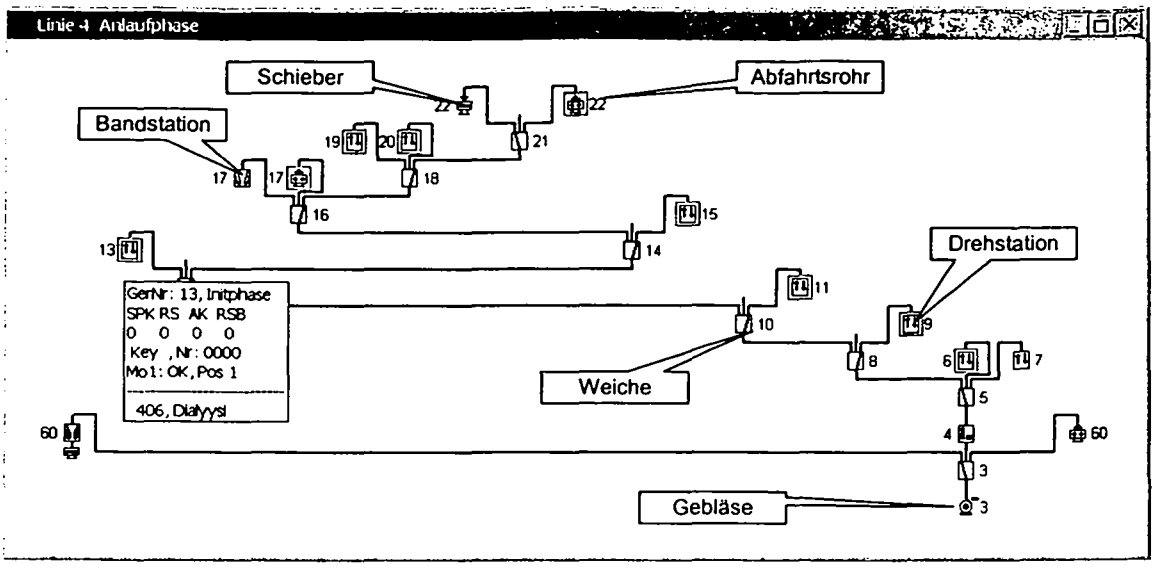


Abb.: 4.10 Anlaufphase

Folgende Rahmen werden benutzt:

- *KEINE farbige Umrandung:* Gerät ist in Betrieb
- *Ein oranger Rahmen:* Gerät wird gerade initialisiert
- *Ein roter Rahmen:* Gerät hat Störung oder wurde noch nicht initialisiert
- *Ein roter durchgestrichener Kreis:* Das Gerät ist außer Betrieb und wird NICHT angesteuert.
- *Ein gelber Rahmen:* Die Tastatur wurde vor kurzem betätigt.
- *Ein violetter Rahmen:* Eine Büchse (Fahrt) wartet auf das Freiwerden der Linie. Zusätzlich wird das Vorhanden-Sein einer Büchse im Aufgabetrichter durch einen roten Punkt in der linken unteren Ecke des Gerätes angezeigt.

Die Nummern neben den Geräten geben die CAN-BUS-Nummern an. Dadurch lassen sich Kombinationsgeräte (2 logische Geräte werden von einer physischen Platine gesteuert) leicht erkennen (vgl. Abfahrtsrohr und Schieber. CAN-Nr. 22). Verweilt man mit der Maus einige Zeit über einem Gerätesymbol, so werden etliche aktuelle Informationen bezüglich dieses Gerätes angezeigt. Im Einzelnen sind dies vor allem Gerätenummer, Gerätestatus, Schaltzustände der Rohrschalter, Motorenstatus, und - falls eine Tastatur vorhanden ist - die Nummer im Display, die zuletzt gedrückte Taste, sowie eine Liste aller zugeordneten Adressen (gelbes Rechteck in obiger Abb.).

Die folgende Abbildung zeigt die Darstellung einer aktiven Fahrt. Der pneumatische Weg wird, wenn das Gebläse läuft, entweder blau (Saug) oder rot (Druck) angezeigt. Auftretende Rohrschaltermeldungen durch eine fahrende Büchse werden bei dem jeweiligen Gerät durch

einen roten Punkt in der linken oberen Ecke signalisiert. Wurde ein Bremsrohrschalter aktiv, so wird dies durch einen roten Punkt in der rechten oberen Ecke des Gerätes angezeigt.

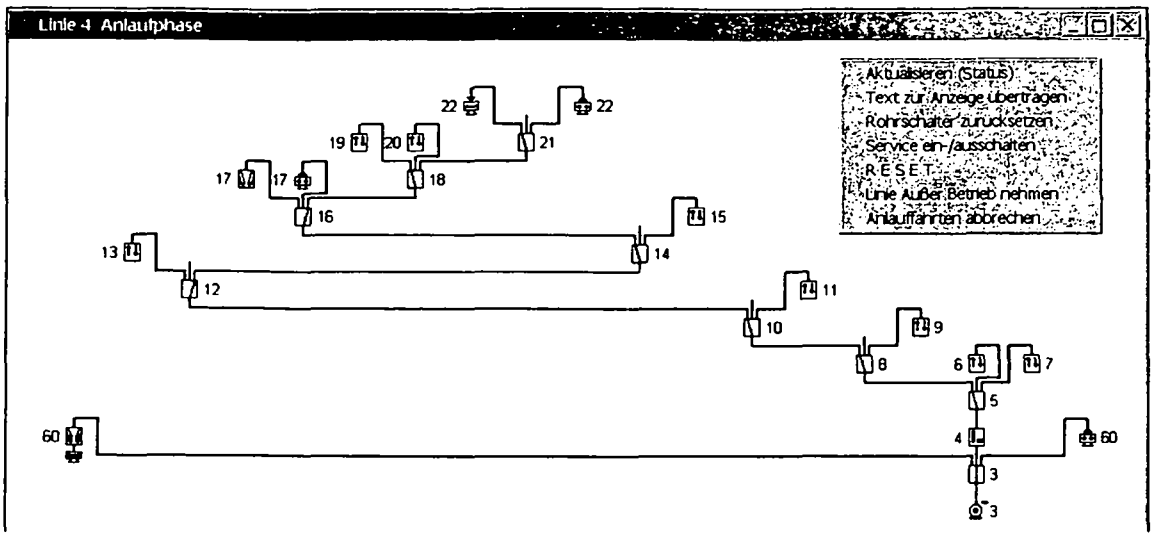


Abb.: 4.11 Anlaufphase

Jedes Linienfenster besitzt Speed-Menüs, die durch Verwendung der rechten Maus geöffnet werden können. Dabei ist der Inhalt der Menüs vom angeklickten Element, dessen Zustand und dem Linienstatus abhängig (vgl. obige Abb. – Klick in freie Fläche).

Im Folgenden werden alle auftretenden Menüeinträge erläutert. Prinzipiell gilt, ist ein Gerät angeklickt, so bezieht sich die folgende Auswahl nur auf dieses Gerät, anderenfalls ist es für die gesamte Linie gültig:

- **`Aktualisieren (Status):`**

  wird immer angezeigt. Der Status eines bzw. aller Geräte wird angefordert und sofort aktualisiert.

- **`Text zur Anzeige übertragen:`**

  wird immer angezeigt. Der Benutzer wird zur Eingabe eines Textes aufgefordert. Gleichzeitig kann auch eine Positionsangabe (wo im Display durchnummeriert beginnend bei 0) angegeben werden, Der eingegebene Text wird sodann an der angegebenen Position angezeigt.

- **`Rohrschalter zurücksetzen:`**

  wird immer angezeigt. Die dynamischen Rohrschalter werden gelöscht.

- **`Service ein-/ausschalten:`**

  Aktiviert bzw. beendet den Service-Mode. In diesem Zustand können alle Geräte individuell bewegt werden, und somit von der Zentrale aus eventuell diverse Fehler behoben werden, oder Servicearbeiten durchgeführt werden.

- **Service aus OHNE Anlaufphase:**

  Beendet den Service-Mode OHNE anschließend eine Anlaufphase zu starten.

  ACHTUNG: Es kann nicht mehr garantiert werden, dass im System keine Büchsen sind ! VORSICHTIG verwenden !

- **RESET:**

  Die Linie führt einen kompletten Neustart durch.

  ACHTUNG: es werden KEINE Werte aus der Datenbank gelesen.

- **Linie außer Betrieb nehmen:**

  Die Linie wird in den Status ‚Außer Betrieb' gesetzt. Dadurch können keine Sendungen mehr von/zu dieser Linie ausgelöst werden. Es muss ein **RESET** durchgeführt werden, damit die Linie wieder den Normalbetrieb aufnimmt.

- **Anlauffahrt abbrechen:**

  Beendet vorzeitig den gesamten Ablauf der Anlaufphase.

  ACHTUNG: Es kann nicht mehr garantiert werden, dass im System keine Büchsen sind ! VORSICHTIG verwenden !

Die folgenden Menüpunkte werden nur angezeigt, wenn ein einzelnes Gerät angeklickt wurde, und die Linie im Service-Mode ist.

- **Neue Position für Motor 1:**

  wird nur angezeigt, wenn dieser Motor vorhanden ist. Nach Auswahl einer neuen Motorposition aus dem angezeigten Dialog wird der entsprechende Steuerbefehl zu diesem Gerät geschickt. Die Reaktion kann entweder direkt am Bildschirm abgelesen werden, oder im Informationsfeld (mit Maus über Gerät warten) festgestellt werden.

- **Neue Position für Motor 2:**

  identisch mit obigem Punkt, nur für Motor2.

- **Bandmotor einschalten:**

  wird nur angezeigt, wenn dieser Motor vorhanden ist. Durch anklicken wird das Bandlaufwerk gestartet.

- **Büchse einschleusen:**

  startet einen kompletten Einschleusvorgang für dieses Gerät.

  ACHTUNG: Eine Büchse ist anschließend im Rohr !

- **Büchse ausschleusen:**

  startet einen kompletten Ausschleusvorgang für eine Büchse bei angewähltem Gerät.

- **Weg einstellen:**

  Es wird eine pneumatische Verbindung vom Gebläse zum ausgewählten Gerät hergestellt, indem alle Weichen entsprechend gestellt werden.

- **Gebläse aus:**
- **Gebläse Saug:**
- **Gebläse Druck:**
- **Gerätedaten ändern:**

  Im angezeigten Dialog können einige wesentliche Gerätedaten geändert werden. Die Änderungen werden auch in der Datenbank berücksichtigt.

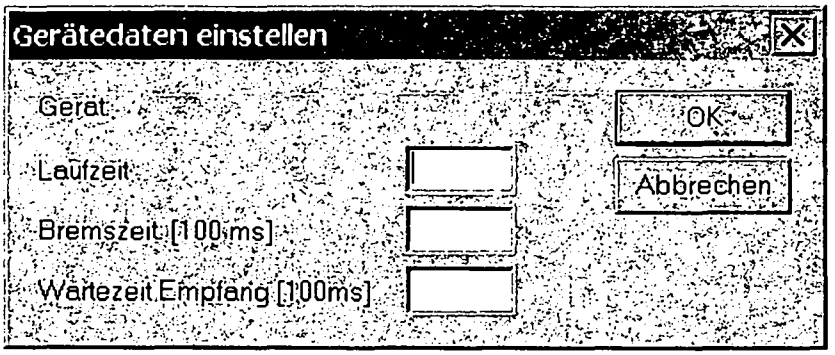


Abb.: 4.12 Gerätedaten

- **Set/Reset AKS:**

  Im angezeigten Dialog kann das Ankunftssignal ausgewählt werden, das gelöscht oder gesetzt werden soll.

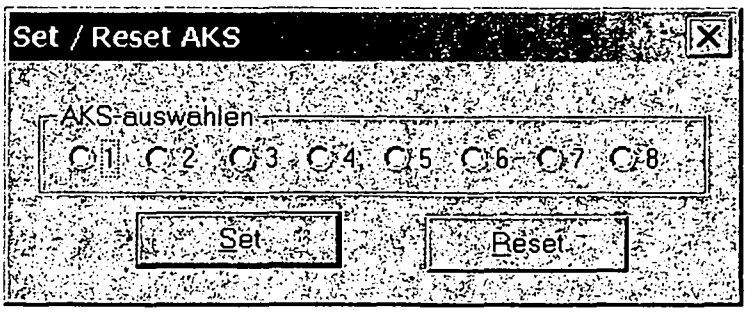


Abb.: 4.13 Ankunftssignale

- **Gerätereset:**

  Reset bei ausgewähltem Gerät durchführen.

4.3.2 Kommunikationsrechner

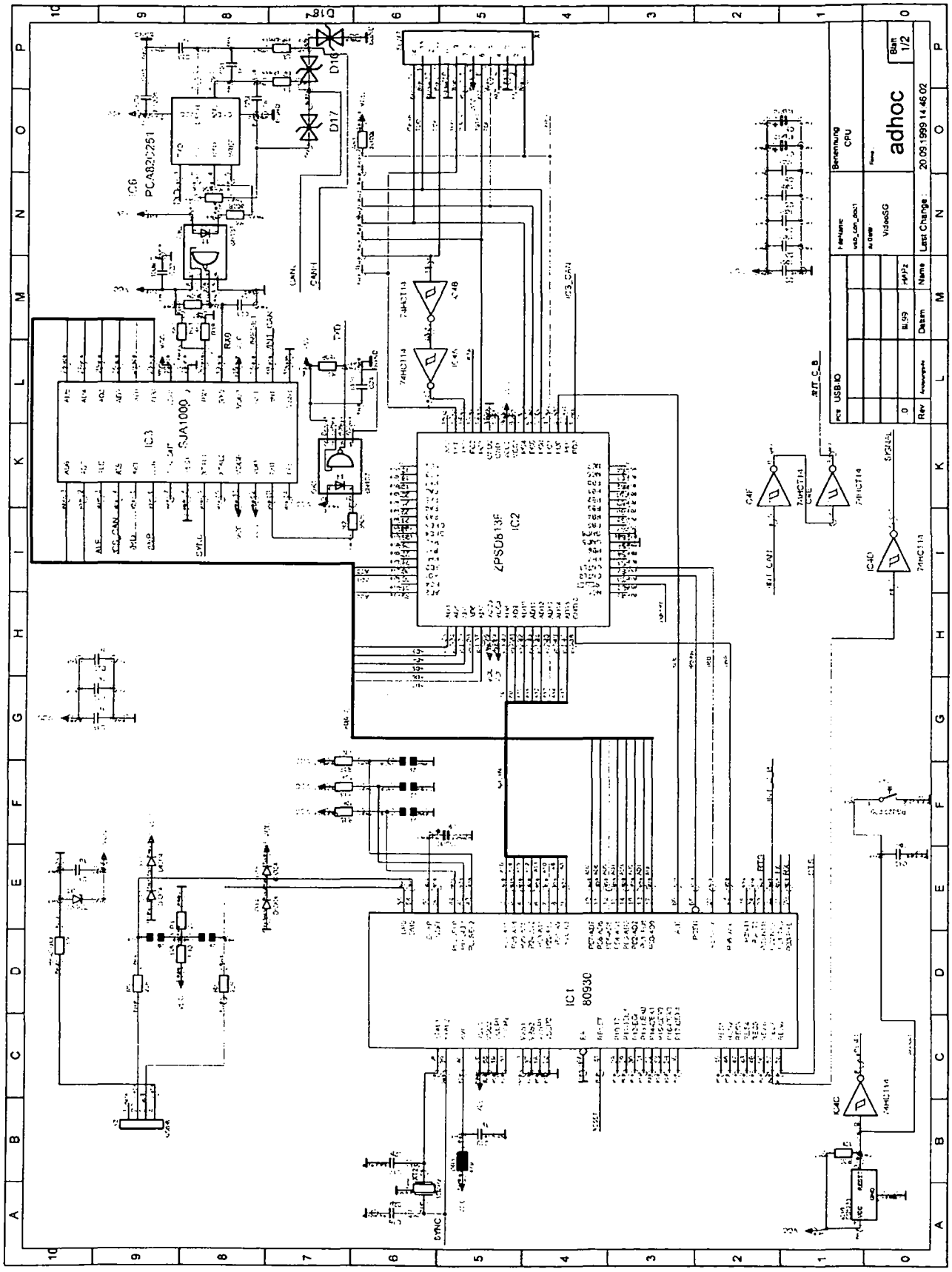
Die Hauptaufgabe des Kommunikationsrechners besteht aus dem Datentransfer zwischen Leitrechner und Stationsrechnern. Die Datenübertragung zum Leitrechner wird immer in Blöcken durchgeführt. Der Kommunikationsrechner speichert diese Blöcke in Puffern, zerlegt diese Blöcke in einzelne CAN– Meldungen und verschickt sie über den CAN– BUS zu den Stationsrechnern. Ebenso werden alle CAN– Meldungen von den Stationsrechnern in einem Puffer gespeichert und auf Anforderung zum Leitrechner übertragen. Neben dieser Aufgabe stellt der Kommunikationsrechner die hardwaremäßige Trennung zwischen störungsarmer Umgebung und störungsbehafteter Umgebung her. Die gesamte Recheneinheit des Kommunikationsrechners liegt in der störungsarmen Umgebung, was einen enormen Vorteil bringt und unnötige Ausfälle des Kommunikationsrechners durch Entladungen im Transportsystem vermeiden soll. Die CAN- BUS Schnittstelle ist das eigentliche Bindeglied zwischen diesen beiden Umgebungen.

4.3.2.1 Hardware

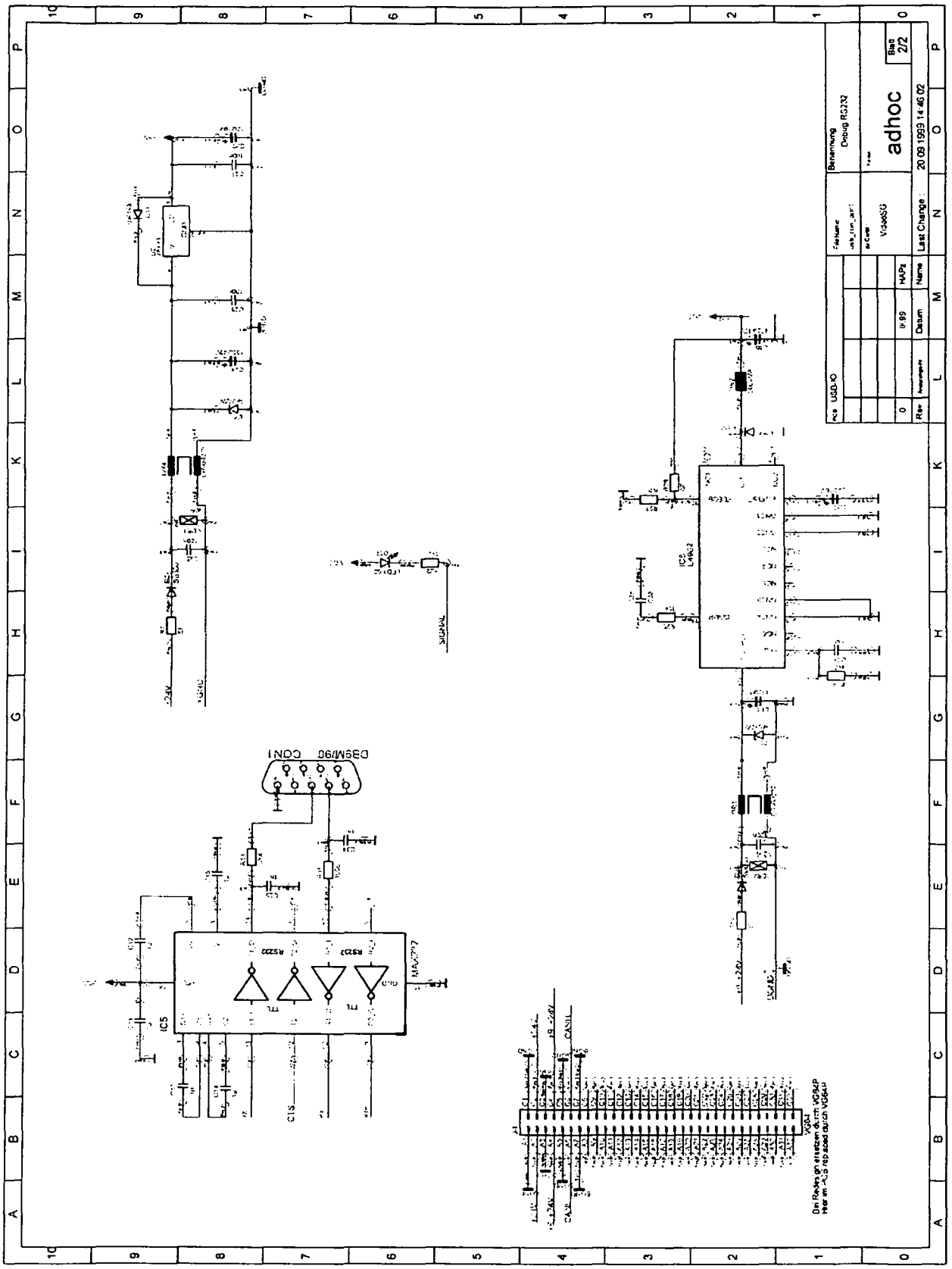
Für die Entwicklung des Kommunikationsrechners wurde das CAD – Paket „EAGLE" eingesetzt. EAGLE unterstützt einen vollständigen Entwurf, vom Schaltplan ausgehend bis zum fertigen Layout. Als Layoutgröße wurde das genormte Europakarten- Format (160mm x 100mm) verwendet. Für dieses Format gibt es eine Menge Standardgehäuse (19" Gehäuse). Für den Anschluss der Kabel wurde eine Anschlussplatine entwickelt. Alle Leitungen sind auf dieser Anschlussplatine steckbar ausgeführt. Um ein Verwechseln der Stecker zu verhindern wurden codierte Stecker verwendet. Die Kosten für codierte Stecker sind etwas höher, aber diese spart man bei der Montage der Anlage sicher ein. Die beiden Platinen sind über eine 96- polige Steckleiste miteinander verbunden.
Die Auswahl der Hardwarekomponenten erfolgte nach Verfügbarkeit, Funktionalität und Kosten. Die Entwicklung der gesamten Hardware dauerte einige Monate, anschließend wurde eine Prototypenfertigung durchgeführt. Nach der Testphase der Prototypen wurde eine größere Vorserie bestellt und hier war auf einmal die Verfügbarkeit der Bauteile das große Problem. Bei diversen Bauteilen betrug die Lieferzeit bis zu 12 Wochen, bei dem eingesetzten Flash– Speicher wurde keine Lieferzeit angegeben. Hier zeigte sich, dass eine Auswahl nur nach der Verfügbarkeit zu wenig ist. Für sämtliche Bauteile sollte es mindestens zwei wenn nicht mehrere Hersteller geben.

4.3.2.2 Aufbau der Hardware

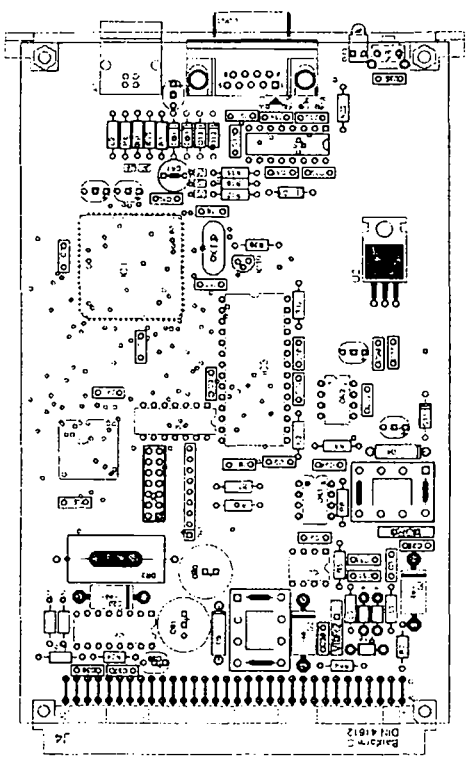
Im Folgenden sind die Schaltpläne abgebildet, die mit dem Programm „EAGLE“ der Firma Cadsoft erstellt wurden.



10.08.2005 13:28:04 C:/CanCon1.0/Leiterplatten/usb_con_1/usb_con_doc1.sch (Sheet: 1/2)



10.09.2005 13:26:53 C:/CanCon1.0/Leiterplatten/usb_con_1/usb_con_doc1.sch (Sheet: 2/2)



10.09.2005 13:28:23 C:/CanCon1.0/Leiterplatten/usb_con_1/usb_con_doc1.brd

4.3.2.2.1 Hardwarekomponenten

Als Mikrokontroller wurde die Intel CPU 80930 verwendet. Der Mikrokontroller hat zusätzliche Hardwarefunktionen integriert, jene die im Kommunikationsrechner verwendet werden, werden im folgenden Abschnitt beschrieben.

- Befehlskompatibel mit einer 8051 CPU.
  Die Programmierung der CPU erfolgt mit einem 8051 C – Compiler oder Assembler.
- USB – Transceiver
  Die gesamte Logik für den USB- Bus ist im Kontroller bereits integriert und die beiden Anschlüsse Dp0 und Dm0 müssen mit externen Widerständen beschalten werden.
- Serielle Schnittstelle
  Für die serielle Schnittstelle wird noch ein externer Treiber (MAX232) benötigt und die Anschlüsse sind auf einen 9 poligen D –SUB Stecker aufgeführt. Die Übertragungsrate der seriellen Schnittstelle wird über einen internen Timer verwaltet und softwaremäßig eingestellt. Die serielle Schnittstelle wird nur für Testzwecke verwendet.
- Internes RAM
  1024 Byte on chip RAM reichen bei diesem Projekt aus. Für Sende- und Empfangspuffer werden je 256 Byte benötigt, die restlichen 512 Byte stehen für Variablen zur Verfügung.
- Wach Dog Timer
  Der integrierte Watch Dog Timer wird für die Überwachung des Programmablaufs verwendet.
- Timer / Counter
  Ein integrierter Timer liefert die Zeitsignale für alle Softwarezeitglieder. Der Hardware Timer löst in bestimmten Zeitintervallen einen Interrupt aus, das laufende Programm wird unterbrochen und es wird in die Interruptroutine zur Zeitgliedbearbeitung verzweigt, diese ausgeführt und anschließend wieder zum unterbrochenen Programm zurückgesprungen. Der Einstellbereich der Zeitglieder ist immer ein Vielfaches der Zeit des Timerintervalls.

Nicht alle Funktionen des Kommunikationsrechners werden durch den Mikrokontroller abgedeckt, daher sind zusätzliche Hardwarebausteine erforderlich.

- Für den Programmspeicher und die Decoder – Logik wird ein spezieller Chip PSD813F eingesetzt. Dieser Baustein ist speziell entwickelt für Mikrokontrolleranwendungen und hat daher die Module Flash-Speicher, RAM– Speicher, EEPROM– Speicher, Decoderlogik und digitale Ein- und Ausgänge bereits integriert. Die Programmierung des Chips erfolgt in der Baugruppe über einen JTEG – Port. Der Flashspeicher ist in Seiten unterteilt, durch diese Seitenteilung ist eine Programmierung während des Betriebs möglich. Die gesamte Decodierung der Signale erfolgt in einem internen PAL, welches auch über den JTEG– Port programmierbar ist.
- Für den CAN– Bus wird der Stand alone CAN– Controller SJA1000 der Firma Philips eingesetzt. Dieser Kontroller unterstützt das CAN 2.0B– Protokoll. Als CAN– Transceiver wird der Baustein PCA82C251 eingesetzt. Dieser arbeitet nach dem ISO– 11898-24V Standard und ist für höhere elektrische Störungen ausgelegt. Die beiden Bausteine CAN– Kontroller und CAN– Transceiver sind galvanisch getrennt und bilden somit die Schnittstelle zwischen den Umgebungen. Die galvanische Trennung erfolgt über Optokoppler.

Betriebsdaten

Die Spannungsversorgung der Baugruppe erfolgt über die USB– Schnittstelle. Die Betriebsdaten der Baugruppe sind:

- Spannung 5V
- Strom 300mA

Für die Stromversorgung der CAN– BUS Schnittstelle ist ein eigener DC/DC Wandler vorgesehen, der über die Anlagenspannung die 24V erhält und 5V für die CA – Schnittstelle zur Verfügung stellt.

4.3.2.2.2 Schnittstellen

4.3.2.2.2.1 Serielle Schnittstelle

Die serielle Schnittstelle wird nur für Testzwecke verwendet, die Übertragungsrate ist im Programm fix auf 9600 Baud eingestellt. Am D– SUB Stecker sind nur die beiden Leitungen RxD und TxD ausgeführt. Auf Handshake - Leitungen wurde verzichtet.

4.3.2.2.2.2 USB – Schnittstelle

Die USB– Schnittstelle hat die Protokollversion 1.1 und eine Übertragungsrate von 12 Mbit/s. Über den USB– Anschluss erhält der Kommunikationsrechner die Versorgungsspannung. Die Standard USB– Komponenten liefern pro USB– Anschluss 5V und 500mA, das sollte für den Kommunikationskontroller reichen.
Die Adressierung der Kommunikationskontroller erfolgt über die Vendor ID und die Product ID die im Device_Descriptor gespeichert sind. Die Product ID muss pro Kommunikationskontroller eindeutig sein.

```
BEGIN_DEVICE_DESCRIPTOR:
  gDevice_bLength:             db  12h     ;DEVICE_DESCRIPTOR_LENGTH
  gDevice_bDescriptorType:     db  DEVICE_DESCR
  gDevice_bcdUSB:              dw  0001h   ;Version 1.00 compliant

  ;; gDevice_bDeviceClass:     db  00h
  gDevice_bDeviceClass:        db  04h     ;for testing

  gDevice_bDeviceSubClass:     db  01h
  gDevice_bDeviceProtocol:     db  00h
  gDevice_wMaxPacketSize0:     db  08h     ;8 byte max for EP0

  gDevice_widVendor:           dw  5E04h   ;Micros. Vendor ID (045E)
  gDevice_widProduct:          dw  0A93h   ;product ID

  gDevice_bcdDevice:           dw  0001h   ;device version 1.00
  gDevice_iManufacturer:       db  0h      ;These three fields are supposed
  gDevice_iProduct:            db  0h      ;to contain the index of strings
  gDevice_iSerialNumber:       db  06h     ;describing device.
  gDevice_bNumConfigurations: db  1
END_DEVICE_DESCRIPTOR:
```

4.3.2.2.2.3 Can – Bus

Der CAN – Bus stellt die Verbindung zur Anlage her. Die Übertragungsgeschwindigkeit ist in der Baugruppe einstellbar. Die möglichen Übertragungsgeschwindigkeiten sind in dieser Anlage 10000, 20000 und 50000 Baud. Das heißt, dass die maximale Leitungslänge 6Km sein kann bei 10000 Baud Übertragungsgeschwindigkeit. Für die CAN– Schnittstelle ist keine Adressierung vorgesehen, alle CAN– Pakete die empfangen werden, werden an den

Leitrechner weitergeleitet. In diesem Softwarestand ist nur eine Datenübertragung zwischen Leitrechner und Stationsrechner vorgesehen.

4.3.2.3 Software

Die gesamte Software für den Kommunikationsrechner ist in Assembler geschrieben. Die Datenübertragung zwischen Leitrechner und Kommunikationsrechner erfolgt im Bulk – Transfer (blockweise). Der Kommunikationsrechner wartet auf einen Datenblock vom Leitrechner, speichert diesen in den Empfangspuffer und sendet anschließend die Daten des Sendpuffers zum Leitrechner. Dieser Vorgang wird mittels Interrupt synchronisiert, immer wenn ein Datenblock im Endpoint der Usb – Schnittstelle steht wird ein Interrupt ausgelöst und dieser stößt den Datentransfer an. Um Engpässe bei der Datenübertragung zu vermeiden sie die internen Puffer immer doppelt ausgelegt und es werden bei jeder Übertragung die Puffer umgeschalten. Diese Umschaltung hat den Vorteil, dass jeder Schnittstelle immer ein freier Puffer zur Verfügung steht.

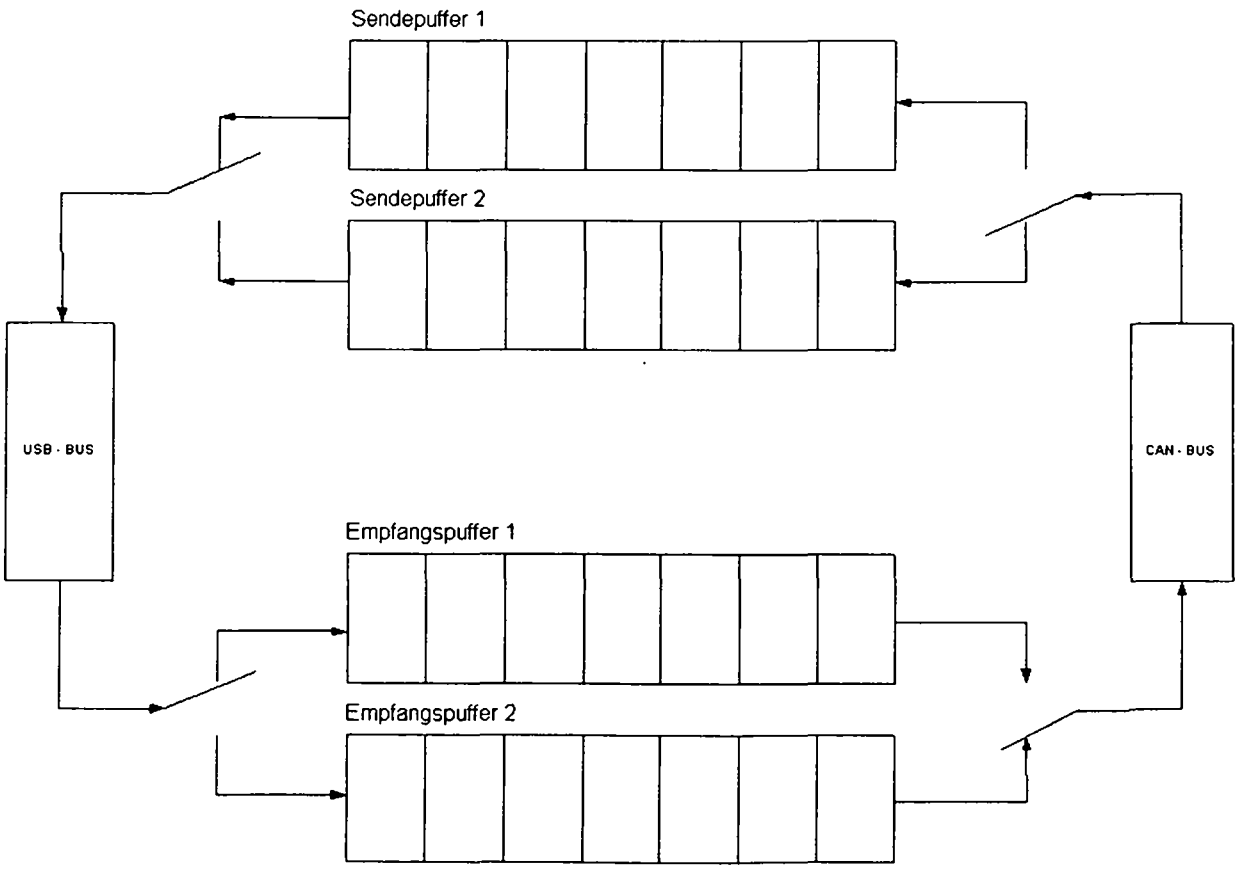


Abb.: 4.14 Pufferverwaltung

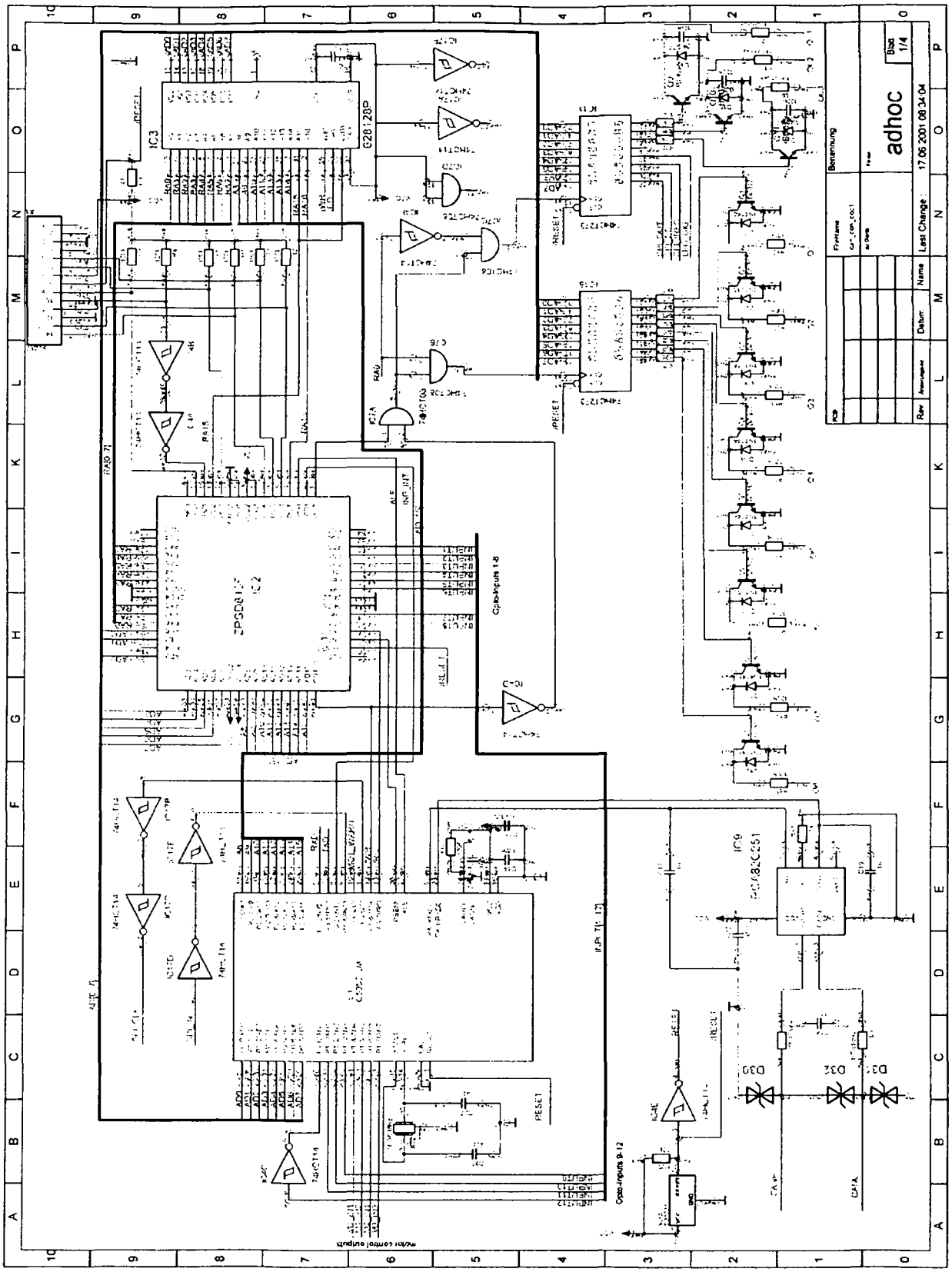
Wenn nun Daten vom Leitrechner kommen, so werden diese in den Sendepuffer 1 übertragen. Anschließend werden die Schalter S1, S2, S3 und S4 umgeschalten und die Daten des Sendpuffers 1 zur Versendung freigegeben. Die Daten des Empfangspuffers 2 werden zum Leitrechner übertragen und im Empfangspuffer 1 werden die Daten von der CAN– Schnittstelle zwischengespeichert. Der Datenaustausch zwischen Kommunikationsrechner und Leitrechner erfolgt alle 500mS. Dieser Wert wurde bei der Einschaltung der ersten großen Anlage durch zusätzliche Softwareroutinen im Kommunikationskontroller verifiziert und als geeignet empfunden. Die Verifikation zeigte, dass die Puffer im normalen Betrieb maximal bis zu 30% und bei Stoßzeiten bis zu 85% ausgelastet sind.
Die Datenübertragung zur und von der CAN– Schnittstelle wird über Interrupt gesteuert. Wird ein Datenpaket empfangen, so wird von der CAN– Schnittstelle ein Interrupt ausgelöst und das Datenpaket zum Empfangspuffer übertragen. Die Datenpakete auf der CAN– Schnittstelle haben immer eine konstante Länge von 10 Byte. Wenn ein neuer Sendepuffer gefüllt wurde, wird die Interruptroutine zum Datenversenden aktiviert. Die CAN– Schnittstelle löst einen Interrupt aus und die Daten werden vom Sendepuffer zum CAN– Kontroller transferiert.

4.3.3 Stationsrechner

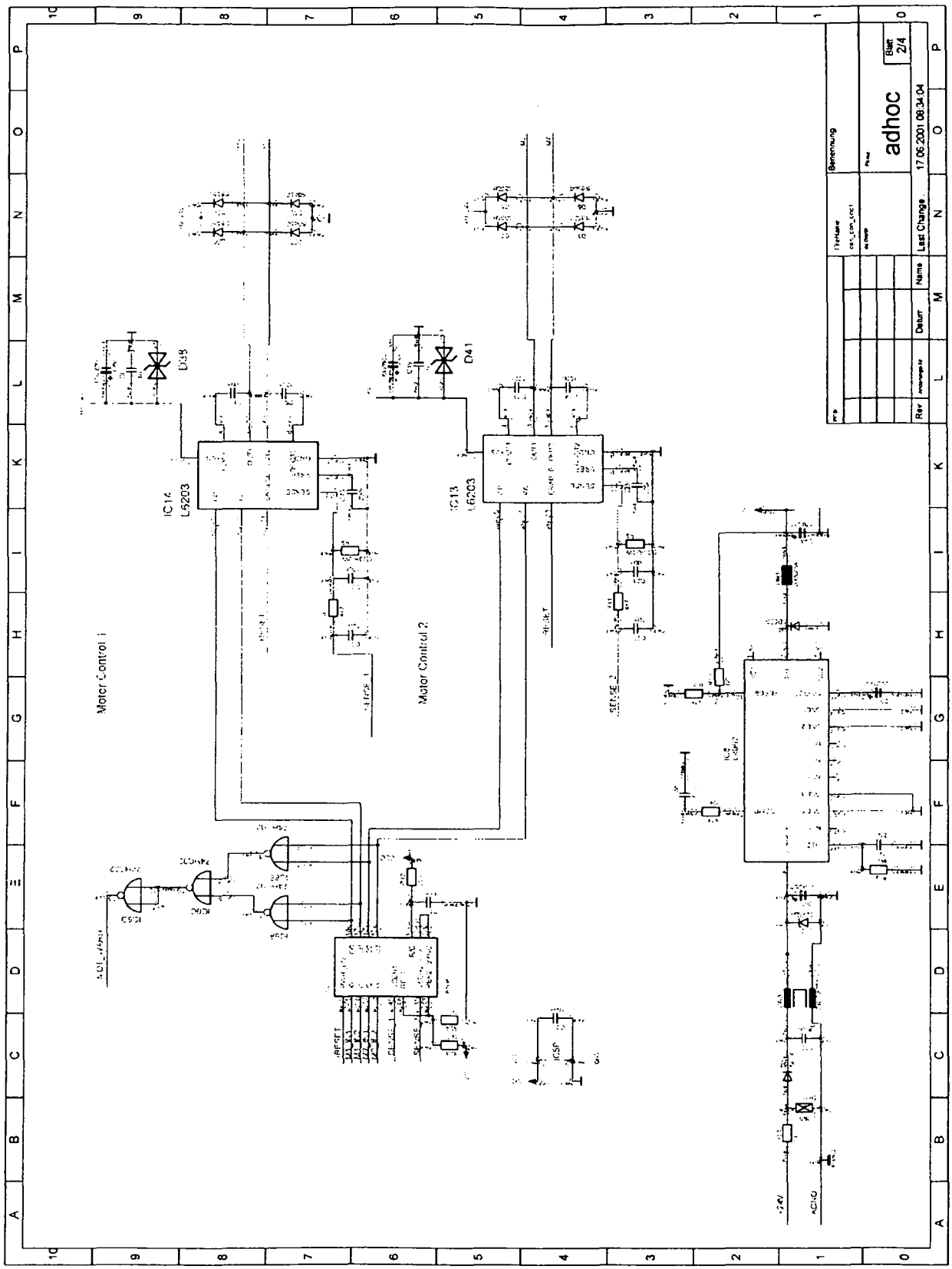
Der Stationsrechner ist jener Teil der Anlage, der alle Befehle vom Leitrechner an der Station umsetzt und die Benutzereingaben an den Leitrechner übermittelt. Der Stationsrechner ist jene Komponente, die den meisten Störeinflüssen ausgesetzt ist. Der Stationsrechner verändert nach einem Reset nichts an den Motorpositionen, die digitalen Ausgänge werden alle ausgeschalten und in der Anzeige erscheint eine Standardmeldung. Alle Initialisierungsvorgänge die an der Station durchgeführt werden müssen, werden vom Leitrechner übernommen. Diese Vorgangsweise ist auch ein Teil des Wiederanlaufkonzepts. Fällt während eines Transportvorgangs ein Stationsrechner aus, so sollen nach Möglichkeit keine Aktoren vom Stationsrechner aus aktiviert werden. Nach einem Rücksetzen des Stationsrechners im Fehlerfall und Beenden der lokalen Initialisierungsroutine kann nun der Leitrechner gezielt weitere Schritte durchführen.

4.3.3.1 Hardware

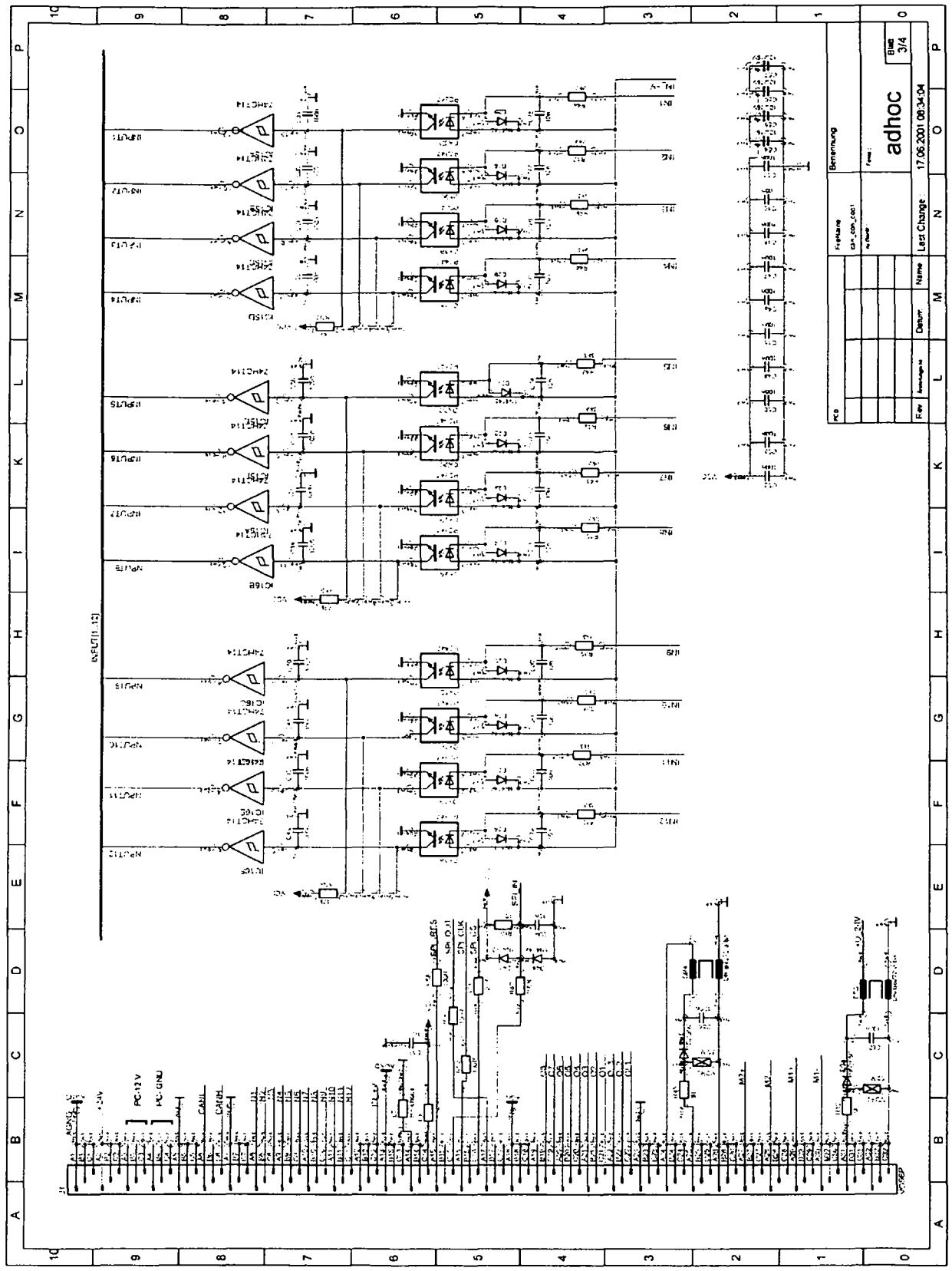
Im Folgenden sind die Schaltpläne die mit dem Programm „EAGLE“ der Firma Cadsoft erstellt wurden.



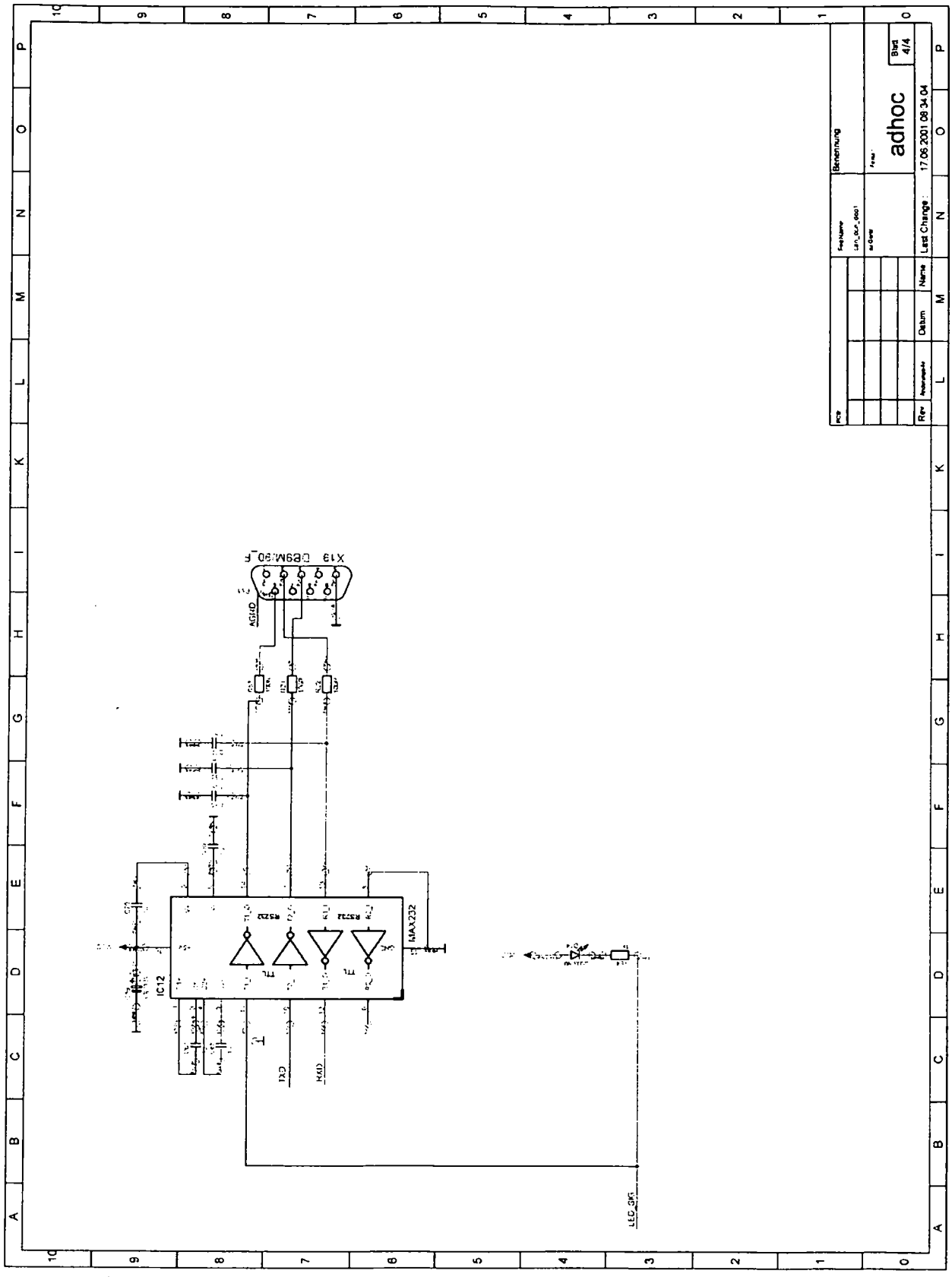
10.09.2005 12:28:41 C:/CanCon1.0/Leiterplatten/cancon/can_con_doc1.sch (Sheet: 1/4)



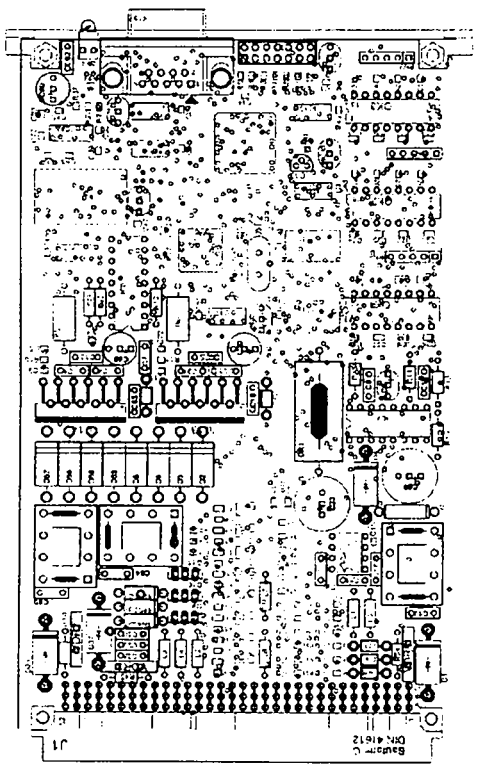
10.09.2005 12:38:17 C:/CanCon1 0/Leiterplatten/cancon/can_con_doc1.sch (Sheet: 2/4)



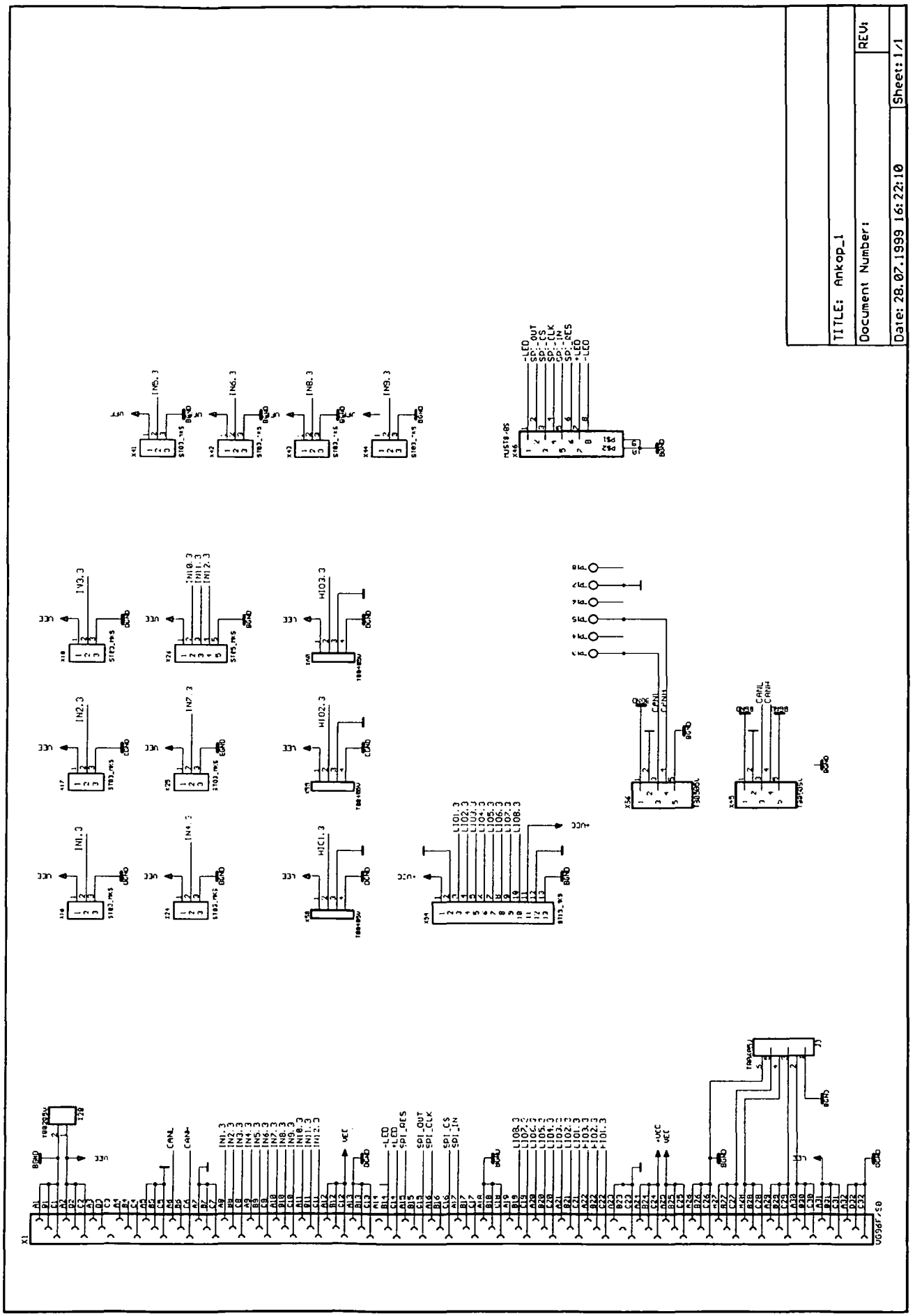
10.09.2005 12:37:16 C:/CanCon1.0/Leiterplatten/cancon/can_con_doc1.sch (Sheet: 3/4)

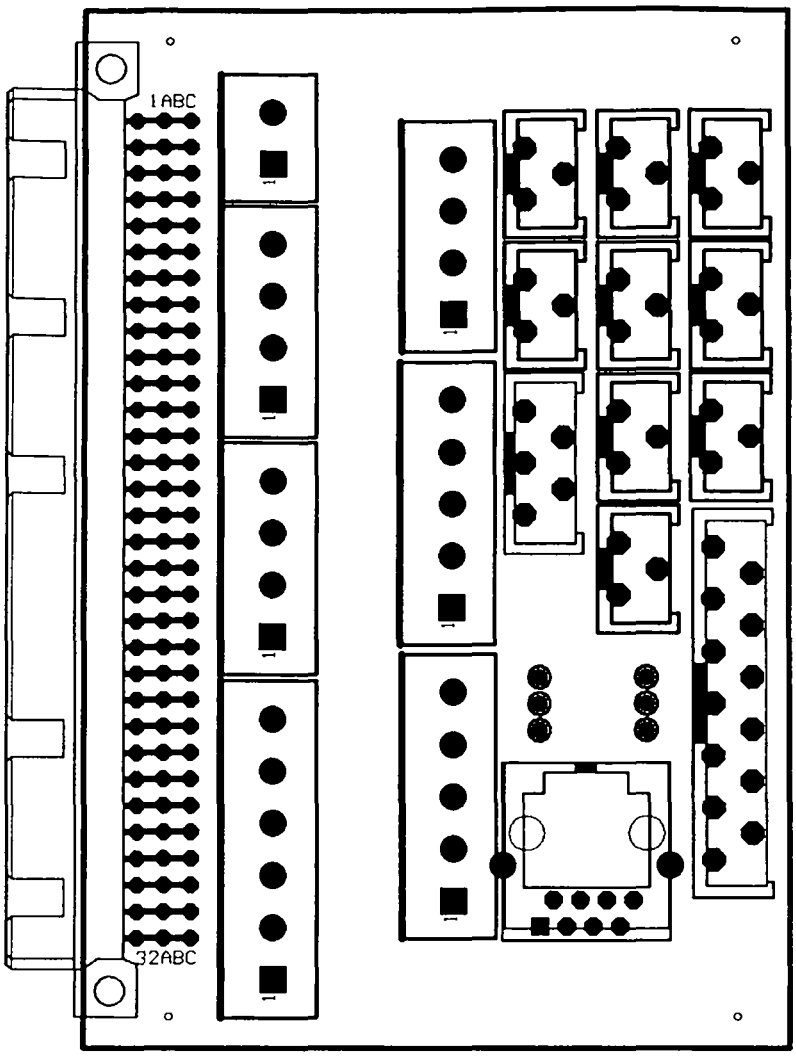


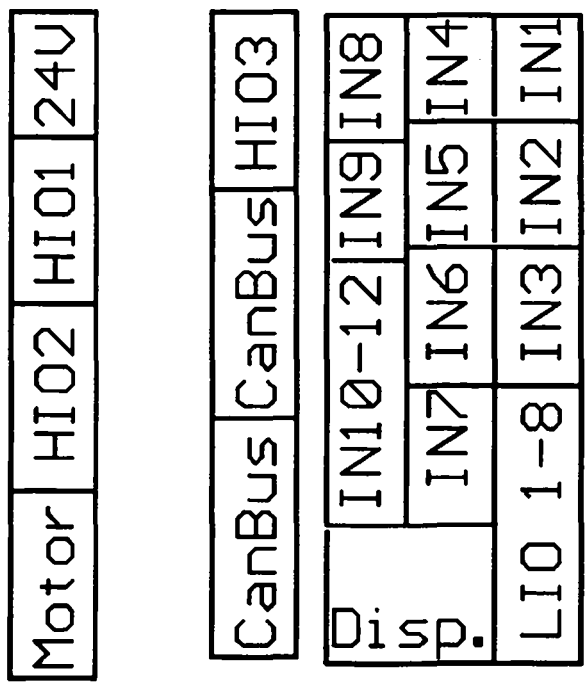
10.08.2005 12:37:57 C:/CanCon1.0/Leiterplatten/cancon/can_con_doc1.sch (Sheet: 4/4)

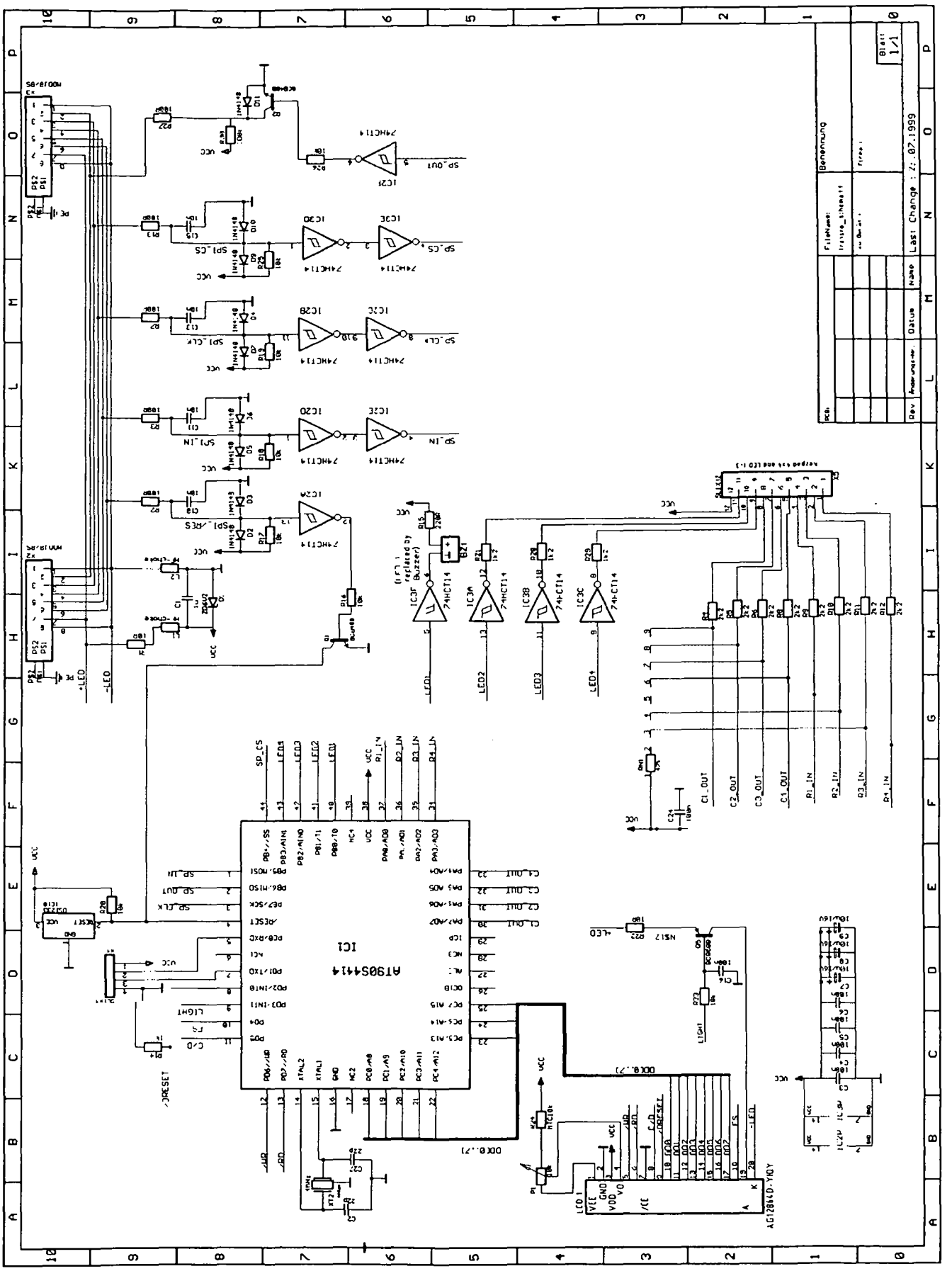


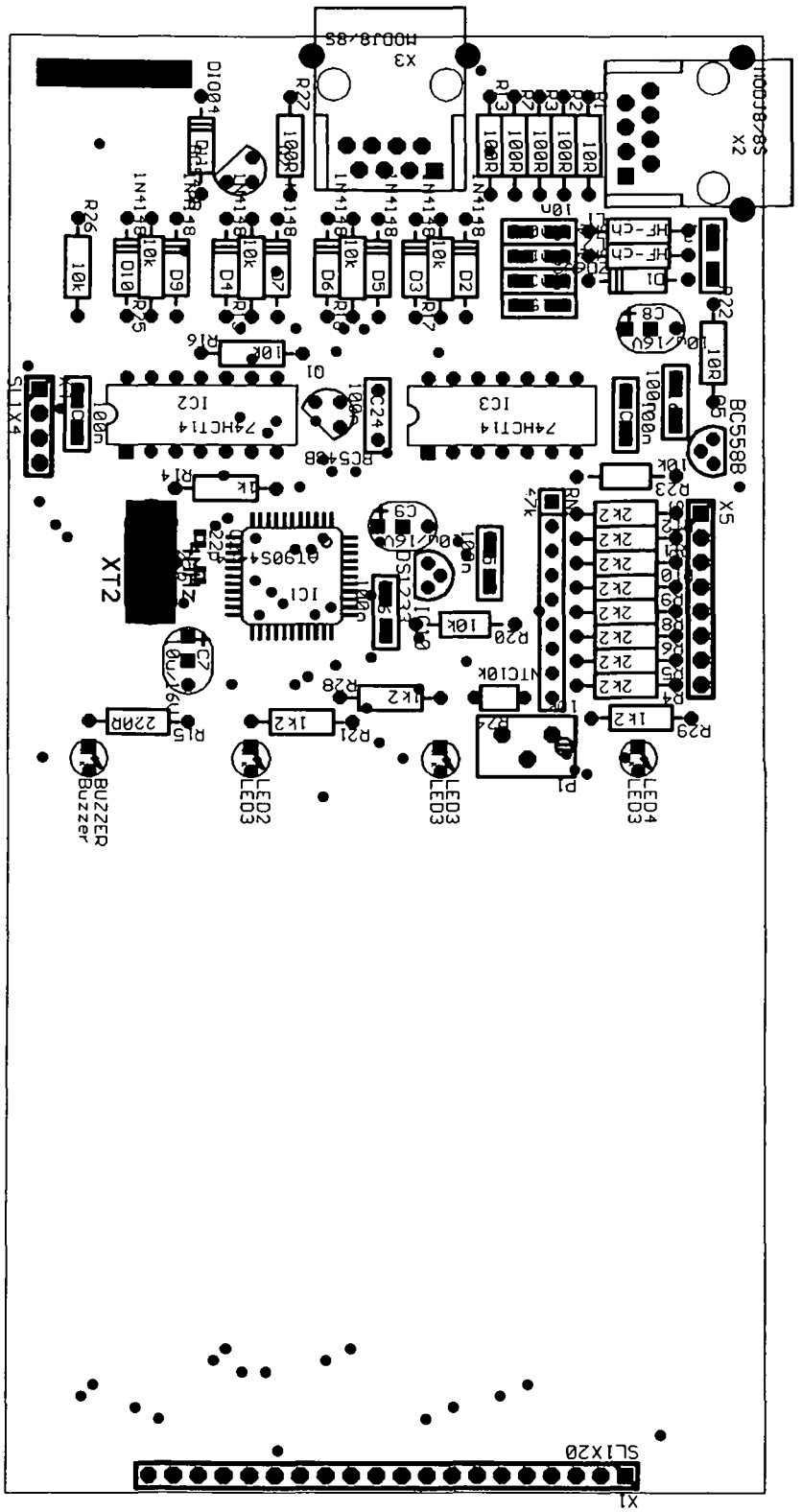
10.08.2005 12:40:17 C:/CanCon1.0/Leiterplatten/cancon/can_con_doc1.brd


TITLE: Ankop_1
Document Number:
REV:
Date: 28.07.1999 16:22:10
Sheet: 1/1


1 ABC
32ABC


Motor
HI02
HI01
24V
CanBus
CanBus
CanBus
HI03
Disp.
IN7
IN10-12
IN6
IN9
IN5
IN8
IN4
LIO 1-8
IN3
IN2
IN1




MODJ8/8S
X3
X2
X5
X1
XT2
74HCT14
IC2
IC3
BC558B
BUZZER
Buzzer
LED2
LED3
LED4
SL1X20
SL1X4

4.3.3.2 Hardwareaufbau

4.3.3.2.1 Hardwarekomponenten

Die Auswahl der Hardwarekomponenten erfolgte nach dem gleichen Prinzip wie beim Kommunikationsrechner, nur die technischen Anforderungen sind etwas anders.

- Die verwendete CPU C505C von Infineon ist ein 8051 Derivat und hat einen 8051 Befehlssatz.
- CAN– Transceiver
  Die gesamte Logik für den CAN- Bus ist im Kontroller bereits integriert, für die Leitungsanschaltung wird nur noch ein CAN- Transceiver benötigt.
- Serielle Schnittstelle
  Für die serielle Schnittstelle wird noch ein externer Treiber (MAX232) benötigt und die Anschlüsse sind auf einen 9 poligen D– SUB Stecker aufgeführt. Die Übertragungsrate der seriellen Schnittstelle wird über einen internen Timer verwaltet und softwaremäßig eingestellt. Die serielle Schnittstelle wird nur für Testzwecke verwendet.
- Internes RAM
  On Chip stehen 1024 Byte RAM zur Verfügung
- Wach Dog Timer
  Der integrierte Watch Dog Timer wird für die Überwachung des Programmablaufs verwendet.
- Timer / Counter
  Ein integrierter Timer liefert die Zeitsignale für alle Softwarezeitglieder. Der Hardware Timer löst in bestimmten Zeitintervallen einen Interrupt aus, das laufende Programm wird unterbrochen und es wird in die Interruptroutine zur Zeitgliedbearbeitung verzweigt, diese ausgeführt und anschließend wieder zum unterbrochenen Programm zurückgesprungen. Der Einstellbereich der Zeitglieder ist immer ein Vielfaches der Zeit des Timerintervalls.

4.3.3.2.2 Externes RAM

Zusätzlich wurde ein externer RAM– Speicher mit einer Speichergröße von 128KBye verwendet. Gewisse Texte werden bereits bei der Anlageninitialisierung in den RAM– Speicher eines jeden Stationsrechners übertragen und bei Gebrauch nur noch über einen

Schalter aktiviert. Die Größe des benötigten Speichers hängt von der Anzahl der verwendeten Sprachen ab.

4.3.3.2.3 Motoransteuerung

Die Motoransteuerung erfolgt mit einer Transistorvollbrücke L6203. Es können 24V Gleichstrommotore bis maximal 3A Dauerstrom angesteuert werden. Der Motorstrom wird fortlaufend überwacht, bei einer Überschreitung der 3A wird der Motor über einen Taktgeber aus- und wieder eingeschaltet. Für diese Überwachung ist der Baustein L6506 eingesetzt.

4.3.3.2.4 Digitale Ausgänge

Die digitalen Ausgänge sind Transistorausgänge, mit zwei verschiedenen Stromstärken. Acht Transistorausgänge können maximal 1A schalten. Die restlichen drei Transistorausgänge können kurzfristig 3A schalten, im Dauerbetrieb aber maximal 1,5A. Die maximale Stromstärke aller Ausgänge darf bei Dauerbetrieb 3A nicht überschreiten. Die großzügige Überdimensionierung ist wegen der Einschaltströme diverser Aktoren erforderlich. Die Ansteuerung der Ausgänge erfolgt über die CPU, die vom Leitrechner die benötigten Informationen dafür erhält.

4.3.3.2.5 Digitale Eingänge

Die digitalen Eingänge sind alle über Optokoppler geführt. Der Optokoppler schaltet bei einem Strom von 5mA vollständig durch. Damit die CPU ein eindeutiges „0" oder „1" Signal bekommt ist dem Optokoppler ein Schmitttrigger nachgeschaltet. Der Schmitttrigger hat am Ausgang immer ein eindeutiges Signal. Die Eingänge werden von der CPU alle 5mS eingelesen und an den Leitrechner weitergeleitet.

4.3.3.2.6 Bedieneinheit

Die Bedieneinheit ist für den Benutzerdialog. Die Anzeige ist ein monochromes Grafikdisplay mit einer Auflösung von 128 x 64 Bildpunkten. Die Eingabe erfolgt über Folientasten. Über eine SPI – Schnittsteller erfolgt der Datenaustausch zwischen Anzeigeeinheit und Stationsrechner. Die angesteuerten Bildpunkte werden zusätzlich im Stationsrechner und auch im Leitrechner redundant gespeichert.

4.3.3.2.7 Anschlussplatine

Der Kabelanschluss erfolgt über eine eigene Platine. Die Steckverbinder für den Kabelanschluss können codiert werden um ein verwechseln der Kabel zu vermeiden.

4.3.3.3 Software

Für die Programmierung der Stationsrechner wurde ein C- Compiler der Firma „Tasking" verwendet. Auf ein Betriebssystem wurde bewusst verzichtet, um die Programmkomplexität so gering als möglich zu gestalten.
Das Hauptprogramm geht nach der Initialisierung in eine Endlosschleife, in der bei jedem Durchlauf alle Einstellungen der Aktoren durchgeführt werden. Weiters wird bei den Stationsrechnern die Tastatureingabe bearbeitet und nach Bedarf Daten zum Display geschickt.
Die Abfrage und Auswertung der Sensoren wird in der Timerinterruptroutine durchgeführt. Die Timerroutine wird in einem Zeitintervall von 2,5ms aufgerufen. Von diesem Zeitintervall werden alle weiteren Zeitglieder des Stationsrechners hergeleitet, diese können immer nur ein ganzes Vielfaches von 2,5ms sein.
Die Synchronisation für den Datenaustausch zwischen Timerroutine und Hauptprogramm wird über eine „Critical Region"[5,8] abgewickelt.
Die Datenübertragung vom CAN- Bus zum Rechner erfolgt ebenfalls mittels Interrupt. Der CAN- Kontroller hat insgesamt 15 Kanäle, von diesen 15 Kanälen werden 5 Kanäle zum Senden verwendet und die restlichen 10 Kanäle stehen zum Empfang der Daten zur Verfügung. Durch die effiziente Aufteilung der Kanäle bedarf es keiner weiteren Pufferung.
Alle Meldungen die der Leitrechner verschickt empfängt jeder Stationsrechner. Die Selektierung der Meldungen selbst erfolgt in der Interruptroutine. Hier wird aus der Meldung die Stationsadresse gefiltert und mit der eingestellten Stationsadresse verglichen. Stimmen diese beiden Adressen überein, so wird die Meldung verarbeitet. Stimmen sie nicht überein so wird die Meldung verworfen.

4.3.3.4 Verkabelung der Anlage

Die Verkabelung der Testanlage wurde von mir sehr sorgfältig durchgeführt. Diese Anlage ist relativ klein, sie besteht aus drei Stationen, einer Weiche und dem Gebläse, daher hielt sich der Verkabelungsaufwand auch in Grenzen. Die Kabeln wurden in eigenen Rohren verlegt.

Diese Anlage diente als Testanlage, die diversen Anlagenteile wurden immer so modifiziert, dass ein Großteil der Software getestet werden konnte. Die erste Anlage in der dieser Prototyp von Steuerung zum Einsatz kam, war eine große Anlage mit fünf Linien und bis zu 20 Stationen pro Linie. Die Verkabelung wurde dort durch eine Installationsfirma durchgeführt. Nach der Komplettierung der einzelnen Stationen mit den Rechnern und einer generellen Überprüfung der Versorgungsspannung wurde der Leitrechner in Betrieb genommen. Das Ergebnis war ernüchternd, es funktionierte keine Kommunikation. Die Kommunikation zwischen Leitrechner und Kommunikationsrechner war in Ordnung, zu und von den Stationen gab es keine Datenübertragung. Aus diesem Grund wurde ein Softwaretool entwickelt, ein so genanntes „CAN_PING", mit dessen Hilfe es möglich war, die Kommunikation zwischen den einzelne Stationen und dem Leitrechner zu testen. Der restliche Teil der Arbeit bestand darin, die Leitung des CAN- BUSes vom Kommunikationsrechner aus abschnittsweise zu verfolgen und diese Teilabschnitte zu testen bis die Kommunikation der Gesamtanlage vollständig funktionierte. Der CAN- BUS ist ein genormtes und vollständig getestetes System in der Automatisierungstechnik. Wenn es bei diesem Bussystem zu Übertragungsfehlern kommt, liegen diese häufig in der Verkabelung.

4.3.3.5 Kabeltypen

Für die Datenübertragung beim CAN –Bus müssen Kabeltypen verwendet werden, die dem ISO 11898 Standard entsprechen.

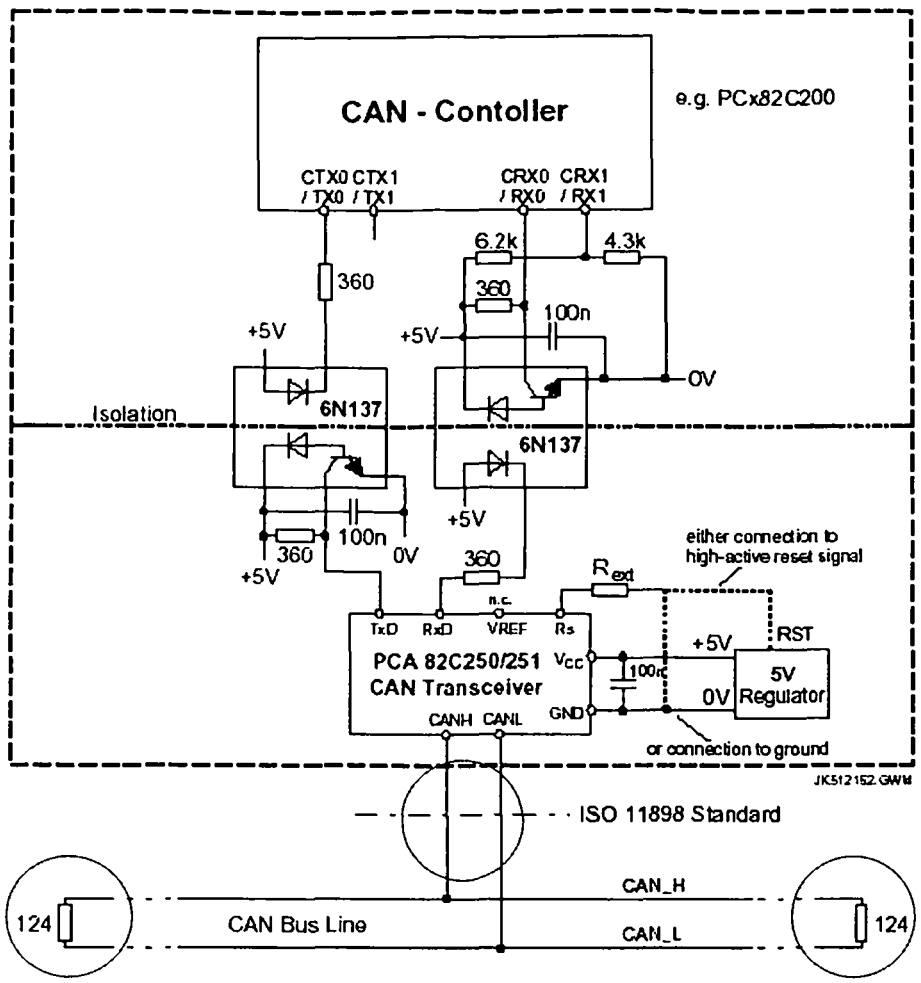


Abb.: 4.15 CAN-Bus Anschaltung

Diese Leitung muss an den beiden Enden mit je einem Widerstand von 120Ohm abgeschlossen sein. Der Anschlussleitung der Stationen an den Bus kann laut Definition bis zu einem Meter lang sein (Philips Semiconductors APPLICATION NOTE PCA82C250 / 251 CAN Transceiver AN96116):

Für die Stromversorgung kann ein Standardkabel eingesetzt werden, es ist dabei auf den Querschnitt zu achten, dass der Spannungsabfall auf der Leitung minimal ist.

4.3.3.6 Verlegung der Kabel

Nach Möglichkeit sollten das Datenübertragungskabel und das Versorgungskabel getrennt verlegt werden. Weiters sollten diese beiden Kabeln in einem gewissen Abstand vom Luftrohr der Anlage verlegt werden, um den elektrischen Störungen, die durch die elektrisch geladenen Büchsen entstehen, nicht unmittelbar ausgesetzt zu sein.

4.3.3.7 Abschirmung und Erdung

Für die Datenübertragung ist ein geschirmtes Kabel vorteilhaft. Der Schirm muss bis ans Ende des Kabels gehen. Weiters ist der Schirm auf beiden Seiten zu erden. Für die Erdung sollte eine Systemerde vorhanden sein. Bei pneumatischen Transportsystemen die sich über mehrere Gebäude ausbreiten, sind Lichtwellenleiter zwischen den Gebäuden einzusetzen. Diese Variante ist etwas teurer, bietet gegenüber den Kupferleitungen aber einen wesentlichen Vorteil. Bei einem Einsatz von Kupferleitungen kann es durch geringe Potentialunterschiede zwischen den Gebäuden zu Ausgleichsströmen auf dem Schirm kommen . Diese wirken störend auf die Datenübertragung. Die größte Gefahr besteht bei einem Blitzschlag, der sich über diese Kupferleitungen in alle Gebäude ausbreiten kann.

4.3.3.8 Stromversorgung

Die Stromversorgung für alle Komponenten beträgt 24V-. Momentane Spitzen in der Stromversorgung entstehen, wenn ein oder mehrere Motoren eingeschalten werden. Bei verschiedenen Messungen an der Prototypenanlage wurden Stromspitzen bis zu 7A aufgezeichnet. Die Stromversorgung sollte auf jeden Fall einen höheren Spitzenstrom liefern können. Liegt der Spitzenstrom unter diesen Wert, so kann es zu Spannungseinbrüchen kommen und die Resetbausteine auf den Stationsrechnern lösen auf Grund dieses Spannungseinbruchs einen Reset aus, was zur Folge hat, dass der Stationsrechner neu initialisiert wird. Die Wiederanlauftechnik bringt in diesem Fall auch keinen Vorteil, da es bei der nächsten Motorbewegung wiederum zu einer Unterspannung kommt und der Vorgang von vorne beginntl.

4.3.3.9 Zentrale oder getrennte Stromversorgung

Meiner Meinung hat eine zentrale Stromversorgung einige gravierende Nachteile. Das sind erstens die langen Leitungen und weiters die Störungen die sich über diese Leitungen ausbreiten können. Dieses System kann durch mehrere getrennte Stromquellen versorgt werden. Diese Stromversorgungen müssen untereinander nicht verbunden sein.

4.4 Betriebserfahrungen

Für das vorgeschlagene Konzept ergaben sich folgende typische Inbetriebnahme und Betriebserfahrungen.

- Planungsphase

  Das Konzept lässt sich auf vergleichbare Anlagen mit minimalem Aufwand übertragen. Der wesentliche Schlüssel liegt in der frei konfigurierbaren Software des Leitrechners (In der vorliegenden Version ist das System bis zu 16 Linien mit je 63 Stationen ausgelegt). Die Auslegung der Stationsrechner bzw. der Kommunikationsrechner ist universell und damit direkt übertragbar. Die Standardisierung dieser Komponenten ist ein beträchtlicher Kostenvorteil. Für die Planung sind ein Stations- und Rohrschema der Anlage so wie die Basisinformationen über die verwendeten Sensoren und Aktoren ausreichend.

- Beschaffung der Komponenten

  Es werden ausschließlich handelsübliche und in der Industrie verbreitete Komponenten verwendet. Die prinzipielle Verfügbarkeit stellt damit kein Problem dar. Auf Grund der Vielfalt der Hersteller sollten Einkaufserfolge erzielbar sein. Die Praxis hat gezeigt, dass zeitweise deutlich erhöhte Lieferzeiten auftreten können. Diese Risiken sind bei der Planung zu berücksichtigen.

- Installation

  Die Hardware ist als Stecksystem ausgeführt und daher effizient zu handhaben. Bei Neuanlagen kann die Anschlussplatine bereits im Werk montiert werden. Der Stationsrechner kann anschließend Vorort gesteckt werden. Sämtliche Kabel sind über Steckverbinder angeschlossen.

  Bei einer Anlagenumrüstung ist der Aufwand etwas größer, da die Halterung und die Anschlussplatine Vorort montiert werden müssen. Die restlichen Arbeiten gleichen der Neuinstallation.

- Inbetriebnahme

  Für die Inbetriebnahme wurden zusätzliche Tools entwickelt. Mit dem Tool „CanPing" kann die Kommunikation zu allen Stationsrechnern gezielt getestet und in Betrieb genommen werden. Die Inbetriebnahme der Kommunikation stellte den

größten Zeitaufwand dar.
Die Justierung der Stationen kann direkt an der Station mit dem Stationsrechner vorgenommen werden .
Nachdem am Leitrechner die gesamte Topologie der Anlage konfiguriert ist, kann der Testbetrieb beginnen.

- Betriebserfahrungen hinsichtlich der Störungstoleranz
  Fehlfunktionen in der Anlage wurden bei der Inbetriebnahme simuliert. Die einfachste Möglichkeit dazu war, während des Betriebes an einem Stationsrechner eine Resettaste zu betätigen und die Systemreaktion genau zu verfolgen. Eine weitere Möglichkeit war, eine elektrische Störung mit einer ESD- Pistole direkt neben einem Stationsrechner zu verursachen, so dass dieser ausfällt. Diese simulierten Störungen wurden alle ohne merkbare Unterbrechung behoben.

- Einarbeitung des lokal verantwortlichen Servicepersonals
  Das Servicepersonal war während der gesamten Installation und Inbetriebnahme schon dabei, daher war eine weitere Einschulung nicht notwendig.
  Einen wesentlichen Vorteil für das Servicepersonal brachte das automatische Anrufsystem (AAS). Durch dieses System wird das Servicepersonal bei dem Auftreten eines nicht automatisch behebbaren Fehlers (ev. mechanische Verklemmung, etc.) sofort telefonisch verständigt und der Ort des Fehlers so wie die Fehlerursache in gesprochener Form übermittelt.

- Akzeptanz der Gesamtsystems durch den Anwender
  Die Akzeptanz beim Anwender war bereits am ersten Tag nach der Inbetriebnahme sehr hoch. Die Anzahl der Sendungen stieg von Tag zu Tag und pendelte sich nach ein paar Tagen auf einen akzeptablen Wert ein.
  Wenn man sich die Verteilung der Sendungen über einen Tag anschaute, so ergaben sich darin 3 Kernzeiten (ca. 8 Uhr 30 bis 9 Uhr, 11 Uhr bis 12 Uhr und 15 Uhr bis 16 Uhr 30) in denen die Auslastung sehr hoch war. In den übrigen Zeiten war eine mittelmäßige Auslastung der Anlage zu beobachten.

- Resumee
  Schon bei Projektbeginn stand fest, dass es sich nicht nur um eine theoretische Arbeit

handeln würde, sondern dass das Projekt realisiert, an einer Testanlage installiert und in Betrieb genommen werden muss. Der Projektplan wurde bis zur Fertigung weitestgehend eingehalten. Durch den Lieferverzug einzelner Bauelemente verzögerte sich die Fertigung (Lieferzeiten der Bauteile stiegen von 4 Wochen auf 12 bis 16 Wochen an) der Baugruppen um drei Monate. Durch diese Verzögerung konnte die Testanlage nicht realisiert werden. Der erste Einsatz des Prototypen fand direkt beim Kunden statt.

Nach Behebung der in der ersten Phase aufgetretenen Softwarefehler, lief das gesamte System ohne nennenswerte Unterbrechungen. In der darauf folgenden Testphase (ca. 2 Monate) wurden weitere Softwarefehler behoben, so dass das System anschließend die gestellten Erwartungen erfüllte.

Im Nachhinein betrachtet war die notgedrungen gewählte Vorgangsweise, ohne einer größeren Testanlage, sehr riskant und mit hohen Kosten verbunden. Ein System, dass nicht ausreichend getestet ist, sollte aus wirtschaftlichen Aspekten aber auch aus Prestigegründen nicht ausgeliefert werden.

5 Zusammenfassung und Ausblick

Ziel dieser Arbeit war, die Grundlage für ein Rechensystem mit hoher Störungstoleranz und automatischer Ausfallsbehebung im Bereich von Transportsystemen zu erarbeiten.
Für die Transportsysteme wurde eine Steuerung entwickelt, implementiert und erprobt, welche auf Basis von verteilten Rechnersystemen arbeitet. Der Ausfall einer oder mehrerer Systemkomponenten wird automatisch erkannt und das System durch geeignete Maßnahmen wieder in einen konsistenten Zustand gebracht, so dass es für die gesamten funktionellen Abläufe keine nennenswerten Unterbrechungen gibt.

Im Rahmen dieser Arbeit wurden verschiedene Ansätze für die Realisierung dieser Wiederanlauftechnik diskutiert. Für das vorliegende Fallbeispiel wurde das Prinzip der Suchstrategie unter Einbeziehung aller verfügbarer Vorinformationen gewählt.

Der Einsatz erfolgte in einer Mehrlinien Rohrpostanlage. Das Projekt umfasst die Projektierung und die gesamte Hard- und Softwareentwicklung. Das Beispiel, der in dieser Arbeit ausgewählten pneumatischen Transportanlage, ist ein typischer Vertreter solcher Systeme.
Alle wesentlichen Aspekte werden im Detail diskutiert; dies sind insbesondere der Systementwurf, die Hardwareentwicklung, das Datenbankdesign, die Softwareentwicklung für die einzelnen Hardwarekomponenten, sowie das automatische Anrufsystem. Der Detaillierungsgrad ist so gewählt, dass das vorgestellte Material als Referenz für vergleichbare Projekte im Bereich fehlertoleranter verteilter Rechnersysteme dienen kann.

Die Steuerungssoftware ist frei konfigurierbar und damit auf andere Anlagenarchitekturen übertragbar. Die Auslegung der Stationsrechner und Kommunikationsrechner ist universell. Die Standardisierung dieser Komponenten kann bei Folgeprojekten ein beträchtlicher Kostenvorteil sein.

Die Betriebserfahrung zeigte, dass die Spezifikationen an die Fehlertoleranz voll erfüllt wurden. Die Akzeptanz der Gesamtlösung sowohl bei den Anwendern als auch beim Servicepersonal war hoch.

6 Anhang

6.1 Netzwerkgrundlagen

6.1.1 Controller Area Network – CAN

Controller Area Network (CAN) ist ein serielles Netzwerk, das ursprünglich für die "Automotive – Industrie" entwickelt wurde. Es dauerte jedoch nicht lange, bis CAN sich in der gesamten Automatisierungsindustrie etabliert hatte. Der CAN- Bus wird in erster Linie in eingebetteten Systemen eingesetzt und ist daher die Kommunikationsschiene zwischen Microcontrollern. Physikalisch besteht CAN aus einem Zweidraht-, Halbduplex-, Hochgeschwindigkeitsnetzwerk, das sich gut für Applikationen eignet, die hohe Geschwindigkeiten verlangen, da der Datenaustausch mittels kurzer Nachrichten erfolgt. Weitere Vorteile sind Stabilität, Zuverlässigkeit und die weite Verbreitung in der Halbleiterindustrie.
Mit CAN können theoretisch bis zu 2032 Devices auf einem einzigen Netzwerk angesprochen werden. Letztendlich sind es in der Praxis durch die Beschränkungen der Hardware dann aber nur 110 Knoten. Durch die mögliche Kommunikationsrate von bis zu 1 Mbit/sec ist Echtzeit-Kontrolle möglich.

Geschichte

CAN wurde1986 von der Robert Bosch GmbH in Deutschland entwickelt um ein Kommunikationssystem zwischen drei ECUs (electronic control units) in Autos (Mercedes) zu erschaffen. Für diese spezielle Situation musste ein „Multi-Master" Kommunikationssystem konzipiert werden. 1987 wurde der erste CAN – Chip von Intel produziert.

CAN Standards

Das Original war die Bosch – Spezifikation. Version 2.0 ist in zwei Teile unterteilt:

- Standard CAN (Version 2.0A): benutzt 11 Bit zur Identifikation
- Extended CAN (Version 2.0B): benutzt 29 Bit zur Identifikation

Diese beiden Teile definieren unterschiedliche Formate für den Nachrichtenrahmen, wobei die Hauptdifferenz in der Länge liegt.

Für CAN gibt es zwei ISO-Standards. Der Unterschied liegt dabei in der physikalischen Schicht. ISO 11898 definiert die Übertragung für Hochgeschwindigkeitsapplikationen bis zu 1 Mbit/sec. ISO 11519 hat ein Limit bei der Übertragung von 125 kbit/sec.

Kompatibilität zwischen Part A und Part B

Es gibt drei Typen von CAN Kontrollern: Part A
Part B passive
Part B

Sie können die verschiedenen Teile des Standards wie folgt bedienen:

Message format \ CAN chip type	Part A	Part B Passive	Part B
11 bit ID	ok	ok	ok
29 bit ID	Fehler	Fehler wird einfach ignoriert	ok

Die meisten 2.0A Controller schicken und verarbeiten nur Standardformatnachrichten. Einige (2.0B passiv) können auch erweiterte Nachrichtenformate erhalten, diese werden aber ignoriert. 2.0B Controller können Nachrichten in beiden Formaten senden und erhalten. Erhält ein Part A Controller eine 29 Bit Nachricht, so kann der Bus nicht mehr arbeiten.

Wie arbeitet CAN?

CAN ist ein Multimaster Netzwerk, das CSMA/CD+AMP (Carrier Sense Multiple Access/Collision Detection with Arbitration on Message Priority) verwendet. Bevor eine Nachricht verschickt wird, prüft der CAN-Knoten, ob der Bus beschäftigt ist. Mögliche Kollisionen werden ebenfalls entdeckt und bereinigt, ähnlich dem Ethernet. Anders als beim Ethernet wird aber nicht das Senden der Nachrichten sofort gestoppt und erst nach einer zufälligen Zeit wieder gesendet, sondern es wird das Prinzip der Entscheidungsfindung verwendet.

Daten, die von einem Knoten über einen CAN Bus verschickt werden, haben weder Absender- noch Zieladresse. Stattdessen hat jede Nachricht eine netzweit eindeutige Identifikation. Jeder Knoten prüft die Nachricht um zu evaluieren, ob die Nachricht für den jeweiligen Knoten relevant ist. Entweder wird sie dann verarbeitet oder ignoriert.

Identifikation und Entscheidungsfindung

Die eindeutige Identifikation bestimmt auch die Priorität der Nachricht. Je kleiner der numerische Wert der Identifikation ist, desto höher ist die Priorität. Dadurch kann entschieden werden, wer senden darf, wenn zwei oder mehrere Knoten gleichzeitig senden wollen.
Die Nachricht mit der höchsten Priorität darf daher den Bus benutzen. Niedrigere Prioritäten werden automatisch entsprechend ihrer Reihenfolge in den nächsten Buszyklen verarbeitet.
Jede CAN - Nachricht hat eine Identifikation mit 11 oder 29 Bits. Diese Identifikation ist der grundlegende Teil des CAN-Entscheidungsfeldes, welches am Beginn jeder CAN Nachricht steht. Die Identifikation identifiziert gleichzeitig den Typ der Nachricht und die Priorität.
Die Bits in einer CAN Nachricht können "Low" oder "High" sein. Die low Bits sind dominant, d.h. wenn ein Knoten low senden will und ein Knoten high, so ist das Resultat am Bus low. Während dessen ein Knoten sendet, hört er den Bus ab. Ein Knoten der high im Entscheidungsfeld sendet und gleichzeitig eine Nachricht mit low entdeckt, weiß, dass er nicht mehr weitersenden darf, so dass der andere ungestört weiter übertragen kann.
Es ist nicht erlaubt, dass zwei Knoten am Netz eine Nachricht mit derselben ID versenden. Falls das doch passiert, kann keine Entscheidung mehr getroffen werden, wer senden darf. Ein Knoten erkennt aber in jedem Fall, dass etwas nicht stimmt und verwendet die Fehlerbehandlung des CAN Bus, sodass einer der Knoten vom Bus deaktiviert wird (bus-off mode).

Remote Frames

Bei CAN gibt es remote Frames und Datenframes. Datenframes werden verwendet, wenn ein Knoten Daten verschicken will.
Remote Frames werden als Anforderung für eine Information angesehen. Ein Frame, der das RTR Bit gesetzt hat, fordert jene Information an, die durch die Identifikation spezifiziert wird. Derjenige Knoten, der die entsprechende Information hat, sollte darauf hin antworten.
Je nach Implementierung des CAN Controllers, wird die Antwort entweder automatisch geschickt oder falls das nicht möglich ist, muss der Microcontroller die Nachricht schicken.

Nachrichtenformate

Es gibt zwei Formate:

- Standard CAN (Version 2.0A)
- Extended CAN (Version 2.0B)

A- Format

Ein Standard - CAN (Version 2.0A) Nachrichtenframe besteht aus sieben verschiedenen Bits:

- Start of Frame (SOF) field: Dabei handelt es sich um ein dominantes (logic 0) Bit, das den Beginn der Nachricht identifiziert.
- Arbitration field: Beinhaltet eine 11 Bit Nachrichten Identifikation. Und das Remote Transmission Request (RTR) Bit. Ein dominantes RTR Bit zeigt an, dass die Nachricht ein Datenframe ist. Ein rezessiver Wert (logic 1) zeigt an, dass die Nachricht ein Remote Frame ist.

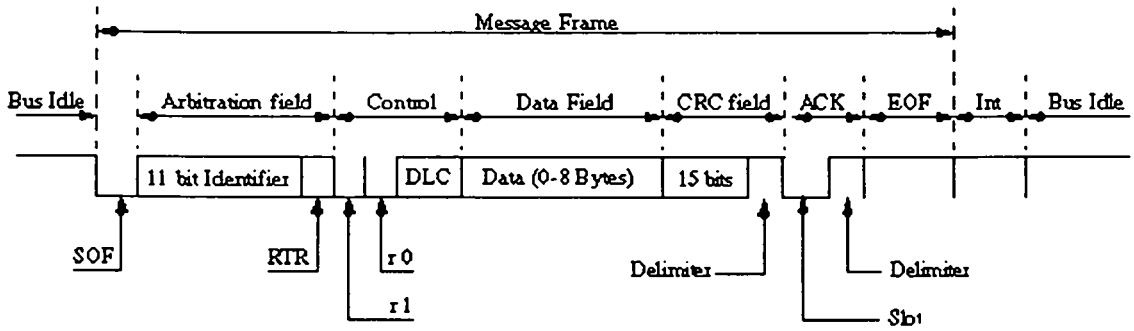


Abb.: 6.1 CAN 2.0A Message Frame

- Control Field mit 6 Bits: zwei dominanten Bits (r0 and r1), reserviert für die Zukunft ein vier Bit Data Length Code (DLC). DLC gibt die Anzahl von Bytes an, die im Datenfeld folgen.
- Data Field: enthält 0 bis 8 Byte
- CRC field: enthält ein 15 bit CRC-Code (cyclic redundancy check) und ein rezessives Trennbit
- ACKnowledge field: Enthält zwei Bits. Das erste ist das Slot Bit (rezessive). Es wird nach und nach durch dominate Bits von allen anderen Knoten überschrieben, die die Nachricht erhalten. Das zweite Bit ist ein rezessives Trennbit.
- End of Frame field: Besteht aus sieben rezessiven Bits.

Nach dem „end of a frame" kommt das INTermission field, das aus drei rezessiven Bits besteht. Danach ist der Bus wieder frei.

B - Format

Das CAN 2.0B Format besteht aus 29 Bits.

Version 2.0B soll die Kompatibilität zu anderen seriellen Kommunikationsprotokollen, die im automotive Bereich eingesetzt werden, gewährleisten.

Unterschiede:

- In Version 2.0B hat das Entscheidungsfeld zwei Indentifizierungsbitfelder. Das erste ist 11 Bits lang (Kompatibilität mit Version 2.0A). Das zweite ist 18 Bits lang.
- Die Unterscheidung wird durch ein „Identifier Extension" (IDE) Bit getroffen.

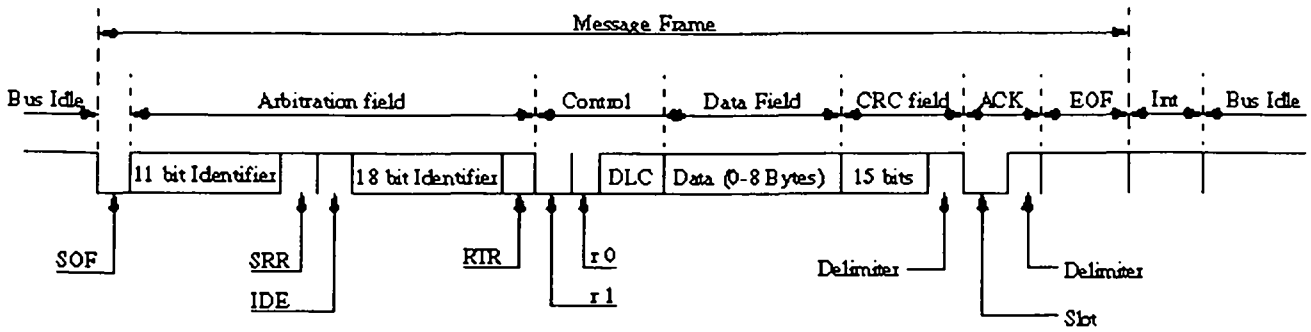


Abb.: 6.2 CAN 2.0B Message Frame

- Ein Substitute Remote Request (SRR) Bit ist im Arbitration Field inkludiert. Das SRR Bit wird immer rezessiv übertragen, um sicherzustellen, dass der Standard Datenframe immer höhere Priorität hat, falls ein Standarddatenframe und ein extended Datenframe die selbe Basisidentifikation haben. Alle anderen Felder im 2.0B Frame sind identisch mit dem Standardformat.

Fehlerbehandlung

Der CAN – BUS ist sehr sicher, da die gesamte Fehlerbehandlung standardisiert ist. Fehlerhafte Knoten gehen automatisch in einen Status, in dem sie den Verkehr am Bus nicht stören.

Mögliche Fehler:

- Bitfehler:

1. Bit stuffing error – nach fünf aufeinanderfolgenden Highs muss ein Low kommen, ansonsten tritt "bit stuffing" auf.
2. Bit error: Ein sendender Knoten liest die Nachricht, die er schickt noch einmal durch. Falls ein unterschiedliches Bit außerhalb des des „arbitration" Feldes oder des "acknowledgement" Feldes auftritt, entsteht ein Bit error.

- Nachrichtenfehler:

1. Checksum error – Jeder empfangende Knoten überprüft die Prüfsumme.

2. Frame error – An bestimmten Position innerhalb eines Frames gibt es definierte Bits, die einen voreingestellten Wert haben müssen, andernfalls ist ein Frame error aufgetreten.
3. Acknowledgement Error – Wird eine Nachricht nicht mittels ACK bestätigt, tritt ein Fehler auf.

Ein CAN - Kontroller kann drei verschiedene Stati annehmen:

- Error active – normaler Modus zum Senden und Empfangen von Nachrichten. Wird eine Fehler entdeckt, wird ein Error-Flag gesendet.
- Error passive – Modus der eintritt, wenn massive Probleme beim Senden oder Empfangen auftreten. Wird ein Fehler entdeckt, wird das passive error flag gesendet.
- Bus off – Modus, der bei schwerwiegenden Problemen auftritt. Ein Senden oder Empfangen ist nicht mehr möglich bis der CAN-Controller durch den Mircrocontroller bereinigt wurde.
-

6.1.2 Universal Serial Bus – USB

Ursprünglich gab es in der Informationstechnologie für jedes Gerät eigene Schnittstellen. Das war nicht nur unübersichtlich sondern auch ineffizient und teuer. So gab es unterschiedliche Schnittstellen für PC-Tastaturen, Bus- und serielle Maus und PS/2. Erst beim Apple Desktop Bus hing die Maus an der PC-Tastatur und es konnten weitere 16 Geräte angehängt werden. Einen weiteren Fortschritt stelle der AccessBus (I2C-Bus mit DEC-Protokoll) dar, der theoretisch bis zu 127 Geräte erlaubte.

Erst der 1996 veröffentlichte USB Standard 1.0 USBSTD sollte nach einigen Problemen zu einem wirklichen Durchbruch verhelfen.

Vorteile:

- Einheitliche Steckverbinder für alle Eingabegeräte
- Stromversorgung über den Bus möglich
- echtes Plug and Play während des Betriebs möglich
- synchrone und asynchrone Datenübertragung
- Gerät gibt Informationen an Treiber
- Nutzung der Technologie ohne Lizenzgebühr

Nachteile:

- Versorgung mit Treibern unter einigen Betriebssystem ist schlecht
- Hubs und Kabel sind teuer
- Einige Geräte (Scanner, Drucker, etc) würden mehr Bandbreite (derzeit max. 1,5 MB) benötigen

Für die Version 2.0 ist eine Übertragung mit einer Datenrate in den MBit- Bereich geplant. Der USB definiert verschiedene Geräteklassen. Geräteklassen haben gemeinsame Merkmale, die es erlauben sollen, generische Treiber für die verschiedenen Klassen zu erstellen. Folgenden Geräteklassen sind definiert:

- Monitore
- Eingabegeräte
- Audiogeräte
- Kommunikationsgeräte
- Drucker
- Massenspeicher

Physikalischer Aufbau des USB

Der USB ist ein bitserieller Bus mit einem hierarchischen Verbindungsschema. Insgesamt können 127 Geräte, angeordnet in 7 Schichten, angehängt werden. Die maximale Kabellänge beträgt 5 m bei Highspeed-Geräten und 3 m bei Lowspeed-Geräten. Kabel für Highspeed-Geräte können länger sein, weil sie eine Schirmung besitzen.

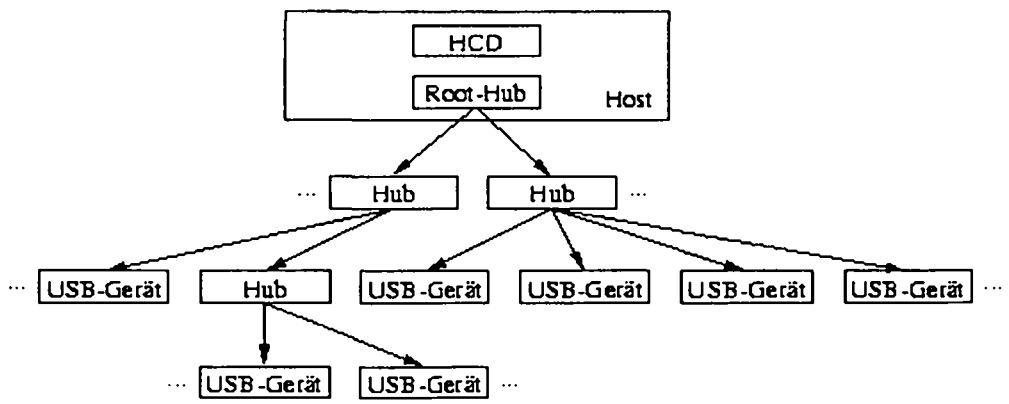


Abb.: 6.3 Physikalische Struktur des USB

Durch die Punkt-zu-Punkt-Verbindung ist dabei immer ein Busabschluss gewährleistet. Die Hubs wirken als Verstärker. Sie müssen sowohl 12 MBit, als auch den 1,5 MBit Datenstrom weiterleiten. Datenpakete mit 12 MBit werden nicht an Geräte mit 1,5 MBit weitergeleitet, um Fehlfunktionen und eine Störabstrahlung über das ungeschirmte Kabel zu vermeiden.

Kabel und Stecker

Der USB hat eine 4-polige Leitung mit unterschiedlichen Steckern an den beiden Enden. Die unterschiedlichen Stecker sollen unerlaubte zyklische Verbindungen verhindern.

Nr	Farbe	Name	Beschreibung
1	rot	VCC	5V Spannungsversorgung
2	weiss	D-	negative Datenleitung
3	grün	D+	positive Datenleitung
4	schwarz	GND	Masse

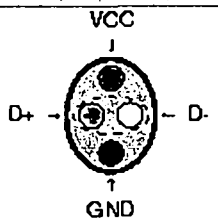
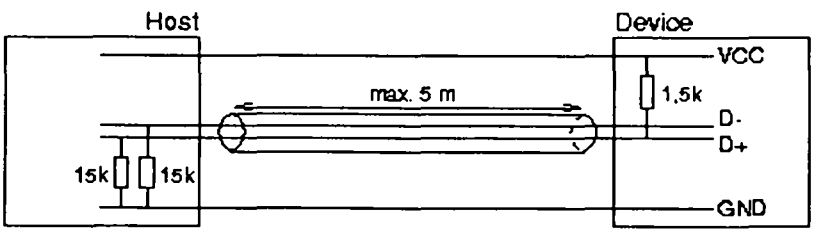


Abb.: 6.4 Fullspeed USB-Kabel

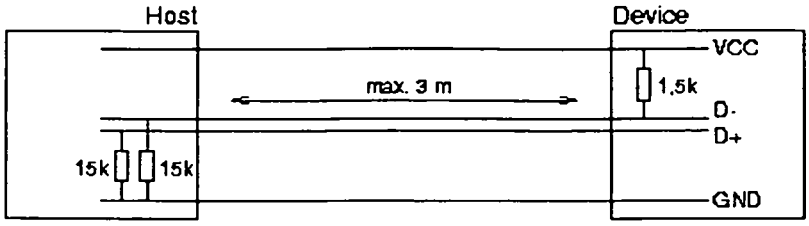
Man unterscheidet zwischen Low- und Highspeed-Kabel.

Highspeed Kabel:
- Abschirmung um die Datenleitungen
- Kabellänge 5 Meter
- geschirmte Twisted-Pair-Kabel mit einer maximalen Impedanz von 90 Ohm und einer maximalen Signalverzögerung von 26 ns

Lowspeed Kabel:
- keine Abschirmung um die Datenleitung
- Kabellänge 3 Meter
- maximale Signalverzögerung von Lowspeed-Kabeln darf 18 ns nicht überschreiten.



High Speed Verbindung mit Schirmung



Low Speed Verbindung

Abb.: 6.5 High und Low Speed Verbindung

Übertragungsvorgang

Daten werden in binärer Form übertragen. Die beiden definierten logischen Zustände werden als J und K bezeichnet. Sie setzen sich aus den Zuständen der zwei Signalleitungen D+ und D- zusammen.

Lowspeed Signale:

Signal	Pegel	D-	D+
J	Differentiell 0	H	L
K	Differentiell 1	L	H

Fullspeed Signale:

Signal	Pegel	D-	D+
J	Differentiell 1	L	H
K	Differentiell 0	H	L

Zusätzlich gibt es einen dritten Zustand SE0. Dieses Signal wird als End-of-Paket-Markierung (EOP) eingesetzt .

Pegel	D-	D+	Anmerkung
SE0	L	L	(D+) = (D-)
SE0-Signal			

Wesentlich für die Übertragung sind auch die Zeiten und die Bandbreite.

USB-Protokoll

Der USB funktioniert nach dem Master-Slave-Prinzip. Der Master (Host) gibt eine Adresse (bestehend aus Gerätenummer und Endpoint) und die Richtung auf den Bus. Anschließend werden die Daten gesendet bzw. könnte auch das angesprochene Gerät Daten senden. Dann kommt die Handshakephase, die die erfolgreiche Übertragung der Daten signalisiert. Bei isochronem Transfer gibt es keine Handshakephase.
Es gibt zwei Arten von Endpunkten:

- Control Endpoints: bidirektional, Halbduplexbetrieb
- Data Endpoints: unidirektional, können nur senden oder empfangen

Bitebene: Die Datenübertragung erfolgt blockweise bitseriell. Dabei wird das niederwertigste Bit zuerst übertragen. (Ausnahme: CRC-Feld, höchstwertigstes Bit wird zuerst übertragen). Zur Fehlererkennung wird jeweils über Header und Daten eine CRC-Prüfsumme gebildet. Damit ist eine Erkennung von Fehlern möglich. Das Daten-CRC ist ein 16-Bit-CRC-Feld und wird demzufolge mit einem Polynom 16. Grades gebildet ($G(X) = X^{16}+X^{15}+X^2+1$).

Blockebene: Jeder Block beginnt mit dem Sync-Feld. Dieses Feld dient der Synchronisation der folgenden Daten. Dann kommt eine Kennung für den Pakettyp. Bei unvollständigen oder fehlerhaften Paketkennungen, wird das Paket ignoriert.

Die Adressierung erfolgt über „function adress field" und „endpoint field", vergleichbar mit IP-Adresse und der Portnummer bei TCP/IP. Das function adress field ist 7 Bit breit. Adresse 0 ist dabei die Standardadresse die nur während der Zuweisung der individuellen Geräteadresse benutzt wird .

Paket-Formate

Sämtliche Pakete werden durch ein SYNC eingeleitet und mit einem EOP abgeschlossen. Es gibt verschiedene Paket-Arten:

Token-Paket:	besteht aus PID (Processidentifikation), Adresse und Endpoint
Start-of-Frame-Paket:	besteht aus PID und Frame-Nummer
Daten-Paket:	besteht aus PID und anschließend die Datenbytes
Handshake-Paket:	besteht aus PID

Pre-Paket: besteht aus PID mit bestimmten Inhalt, signalisiert den Hubs, dass Datenübertragung an Ports mit Low-Speed Geräten beginnen kann

Hardware und Software

Wie beim OSI-Modell gibt es auch für USB ein Schichtenmodell.

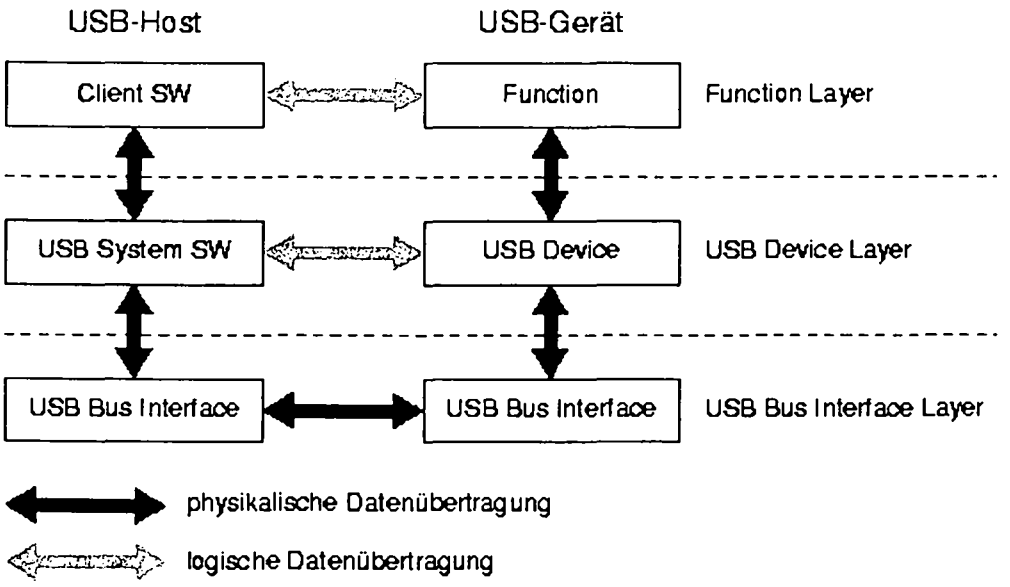


Abb.: 6.6 USB Schichtenmodell

Die logische Kommunikation findet immer nur zwischen gleichen Schichten statt.

USB-Host

Bus-Interface : Das Bus-Interface ist auf der Host-Seite als Host Controller implementiert. Der Host Controller hat die Aufgabe die Daten - entsprechend den elektrischen Anforderungen - in Bitströme auf der Datenleitung umzuwandeln.

USB-System : Beim USB-System handelt es sich um die Software, die vom Betriebssystem mitgeliefert wird. Das USB-System benutzt den Host Controller für die Datenübertragung zum USB-Gerät.

Client : Die Client-Schicht umfasst sämtliche Softwareteile, die für die Interaktion mit dem USB-Gerät verantwortlich sind. Die Client-Schicht ist nicht in der Lage, direkt auf die Hardeware eines USB-Gerätes zuzugreifen. Die Client-Software ist der eigentliche Geräte-Treiber, der vom Geräte-Hersteller mitgeliefert wird.

USB-Gerät
Die unterste Schicht - das Bus Interface - sendet und empfängt Datenpakete.

Die mittlere Schicht ist für das Routing zuständig. Das heißt die empfangenen Daten werden den verschiedenen Endpoints zugeordnet.

Die oberste Schicht stellt die Funktionalität des Gerätes dar und ist dementsprechend geräteabhängig.

6.1.3 LAN

TCP/IP wurde in den 70-er Jahren entwickelt um Datenaustausch zwischen heterogenen Rechnernetzen zu ermöglichen. Bei TCP/IP handelt es sich um eine Protokollfamilie, wobei zwei sehr wichtige Protokolle aus dieser Familie für den Namen verantwortlich sind: TCP - 'Transmission Control Protocol' (Schicht 4) und IP - 'Internet Protocol' (Schicht 3). Auf Grund der großen Verbreitung stellt TCP/IP eigentlich bereits einen Standard dar. Die Spezifikationen sind in so genannten RFC – (Request for Comment) Dokumenten niedergelegt.

Nachfolgend ist eine Gegenüberstellung zwischen dem OSI- Modell und dem TCP/IP dargestellt.

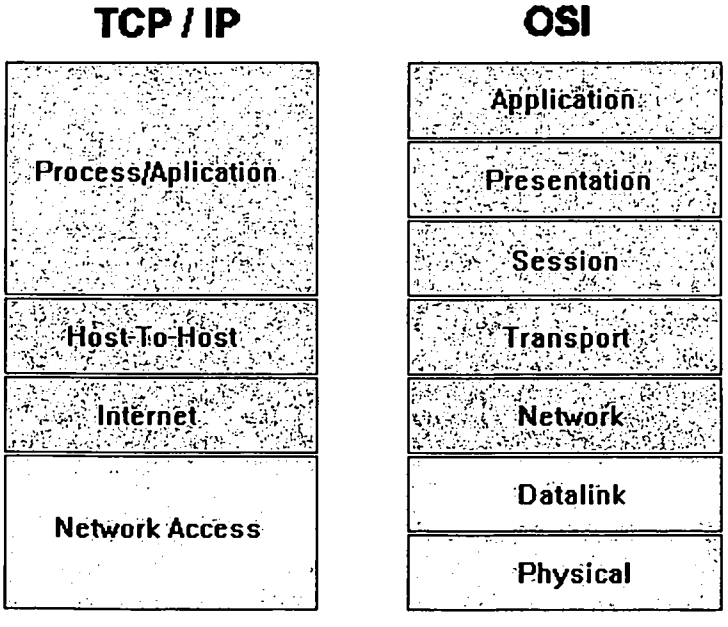


Abb.: 6.7 TCP/IP – OSI Modell

Die Schichten 5 - 7 des OSI- Modells sind beim TCP/IP in einer Schicht zusammengefasst, da sie sich mit den Anwendungsprogrammen beschäftigen.

In Schicht 4 (Host to Host) befindet sich außer dem TCP auch noch das UDP (User Datagram Protocol). Das TCP ist dabei für einen verbindungsorientierten, sicheren Transport zuständig (Erhalt von Daten wird mittels Receive und Send- Windows gesteuert), während dessen das UDP für einen verbindungslosen und daher ungesicherten Datentransport zuständig ist. Beide Protokolle verwenden für die eindeutige Identifizierung der Anwendungsprogramme auf ein- und derselben Maschine Ports.

In Schicht 3 (Internet) befindet sich das verbindungslose Internet-Protokoll (IP). Von diesem Protokoll werden die Datenpakete ohne Empfangsbestätigungen weitergereicht. Innerhalb des Kopfteils eines Paketes gibt es das TTL- (time to live) Feld, das dazu dient, Pakete die schon zu lange unterwegs sind, zu vernichten. Dieses Protokoll verwendet IP-Adressen für die Adressierung. Neben einigen anderen Protokollen gehört auch das ARP in diese Ebene. Beim ARP (Address Resolution Protocol) handelt es sich um ein Protokoll, welches für die Auflösung einer logischen IP-Adresse in eine physikalische (z. B. Ethernet-) Adresse zuständig ist. Durchgeführt wird das mittels Broadcasts. Dabei handelt es sich um Datenpakete, die keine spezielle IP- Adresse eingetragen haben und daher alle Stationen erreichen.

Die unterste Schicht ist gegenüber der Internetschicht protokolltransparent. Sie können durch standardisierte Protokolle (z. B. Ethernet, FDDI, SLIP, PPP) oder andere Übertragungsverfahren realisiert werden.

Die TCP/IP-Protokolle

Der große Vorteil der TCP/IP-Protokollfamilie ist die einfache Realisierung von Verbindungen zwischen Netzwerken. Hosts können aus diesem Grund über mehrere Teilnetze hinweg kommunizieren. Für die physikalische Ebene können dabei verschiedene Möglichkeiten verwendet werden:

- LANs (Ethernet, Token-Ring, usw.)
- WANs (X.25, usw.)
- Punkt-zu-Punkt-Verbindungen (SLIP, PPP)

<table>
<tr><th colspan="10">Internet-Protokolle</th></tr>
<tr><td colspan="2">OSI-Schicht</td><td colspan="7">Internet Protokoll Suite</td><td>DOD Schicht</td></tr>
<tr><td>7</td><td>Anwendung</td><td>File Transfer</td><td>Electronic Mail</td><td>Terminal Emulation</td><td>Usenet News</td><td>Domain Name Service</td><td>World Wide Web</td><td>Art der Kommuni-kation</td></tr>
<tr><td>6</td><td>Darstellung</td><td rowspan="2">File Transfer Protocol (FTP) RFC 959</td><td rowspan="2">Simple Mail Transfer Protocol (SMTP) RFC 821</td><td rowspan="2">Telnet Protocol (Telnet) RFC 854</td><td rowspan="2">Usenet News Transfer Protocol (NNTP) RFC 977</td><td rowspan="2">Domain Name Service (DNS) RFC 1034</td><td rowspan="2">World Wide Web (WWW) RFC</td><td rowspan="2">Applikation</td></tr>
<tr><td>5</td><td>Sitzung</td></tr>
<tr><td>4</td><td>Transport</td><td colspan="6">Transmission Control Protocol (TCP) RFC 793</td><td>User Datagram Protocol (UDP) RFC 768</td><td>Host to Host Kommunikation</td></tr>
<tr><td>3</td><td>Netzwerk</td><td>Address Resolution Protocol (ARP) RFC 826</td><td colspan="5">Internet Protocol (IP) RFC 791</td><td>Internet Control Messsage Protocol RFC 792</td><td>Internet</td></tr>
<tr><td>2</td><td>Sicherung</td><td colspan="2">Ethernet</td><td>Token Ring</td><td>DQDB</td><td>FDDI</td><td>ATM</td><td>lokales Netzwerk</td></tr>
<tr><td>1</td><td>Physikalische Übertragung</td><td>Twisted Pair</td><td>Lichtwellenleiter</td><td colspan="2">Coaxkabel</td><td>Funk</td><td>Laser</td><td>Netzzugriff</td></tr>
</table>

Im Internet gibt es drei unabhängige Namens- bzw. Adressierungsebenen:

- Physikalische Adressen (z. B. Ethernet-Adresse)
- Internet-Adressen (Internet-Nummer, IP-Adresse)
- Domain-Namen

ARP

Die Umsetzung einer IP-Adresse in eine Hardware-Adresse erfolgt einerseits mittels Tabellen andererseits auf Hardware-Ebene (z. B. Ethernet) automatisch über ARP.

Beispiel:

Station A will Daten an Station B mit der Internetadresse I(B) senden, deren physikalische Adresse P(B) sie noch nicht kennt. Sie sendet einem ARP-Request an alle Stationen im Netz, der die eigene physikalische Adresse und die IP-Adresse von B enthält.

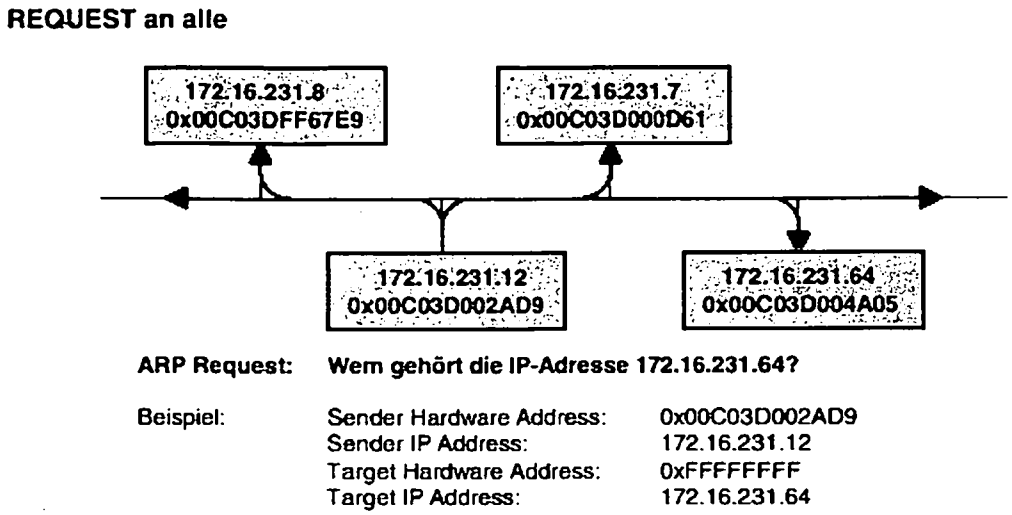


Abb.: 6.8 ARP Auflösung

Alle Stationen erhalten und überprüfen den ARP-Request und die angesprochene Station B antwortet, indem sie einen ARP-Reply mit ihrer eigenen physikalischen Adresse an die Station A sendet. Letztere speichert die Zuordnung in einer Tabelle (Address Resolution Cache).

IP - Internet Protocol

Die Internet-Schicht baut auf die Netzwerkschicht auf. Das IP ist für das Versenden der Datenpakete, die hier Datagramme genannt werden, zuständig. Die Daten selbst, die befördert werden, sind für das IP unrelevant. Relevant ist nur jener Teil, den das IP selbst als Header dazugibt, um die Adressierung vornehmen zu können.

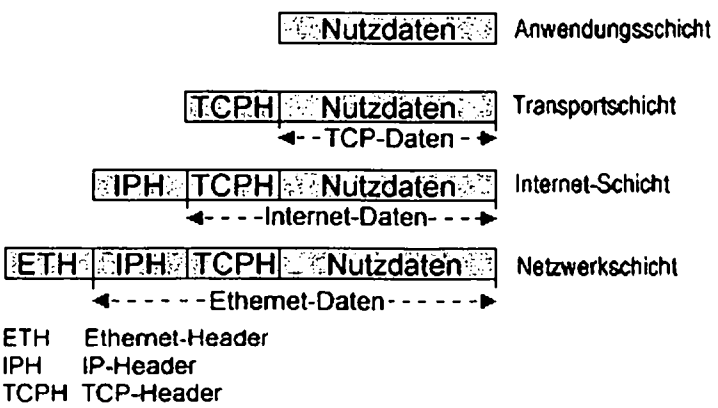


Abb.: 6.9 TCP/IP Layer

Da IP verbindungslos ist, genügt es, die Daten einfach wegzuschicken, in der Hoffnung, dass sie das Ziel erreichen werden. Ebenso wenig garantiert IP, dass die Pakete in der richtigen Reihenfolge ankommen. Garantiert ist jedoch, dass ankommende Pakete fehlerfrei sind. Die einzelnen Pakete müssen auch nicht den gleichen Weg zurücklegen, um das Ziel zu erreichen. Auf diesem Weg kann es natürlich vorkommen, dass Pakete verloren gehen oder sich verdoppeln. Derartige Probleme müssen aber von höheren Schichten erkannt und beseitigt werden.

Ein IP-Datagramm besteht aus einem Header und einem Datenblock, der z. B. in einem Ethernet-Frame "verpackt" wird. Die maximale Datenlänge wird kann variieren und muss durch entsprechende Mechanismen (Fragmentierung) Jedoch sollten Datagramme bis zu einer Länge von 576 Bytes unfragmentiert übertragen werden.

Format des IP-Headers

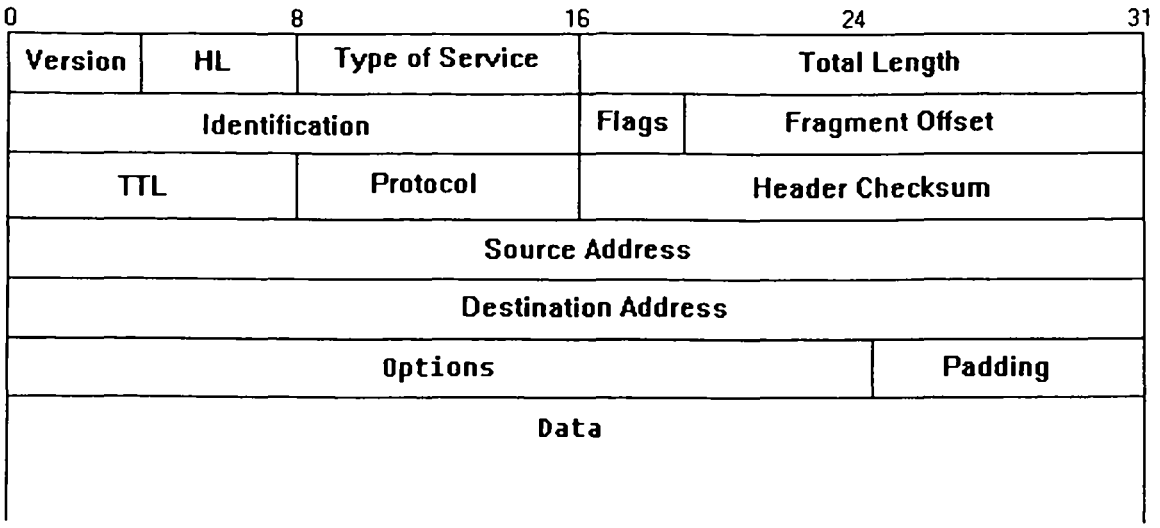


Abb.: 6.10 Format des IP-Headers

Version

IP-Protokollversion

IHL (Internet Header Length)

Die Angabe der Länge des IP-Headers erfolgt in 32-Bit-Worten (normalerweise 5).

Type of Service

Alle Bits haben nur "empfehlenden" Charakter.

Total Length

Gesamtlänge des Datagramms in Bytes (max. 64 KByte).

Identification

Identifizierung von Teilpaketen innerhalb eines Gesamtpakets

Flags

Die beiden niederwertigen Bits haben folgende Bedeutung:

- Don't fragment: Für Hosts, die keine Fragmentierung unterstützen
- More fragments: Zum Erkennen, ob alle Fragmente eines Datagramms empfangen wurden

Fragment Offset

Das erste Fragment hat Offset 0, für alle weiteren erhöht sich der Wert um die Länge des Datenfeldes eines Fragments. Dadurch kann der Empfänger feststellen, ob Fragmente fehlen.

Time-to-live (TTL)

Jedes Datagramm hat eine vorgegebene maximale Lebensdauer,die bei jedem Gatewy decrementiert wird.

Protocol

Da sich unterschiedliche Protokolle auf IP stützen, muss das übergeordnete Protokoll angegeben werden.

Header Checksum

16-Bit-Längsparität über den IP-Header (nicht die Daten)

Source Address

Internet-Adresse der Quellstation

Destinantion Address

Internet-Adresse der Zielstation

Options

Optionales Feld für weitere Informationen

Padding

Füllbits

Die Hauptaufgabe von IP ist es also, die Unterschiede zwischen den verschiedenen, darunterliegenden Netzwerkschichten zu verbergen und eine einheitliche Sicht auf die verschiedensten Netztechniken zu präsentieren.

UDP - User Datagram Protocol

UDP gehört derselben Ebene wie das TCP an, jedoch arbeitet es mittels nicht zuverlässigen, verbindungslosen Transport ohne Flußkontrolle. Es eignet sich dort, wo schneller Datentransport ohne großen Overhead notwendig ist und die gesicherte Datenübertragung keine große Rolle spielt. Auch hier passiert die Identifikation zweier Anwendungen über Port- Nummer. Die Transporteinheiten werden 'UDP-Datagramme' oder 'User Datagramme' genannt. Sie haben folgenden Aufbau:

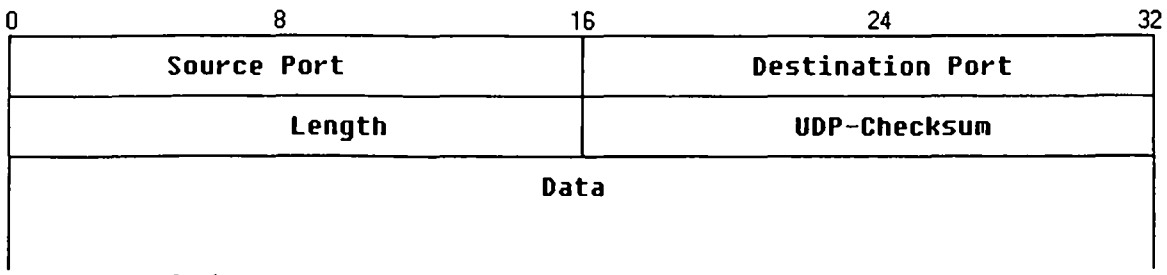


Abb.: 6.11 UDP- Header

Source Port

Identifiziert den sendenden Prozess

Destination Port

Identifiziert den Prozess des Zielknotens.

Length

Länge des UDP-Datagramms in Bytes (mindestens 8 = Headerlänge)

UDP-Checksum

Optionale Angabe einer Prüfsumme. Zu deren Ermittlung wird dem UDP-Datagramm ein Pseudoheader von 12 Byte vorangestellt (aber nicht mit übertragen), der u. a. IP-Source-Address, IP-Destination-Address und Protokoll-Nummer (UDP = 17) enthält.

TCP - Transmission Control Protocol

'Protokoll'-Feld jedes IP-Paketes steht das übergeordnete Protokoll, dass das Datenpaket enthält (z.B.: TCP). Jedes Protokoll der Transportschicht bekommt eine eindeutige Identifikationsnummer zugewiesen, anhand der die IP-Schicht entscheiden kann, was mit dem Paket zu geschehen hat.

TCP muss sich nun um die Aufgaben kümmern, um die sich das IP nicht kümmert. Es muss also dafür Sorge tragen, dass die Daten korrekt transportiert werden, dass alle ankommen und die Reihenfolge wieder hergestellt wird. TCP kann weiters mehrere virtuelle Kanäle (Ports) aufmachen, so dass mehrere Anwendungen gleichzeitig kommunizieren können. Bei den Daten dieser Ebene handelt es sich nun eigentlich um eine Datei und nicht mehr um Teilpakete.

Die Sicherheit wird in durch positive Rückmeldungen (acknowledgements) und Wiederholung fehlerhafter Blöcke erreicht. Zur Realisierung der Flußkontrolle wird ein Fenstermechanismus (sliding windows) verwendet (variable Fenstergröße). TCP-Verbindungen sind vollduplex. Wie bei allen verbindungsorientierten Diensten muss zunächst eine virtuelle Verbindung aufgebaut und bei Beendigung der Kommunikation wieder abgebaut werden. "Verbindungsaufbau" bedeutet hier eine Vereinbarung beider Stationen über die Modalitäten der Übertragung (z. B. Fenstergröße, Akzeptieren eines bestimmten Dienstes, usw.). Ausgangs- und Endpunkte einer virtuellen Verbindung werden wie bei UDP durch Ports identifiziert. Allgemein verfügbare Dienste werden über 'well known' Ports (--> feste zugeordnete Portnummer) erreichbar. Andere Portnummern werden beim Verbindungsaufbau vereinbart.

Damit aber das Bestätigen die Geschwindigkeit nicht zu stark vermindert, wird nicht sofort bestätigt. Es muss lediglich gewährleistet werden, dass innerhalb eines Fensters alle Bytes bis zu einem bestimmten Zeitpunkt bestätigt werden.

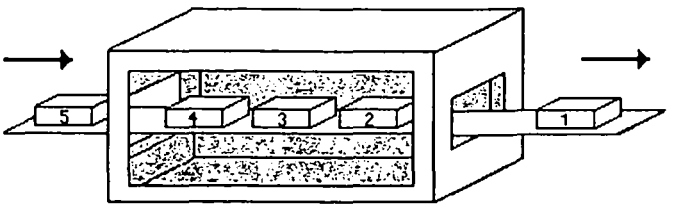


Abb.: 6.12 Sliding window

Das Prinzip des Fenstermechanismus ist eigentlich einfach. Beispiel mit Fenstergröße drei Bytes:

- Byte 1 wurde von der Datenquelle gesendet und vom Empfänger quittiert.
- Die Quelle hat die Bytes 2, 3 und 4 gesendet, sie wurden aber vom Empfänger noch nicht quittiert (Quittung eventuell noch unterwegs).
- Byte 5 wurde von der Quelle noch nicht gesendet. Er geht erst dann auf die Reise, wenn die Quittung für Byte 2 (oder höher) eingetroffen ist.

Das TCP-Paket wird als 'Segment' bezeichnet. Auch das TCP gibt jedem Datenblock einen Header mit bestimmten Informationen:

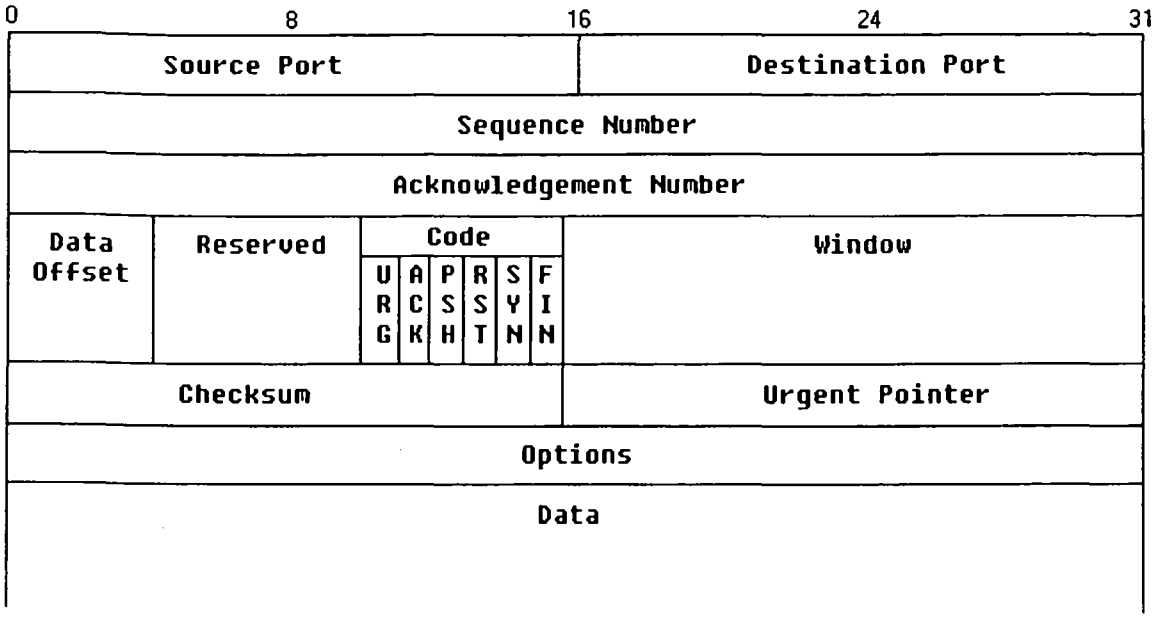


Abb.: 6.13 TCP-Header

Source Port

Identifiziert den sendenden Prozess.

Destination Port

Identifiziert den Prozess des Zielknotens.

Sequence Number

TCP betrachtet die zu übertragenden Daten als nummerierten Bytestrom. Dieser Bytestrom wird bei der Übertragung in TCP-Segmente aufgeteilt.

Acknowledgement Number

Bestätigung, bezieht sich auf Sequence-Nummer

Data Offset

Da der Segment-Header ähnlich dem IP-Header Optionen enthalten kann, wird hier die Länge des Headers in 32-Bit-Worten angegeben.

Reserved

Reserviert für spätere Nutzung

Code

Angabe der Funktion des Segments:

Window

Spezifiziert die Fenstergröße

Checksum

16-Bit Längsparität über Header und Daten.

Urgent Pointer

Markierung eines Teils des Datenteils als dringend. Dieser wird unabhängig von der Reihenfolge im Datenstrom sofort an das Anwenderprogramm weitergegeben .

Options

Dieses Feld dient dem Informationsaustausch zwischen beiden Stationen auf der TCP-Ebene.

TCP

- ist ein verbindungsorientierter Dienst
- ist vollduplexfähig
- hat hohe Zuverlässigkeit
- gewährleistet die Sicherung der Datenübertragung durch Prüfsumme und Quittierung mit Zeitüberwachung
- verfügt über Sliding Window Verfahren
- hat die Möglichkeit von Vorrangdaten
- verfügt über Adressierung der Ende-zu-Ende-Verbindung durch Portnummern in Verbindung mit IP-Adressen

Ports

Server-Prozesse erwarten sowohl bei UDP als auch bei TCP bestimmte Dienste auf bestimmten Portnummern. Für diese Dienste sind niedrige Nummern standardmäßig vergeben. Sie sind notwendig, wenn mehrere Anwendungen auf einem Rechner laufen, die mit dem Netz kommunizieren. Diese Nummern sind 16 Bit groß, damit können bis zu 65535 TCP- Verbindungen mit einer einzigen IP- Adresse aufgebaut werden.

So genannte "well known" Portnummern (0 bis 1023) sind weltweit eindeutig adressiert

Portnummer	Protokoll
20	FTP (Daten)
21	FTP (Befehle)
22	Secure Shell
23	Telnet

25	SMTP
53	DNS-Server
70	Gopher
79	Finger
80	HTTP (Proxy-Server)
110	POP3
119	NNTP
143	IMAP
194	IRC
210	WAIS
256 - 1023	UNIX-spezifische Services
540	UUCP
1024 - 49151	Registered Ports
49152 - 65535	Dynamic / Private Ports

IP-Adresse und Portnummer definieren einen Kommunikationsendpunkt, der in der TCP/IP-Welt "Socket" genannt wird.

7 Literatur

- [1]
  Tanenbaum; Andrew S.
  Computernetzwerke
  1981 Prentice-Hall Inc., Englewood Cliffs N. J. 07632
  ISBN 0-13-165183-8

- [2]
  Comer, Douglas E.
  Computernetzwerke und Internets
  2000 By Pearson Studium
  ISBN 3-8273-7012-4

- [3]
  Comer:, Douglas E.
  Internetworking with TCP/IP
  1988 by Prentice Hall Inc.
  ISBN 0-13-470188-7

- [4]
  Tanenbaum; Andrew S.
  Operating Systems Design and Implementation
  1987 Prentice-Hall Inc., Englewood Cliffs N. J. 07632
  ISBN 0-13-637331-3 025

- [5]
  Tanenbaum; Andrew S.
  Moderne Betriebssysteme. 2., überarbeitete Auflage
  1995 Prentice-Hall Inc.,
  ISBN 3-8273-7019-1

- [6]

  Tanenbaum; Andrew S.

  Verteilte Betriebssysteme

  2000 Prentice-Hall Inc.,

  ISBN 3-930436-23-x

- [7]

  Coulouris, George; Dollimore, Jean; Kindberg, Tim;

  Distributed Systems: Concepts and Design. Second Edition

  1994 Addison – Wesley

  ISBN 0-201-62433-8

- [8]

  Bach, Maurice J.

  THE DESIGN OF THE UNIX OPERATING SYSTEM

  1986 Prentice-Hall Inc., Englewood Cliffs N. J. 07632

  ISBN 0-13-201757-1 025

- [9]

  Durcansky, Georg

  EMV-gerechtes Gerätedesign

  1992 Franzis Verlag GmbH & KG, München

  ISBN 3-7723-5385-1

- [10]

  Blasinger, Frank

  Digitale Schnittstellen und Bussysteme für die Kommunikation

  Firmenliteratur Firma Jumo

- [11]

  Eberhart, Udo; Hildebrandt, Günter; Klem, Hans Jochim; Maik, Otto; Pahnke Karsten;

  Senf, Frank; Schaufelberger, Siebert; Schulze, Stefan; Sommer, Detlef;

  USB 1.1 Universal Serial Bus, Datendienste, Errorhandling, Powermanagement, USB

  Treiber – Struktur, USB Funktions – Bausteine,

2. aktualisierte und erweiterte Ausgabe
2000 Franzis Verlag GmbH, 85586 Poing
ISBN 3-7723-7964-8

➢ [12]
TU Graz, G. Winkler: Schutzschaltungen für elektronische Geräte,
Vortragsskript über EMV
1997 TU-Graz

➢ [13]
Bussysteme in der Automatisierungstechnik und Prozesstechnik. Grundlagen und Systeme der industriellen Kommunikation, 4. Auflage.
2000 Vieweg Verlag
ISBN: 3-528-36569-2

➢ [14]
Etschberger, K.
CAN Controller Area Network - Grundlagen, Protokolle, Bausteine, Anwendungen.
2000 Hanser-Verlag
ISBN 3-446-19431-2

➢ [15]
Etschberger, K.
CAN-based higher Layer Protocols and Profiles
Internet:
http://www.ixxat.de/article_can_based_higher_layer_protocols_de,2448,147.html
2004 IXXAT Automation GmbH.

➢ [16]
Date G. J.
An Introduction to Database Systems Seventh Edition
2000 Addison Wesley Longman, Inc
ISBN 0-201-68419-5

- [17]
  Rolia J., Slonim J., Botsford J.
  Open Distributed Processing and Distributed Platforms
  Proceedings of the IFIP/IEEE international conference on open distributed processing and distributed platforms, Toronto, Canada, May 26-30,
  1997. London: Chapman & Hall, 1997, pp. 15-33
  ISBN 0-412-81230-4

- [18]
  Helal A. A., Heddaya A. A., Bharagava B. B.
  Replication Techniques in Distributed Systems
  1996 Kluwer Academic Publishers
  ISBN 0-7923-9800-9

- [19]
  Garg V. K.
  Prinziples of Distributed Systemes
  1996 Kluwer Academic Publishers
  ISBN 0-7923-9668-5

- [20]
  Kopetz H.
  Real – Time System: Design Principles for Distributed Embedded Applications
  1997 Kluwer Academic Publisher, AH Dordrecht
  ISBN 0-7923-9894-7

- [21] Rall B.
  Probleme der elektromagnetischen Verträglichkeit bei Bus-Kommunikationssystemen.
  Fachtagung „Sensoren – Aktoren – Buskommunikation"
  Magdeburg 1995

- [22]
  H. Kirrmann, „Industrieller Einsatz fehlertoleranter Rechner“,
  1988 Oldenbourg Verlag, it-Informationstechnik 30,
  Nr. 3, S. 186–195.

- [23]
  K.-E. Großpietsch, E. Maehle, „Fehlerbehandlung in komplexen
  nebenläufigen Systemen“
  Informatik-Spektrum, Dezember
  1998, S. 347–355

- [24]
  M. Euringer, W. Reichert, T. Schlierf, „Hochverfügbare und
  fehlersichere Prozessautomatisierung mit redundanten Systemen“
  Siemens Energie & Automation 8 (1986) Heft 5, S. 334–336.

- [25]
  P. David,C. Guidal, „Development of a Fault-Tolerant Computer
  System for the HERMES Space Shuttle“, Proc. FTCS-23,
  Toulouse 1993, S. 641–647.

8 Register

8.1 Abbildungen

Abb.: 1.1 Asiatischer Abakus 5
Abb.: 1.2 Colossus 6
Abb.: 1.3 Photo des ersten „Bugs“ in einem Computer 7
Abb.: 1.4 Hollerith mit Lochkartenmaschine 8
Abb.: 1.5 Diagramm für die 4-Knoten Verbindung 9
Abb.: 1.6 Erstes Diagramm für die Verbindung 9
Abb.: 1.7 Handschriftliches Protokoll vom 29.10.1969 10
Abb.: 1.8 I(ntegrated)C(ircuit) 11
Abb.: 1.9 Multicomputer 16
Abb.: 3.1 Abstrakte Anlagendarstellung 22
Abb.: 3.2 Bearbeitung von Fehlfunktionen 23
Abb.: 3.3 Diagramm zum Lösungsansatz 1 29
Abb.: 3.4 Diagramm zum Lösungsansatz 2 30
Abb.: 3.5 Diagramm zum Lösungsansatz 3 31
Abb.: 4.1 Anlagenschema 50
Abb.: 4.2 Steuerungsübersicht 34
Abb.: 4.3 Flussdiagramm 36
Abb.: 4.5 Grafische Benutzerschnittstelle 60
Abb.: 4.6 Grafische Benutzerschnittstelle 74
Abb.: 4.7 Parameter Trockensaugen 75
Abb.: 4.9 Symbolleiste 76
Abb.: 4.10 Anlaufphase 77
Abb.: 4.11 Anlaufphase 78
Abb.: 4.12 Gerätedaten 80
Abb.: 4.13 Ankunftssignale 80
Abb.: 4.14 Pufferverwaltung 89
Abb.: 4.15 CAN-Bus Anschaltung 104
Abb.: 6.1 CAN 2.0A Message Frame 113
Abb.: 6.2 CAN 2.0B Message Frame 114
Abb.: 6.3 Physikalische Struktur des USB 116
Abb.: 6.4 Fullspeed USB-Kabel 117
Abb.: 6.5 High und Low Speed Verbindung 118
Abb.: 6.6 USB Schichtenmodell 120
Abb.: 6.7 TCP/IP – OSI Modell 121
Abb.: 6.8 ARP Auflösung 124
Abb.: 6.9 TCP/IP Layer 124
Abb.: 6.10 Format des IP-Headers 126
Abb.: 6.11 UDP- Header 127
Abb.: 6.12 Sliding window 129
Abb.: 6.13 TCP-Header 130

8.2 Tabellen

Tabelle 1: Hardwarekonzept und die Taxonomie nach Flynn[1][4][7] 15

Tabelle 2: Speichermodell der Systemabläufe 27

Tabelle 3: tab_sprachen
Generierung der Ressoucefiles für die einzelnen Sprachen 57

Tabelle 4: tab_Adressen
Adressenverwaltung der einzelnen Stationen 62

Tabelle 5: tab_Geraete
Konfiguration der Stationen 63

Tabelle 6: tab_Linien
Konfiguration der Linien 64

Tabelle 7: tab_Verteiler
Konfiguration der Linienübergabe 64

Tabelle 8: tab_Sending_Restrictions
Zuteilung der Adressen zu den Stationen 64

Tabelle 9: tab_Errors
Fehlerprotokoll 65

Tabelle 10: tab_Checkpoint
Checkpointverwaltung 65

Tabelle 11: tab_Fahrt
Eingespeicherte Sendungen 66

Tabelle 12: tab_Fahrtdetail
Protokollierung der Sendungen 66

Tabelle 13: tab_Stati
Namenstabelle für alle Stati 67

Tabelle 14: tab_Language
Spracheneinstellung 67

Tabelle 15: tab_Fahrtdetaildescription
Text für die Sendungsdetails 67

Tabelle 16: tab_Gerätearten
Beschreibung der Stationstypen 67

Tabelle 17: tab_Ankunftssignale
Beschreibung der Ankunftssignale 68

Danksagung

Allen die mich während der Entstehung dieser Arbeit direkt oder indirekt unterstützt haben, möchte ich an dieser Stelle aufrichtig Danke sagen. Im Besonderen

Herrn o. Univ. Prof. Dr. Dr.h.c.mult. Peter Kopacek, der mir die Möglichkeit gab, diese Arbeit durchzuführen und mich dabei in jeder Hinsicht unterstützte,

den Mitarbeitern des Instituts für Handhabungsgeräte und Robotertechnik an der technischen Universität Wien,

meinem alten Freund Dipl.-Ing. Dr. Robert Probst, der mich in manchen verzwickten Situationen immer wieder unterstützte und einen wesentlichen Beitrag zur Realisierung und Installation des Systems leistete,

meinen Freund Dr. Philipp Kohler – Redlich, der mir immer wieder beratend zur Seite gestanden ist,

ein spezielles Danke an meine Frau und Tochter, die mich oft vermissen mussten und mir in Zeiten der Mutlosigkeit Zuspruch und Motivationen gaben.

Lebenslauf

Angaben zu Person:	Dipl.-Ing. Johann Adlmaninger, geb. am 8.9.1955 in Tollet als Sohn von Johann und Theresia Adlmaninger, röm. kath., österreichischer Staatsbürger, verheiratet mit Dipl.-Ing. Gabriele Adlmaninger, geb. Mörtlitsch, eine Tochter.
Schulbildung:	1961-1966 Volksschule in Taufkirchen an der Trattnach 1966-1970 Hauptschule in Grieskirchen 1976-1978 Werkmeisterlehrgang für Elektronik und Regelungstechnik in Linz 1979-1980 Studienberechtigungsprüfung für Informatik an der TU-Wien
Präsenzdienst:	1975-1976 in Hörsching und in Siezenheim bei Salzburg
Studium:	1980-1986 Informatikstudium an der Universität Linz, Diplomarbeit Tabellengesteuerte Assembler für Mikroprozessoren. 1986 Sponsion zum Diplom-Ingenieur.
Beruflicher Werdegang:	1970-1974 Lehre als Elektroinstallateur 1974-1981 Beschäftigt bei der Firma Siemens in der Nachrichtentechnik 1986-1987 Beschäftigt bei der Fa. Chemie Linz AG 1987-1998 Selbstständig im Bereich der Automatisierungstechnik Seit 14.9.1998 beschäftigt beim Landesschulrat für Kärnten als Fachtheoretiker für Automatisierungstechnik