Dokumentation
zur
Raumkomponentensteuerung über TCP/IP

im Rahmen der Technikerarbeit 2002 an der Werner-Siemens-Schule / Stuttgart

von Oliver Barkhofen und Götz Mayer

Erklärung

Hiermit erklären wir, dass die vorliegende Technikerarbeit eine eigenständige Leistung darstellt und nicht auf der Basis einer bereits vorhandenen Techniker-, Diplom- oder ähnlicher Arbeit erstellt wurde. Bei der Durchführung und Ausarbeitung wurden nur die zulässigen Hilfsmittel verwendet.

Uns ist bewusst, dass bei einem Verstoß gegen diese Erklärung innerhalb der gesetzlichen Einspruchsfristen auch im Nachhinein die Leistungsbewertung aberkannt werden kann. Damit erlischt die Berechtigung zum Tragen der Berufsbezeichnung des staatlich geprüften Technikers.

Hiermit versichere ich, Götz Mayer, dass ich folgende praktischen Teile und Gliederungspunkte selbständig bearbeitet und verfasst habe.

- *Layouts und Schaltpläne aller Platinen*
- *Programmierung der Client-Software*
- *Bedienungsanleitung des Clients*

Bedanken möchte ich mich bei meinen Eltern und meinem Bruder für ihre Unterstützung, bei Peter Kampf für die Verwendung seines EAGLE Programms und bei Otto Neff für die Hilfe bei der Einarbeitung in die Visual Basic Winsock Komponente.

Hiermit versichere ich, Oliver Barkhofen, dass ich folgende praktischen Teile und Gliederungspunkte selbständig bearbeitet und verfasst habe.

- *Programmierung der Server-Software und des Assemblerprogramms*
- *Dokumentation außer die Bedienungsanleitung des Clients*

Bedanken möchte ich mich bei Götz Mayer, für die Idee dieser Projektarbeit und für die beinahe unerschöpfliche Energie, die er aufbringen mußte, mich als Partner zu gewinnen, bei Georg Dittrich, für die Hilfe bei dem erneuten Einstieg in die Mikrocontrollerprogrammierung, bei meinem bereits verstorbenen Großvater, für die geistige und finanzielle Unterstützung.

Stuttgart, den _____

Götz Mayer	Oliver Barkhofen

Gestaltung Umschlagfoto: Veith Mayer
Grafische Gestaltung: Oliver Barkhofen
Text: Oliver Barkhofen und Götz Mayer
Druck und Bindearbeit: Digital Competence Center Kästl, 73760 Ostfildern

Diese Dokumentation wurde mit der 60-Tage-Versuchsversion von QuarkXpress Passport 5.0 beta erstellt.
Die Schaltpläne wurden mit Orcad 9.2 Lite erstellt.
Das Umschlagfoto wurde mit Adobe Photoshop erstellt.
Die Bilderausschnitte der einzelnen Bauteile wurden aus den jeweiligen Datenblättern importiert.

Die Server- und Clientanwendungen wurden mit Visual Basic 6 erstellt.
Das Assemblerprogramm wurde mit der Demoversion µVision 2 von Keil erstellt und mit “AT89S PC Based Programmer” von AEC in den Versionen 2 und 3 in den Controller übertragen.

Vorwort

Im zwanzigsten Jahrhundert, ist das "World Wide Web" nicht mehr wegzudenken, es lässt die Kommunikation zwischen Menschen unterschiedlicher Kontinente zu. Doch nicht nur das, es erlaubt sogar das sich Maschinen und Anlagen miteinander verständigen und so ganze Firmenkomplexe über Internet gesteuert werden können. Diese Technologie hat bereits Einzug auf dem freien Markt erhalten, doch noch zu unerschwinglichen Preisen. Die Idee war nun, auch eine Variante für den Durchschnittsbürger zu erschaffen, die trotz geringem Preis auch an die Leistungen der gewerblichen Produkte herankommt.

Als erstes sollte das Projekt mit einem herkömmlichen PC, der in fast jedem Haushalt schon vorhanden ist, realisiert werden können, auf dem das Serverprogramm installiert ist.
Mit einer Client - Applikation (siehe Kapitel 6.1) kann man sich dann in den PC einloggen und sämtliche Daten abrufen bzw. Befehle senden.
An diesem Server - PC (siehe Kapitel 3), der über einen Internet Zugang verfügen sollte, hängt über die serielle Schnittstelle verbunden eine Controllereinheit, die dann über einen I²C Bus mit bis zu max. 16 Raum - Controllereinheiten kommuniziert. Diese Idee wurde jedoch schnell verworfen (siehe Kapitel 10).

Somit beschränkt sich das Projekt auf einen Raum - Controller (Unit). Die Unit ist im Stande, 6 Endgeräte, wie zum Beispiel einen Rollo, ein Fenster und 4 Lampen, zu steuern, zudem ermittelt sie auch die aktuelle Raumtemperatur. Alle Daten, werden dann über die serielle Schnittstelle an den Server - PC weitergegeben, der diese dann an ein eventuell angemeldetes Clientprogramm über TCP/IP weiterleitet. Natürlich kann die Unit auch über ein Eingabe Feld, manuell bedient werden.
Das ganze System hat jedoch noch einen Nachteil: Wenn der Anwender keine feste IP besitzt, sondern nur einen herkömmlichen Zugang mit dynamischer IP, bekommt er bei einem IP-Wechsel keine Verbindung mehr zum Server - PC (siehe Kapitel 9.2).
Dies ist z.B. der Fall wenn man sich im Ausland befindet und der Provider eine Zwangstrennung vornimmt.
An einer Lösung dieses Problems wird gerade gearbeitet, sie wird aber nicht mehr bis in die jetzige Server - Applikation einfließen, sondern später als Updateversion zur Verfügung gestellt.
Die Realisierung wird so aussehen, daß man z.B. mit einem Handy den Server - PC anwählt, der dann über die CLIP-Funktion (Calling Line Identification Presentation = Erkennung der Rufnummer) den User erkennt, sich automatisch ins Internet einwählt und die ihm zugewiesene IP per E-Mail oder SMS an den Benutzer weitergibt. Diese Option erfordert jedoch bei Benutzern von ADSL ein zusätzliches Modem oder ISDN Karte.

Götz Mayer und Oliver Barkhofen

Inhaltsverzeichnis

Pflichtenheft

zum Projekt
Raumkomponentensteuerung über TCP/IP

Betreuender Lehrer Herr Borsum

1.1 Allgemeine Funktionsbeschreibung

Die Komponenten eines Raumes, dazu zählen wir Fenster, Jalousien und Licht werden mit unserem Projekt "ferngesteuert". Hierfür werden 2 Rechner benötigt, die entweder einen Internetanschluss besitzen oder sich beide in einem lokalen Netzwerk befinden.
Einer dieser Rechner muss sich unbedingt in der Nähe des zu steuernden Zimmers befinden, es reicht aber schon aus, wenn er auf derselben Etage oder sogar in derselben Wohnung ist.
Zusätzlich wird an diesem Rechner ein Mikrokontrollersystem angeschlossen, welches die Schnittstelle zwischen dem Rechner an TCP/IP und den einzelnen Komponenten bildet.

Der Einfachheit halber wird untenstehend der Rechner, der die Komponenten steuert als den Server und der andere als Client bezeichnet.

Auf beiden Rechnern werden nun unterschiedliche Software ausgeführt, die es über eine grafische Oberfläche ermöglichen, Befehle vom Client zum Server zu schicken, dort umzuwandeln und dem Mikrokontroller anschließend über die V24 Schnittstelle des Servers in einer verständlichen Sprache zu übermitteln.
Wenn sich nun ein Zustand einer Komponente ändert, z.B. die Jalousie fährt an den unteren Anschlag, dann wird dieses Ereignis dem Client zurück über Mikrokontroller, Server und Internet/LAN mitgeteilt.
Selbstverständlich können die Komponenten auch noch mit ganz normalen Tastern bedient werden.

Des weiteren wird die aktuelle Raumtemperatur über den Server an den Client übermittelt und dort angezeigt.

1.2 Spannungsversorgung

Als Spannungsversorgung für die Mikrocontroller-Einheit wird ein 24V Steckernetzteil mit max. 300 mA Ausgangsstrom verwendet.

1.3 EMV-Maßnahmen

Als Maßnahmen, um den Elektromagnetischen (Un)verträglichkeiten, vorzubeugen, sind RC-Glieder an den Leitungen vorgesehen, welche von dem Microcontroller Board auf die verschiedenen Taster führen.
Sowie eine abgeschirmte V24-Leitung, da der Server sich in einem anderen Zimmer befinden darf.

1.4 Gruppeneinteilung

Der theoretische Teil wurde gemeinsam bearbeitet. Bei der Umsetzung der Teilgebiete übernimmt Götz Mayer die Hardware und die Software für den Client. Oliver Barkhofen kümmert sich um die Software des Servers und die Assemblerprogrammierung.

Unser Betreuungslehrer ist Herr Borsum.

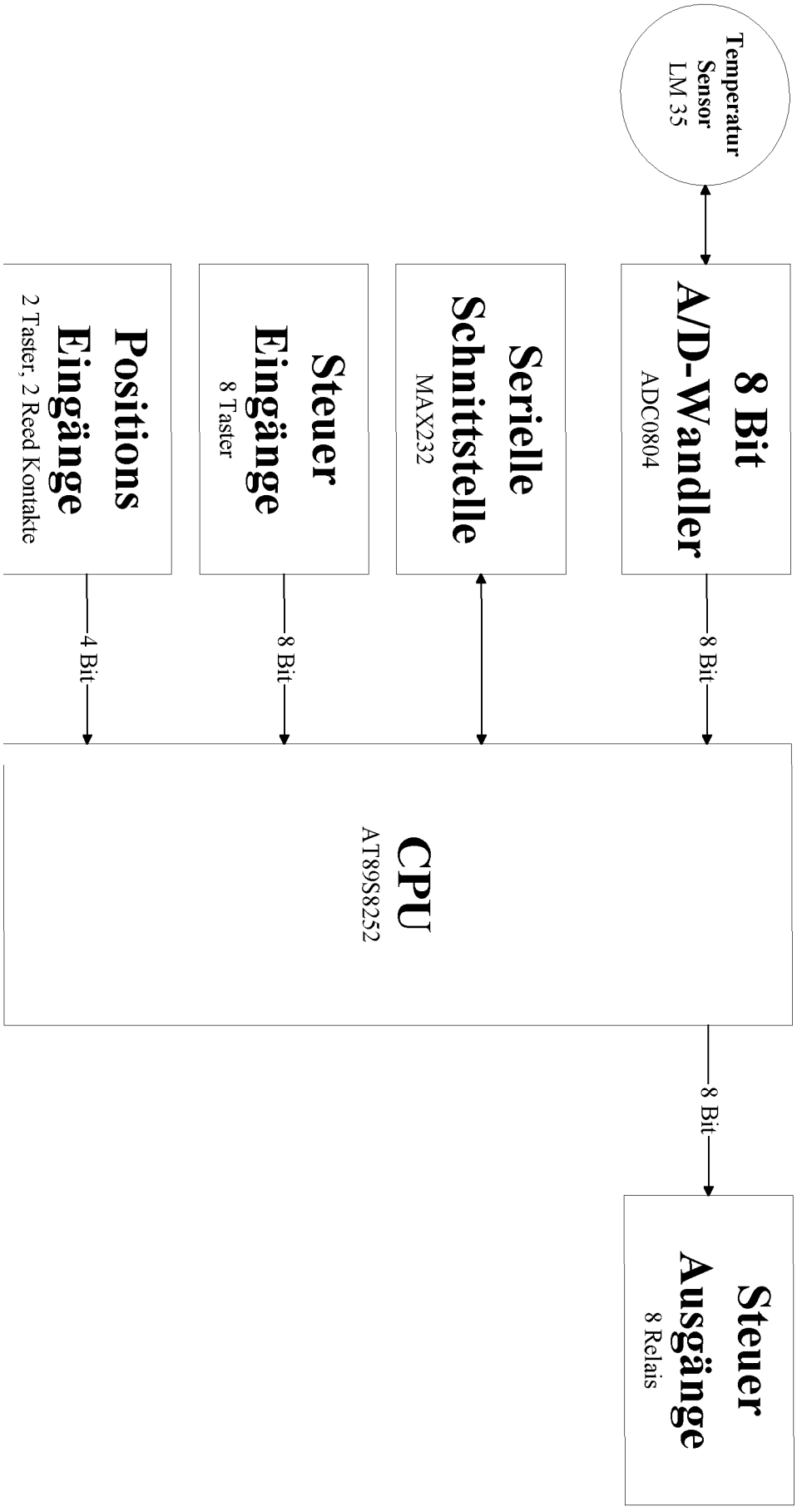
1.5 Kostenaufwand

Einen Preis werden wir nicht angeben, da wir nicht die Absicht haben, der WSS unsere TAR zu überlassen.

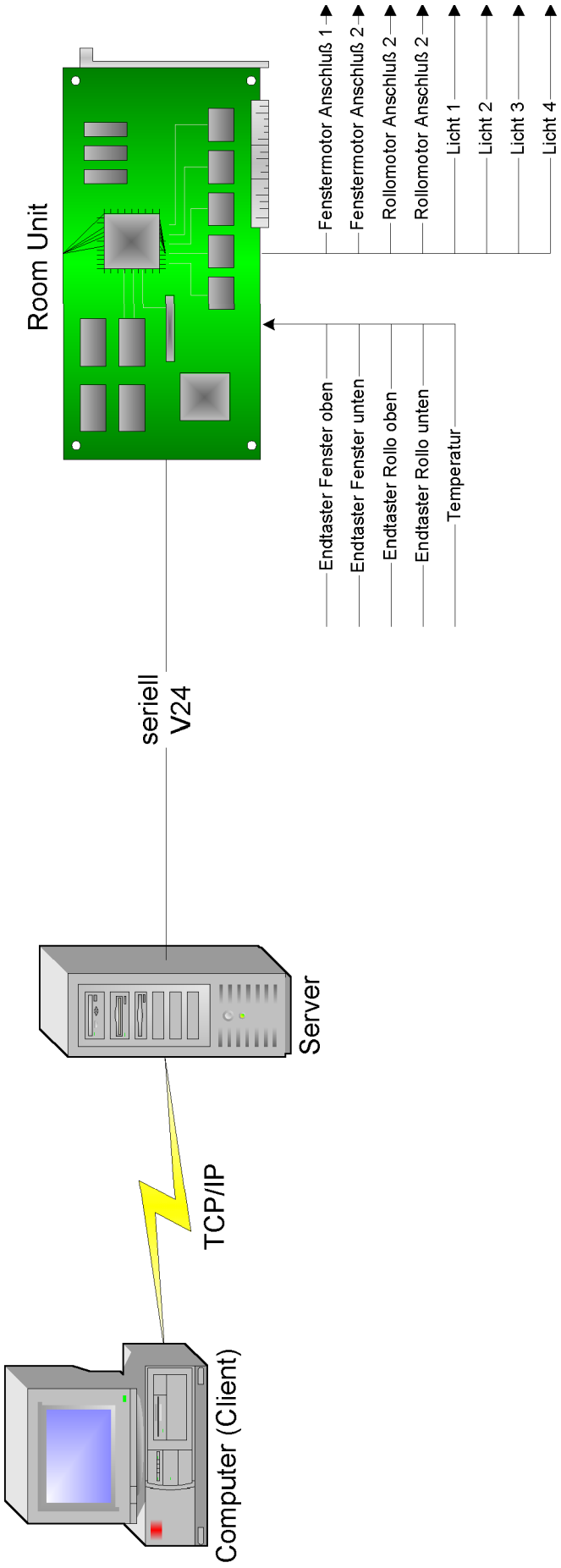
1.6 Blockschaltbild

siehe folgende Doppelseite

Blockschaltbild Teil 1



Blockschaltbild Teil 2



Bedienungsanleitung
für den Server

2. Installation des Servers

2.1 Installation der Software

Um die Installation zu beginnen, starten Sie die Datei “Setup.exe“ im Ordner “HomeControl Server“ auf der CD. Als erstes werden Sie darauf hingewiesen alle Programme, die im Hintergrund weiterlaufen, zu schließen, um mögliche Systemdateien aktualisieren zu können.

In der zweiten Phase der Installation haben Sie die Möglichkeit einen anderen als den Installationspfad anzugeben. Klicken Sie hierbei auf “Verzeichnis wechseln“ und geben Sie bei “Pfad:“ Ihren kompletten Installationspfad an oder verwenden Sie die Verzeichnis- und Laufwerkslisten, um den von Ihnen gewünschten Pfad zu suchen.

Falls Sie den von Ihnen gesuchten Pfad gefunden haben, starten Sie den Kopiervorgang indem Sie auf den Button mit dem Computer klicken. Das Programm und die dazugehörende Komponenten werden nun installiert.

In der abschließenden Phase haben Sie die Möglichkeit den Ort des Eintrags der Startdatei “HomeControl Server.exe“ im Startmenü von Windows zu bestimmen. Mit dieser Auswahl wird das Installationsprogramm beendet.

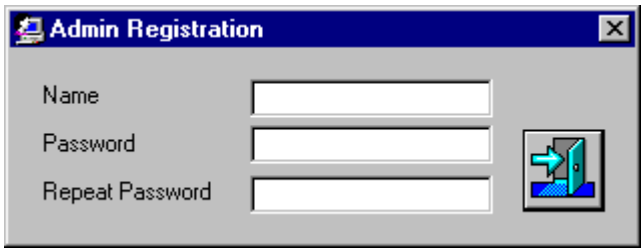
2.2 Deinstallation der Software

Falls Sie das Programm von Ihrem Rechner löschen möchten, öffnen Sie des Fenster “Software“ in dem Systemordner “Systemsteuerung“, der über den “Arbeitsplatz“ erreichbar ist. Wählen Sie dort den Eintrag “HomeControl Server 2.1“. Klicken Sie anschließend auf “Hinzufügen/Entfernen“.

Möglicherweise werden Sie während des Löschvorgangs gefragt, ob Sie eine Komponente, die von mehreren Programmen benutzt wird, wirklich löschen möchten. Kontaktieren Sie in diesem Fall Ihren Administrator. Im Zweifelsfall aber, belassen Sie diese Komponente auf Ihrem System, um Fehlfunktionen bei anderen Programmen zu vermeiden.

Die Einträge in der Registry und die Datei “Config.ini“ werden nicht mitgelöscht, da sie erst nach Installation erstellt wurden. Entfernen Sie bitte manuell den Installationspfad, in dem sich die “Config.ini“ befindet. Die Einträge in der Registry müssen ebenfalls manuell entfernt werden, **dies sollte aber nur vom Administrator durchgeführt werden. Bei unsachgemäßer Durchführung kann es zu erheblichen Fehlfunktionen oder dem kompletten Ausfall von Windows kommen.**
Der Schlüssel, der zu entfernen ist, heißt `HKEY_USERS\.DEFAULT\Software\VB and VBA Program Settings\Server` für Windows9x und `HKEY_CURRENT_USER\Software\VB and VBA Program Settings\Server` für Windows2000. Sie könnn mit dem zu Windows dazugehörenden Programm “Regedit.exe” entfernt werden. Sie können das Programm starten, in dem Sie auf den Start-Button von Windows und anschliessend auf “Ausführen...” klicken. Geben Sie nun `regedit` ein und drücken Sie dann auf `OK.` Suchen Sie nun den Schlüssel und entfernen Sie den Unterschlüssel `Server.`

2.3 Beim ersten Start



Wenn Sie den Server das erste Mal starten, werden Sie aufgefordert einen Namen und zweimal ein Passwort einzugeben. Dies werden zukünftig die Logindaten des Administrators sein.
Der Administratorlogin ist nötig, um die Daten der registrierten Benutzer nur einer Person zu überlassen, um unsachgemäßen Gebrauch zu vermeiden.

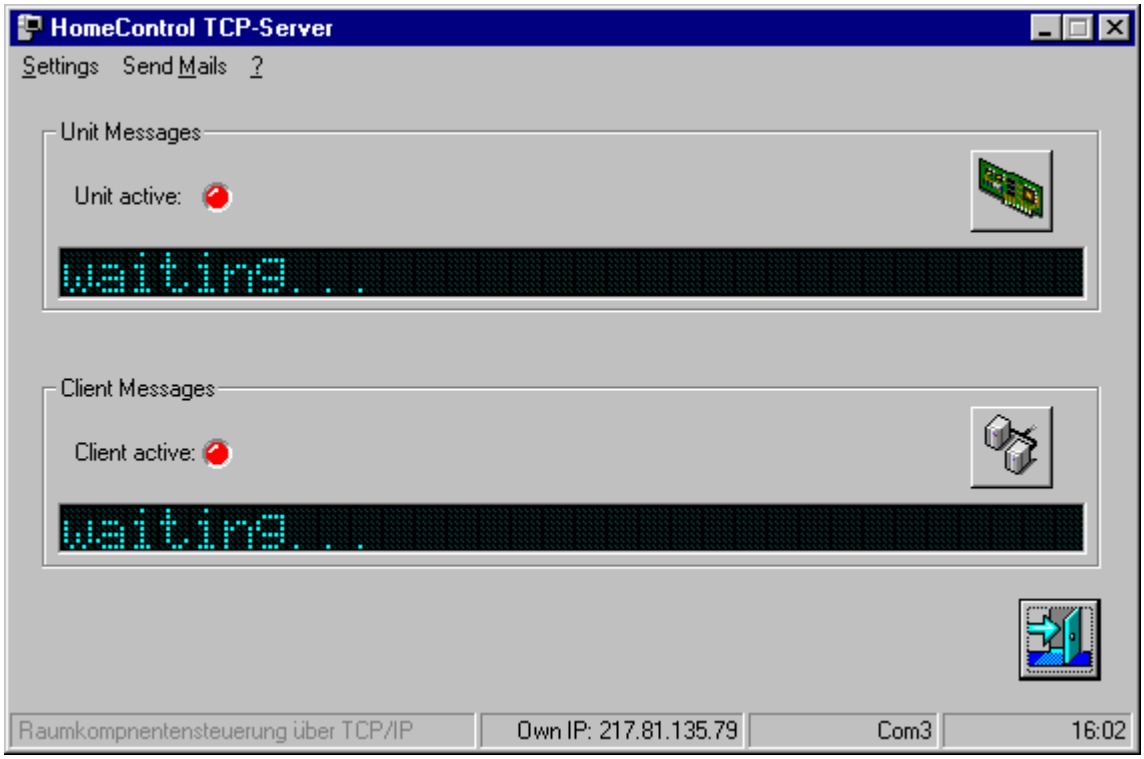
Sollten Sie beim ersten Start dieses Eingabefenster einfach geschlossen haben, so werden Sie während dieser Sitzung nicht in der Lage sein, User anzulegen. Um dennoch einen Administratorlogin festzulegen, schließen Sie den Server und starten Sie ihn anschließend gleich wieder. Sie werden dann erneut nach einem Administratornamen und -passwort gefragt.

Nachdem Sie ein Administratorlogin festgelegt haben, sollten Sie noch mindestens einen User anlegen, da nur die Benutzer sich über den Client am Server anmelden können, die auch in der Benutzerliste im Fenster “User“ enthalten sind. Es reicht also nicht aus nur einen Administrator anzumelden. Dieser ist kein Benutzer im eigentlichen Sinne.

Die Daten des Administratorlogins können jederzeit durch den Administrator selbst geändert werden. Öffnen Sie dazu das “User“-Fenster, drücken Sie im Menü auf “Change Admin“ und geben Sie darauf den neuen Namen und Passwort ein.

3. Dialoge des Servers

3.1 Das Hauptfenster



Das Hauptfenster ist in 4 Bedienungseinheiten aufgeteilt:

- das Menü unterhalb der Titelleiste
- Bedien- und Anzeigeelemente für den Client
- Bedien- und Anzeigeelemente für die Controllereinheit
- den Beendenbutton

Die beiden Displays zeigen jeweils Ereignisse und Fehlermeldungen an, die im Zusammenhang mit der dementsprechenden Gegenstelle steht. Beispiele finden Sie unter
“3.3.Meldungen der Displays“

Die LEDs zeigen die Betriebsbereitschaft der Gegenstelle an. Grün für aktiv, rot für inaktiv.

Der Verbindungsbutton im Bereich des Controllereinheit verbindet den Server mit der Controllereinheit und trennt die zuvor erfolgreich zustande gekommene Verbindung wieder. Der Button im Bereich des Clienten dagegen verbindet den Server nicht mit dem Clienten, sondern veranlaßt den Server in den Wachemodus zu gehen. Ab sofort wartet nun der Server auf eine Verbindungsanforderung seitens des Client. Wird dieser Button nun während des Wachemodus oder während einer bestehenden Verbindung zum Client betätigt, so wird die Verbindung/der Wachemodus augenblicklich abgebrochen. In diesem Zustand ist es keinem Benutzer möglich sich am Server anzumelden.

In der Statuszeile kann man die aktuelle IP, der aktuelle COM-Port und die Zeit entnehmen. Sollte sich der Server zum Zeitpunkt des Startens nicht im Internet befinden, so wird anstatt der IP, das Wort “Offline“ angezeigt. Darüber hinaus wird ein Text eingeblendet, der nochmal darauf hinweist, daß sich der Server nicht im Internet befindet. Sollten Sie jedoch wünschen eine Verbindung über das Internet herzustellen, so beenden Sie bitte den Server, stellen Sie eine Verbindung zu Ihrem Provider her und starten Sie erneut den Server.

Sollten Sie sich Ihr Rechner in einem Netzwerk befinden, so kontaktieren Sie Ihren Netzwerkadministrator.

Die Zeitanzeige auf der rechten Seite der Statuszeile entspricht der unter Windows eingestellten Zeit und wird nur alle 10 Sekunden aktualisiert. Zudem ist das die Zeit, die dem Client übertragen und dort in dem Display angezeigt wird.

3.2 Das Maileinstellungs -Fenster

Mit dem Menüpunkt “Mail“ stellt man die Erscheinungsoptionen des Mailversandfensters, den SMTP-Server und die Option “Beim Start die aktuelle IP versenden“ ein.

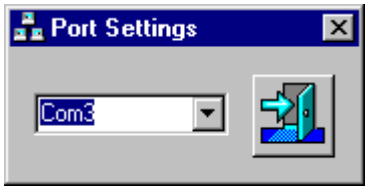
Im Eingabefeld “SMTP Server“ muß ein gültiger Postausgangsserver eingegeben werden, da sonst keine Mails mit der IP oder den Logindaten eines Benutzers versandt werden können.

Das Optionsfeld “Send IP at start-up“ ermöglicht es eine Mail mit der IP an alle registrierten Benutzer zu versenden, nachdem das Serverprogramm gestartet wurde.

Das aktivierte Optionsfeld “Show mail progress windows“ zeigt das Mailversandsfenster an, nachdem auf den Menüpunkt “Send Mail“ geklickt wurde.

Das aktivierte Optionsfeld “Hide mail progress window after IP was sent“ läßt das Mailversandsfenster nach erfolgreichem oder nichterfolgreichem eMailversand wieder verschwinden. Ist das Optionsfeld “Show mail progress windows“ deaktiviert, so kann man dieses Feld nicht verändern.

3.3 Das Porteinstellungs -Fenster

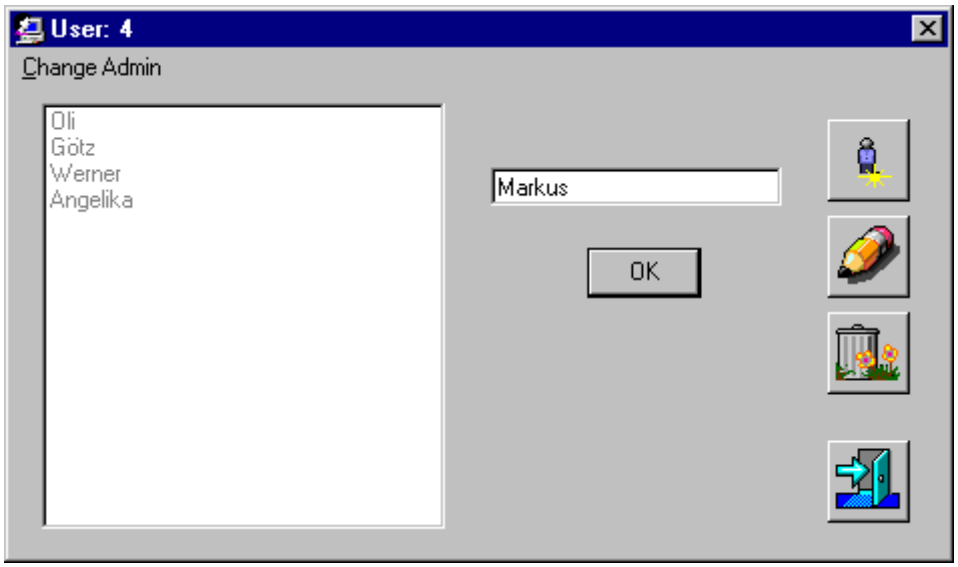


Im Menüpunkt “Port“ kann der COM-Port eingestellt werden, über den der Server mit der Controllereinheit verbunden wird. Sollte bereits eine Verbindung zur Controllereinheit hergestellt worden sein, so kann dieses Fenster erst gar nicht aufgerufen werden. Beenden Sie zuerst die Verbindung und klicken Sie dann auf den Menüpunkt “Port“ und “Settings“.

3.4 Das Roomeinstellungs-Fenster

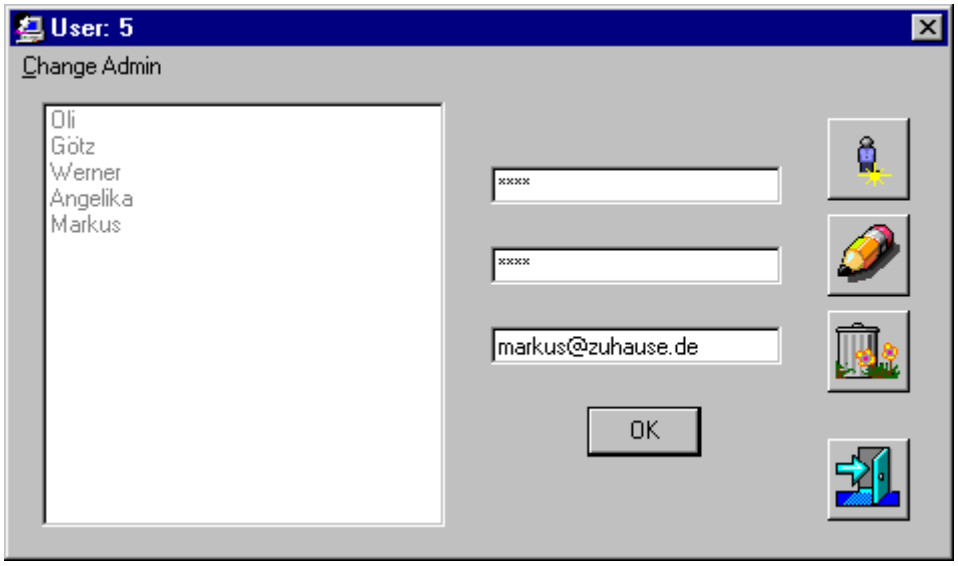
Mit dem Roomeinstellungs-Fenster können die Raumbeschriftungen bestimmt werden, die nach einer aufgebauten Verbindung mit dem Client in die Buttons übernommen werden. Jedes Feld muss ausgefüllt und deren Inhalt darf nicht länger als 13 Zeichen sein. Ansonsten wird das Wort ab dem 13. Zeichen abgeschnitten.
Mit dem “Default“-Button können die Standardbeschriftungen wieder hergestellt werden.

3.5 Das Benutzereinstellungs-Fenster

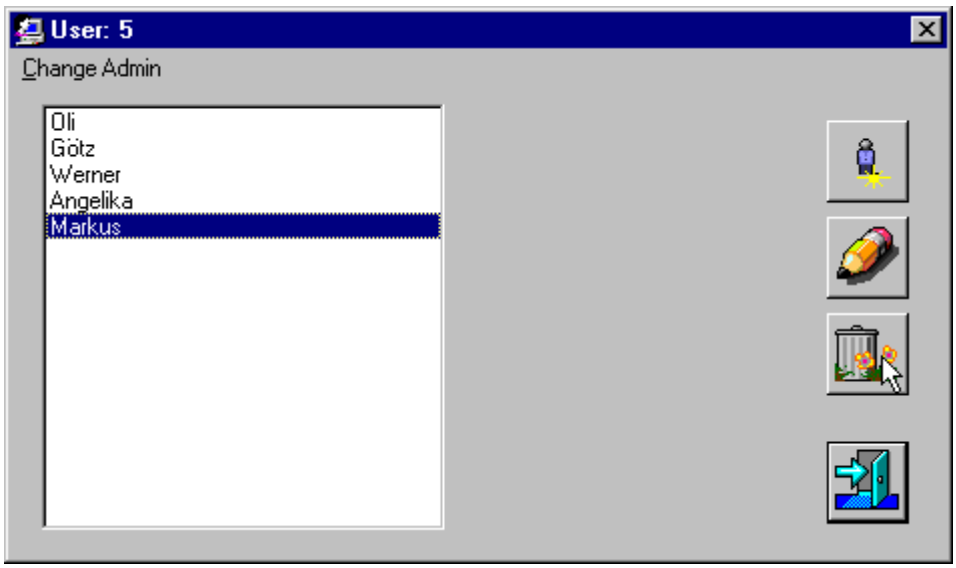


Das Benutzereinstellungs-Fenster wird nach dem Klicken auf den Menüpunkt "User" unter "Settings", und nachdem sich der Administrator mit seinen Logindaten angemeldet hat, angezeigt.

Der oberste Button dient zum Anlegen eines Benutzers. Maximal können 16 Benutzer mit Passwort und eMail-Adresse eingetragen werden. Beim Anlegen müssen alle 4 Felder ausgefüllt werden, die während diesen Vorgangs erscheinen. Bei der eMail-Adresse ist auf korrekte Eingabe zu achten.



Mit dem Stift-Button können Passwort und eMail-Adresse geändert werden. Hierbei brauchen nicht alle Felder audgefüllt werden, sondern nur die zu ändernden. Also entweder nur das eMail-Feld oder die beide Passwortfelder.

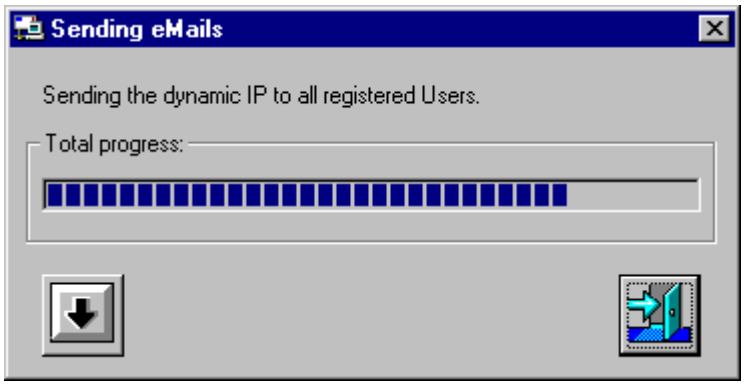


Der “Mülleimer“-Button löscht einen Benutzer, nachdem dieser durch den Administrator markiert wurde.
Vorsicht: Der markierte Benutzer wird beim Klicken auf den “Mülleimer“-Button sofort und ohne Hinweise gelöscht.

Unter dem Menüpunkt “Change Admin“ können die Logindaten des Administrators, d.h. Namen und Passwort geändert werden.
Vorsicht: Es ist zulässig nur den Namen anzugeben, aber ermöglich so unbefugten Dritten leichten Zugriff und Veränderungen.

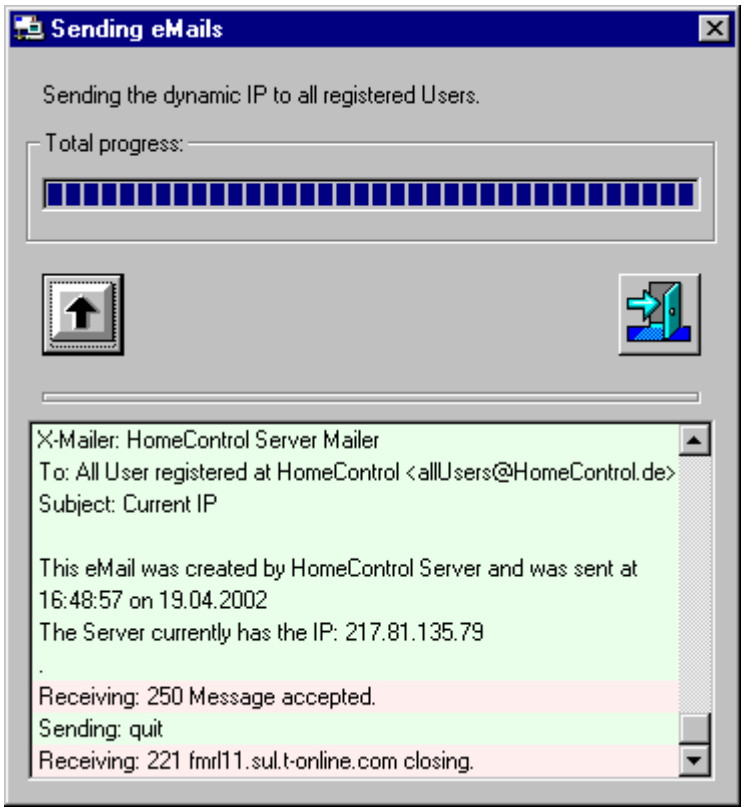
Der “Exit“-Button schließt das “User“-Fenster und kehrt zum Hauptfenster zurück.

3.6 Das Mailversand-Fenster



Das Mailversand-Fenster erscheint nachdem man auf den Menüpunkt “Send Mail“ im Hauptfenster geklickt hat. Vorraussetzung allerdings dafür ist, daß in den “Mail Settings“ die Option “Show mail progress window“ aktiviert wurde. Während des Sendevorgangs füllt sich die Fortschrittsanzeige bis die eMail erfolgreich versandt wurde.

Falls sich die Fortschrittsanzeige nicht weiter füllen sollte, so könnte dafür ein Fehler im Sendevorgang dafür verantwortlich sein.



Wünscht man nun den Grund des Fehler zu erfahren, so kann man sich den Verlauf in detaillierter Aufschlüsselung anzeigen lassen, indem man auf den Button mit dem Pfeil, der nach unten zeigt, drückt.

Die roten Zeilen zeigen die Bestätigungen an, die vom SMTP-Server empfangen wurden.
Die grünen Zeilen enthalten Befehle oder Text, die zum SMTP-Server gesandt wurden.
Nun gibt es noch eine dritte Art von Anzeige: die Fehleranzeige. Sollte ein interner oder ein genereller Fehler auftreten, so wird er in einer blauen Zeile dargestellt.

In dieser Aufschlüsselung läßt sich nun z.B. entnehmen an welche Empfängeradressen die eMail versendet wird oder welchen Text die eMail beinhaltet.

Folgende Punkte werden angezeigt:

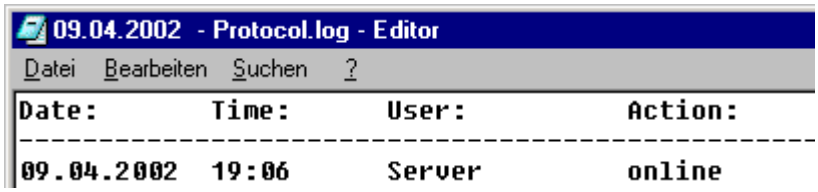
- Verbindungsaufbau zum SMTP-Server
- Übermittlung der Absenderadresse
- Übermittlung der Empfängeradressen
- Übermittlung des eMail-Textes
- Empfang von positiven und negativen Quittungszeichen
- und gegebenenfalls eine Fehlermeldung.

4. Sonstiges

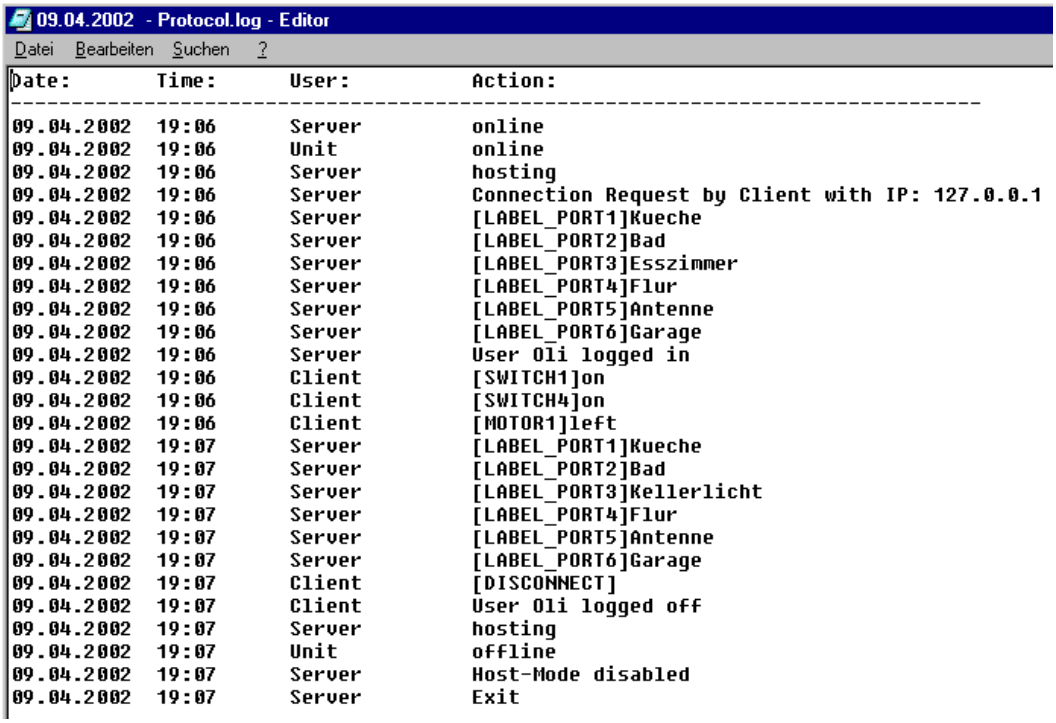
4.1 Protokollierung

Die Protokollierung soll dem Benutzer bzw. dem Administrator helfen, Fehler und Unstimmigkeiten von Verbindungen und Einstellungen zu lokalisieren. Es werden Ereignisse, wie Benutzeranmeldungen oder die Bereitschaften von Controllereinheit und Server.
Neben den Ereignissen werden noch die Zeit und das Datum festgehalten, um diese Vorkommnisse einfacher einer Zeit zu zuordnen.

Alle Protokolle werden in dem Ordner/Verzeichnis abgepeichert, in dem auch das Programm installiert wurde. Für jeden Tag wird ein neues Protokoll erstellt, um die Größe der Dateien, sowie auch die Übersichtlichkeit in Grenzen zu halten. Der Dateiname der Protokolle setzt sich aus dem Datum und das Anhängsel "- Protocol.log" zusammen. Wie im Bild unten in der Titelleiste zu sehen ist.
Wenn ein neues Protokoll erstellt wird, so wird als erstes der Protokollkopf eingefügt, der die unter ihm aufgeführten Daten erläutert.



So könnte zum Beispiel ein Protokoll einer Sitzung mit normalem Verlauf aussehen:



Man sieht hier genau, welchen Verlauf diese Sitzung genommen hat: von dem Start des Servers über die Anmeldung des Clients und der Controllereinheit bis hin zum Aktivieren von 2 Schaltern, eines Motors und dem anschliessenden Beenden des Servers.

Was wird alles protokolliert?

- die Bereitschaft und das Abmelden des Servers, des Clients und der Controllereinheit
- An- und Abmeldung eines Benutzers
- Verbindungsanforderung durch den Client
- Befehle, welche vom Client gesandt und an die Controllereinheit weitergeschickt wurden
- Der Versenden der Logindaten an einen Benutzer
- IP-Adressenversand an alle registrierten Benutzer
- Änderung bzw. Übermittlung der Beschriftungen der Buttons im Client
- Feueralarm
- Fehlermeldungen, wie z.B. Fehler während des Mailversandes oder Fehler während eines Verbindungversuchs

4.2 Meldungen der Displays

Beide Displays zeigen entweder den soeben aktivierten Modus oder Befehle, die vom Client geschickt wurden an.

Folgende Meldungen werden vom Unitdisplay ausgegeben:

- “Trying to connect to Unit ...“
- “Unit disconnected!!!“
- “Int: “ und diverse Fehlerbeschreibungen
- “Connection success!!!“
- “Connection failure!!!“
- “Unit not available!“
- “Fire alert“
- Sämtliche Befehle die Ports betreffend ([SWITCHx]on / off, [MOTORx]left/right, [RESET])

Folgende Meldungen werden vom Clientdisplay ausgegeben:

- “Host-Mode disabled.“
- “hosting...“
- “Disconnected“
- “Sending mail....“
- “Connected to “ und die IP des Client
- “[DISCONNECT]“
- “[ERROR]“ (bei fehlerhafter Übertragung vom Client)
- “Mail sent“

4.3 Konfiguration

Die Einstellungen der Konfiguration werden in der Datei “Config.ini“ im Installationspfad gespeichert. Hierbei ist es Ihnen überlassen, wie Sie die Einstellungen ändern möchten: entweder im Programm selbst oder mit einem Texteditor, wie z.B. dem Notepad unter Windows.

Sollten Sie sich entscheiden, die Konfiguration mit einem Texteditor vorzunehmen, müssen Sie ein paar Punkte beachten, so daß es zu keinen Fehlfunktionen des Programms kommen kann.

- Eine Zeicheneingabe, anstatt einer Zahleneingabe, ist bei “COM“ zwar zulässig, führt aber unweigerlich zu einem Fehler. In diesem Fall kann das Programm erst gar nicht gestartet werden. Ändern Sie den Eintrag, speichern Sie die “Config.ini“ ab und starten Sie das Programm erneut.

- Die Zeilen unterhalb von “SendMail“ entsprechen genau den Einstellungen, die auch in dem Maileinstellungs -Fenster vorgenommen werden können. Es ist darauf zu achten, daß das Wort “True“ für eine aktivierte Funktion und das Wort “False“ für eine deaktivierte Funktion steht.

  Ein Beispiel: Falls Sie die Option “Show mail progress window“ aktiviert und die Option “Hide mail window after IP was sent“ deaktiviert haben, dann steht in der Config.ini hinter “ShowMail“ ein “True“ und hinter “HideMail“ ein “False“.

- Die Einstellungen unter “RoomConfiguration“ müssen alle ausgefüllt werden und die Zeichenketten dürfen nicht länger als 13 Zeichen sein, da sie sonst im Clientfenster in den Buttons auf 13 Zeichen gekürzt angezeigt werden.

- Es können neue Abschnitte und Schlüssel hinzugefügt werden, werden vom Programm aber ignoriert und haben keinen Effekt auf die Funktionen des Servers. So können also Kommentare und Hinweise vom Benutzer oder Administrator eingetragen werden.

Bedienungsanleitung
für den Client

5. Installation des Clients

5.1. Installation der Software

Um die Installation zu beginnen, starten Sie die Datei “Setup.exe“ im Ordner “HomeControl Client“ auf der CD. Als erstes werden Sie darauf hingewiesen alle Programme, die im Hintergrund weiterlaufen, zu schließen, um mögliche Systemdateien aktualisieren zu können.

In der zweiten Phase der Installation haben Sie die Möglichkeit einen anderen als den Installationspfad anzugeben. Klicken Sie hierbei auf “Verzeichnis wechseln“ und geben Sie bei “Pfad:“ Ihren kompletten Installationspfad an oder verwenden Sie die Verzeichnis- und Laufwerkslisten, um den von Ihnen gewünschten Pfad zu suchen.

Falls Sie den von Ihnen gesuchten Pfad gefunden haben, starten Sie den Kopiervorgang indem Sie auf den Button mit dem Computer klicken. Das Programm und die dazugehörende Komponenten werden nun installiert.

In der abschließenden Phase haben Sie die Möglichkeit den Ort des Eintrags der Startdatei “Client.exe“ im Startmenü von Windows zu bestimmen. Mit dieser Auswahl wird das Installationsprogramm beendet.

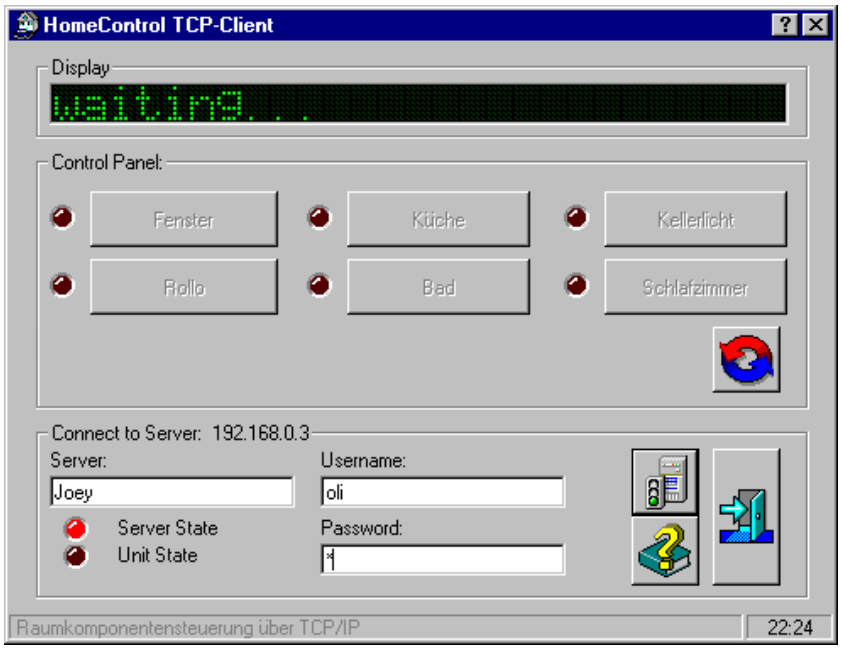
5.2 Deinstallation der Software

Falls Sie das Programm von Ihrem Rechner löschen möchten, öffnen Sie des Fenster “Software“ in dem Systemordner “Systemsteuerung“, der über den “Arbeitsplatz“ erreichbar ist. Wählen Sie dort den Eintrag “HomeControl Client 1.0“. Klicken Sie anschließend auf “Hinzufügen/Entfernen“.

Möglicherweise werden Sie während des Löschvorgangs gefragt, ob Sie eine Komponente, die von mehreren Programmen benutzt wird, wirklich löschen möchten. Kontaktieren Sie in diesem Fall Ihren Administrator. Im Zweifelsfall aber, belassen Sie diese Komponente auf Ihrem System, um Fehlfunktionen bei anderen Programmen zu vermeiden.

Die Einträge in der Datei “Config.ini“ werden nicht mitgelöscht, da sie erst nach Installation erstellt wurden. Entfernen Sie bitte manuell den Installationspfad, in dem sich die “Config.ini“ befindet.

6.1 Der Dialog des Clients



Der Dialog des Clients ist in folgende Bereiche aufgeteilt:

- ein Display für Status-, Temperatur- und Zeitanzeigen
- ein Bereich mit Buttons für die Steuerung der der Komponenten mit deren Statusanzeige
- ein Bereich für die Anmeldung

Um die Komponenten per Fernsteuerung bedienen zu können, muß ein Benutzer mit Namen und Passwort am Server angemeldet sein. Hierzu gibt man seinen Benutzernamen bei ‘Username’ und das Passwort bei ‘Password’ ein. Nachdem man noch den Namen oder die IP des Servers bei ‘Server’ eingegeben hat, baut man eine Verbindung auf indem man auf den ‘Connect/Disconnect’-Button drückt.

Sollte dem Server bei jeder Einwahl in das Internet eine neue IP zugewiesen werden (dynamische IP), so gibt es in der Serversoftware eine Funktion, die auf Knopfdruck oder mit jedem Start der Serversoftware die aktuelle IP an alle registrierten Benutzer versendet wird.
Ist keine dynamische IP vorhanden, z.B. falls keine Verbindung besteht oder in einem LAN, so wird diese Funktion gesperrt.

Falls der Benutzer einmal sein Passwort vergessen hat, so wird ihm eine Möglichkeit geboten dieses direkt anzufordern: man gibt seine eMail-Adresse in das ‘Username’-Feld ein, läßt das ‘Password’-Feld leer und verbindet nun zu dem Server, auf dem der Benutzername und Passwort zu dieser eMail-Adresse hinterlegt sind. Die kompletten Logindaten werden dann an diese eMail geschickt.

Ist eine erfolgreiche Verbindung zustande gekommen, so werden die Buttons, gemäß den Abspeicherungen auf dem Server, beschriftet und die aktuellen Zustände der Komponononenten in die LED-Anzeigen übernommen.

Der ‘Reset’-Button läßt alle Komponenten in Grundstellung zurückfahren.

Das Display zeigt im normalen Betrieb die Zeit des Servers und die Temperatur im 2-Sekunden-Takt abwechslend an.

Außerdem werden folgende Meldungen über das Display angezeigt:

- "Welcome to your Room Control Center"
- "connecting..."
- "waiting..."
- "Unit reseted"
- "connection failed"
- "disconnected" & der Grund
- "Fire Alert!!!"
- vom Server gesendeter Text

7. Fehlerbeseitigung für Client und Server

Leider funktionieren nicht alle Geräte oder Programme auf Anhieb oder so, wie man sich es vorstellt. Es kann ein Fehler eines Bauteils in einer Baugruppe sein oder, wie es häufig mit unbekannten, neuen Geräten ist, ein einfacher Bedienungsfehler. Um den anfänglichen Startschwierigkeiten etwas zu entgehen, werden hier ein paar Tipps und Fehlerbeseitigungsvorschläge angeboten.

F: Das Modell funktioniert gar nicht. Ich habe alles gemäß den Installationsanleitungen aufgebaut und installiert, kann das Modell aber weder vom Client aus, noch über die Tasten am Modell selbst bedienen.

A: Hierfür kann es mehrere Gründe geben:

- Prüfen Sie als erstes, ob das Modell auch tatsächlich mit dem 230V-Netz verbunden ist. Falls das Modell an einer Steckdosenleiste betrieben wird, stecken Sie den Netzstecker des Modells direkt in eine Steckdose an der Wand.

- Überprüfen Sie die Glasrohrsicherung, die zum Schutz der Baugruppen eingebaut ist. Der Faden in der Glasrohrsicherung darf keine Unterbrechung aufweisen. Lassen Sie die Sicherung am besten von einem Fachmann überprüfen.
  VORSICHT! Ziehen Sie vor irgendwelchen Handlungen an der Netzteilplatine den Netzstecker. Ansonsten drohen Ihnen ein elektrischer Schock und eventuelle Folgen.

- Prüfen Sie die Leitungen der Motoren und der Controllereinheit, die mittels Schraubklemmen mit der Netzteilplatine verbunden sind. Ziehen Sie die Schrauben der Schraubklemmen mit einem kleinen Schlitzschraubendreher an, falls sich dort Schrauben gelockert haben sollten.
  VORSICHT! Ziehen Sie vor irgendwelchen Handlungen an der Netzteilplatine den Netzstecker. Ansonsten drohen Ihnen ein elektrischer Schock und eventuelle Folgen.

- Vergewissern Sie sich, daß das Verbindungskabel zwischen Tastenplatine und Controllereinheit richtig in den dafür vorgesehenen Sockeln sitzt.

- Falls sich der Betriebswahlschalter in der Stellung “Run“ befindet, schalten Sie ihn auf “on“.

F: Das Modell kann mit den Tasten bedient werden, nicht aber über den Client. Server- und Clientsoftware sind gemäß den Anleitungen installiert worden.

A: Überprüfen Sie folgende Punkte:

- Ist die Leitung für die serielle Datenübertragung auf beiden Seiten richtig eingesteckt?

- Leuchtet in dem Clientfenster die Bereitschafts-LED für den Server? Falls nicht, überprüfen Sie, ob sich der Server in Bereitschaft befindet.

- Leuchtet in dem Clientfenster die Bereitschafts-LED für die Unit? Falls nicht, überprüfen

Sie, ob die Bereitschafts-LED der Unit in dem Serverfenster grün leuchtet. Stellen Sie gegenbenfalls Sie eine Verbindung mit der Controllereinheit her.

F: Ich bekomme keine Verbindung zwischen Client und Server zustande. Das Display zeigt immer “connection failed“ an. Das Modell funktioniert ansonsten einwandfrei.

A: Überprüfen Sie folgende Punkte:

- Stellen Sie sicher, daß falls die Verbindung über das Internet erfolgen soll, sich beide Rechner - Server und Client - direkten Zugang zum Internet haben und eingewählt sind.

- Stellen Sie sicher, daß falls auf dem Server eine Firewall installiert ist, der Port 35400 durch die Firewall freigeschaltet ist.

- Die im Clientfenster eingegebene IP muß der IP des Servers entsprechen. Ersatzweise kann der DNS-Name eingegeben werden. Dies kann aber bei dynamisch vergebenen IPs zu Problemen führen.

F: Ich möchte mich mit meinem Benutzernamen und Passwort anmelden. Die Anmeldung schlägt aber immer mit der Fehlermeldung “Wrong username or password“ fehl.

A: Stellen Sie sicher, daß Sie ihren Benutzernamen und Passwort richtig eingegeben haben. Achten Sie besonders auf die Groß-/Kleinschreibung des Passworts.
Sollten Sie Ihre Logindaten vergessen haben, so tragen Sie bei “Username“ die eMail-Adresse, die Sie von Ihrem Administrator erhalten haben, ein und drücken Sie anschließend den Verbindungsbutton. Ihre Logindaten werden dann an diese eMail-Adresse gesendet.
Sollte auch das nicht helfen, kontaktieren Sie Ihren Administrator.

F: Ich möchte eine Verbindung von Server zur Controllereinheit herstellen. Es erscheint aber jedesmal nur die Fehlermeldung “Unit not available“.

A: Stellen Sie sicher, daß das Modell am 230V-Netz angeschlossen und über die Tasten bedienbar ist. Sollten Sie nicht bedienbar sein, besteht entweder ein Feueralarm oder der Temperatursensor ist defekt.
Ist das Modell aber über die Tasten bedienbar, dann ist möglicherweise nur das Datenübertragungskabel (3,5mm Klinke/9 pol. Sub-D) nicht richtig eingesteckt. Kontrollieren Sie an Ihrem Rechner und an der Controllereinheit den Sitz der Stecker.

F: Jedesmal, wenn ich auf den “Send Mail“-Button klicke und sich das Fortschrittsfenster öffnet, erscheint nach ein paar Sekunden nur ein “SMTP - Time out“-Fehler.

A: Überprüfen Sie die SMTP-Servereinstellung im Maileinstellungsfenster. Ändern Sie dort gegebenenfalls die Angaben.

F: Jedesmal, wenn ich den Server starte und auf “Send Mail“ klicken möchte, ist dieser Menüpunkt verschwunden.

A: Wenn Sie nicht direkt mit dem Internet verbunden sind, d.h. wenn keine direkte Verbindung zwischen Ihrem Rechner und dem Provider besteht, dann ist die Option, die dynamische IP an alle registrierten Benutzer zu schicken, deaktiviert.
Schließen Sie in diesem Fall den Server, stellen Sie eine Verbindung mit dem Internet her und starten Sie den Server erneut.

F: Woher weiß ich, welcher Button welche Funktion hat? Es ist kein Button beschriftet.

A: Das stimmt. Es gibt aber eine Möglichkeit, jede Funktion eines Buttons herauszufinden. Bewegen Sie Ihren Mauszeiger auf einen der Buttons und warten Sie eine kurze Zeit. Es erscheint ein kleines, gelbes Fenster in der Nähe des Mauszeigers, welches die Funktion des Buttons beschreibt.

F: Nachdem ich beim Anlegen eines Benutzers nur den Namen angegeben hatte, klickte ich aus Versehen auf den “Exit“-Button. Was soll ich jetzt machen?

A: Öffnen Sie das “User“-Fenster erneut. Jetzt können Sie entweder die Daten des Benutzers vervollständigen, indem Sie diesen Benutzer markieren und auf den “Edit“-Button klicken, oder Sie löschen diesen Benutzer

F: Ich habe aus Versehen auf den Punkt “User...“ im Menü “Settings“ geklickt. Nun werde ich nach einem Administratornamen und -passwort gefragt. Wie bekomme ich das Fenster wieder weg?

A: Entweder Sie klicken auf das kleine “X“ rechts oben in der Fensterecke oder Sie löschen bereits eingegeben Text aus den Feldern und drücken den “Exit“-Button.

Beschreibungen
der Soft- und Hardware

8. Hardware

8.1 Der Controller

Das Herzstück der Raumkomponentensteuerung über TCP/IP ist ein AT89S8252 von ATMEL. Dieser Controller ist in einem 44-Pin PLCC-Gehäuse untergebracht, kann aber auch in einem 40-Pin PDIP-Gehäuse oder 44-Pin PQFP/TQFP-Gehäuse bezogen werden. Er enthält den Kern eines 8051 von Intel und kann auch mit dessen Befehlssatz programmiert werden.
Auswahlkriterien für diesen Controller waren eine integrierte serielle Schnittstelle, ein integriertes EEPROM und die ISP-Fähigkeit (In-System-Programming).

Ein 12MHz-Quarz liefert den Takt, wodurch sich ein Maschinenzyklus von 1 µs ergibt. Der Takt für die serielle Schnittstelle kann in diesem speziellen Fall nicht Timer 1 übernehmen, sondern muß von Timer 2 gewonnen werden. Der Timer 1 ist normalerweise für die Baudraten-Generierung zuständig, da er leichter zu intialisieren ist. Weil aber ein 12MHz Quarz in diesem System arbeitet, ist es nicht möglich mit den einstellbaren Teilerverhältnissen des Timer 1 auf eine Baudrate von 9600 Bits/Sekunde zu kommen.

Mit der Formel

$$\text{Baudrate} = \frac{\text{Oszillatorfrequenz}}{32 \cdot (65536 - \text{Nachladewerte})} = \frac{12 \cdot 10^6 \text{Hz}}{32 \cdot (65536 - 65497)} = 9615 \text{ Bits/Sekunde}$$

kann ein Baudratenwert errechnet werden. Da man aber meistens schon die Baudrate hat, möchte man nun wissen, mit welchem Nachladewert diese zu erreichen ist. Für diesen Zweck muß man einfach nur die Formel umstellen. Die Nachladerwerte sind die Werte, die in die Register geschrieben wurden, die nach jedem Timerüberlauf als aktueller Zählwert in den Timer selbst kopiert werden. Der Divisor „32“ entsteht durch den Modus, in dem der Timer betrieben wird.

Da die Nachladewerte nur aus natürlichen Zahlen bestehen können, ist es nicht möglich auf genau 9600 Baud zu kommen. 9615 Baud ist der nächstmögliche Wert, der mit dieser Timerschaltung realisiert werden kann. Für die „einfache“ Übertragung zum PC reicht dies aber vollkommen aus.

Timer 2 ist somit ausschließlich für die Baudratengenerierung zuständig. Um aber für andere Funktionen feste Zeiten bereitzustellen, werden noch zusätzlich Timer1 und Timer 0 in Anspruch genommen.
Timer 0 ist für den A/D-Wandler, und Timer 1 für die Zeitüberwachung der beiden Motoren verantwortlich.
Beide Timer werden im Modus 1 betrieben, als 16-Bit Timer ohne Vorteilung und Nachladewerte.

Wenn der Server angemeldet ist, schickt er in 2 Sekunden-Schritten den Befehl zur Temperaturanforderung. Mit dem Empfang dieses Befehls wird Timer 0 und die A/D-Wandlung gestartet. Nachdem der Zähler „übergelaufen“ ist, nach ca. 65ms, wird der gewandelte analoge Wert eingelesen und an den Server übertragen.

Ist der Server nicht angemeldet, wird die A/D-Wandlung mit jedem erneuten Durchlauf des Hauptprogramms gestartet.
Die Wandlungszeit von mehr als 65 ms ist deshalb so groß gewählt, daß der Controller nicht mit “unwichtigen“ Aufgaben, wie Temperaturwerte einlesen und wieder ausgeben, beschäftigt ist.

Die Zeitüberwachung der Motoren, und somit der Timer 1, wird mit Drücken einer der Tasten für die Motoren oder per Befehlsempfang über die serielle Schnittstelle aktiviert. Da nun ein Zählerüberlauf nur ca. 65ms (exakt: 65536 µs) dauert, muß über zusätzliche Register ein Zähler eingerichtet werden, um eine Überwachung über mehrere Sekunden zu gewährleisten.
Man kann von dieser Zeitüberwachung sagen, daß sie die Funktion eines Watchdogtimers hat. Jedoch mit dem Unterschied, daß der Überlauf des Watchdogtimer einen Reset des Controllers bewirkt. Wo hingegen bei der Motorenüberwachung lediglich die Motoren angehalten werden und der Controller dann auf keine Eingabe mehr reagiert.
Für die Überwachung des Rollomotors sind zwei 8-Bit Register reserviert, die bis zu einem festen, vom Endanwender unbeeinflussbaren Wert hochzählen. Mit diesen beiden Registern ist es möglich bis zu ca. 71 Minuten diesen Motor zu überwachen, bevor er abgestellt wird. Mit einem 8-Bit Register ist es möglich ca. 16.7 Sekunden zu überwachen. Für den Fenstermotor wird nur ein solches Register verwendet.
Wurde ein Motor durch die Zeitüberwachung abgestellt, nicht über die Endtaster, dann kann die Controllereinheit nur noch über den Resettaster neu gestartet werde.

Es werden alle Portpins des Controller verwendet. Hier nun ein paar Erläuterungen zu den Funktionen der Pins und weitere nötige Maßnahmen:

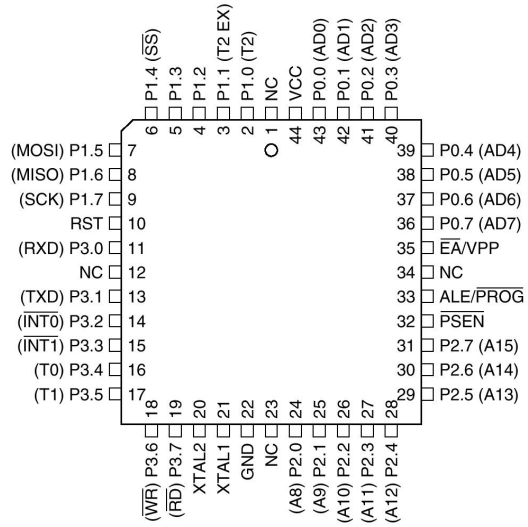
- Port 0 stellt den Eingabeport der Tasten dar. Da dieser Port nur dann als Eingabeport funktionieren kann, wenn die Pins alle High-Pegel führen, ist dieser Port Low-aktiv. An diesem Port ist desweiteren unbedingt nötig, Pull-Up Widerstände davor zu schalt

- Port 1 empfängt zum einen das 8-Bit Wort vom A/D-Wandler kommend, zum anderen werden die letzten drei Pins (Pin 5 - 7) zum Downloaden des Programms in das Flash- oder EEPROM geladen. Bei dem Anschluß des A/D-Wandlers gibt es nicht viel zu beachten, weil der A/D-Wandler direkt für den Betrieb an µController konzipiert wurde. Auch die Tatsache, daß der Controller über die parallele Schnittstelle (LPT) des PCs programmierbar ist, welche auf TTL-Basis arbeitet, bedeutet, daß keine weiteren Maßnahmen nötig sind. Jedoch sollte die hohe Empfindlichkeit der paralellen Schnittstelle und die Tatsache, daß es zu möglichen Schäden der LPT kommen kann, erwähnt werden. Es wird daher dringend empfohlen, nach dem Programmieren des EEPROMs den Programmierstecker zu ziehen, bevor der Schalter wird auf die Stellung “Run” geschaltet wird. Vermeiden Sie es ebenfalls die Pins des Programmiersteckers mit den Fingern zu berühren, und den Kontakt mit anderen elektrischen Geräten.

- Port 2 dient nur als Ausgang für die 8 Relais. An ihm ist nachfolgend ein Treiberbaustein (U4) geschaltet, der wegen den höheren Schaltströmen der Relais benötigt wird. Die Invertierung der Zustände, die der Treiberbaustein vornimmt, benötigt einen weiteren Inverter-Baustein (U5) direkt am Port 2.

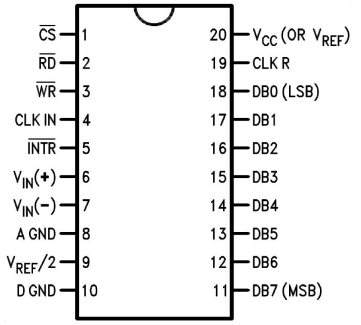
- Die Pins des Port 3 werden zwar ebenfalls komplett benutzt, allerdings einzeln für verschiedene Funktionen. Pin 0 und Pin 1 bilden zusammen den Aus-/Eingang der seriellen Schnittstelle. An den Pins 2 bis 5 sind die Endtaster angeschlossen. Für den Beginn der

A/D-Wandlung und der anschliessenden Ausgabe der Zustände auf den Port 1 sind die Steuerbits Pin 6 und 7 zuständig.



Das Pinlayout läßt vermuten, wie kompakt die gesamte Controllereinheit aufgebaut werden könnte, da solche Zusatzbausteine, wie Speicher (EEPROM und RAM) und erforderliche GAL-ICs, entfallen. Tatsächlich wurde die Kompaktheit nicht ausgenutzt um die Übersichtlichkeit für Präsentationszwecke zu wahren.

8.2 Der A/D-Wandler



Der A/D-Wandler ADC804 von National Semiconductor ist ein 8-Bit Wandler, der eine analoge Spannung in 100 µSekunden umwandelt. D.h. die höchste in der analogen Spannung vorkommende Frequenz darf nach dem Shannon'schen Abtasttheorem mit der Formel

$$f_0 = \frac{1}{\frac{\text{Umwandlungszeit}}{2}} = \frac{2}{100 \cdot 10^{-6}\,\text{s}} = 2 \cdot 10^4 \frac{1}{\text{s}} = 20\,\text{kHz}$$

maximal 20kHz betragen. Da ansonsten mögliche Informationen, die im Spektrum über der 20kHz-Grenze liegen, einfach eliminiert werden.

Die Frequenz des Taktes, mit der der Wandler betrieben wird, wird mit einem Widerstand-Kondensator-Glied festgelegt, das extern an den A/D-Wandler angeschaltet wird. Mit dem im Wandler selbst befindlichen Schmitt-Trigger wird nun ein Rechtecksignal mit der Frequenz

$$f_T = \frac{1}{1.1 \cdot \text{Widerstand} \cdot \text{Kondensator}} = \frac{1}{1.1 \cdot 30\,\text{k}\Omega \cdot 150\,\text{pF}} = 202\,\text{kHz}$$

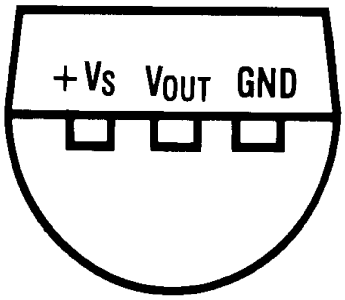
von ca. 202kHz generiert. Der Bereich der Frequenz, in der der A/D-Wandler betrieben werden kann, beträgt 100 bis 1406 kHz. Die Taktfrequenz, bei der Wandlungen mit den geringsten Fehlern stattfinden liegt bei 640kHz.

Um eine Quantisierung in 10 mV-Schritten zu ermöglichen muß ein Referenzspannung von

$$U_{Ref} = \text{Quantisierungsschritt} \cdot \text{Stufenanzahl} = 10\text{mV} \cdot 256 = 2.56\text{V}$$

2.56V verwendet werden. Da der A/D-Wandler die Spannung, die an Pin 9 angelegt wird, noch um den Faktor zwei verstärkt, muß sie um die Hälfte reduziert werden. Somit werden dem A/D-Wandler 1.28V am Pin9 ($V_{Ref/2}$) zugeführt. In diesem speziellen Fall wird die Referenzspannung über ein 10kOhm Potentiometer eingestellt, da mit Hinsicht auf die Genauigkeit "nur" die Temperatur im Client angezeigt werden soll. Da wirken sich also die Toleranzen des Potentiometers nicht besonders stark aus.

8.3 Der Temperatursensor

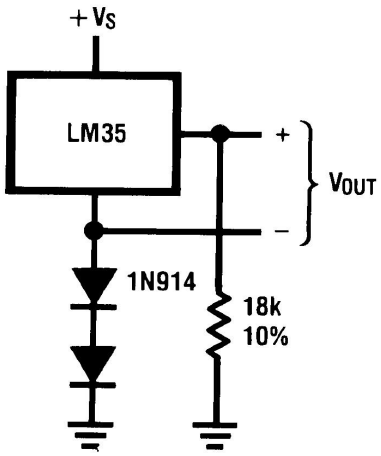


Da für den speziellen Zweck, die Temperatur zu messen und auszuwerten ein temperaturabhängiger Widerstand zu ungenau bzw. nichtlinear ist, traf die Wahl auf den LM35. Dieses IC ermöglicht es einfach die Spannung, die es ausgibt direkt auszuwerten (die richtig dimensionierte Randbeschaltung vorrausgesetzt!), da sich diese Spannung fast hundertprozentig linear zur Temperaturänderung verhält. Aufgrund dieser hohen Genauigkeit kann man die Formel

$$\Delta V_{out} = 10\,mV \cdot \Delta\,°C$$

zur Berechnung der zu erwartenden Spannung verwenden.

Da es aber nun aus technischen Gründen und ohne weitere Randbeschaltung nicht möglich ist niedrigere Temperaturen als 2 °C (entspricht 20mV, wenn Masseanschluß des LM35 mit der Masse der restlichen Schaltung verbunden ist) anzuzeigen, sind wenige Änderungen nötig. Hierzu hebt man den Masseanschluss des ICs auf ein etwas höheres Potential und speist die nun so “neu” enstandene Masse direkt an den $V_{in(-)}$- Pin des A/D-Wandlers, der den Bezug zu dem zu wandelnden Analogwert darstellt.
Außerdem muß der Ausgang des Temperatursensors über einen 18kOhm Widerstand auf Masse gelegt werden.

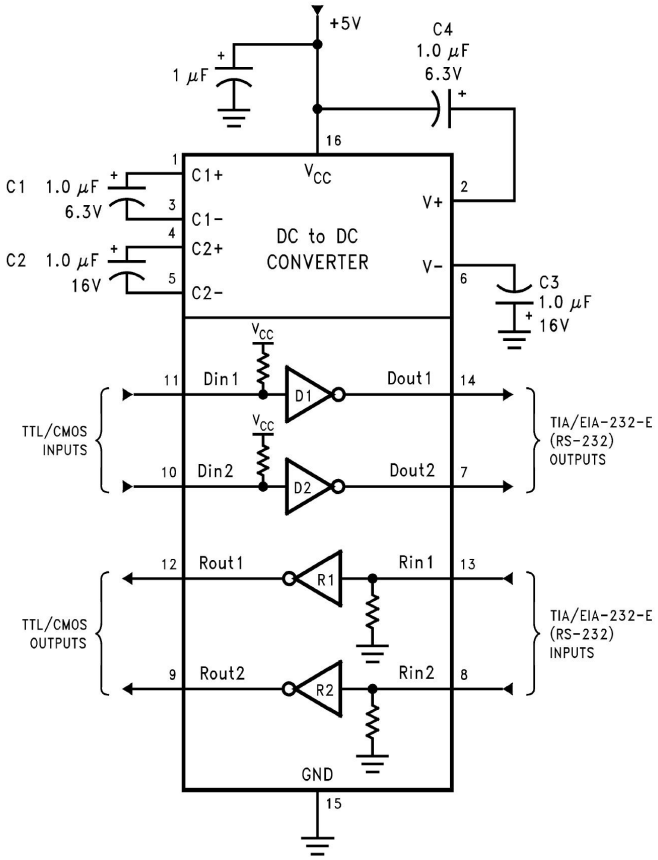


Man hebt den Fußpunkt des Sensors an, indem man zwei Dioden zwischen dem GND-Anschluß und der Masse schaltet. Die Stabilität der Dioden ist hierbei nicht von wesentlichem Wert, solange der Diodenstrom nicht unter ca. 10mA fällt. Die Stabilität ist deshalb nicht so wichtig, da die Spannung nicht über den Dioden abgenommen und gewandelt wird, sondern die Spannung über V_{OUT} und GND des Sensors. Ändern sich nun die Spannungen der Dioden, so wird nur die Spannung über V_{OUT} und GND angehoben bzw. abgesenkt.
Der 18kOhm Widerstand ermöglicht es jetzt, eine Temperatur kleiner als 2 °C zu messen, da nun die Ausgangspannung gegenüber dem Fußpunkt des Sensors tiefer absinken kann.

8.4 Die serielle Schnittstelle

Die einfachste Art eine Kommunikationsmöglichkeit herzustellen, ist eine serielle Schnittstelle zu verwenden. Das Problem dabei ist, daß die serielle Schnittstelle des Controllers und die des PCs unterschiedliche Pegel führen.
Die Pegel des Controllers betragen 0V und 5V, die Pegel des PCs -12V und +12V. Zudem handelt es sich bei den Datenleitungen bei der Schnittstelle des PC um negative Logik. Um nun eine einfache Pegelwandlung zu gewährleisten, setzt man hier das IC MAX232 der Firma MAXIM ein.

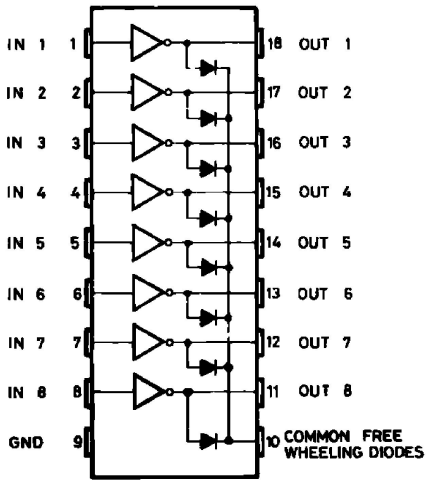
Dieses IC wird mit +5V betrieben und enthält jeweils zwei Sende- und Empfangsleitungen. Um nun die Spannungen für die V24-Schnittstelle zu erhalten, werden vier Elektrolytkondensatoren eingesetzt, die als Ladungspumpen dienen.



Anders als bei einer seriellen Verbindung zwischen einem Modem und einem PC läuft die Datenübertragung zwischen Controllereinheit und PC vollkommen ereignislos ab. D.h. daß die Daten ohne zusätzliche Steuerbits gesendet und empfangen werden. Falls also die Controllereinheit vom PC ein Datenwort empfängt unterbricht der Controller **nicht** sein aktuell laufendes Programm, um das Datenwort auszuwerten, sondern fährt fort bis er eine Routine abarbeitet, die den Empfangspuffer auf ein neu eingegangenes Datenwort überprüft.

8.5 Der invertierende Ausgangstreiber

Da induktive Lasten - in diesem Projekt sind es Relais - nicht direkt an einem Mirco-Controller betrieben werden sollten, werden verschiedene Typen von Treiber-ICs angeboten, die den nötigen Strom für die Lasten liefern. In diesem Fall wurde ein ULN2803A eingesetzt.

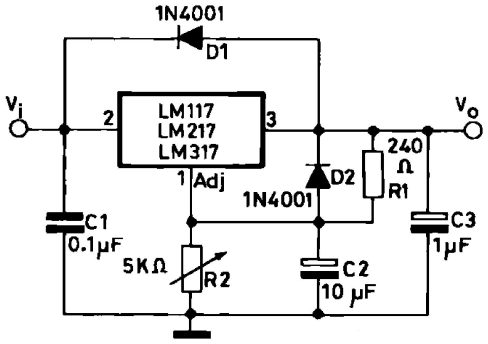


Jeder der acht Ausgänge kann einen konstanten Strom von 500mA liefern, spitzenmäßig sogar 600mA. Die Ausgänge haben jeweils eine Diode, um speziell für induktive Lasten, die beim Abschalten eine hohe Induktionsspannung erzeugen, den dadurch entstehenden Strom kurzschließen (Freilaufdiode).
Zusätzlich bietet dieses IC die Möglichkeit mehrere Ausgänge zu einem gemeinsamen zu schalten, um eine höhere Schaltleistung zu erreichen.

8.6 Die Spannungsversorgung

Das Modell wird an das 230V/50Hz angeschlossen. Diese Spannung wird über einen Transformator auf 24V effektive Wechselspannung runter transformiert und anschliessend von 2 parallel geschalteten Brückengleichrichtern in DIP-Gehäuseform gleichgerichtet.

Die Spannung für die Motoren und die Lämpchen wird von einem Gleichrichter geliefert und wird danach von drei steuerbaren Spannungsreglern (LM317) stabilisert. Es sind jeweils für die beiden Motoren und eine Lämpchenbatterie ein Spannungsregler vorhanden, da die Motoren und Lämpchen unterschiedliche Spannungen benötigen.

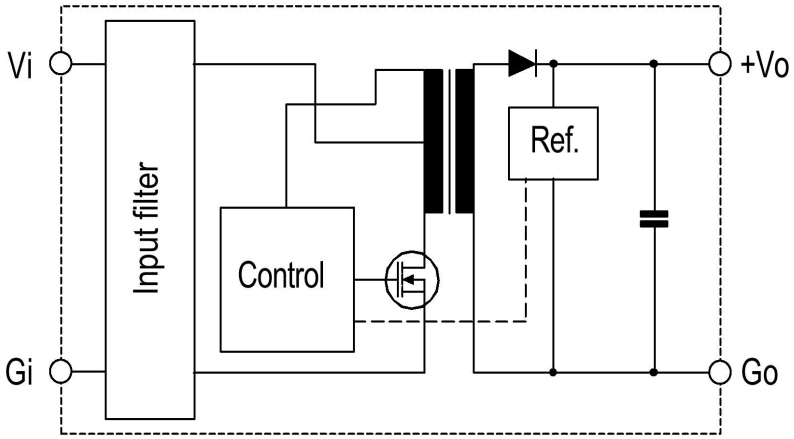


Während die Dioden den Regler vor Verpolung schützen, dienen die Kontensatoren der Schwingneigungsunterdrückung. An dem Potentiometer wird die gewünschte Spannung am Ausgang eingestellt.

Die Ausgangsspannung des Reglers für den Motor beträgt 8,65V, die des Fensters ca 1,2V (die kleinstmögliche Einstellung, da das Getriebe des Motors eine zu hohe Übersetzung besitzt).
Für die Lämpchen sollte man eine der Leistung der Lämpchen entsprechende Spannung einstellen. Es ist unbedingt darauf zu achten, daß die Spannungen immer die oben angegebenen Werte haben, da die Dauer des Öffnens und des Schließens von Rollo und Fenster vom Controller überwacht werden.
Werden diese Spannungen unterschritten, stoppen die Sicherheitsroutinen immer frühzeitig die Motoren. Werden hingegen die Spannungen überschritten entsteht ein zu großer Zeitraum zwischen erwarteter Dauer und eingestellter Sicherheitsdauer. Die Zeitüberwachung wäre somit überflüssig.

Die Spannung für die Controllereinheit selbst wird von dem zweiten Gleichrichter bezogen. Der dem Gleichrichter nachgeschaltete integrierte DC/DC-Wandler schaltet die 24V vom Transformator kommend auf 5V.

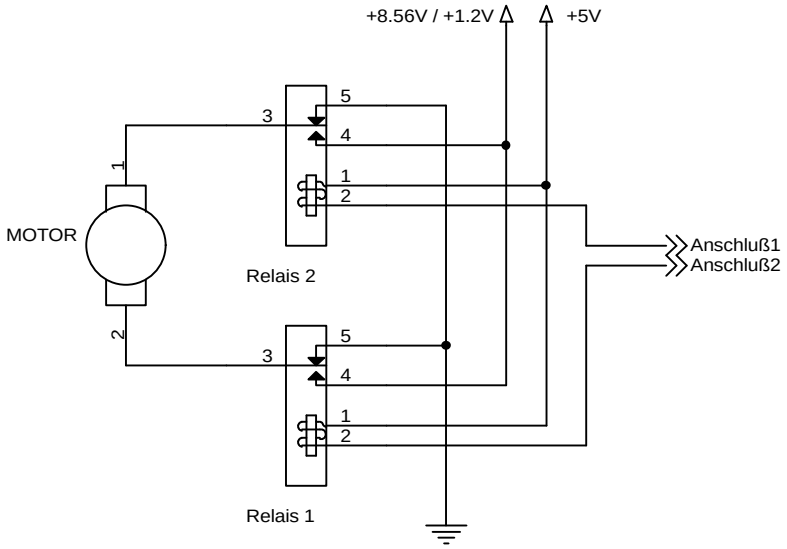
Da die Leistung der gesamten Controllereinheit (Controller, Relais, LEDs, A/D-Wandler und MAX232) und der Spannungsunterschied nur von einem stark gekühlten Festspannungsregler zu bewältigen wäre, fiel die Wahl gleich auf einen DC/DC-Wandler. Dieser schafft die Leistung und den Spannungsunterschied mit nur leichter Erwärmung. Eine schnelle Alterung eines Festspannungsreglers ist somit ausgeschlossen.



Ein Nachteil des DC/DC-Wandlers entsteht durch die Methode der “Wandlung”. Da in diesem integrierten Baustein ein Schaltnetzteil enthalten ist, d.h. die Ausgangsspannung wird über eine Puls-Weiten-Modulation (PWM) der vorne angelegten Gleichspannung erzeugt, entstehen Störungen, die für vor- und nachstehende Schaltungen wieder eliminiert werden müssen. Weil nach den Gesetzen von Fourier die Störungen zum Teil größere Bereiche des Spektrums abdecken, ist auf eine sorgfältige Eliminierung (sprich: Dimensionierung der Filter) zu achten.

8..8 Die Ansteuerung der Motoren

Die Ansteuerung der Motoren erfolgt über jeweils zwei Relais um drei mögliche Zustände der Motoren zu erreichen (Motor links, Motor rechts und Motor stopp).



Wenn beide Anschlüsse nun High-Pegel (+5V) führen, so befindet sich der Motor im Ruhezustand. Der gleiche Zustand kann durch anlegen von Low-Pegeln erreicht werden. Allerdings führen dann beide Anschlussklemmen des Motors Spannung, was bei einem Ruhezustand vermieden werden sollte.
Führt Anschluß 1 nun High- und Anschluß 2 einen Low-Pegel, dann dreht der Motor rechts rum. Liegen die Pegel genau verkehrt herum an, dann dreht der Motor nach links.

8.9 Die Taster und Schalter

Die gelben Taster steuern die vier Lichter und die zwei Motoren. Im folgenden werden die einzelnen Funktionen der Taster auf dem Tastenboard von links nach rechts aufgelistet:

- Licht 1 (an/aus)
- Licht 2 (an/aus)
- Licht 3 (an/aus)
- Licht 4 (an/aus)
- Rollo hoch
- Rollo runter
- Fenster auf
- Fenster zu

Die Tasten für die Lichter können jederzeit gedrückt werden, nicht jedoch die Tasten für das Fenster und das Rollo. Da die Öffnungs- und Schließungsvorgänge einige Sekunden Zeit in Anspruch nehmen, werden die Tasten nur dann mit ihrer Funktion belegt, wenn sich das Fenster oder Rollo in einer Ruheposition befindet. Desweiteren ist es aufgrund der internen Tastenverriegelung nicht möglich zwei Komponenten gleichzeitig zu steuern. Es wurde ebenfalls eine Tastenentprellung eingebaut, die verhindert, daß die Lichter nur kurz aufflackern oder nicht ausgehen.

Der Betriebswahlschalter ist an der Controllereinheit selbst angebracht . Über ihn lassen sich die Betriebsarten “Programmierung”, “Aktiv” und “Reset” anwählen, wobei “Programmierung nur von einem Service-Techniker bedient werden darf. Die Stellung “Reset” ermöglicht es im Falle eines falsch ausgelösten Feueralarms, oder Time-Out Überwachung der Motoren die gesamte Controllereinheit in Anfangszustand zu versetzen.
Die Stellung “Aktiv” ist für den normalen Gebrauch anzuwählen.

Mit dem Taster “Feueralarm-Test” kann im normalen Betrieb ein Feuer simuliert werden. Im Falle eines Feueralarms (die Controllereinheit löst einen bei über 75°C aus, oder über den Feueralarm-Test) werden alle Lichter abgeschaltet, das Rollo hochgefahren und das Fenster geschlossen. Danach können die Komponenten nicht mehr über die Tasten oder die Client-Software bis zu einem “Reset” bedient werden.

Im Programm des Controllers ist eine Art Kalibriermodus eingerichtet, der es ermöglicht das Rollo oder Fenster in jede gewünschte Position zu bringen. Dieser Modus ist allerdings nicht für den normalen Betrieb konzipiert, sondern nur für die Einrichtung der Endtaster gedacht.
Um in den Kalibriermodus zu gelangen, muß die Resettaste gedrückt werden, um die Controllereinheit zurückzusetzen, und gedrückt bleiben. Nun müssen die Tasten “Fenster zu” und “Rollo runter” betätigt werden. Nun läßt man nacheinander die Resettaste und um eine Sekunde verzögert die beiden anderen Tasten los. Jetzt befindet sich der Controller im Kalibriermodus. Die Controllereinheit kann mit den Tasten für die Motoren wie gewohnt bedient werden.
Vorsicht: Im Kalibriermodus haben die Endtaster und Zeitüberwachung keine Funktion. Es muß also darauf geachtet werden, daß sich die Rolloantriebswelle nicht überdreht und der Motor für das Fenster nicht in die Richtung gesteuert wird, in der sich das Fenster bereits in der Endposition befindet.

9. Software

9.1 Die Software des Servers

Die Software des Servers wurde, wie die des Clients, komplett in Visual Basic geschrieben. Nach dem Start der Serversoftware wird das Hauptfenster dargestellt, das die Betriebszustände des Client und der Controllereinheit anzeigt. Von hier aus können die Einstellungfenster für Benutzer-, Mail-, COM-Port- und Raumeinstellungen erreicht werden.

9.1.1 Das Benutzerfenster

Das Benutzerfenster enthält links ein Listenfeld, in dem die registrierten Benutzer aufgelistet werden. Auf der rechten Seite befinden sich diverse Werkzeuge zum Administrieren der Benutzer. Das leere Feld in der Mitte enthält momentan noch unsichtbare Textboxen und Buttons, die je nach gewähltem Werkzeug erscheinen und verschwinden.
Die Benutzer sind in der Registrierung von Windows unter dem Schlüssel „HKEY_USERS\.DEFAULT\Software\VB and VBA Program Settings\Server“ abgelegt. Ebenso, wie die Logindaten des Administrators selbst. Jeder Benutzer ist dort mit Namen, Passwort und eMail-Adresse abgelegt, wobei der Name und die eMail-Adresse unverschlüsselt hinterlegt werden. Der Administrator besitzt hier nur Loginnamen und -passwort.

Die Passwörter werden dort nur in einer doppelt verschlüsselten Form abgespeichert, um den Missbrauch und den Zutritt unbefugter zu verhindern.
Die Passwortverschlüsselung basiert auf der XOR-Methode, bei der zum Ver- und Entschlüsseln ein Schlüssel benötigt wird.
Wird ein Passwort verschlüsselt, so wird jeder ASCII-Binärwert eines Zeichens mit jedem ASCII-Binärwert eines Zeichens des Schlüssels an derselben Position mit XOR verknüpft. Nun ist es bereits unmöglich mit normalen Mitteln und ohne Schlüssel das Passwort herauszufinden. Um aber noch die Verschlüsselungsmethode selbst zu verschleiern -bei der XOR-Methode entstehen neue Zeichen mit einer wirren und unlogischen Zusammensetzung- wird nun jedes durch die Verschlüsselung neu entstandene Zeichen in seinen ASCII-Hexwert umgewandelt und abgespeichert.

Ein Beispiel:

Das Passwort lautet: Paß. Und der Schlüssel lautet: Key.
Die Binärwerte der Zeichen in ‘Paß’ lauten:

```
P=0101 0000; a=0110 0010; ß=1110 0001
```

Die Binärwerte der Zeichen in ‘Key’ lauten:

```
K=0100 1011; e=0110 0101; y=0111 1001
```

Somit ergeben sich aus der XOR-Verknüpfung diese neuen Zeichen:

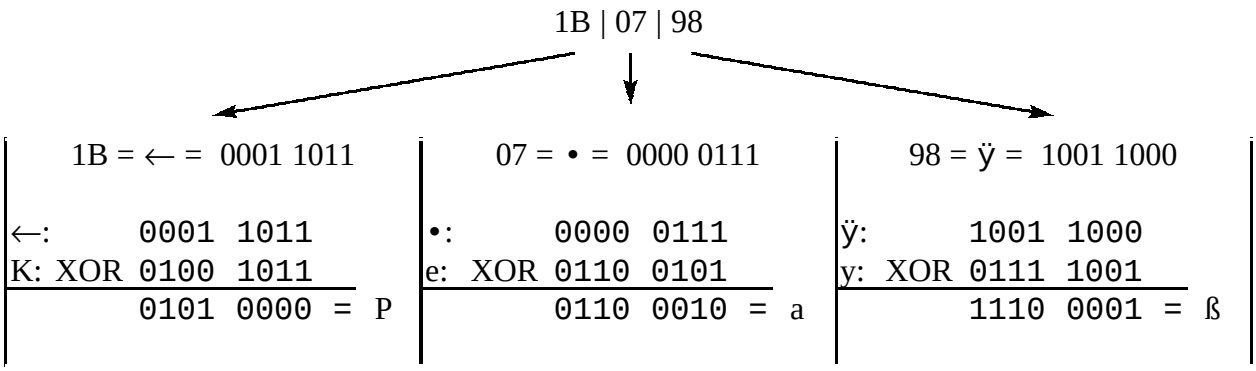
```
P:        0101 0000
K:  XOR   0100 1011
          0001 1011 = ←
```

```
a:        0110 0010
e:  XOR   0110 0101
          0000 0111 = •
```

```
ß:        1110 0001
y:  XOR   0111 1001
          1001 1000 = ÿ
```

Das verschlüsselte Passwort heißt also: ←•ÿ. Diese wirre Zeichen werden nun ihrerseits mit ihrem ASCII-Hexwert in der Registry abgespeichert. Dort steht nun die Zeichenkette: 1B0798.

Bei der Entschlüsselung läuft dieser Algorithmus in anderer Richtung: Zuerst werden die Hexwerte zweistellig in die dazugehörenden ASCII-Zeichen umgewandelt. Und die daraus resultierenden in Binärform wird wiederum mit dem Schlüssel XOR veknüpft.



Die Benutzerkonten werden nicht erst bei Anlegen eines Benutzers erstellt, sondern beim ersten Start des Serverprogramms. Wenn ein Konto keinen Benutzer enthält, so steht bei Namen, Passwort und eMail “leer” in den Feldern.

Die Visual-Basic Befehle SaveSetting und GetSetting ermöglichen einen einfachen Umgang mit der Windowsregistrierung ohne viel Basiswissen über die Registrierung zu benötigen. Mit diesen beiden Befehlen ist es aber nur möglich in einem speziell dafür reservierten Bereich in der Registrierung zu arbeiten.

9.1.2 Die Fenster der Mail-, COM-, Porteinstellung und des Clients

Diese vier Fenster verwenden als Speicherort die Datei “Config.ini” im Installationspfad des Server- bzw des Clientprogramms. Dabei greift das Raumeinstellungsfenster nur auf den Absatz “[RoomConfiguration]”, das Maileinstellungsfentser auf “[SendMail]” und das COM-Porteinstellungs-, sowie Clientfenster auf “[Config]” zu.

Der Zugriff läuft über extern programmierte Routine, die aber auch nur die Unterroutinen der `Kernel32.dll` verwendet.

Auf diese vier Fenster wird hier nicht weiter eingegangen, da in den Kapiteln 2 bis 7 die Funktionen bereits beschrieben wurden.

9.2 Die Ermittlung der dynamischen IP

Auf die genaue Vorgehensweise zur Ermittlung der dynamischen IP kann hier ebenfalls nicht weiter eingegangen werden, da die Funktionen dieses Moduls von Dritten stammen.
Allerdings kann gesagt werden, daß die Ermittlung der dynamischen IP problemlematisch ist.
Sollte sich der Rechner (Server) in einem lokalen Netzwerk (LAN) befinden, welches aber einen Router zu einem ISP besitzt, so ist es nicht möglich die dynamische IP des Routers mittels dieses Moduls zu erhalten.

Eine Möglichkeit dieses Problem zu beheben ist, das Port-Forwarding des Routers zu aktivieren und den Port 35400 zu der LAN-IP durchzuschleifen, die den Server repräsentiert.
Nun gilt es noch die dynamische IP des Servers herauszufinden. Dafür gibt es einige kostenlose Anbieter, die es ermöglichen nach Einwahl des Servers dessen IP über DNS auflösen. Dazu müßte der Domänennamen erst bei dem Anbieter eingetragen sein. Somit braucht man sich nur einen Domänennamen zu merken, wie z.B. www.homecontrolserver.dns2go

Unter http://www.deerfield.com/support/dns2go/ kann man sich registrieren und danach das Programm DNS2go runterladen.

Soll nur innerhalb des LANs eine Verbindung zwischen Client und Server hergestellt werden, so entfällt der Aspekt einer dynamischen IP. In kleineren Netzwerken werden normalerweise statische IPs vergeben, welche dann im Client-Fenster unter “Server” eingetragen werden. Ersatzweise kann aber auch der Computername des Servers eingetragen werden, sofern der Client entweder in der Datei `lmhost` unter Windows 2000 im Verzeichnis `C:\WINNT\system32\drivers\etc` oder in der Datei `Hosts` unter Win9x/ME im Verzeichnis `C:\Windows\` der Computername des Servers zu seiner IP zugeordnet ist.

Falls dies nicht der Fall ist, sollte dieser eingetragen werden. So könnte z.B. der Eintrag aussehen:

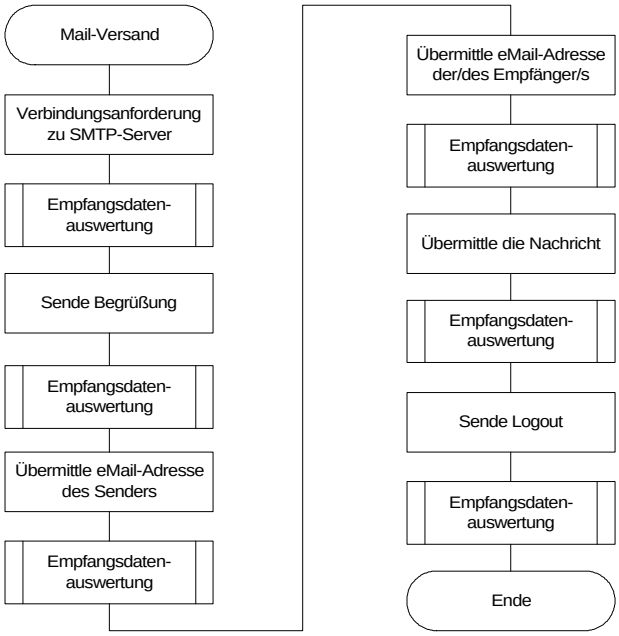
```
192.168.0.15   HomeControlServer
```

9.3 Die SendMail-Funktion

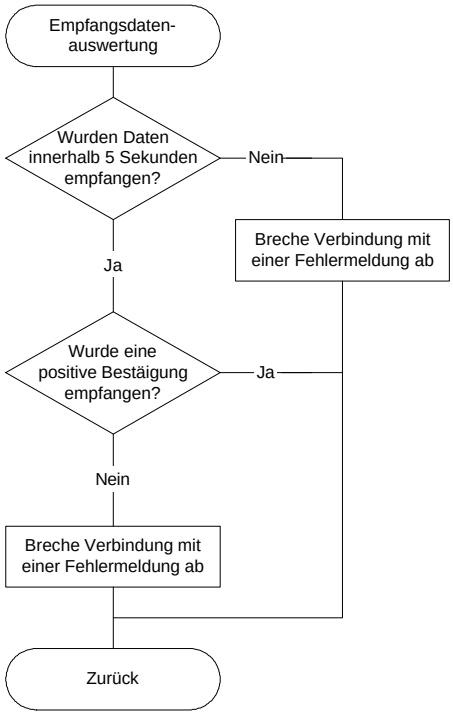
Es gibt bei dem Mailversand zwei sich voneinander unterscheidenden Varianten: eine Mail wird an alle registrierten Benutzer mit der dynamische IP verschickt und die Versendung von Logindaten (Name und Passwort) eines Benutzers.

Beide Varianten haben diesselbe Funktionsreihenfolge was den Ablauf von Einwahl beim SMTP-Server, Handshake, Übermittlung der Daten und Logout betrifft. Sie unterscheiden sich nur im Inhalt bei der Übermittlung der Daten.

Die Reihenfolge eines solchen Ablaufs sieht folgendermaßen aus:



Die Unterroutine ‘Empfangsdatenauswertung’ hat folgende Struktur:



Jeder der Punkte des Ablaufs “Mail-Versand” wird protokolliert und dem Benutzer im Mail-Versandfenster angezeigt, sofern diese Funktion aktiviert ist.

Neben den Befehlen, die an den entfernten SMTP-Server geschickt werden, werden auch Quittungen entgegengenommen. Diese Quittungen bestehen immer aus einer dreistelligen Zahl und geben schon mit der dritten Stelle (Hunderterstelle) an, ob die Aktion erfolgreich war oder nicht.

Kategorien

- 1xx vorläufiges positives Feedback in einer vorbereitenden Phase der Operation
- 2xx positives Feedback
- 3xx vorläufiges positives Feedback
- 4xx vorübergehende Ablehnung eines Kommandos
- 5xx negatives Feedback, permanenter Fehlercode

Unterkaterogien

- x0x Syntaxfehler: Wird bei einem fehlerhaften Kommando eingesetzt, das nicht definiert ist. Da es sich um einen Hinweis auf einen echten Fehler handelt, kann diese Kategorie lediglich in der Klasse 5 (50x) vorkommen

- x1x Information: Antwortcodes dieser Art werden gesendet, um weitere Informationen anzufordern

- x2x Verbindungsbezogenes Feedback: Antwortcodes dieser Art werden vorzugsweise in der Klasse 2 (positives Feedback) und in der Klasse 4 (temporäre Fehler) verwendet, um Statusmeldungen zur aktuellen Session zu übermitteln

- x3x undefiniert
- x4x undefiniert
- x5x Meldungen des Mailsystems

Beispiel für Quittungen

- 220 <\<>homecontrolserver.de<\>> Service Ready (unmittelbar nach Session-Start)
- 221 <\<>homecontrolserver.de<\>> Service Closing Transmission Channel (positive Quittung nach Ende der Übertragung des E-Mail-Blocks)

- 250 positives Feedback auf ein SMTP-Kommando
- 354 Start Mail Input (Anweisung, mit der Sendung des E-Mail-Blocks zu beginnen)
- 451 Requested Action Aborted (Fehler im Verlauf der Operation, Abbruch der Aktion)
- 500 Syntax Error (unbekanntes Kommando)
- 501 Syntax Error in Arguments (fehlerhafte Übergabe der Argumente)
- 554 Transaction Failed (Beispielsweise bei fehlerhaftem Versuch, die Sitzung herzustellen)

9.4 Übertragungsprotokoll von Client zu Server

Die Übertragung zwischen Server und Client besteht nicht dauernd, sondern läuft ereignis-gesteuert ab. D.h. erst wenn z.B. der Client ein Befehl schickt, aufgrund eines gedrückten Buttons im Client, wird eine Übertragung gestartet und danach wieder abgeschlossen, eine Verbindung bleibt aber bestehen.

Folgende Befehle werden vom Client zum Server übermittelt:

- [MOTOR1]left / right
- [MOTOR2]left / right
- [SWITCH1]on / off
- [SWITCH2]on / off
- [SWITCH3]on / off
- [SWITCH4]on / off
- [DISCONNECT]
- [RESET]
- [AUTHORIZE] "Username, Password," oder "E-Mail Adresse"
- [ERROR] "code"

Die ersten sechs Befehle werden gesendet, wenn einer der sechs Buttons gedrückt wird.

[DISCONNECT] wird gesendet, wenn der Benutzer den Client per Exit-Button schließen möchte. Der Client wird allerdings erst dann geschlossen, wenn der Server mit einem [DISCONNECT] geantwortet hat.

Der Befehl [RESET] wird vom Server zur Controllereinheit umgeleitet und veranlasst sie die Komponenten in Grundstellungen zu fahren.

Mit [AUTHORIZE] und folgendem Benutzername mit Passwort kann ein Anmeldeversuch gestartet werden. Ist serverseitig der Benutzer mit diesem Passwort registriert, dann antwortet der Server nur mit einem [AUTHORIZE].
Schlägt die Anmeldung fehl, so sendet der Server ‘[ERROR]Unknown User or Password ‘ und es erscheint beim Client eine DialogBox, in der auf die Fehlanmeldung hingewiesen wird.

Die Zeichenkette [ERROR]xxxx teilt dem Server eine Fehlübertragung mit. Hinter dem [ERROR] steht die Fehlerinformation.

9.5 Übertragungsprotokoll von Server zu Client

Folgende Befehle werden übertragen:

- [PORT1]on / off
- [PORT2]on / off
- [PORT3]on / off
- [PORT4]on / off
- [PORT5]left / right / movingleft / movingright
- [PORT6]left / right / movingleft / movingright
- [LABEL_PORT1]xxxx
- [LABEL_PORT2]xxxx
- [LABEL_PORT3]xxxx
- [LABEL_PORT4]xxxx
- [LABEL_PORT5]xxxx
- [LABEL_PORT6]xxxx
- [AUTHORIZE]
- [TEMP]xx
- [TIME]xx:xx
- [MAILER]xxxx
- [STATE]UnitOn / UnitOff / Reset
- [ERROR]"Unknown User or Password"
- [DISCONNECT]xxxx

Die Befehle [PORTx]xxxx werden zum Client durchgeschleift von der Controllereinheit kommend und zeigen die Zustände und Positionen der Lichter 1- 4 und die beiden Motoren an.

Nachdem im Raumeinstellungsmenü die Namen für die einzelnen Ports vergeben wurden, werden die Befehle [LABEL_PORTx] mit dem zuvor eingegeben Namen zum Client geschickt. Dort werden darauf die Buttons umbenannt.

[AUTHORIZE] wird nach einem durch den Client erfolgten Anmeldeversuch gesendet, falls der angegebene Benutzer mit dem Passwort registriert ist.

[TIME] mit der Zeit und [TEMP] mit der Temperatur werden im 2 Sekunden-Takt an den Client übertragen. Der Client wechselt somit im 2-Sekunden-Takt die Temperatur- und Zeitanzeige durch.

[MAILER] mit angehängtem Text wird immer dann gesendet, wenn der Server die Logindaten eines Benutzers versendet.

[STATE] zeigt den Status der Controllereinheit an. “Einheit an”, “Einheit aus” oder “Einheit fährt” in Grundstellung sind die drei Möglichkeiten.

Die Fehlermeldung [ERROR]"Unknown User or Password" wird gesendet, wenn sich ein Benutzer mit falschem Benutzernamen oder -passwort versucht hat anzumelden.

Falls ein Benutzer oder der Administrator bei bestehender Verbindung zu einem Client das Serverprogramm schließt, dann sendet der Server vor dem Schließen ein [DISCONNECT]xxxx.

10. Ideen, die nicht in diesem Projekt realisiert wurden

Wenn es anfangs der Weiterbildung zum Techniker heißt “Sie müssen am Ende des zweiten Jahres ein Projekt abgeben, das bewertet wird”, dann schwirren einem viele Ideen und Gedanken durch den Kopf. Das Problem dabei ist, daß nicht alle Ideen in diesem Projekt Platz finden können, da zum einen sehr wenig Zeit zu Verfügung steht und zum anderen das Wissen zur Umsetzung fehlt.

Nachfolgend werden aus diesem Grund genau diese Ideen aufgelistet, die hauptsächlich aus Zeitgründen verworfen wurden.

- Multi-Raumkomponentensteuerung

  Die Grundidee der Raumkomponentensteuerung, nachfolgend RKS genannt, bestand darin, nicht nur einen Raum steuern und überwachen zu können, sondern bis zu 16 Räume. Dazu waren für jeden Raum eine eigene Controllereinheit vorgesehen, die über einen I^2C-Bus gemeinsam an eine Hauptcontrollereinheit angeschlossen sind. Jede Raumeinheit kann für sich betrieben werden, d.h. bei einem Ausfall der Hauptcontrollereinheit oder anderen Raumeinheiten funktioniert sie, bis auf wenige Einschränkungen, normal weiter.

  Natürlich hätten auch auch die Server- und Clientprogramme auf diese Multi-RKS abgestimmt werden müssen.
  Auf den Server wären oberflächenmäßig keine großen Veränderungen zugekommen, jedoch allerdings auf den Client. Immerhin hätte er nun die Aufgabe besessen 16 RKS’ zu steuern und zu überwachen.

- Kameraüberwachung

  Eine weitere Idee der ersten Stunde war die visuelle Überwachung per Kamera. Dies sollte allerdings nicht in “Big Brother”- Manier von statten gehen, sondern es sollte einfach nur eine Kamera in einem Raum oder Teil des Garten installiert sein, die ein sensibles Objekt beobachtet und die Bilddaten an den entfernten Benutzer übermittelt.
  Das Problem hierbei war die Datenübertragungsrate der seriellen Schnittstelle. Die heutigen “Webcams” verwenden meist den USB-Bus, um die aufgenommenen Bilder weiterzugeben. Da es aber noch zu wenige Controller mit integrierter USB-Schnittstelle gibt, hätte dies eine aufwendigere und teurere Realisierung bedeutet.

- Chatfunktion

  Um sich mit einem eingeloggten Benutzer über Tastatur und Bildschrim absprechen zu können, bedarf es einer Funktion im Server- und Clientprogramm, die genau dies erlaubt. So kann man sich über eine soweiso schon aufgebaute Verbindung unterhalten ohne noch zusätzlich Telefongebühren zahlen zu müssen.
  Diese Funktion wurde als absoluter Luxus eingestuft und hatte somit die niedrigste Priorität, weshalb sie nun auch nicht im Projekt umgesetzt wurde.

- Entfernter Administratorlogin (Remote-Administration)

  Einen Administrator, der sich immer an einem einzigen PC-Arbeitsplatz befindet gibt es nicht. Daher kam der Gedanke auf, dem Administrator die Möglichkeit zu geben, sich auch von den Rechnern aus anmelden zu können, an denen auch “normale” Benutzer arbeiten.
  Da diese Funktion ebenfalls keinen Stellenwert im Pflichtenheft besitzt, entfiel sie kurzerhand.

- IP- und Loginversand nach telefonischer Anmeldung

  Die momentane Versandfunktion der IP ist zwar nützlich, es bekommt aber jeder der registrierten Benutzer die dynamische IP zugesandt, ob er sie braucht oder nicht. Da diese Art der Versendung ein Sicherheitsrisiko birgt, kam -zugegebenermaßen- sehr spät die Idee, sich die IP erst nach einem telefonischen Anruf zukommen zulassen.
  Die digitalen Ortsvermittlungstellen der heutigen Telefonnetze erlauben es die Rufnummer des Anrufenden an den Angerufenen zu übermitteln (CLIP-Funktion).
  So kann man nun diese Rufnummer mit einer in der Benutzerkartei hinterlegten Nummer vergleichen und bei Übereinstimmung die aktuelle IP an die ebenfalls hinterlegte eMail-Adresse versenden.
  Diese Funktion kann man als sehr sichere Anmeldung bezeichnen, da die Rufnummer, die anzurufen ist, eine Geheimnummer sein kann und somit nicht der breiten Öffentlichkeit zugänglich ist. Es wird derzeit darüber nachgedacht diese Funktion in einem Updatepaket anzubieten.

- Benutzerdefinierte Portfunktionen

  Flexibilität ist heutzutage überall ein großgeschriebenes Gut, daß von allen geschätzt wird. So war anfangs die Idee, die Ausgangsports der Controllereinheit von dem Benutzer selbst definieren zu lassen, ein guter Gedanke. Es sollte dem Benutzer überlassen sein, welche Ausgangsports er für welche Geräte verwenden möchte.
  Es stellte sich aber schnell heraus, daß dies nur mit einem hardwaretechnischen Mehraufwand zu realisieren ist, da die Definierungen selbst auf der Controllereinheit in nichtflüchtigen Speichern abgespeichert werden müssten. Der programmiertechnische Aufwand wäre ebenfalls sehr hoch, da der Controller in Assembler programmiert wird.

- Kalenderfunktion

  Während der Umsetzung der Client- und Serversoftware kam die Idee auf, eine Art zeitunterstützte Steuerung miteinzubauen. Hardwaremäßig wären dafür wieder ein nichtflüchtiger Speicher und ein Timer-IC mit Datumsfunktion notwendig.
  Eine rein softwaretechnische Lösung kann hier nicht gefunden werden, da die Controllereinheit auch bei abgeschaltetem Server weiterhin funktionieren muß.

- Display

  Um die Temperatur auch unhabhängig von Server und Client ablesen zu können, ebenfalls die Stati der Komponenten, ist hier ein LCD-Display die erste Wahl. Da aber das Layout zu dem Zeitpunkt, als dieser Gedanke aufkam, bereits fertig war und nur auf den Versand zum Platinenhersteller wartete, wurde der Gedanke für dieses Projekt verworfen.

Anhang
Source Codes und Struktogramme

11. Source Codes des AT89S8151 Controller

```
; Programm: RoomControl V1.45b
; Letzte Änderung: 01/05/2002
; Autor: O. Barkhofen
; Projekt: TAR Raumkomponentensteuerung über TCP/IP (RKS2002) an der Werner-Siemens-Schule Stuttgart
; Homepage: http://www.rks2002.de.vu/
;
; Beschreibung:
; Programm zur Steuerung und Überwachung von Rolladen, Fenster und vier Lichter über
; angeschlossene Tasten und über serielle Schnittstelle. Ermittlung der Zimmertemperatur über
; einen angeschlossenen Temperatursensor (mit Feueralarm). Übermittlung der Zustände an den Client.
; Softwarereset über serielle Schnittstelle


$NOMOD51
$include (at898252.inc)


;***************************************************************************
;***************************************************************************
;**                                                                       **
;**                            Deklarationen                              **
;**                                                                       **
;***************************************************************************
;***************************************************************************


flzahl equ 20h           ;Speicher für Tastenentprellung
gerzstd equ 21h          ;Zustand der Geräte am Port2
tasten equ 22h           ;Tasteneingangssignalabbild
freischalt equ 23h       ;Freischalt- und Sperrbits der Endtaster
status_RoFe equ 24h      ;
convStart equ P3.6       ;Port 3.6 = /RD
readValue equ P3.7       ;Port 3.7 = /WR
ZeitrolloL equ 04h       ;TimeOut für Rollo (0104h = 17s)
ZeitrolloH equ 01h       ;
ZeitFenster equ 17       ;TimeOut für Fenster (17h = 1,5s)
r_timer_stepL equ R1     ;unteres 8-Bit Zählregister für das Rollo
r_timer_stepH equ R2     ;oberes 8-Bit Zählregister für das Rollo
f_timer_step equ R3      ;8-Bit Zählregister für das Fenster
zeitzL equ R4            ;unteres 8-Bit Zählregister für Tastenentprellzeit
zeitzH equ R5            ;oberes 8-Bit Zählregister für Tastenentprellzeit
char_count equ R6        ;Zeichenzählregister
entprell_zh equ 03       ;Tastenentprellzeit optimale Einstellung   03d
entprell_zl equ 64       ;Tastenentprellzeit     “       “        64d
endRolloO equ P3.2       ;Endtaster Rollo oben
endRolloU equ P3.3       ;Endtaster Rollo unten
endFensterZ equ P3.4     ;Endtaster Fenster zu
endFensterA equ P3.5     ;Endtaster Fenster auf



;***************************************************************************
; Um eine Wechselschaltung mit einem Taster realisieren zu können, muß
; man die Flanken des Signalverlaufs eines betätigten Tasters feststel-
; len. Da es nun mit einfachsten Mitteln nicht möglich ist, die Flanken
; auszuwerten, werden die Zustände vor und nach einer Flanke ausgewertet.
; Wenn z.B. vor Betätigung eines Tasters das Signal einen High-Pegel
; und während Betätigung einen Low-Pegel führt, so kann man sagen, daß
; der Signalverlauf eine negative Flanke besitzt.

lt1neu equ flzahl.0      ; * Bitzuordnugen 0-3 = Neue Zustände
lt2neu equ flzahl.1      ;
lt3neu equ flzahl.2      ;
lt4neu equ flzahl.3      ;
lt1alt equ flzahl.4      ; * Bitzuordnungen 4-7 = Alte Zustände
lt2alt equ flzahl.5      ;
lt3alt equ flzahl.6      ;
lt4alt equ flzahl.7      ;


;***************************************************************************
; Diese acht Bits dienen zum Einen als sogenannte Befehlsbits. Zum Anderen
; werden sie als Statusbits verwendet. Während die Auswertung der vier
; ‘Licht’-Bits für Statuszwecke kein Problem darstellt, kann bei den
; Motoren nicht gesagt werden, an welche Position sie das Fenster oder
; Rollo gebracht haben.
; Ein Bespiel: Zu Anfang befindet sich das Rollo an der oberen Position.
```

```
; Die Bits ‘motor1_1’ und ‘motor1_2’ sind nicht gesetzt. Nun fährt das
; Rollo nach unten. Hier wurde nur das ‘motor1_2’-Bit gesetzt. Unten
; angekommen wird das Bit ‘motor1_2’ wieder zurückgesetzt. Man kann
; jetzt also durch alleinige Auswertung der beiden ‘motor1’-Bits nicht
; sagen, wo sich das Rollo befindet, da die Bitkombinationen für beide
; Rollopositionen dieseöben sind. Um dieses Problem abzuschaffen, werden
; weiter unten extra vier Bits zur Verfügung gestellt (siehe
; ‘status_RoFe’).

licht1 equ gerzstd.0     ; * Bitzuordnungen 0-3 = Lichter 1-4
licht2 equ gerzstd.1
licht3 equ gerzstd.2
licht4 equ gerzstd.3
motor1_1 equ gerzstd.4   ; * Bitzuordnungen 4,5 = Motor Rollo
motor1_2 equ gerzstd.5   ;
motor2_1 equ gerzstd.6   ; * Bitzuordnungen 6,7 = Motor Fenster
motor2_2 equ gerzstd.7


;*************************************************************************
; Die acht Tasten, die am Modell zur Bedienung des Modells da sind.

ta_lt1 equ tasten.0      ; * Bitzuordnungen 0-3 = Tasten für Lichter 1 - 4
ta_lt2 equ tasten.1
ta_lt3 equ tasten.2
ta_lt4 equ tasten.3
ta_r_e equ tasten.4      ; * Bitzuordnungen 4-7 = Tasten für Motoren (Rollo/Fenster)
ta_r_a equ tasten.5
ta_f_a equ tasten.6
ta_f_z equ tasten.7


;*************************************************************************
; Das Problem, zu erkennen in welche Richtung das Fenster oder Rollo unter-
; wegs ist, kann nur gelöst werden, indem man alle möglichen Positionen
; Bitmustern zuordnet, die bei Drücken einer Taste oder Empfang eines
; entsprechenden Befehls in den Speicher geschrieben werden und bei
; Betätigung eines Endtasters wieder verändert in den Speicher abgelegt
; werden.
; Wenn sich nun das Rollo z.B. nach unten bewegt, so wird dies bereits als
; ‘eine’ Position bezeichnet und erhält ein bestimmtes Bitmuster aus
; zwei Bits bestehend. Für das Fenster gilt dasselbe.
; Wie weit sich das Rollo bereits abgewickelt hat kann man mit dieser
; Methode aber nicht feststellen. Das ist für diese Aufgabe aber auch nicht
; nötig.

stat_Ro1 equ status_RoFe.0
stat_Ro2 equ status_RoFe.1
stat_Fe1 equ status_RoFe.2
stat_Fe2 equ status_RoFe.3

;*************************************************************************
; Die Freischaltbits, abgesehen von ‘client_aktiv’, ‘alarm’ und ‘reset’,
; dienen der sicheren Steuerung der Komponenten und Tastenverriegelung.
; Wird z.B. das Bit für den oberen Endtaster des Rollos gesetzt, so ist
; es nicht mehr möglich, das Rollo nach unten fahren zu lassen, solange
; bis der obere Endtaster betätigt wird. Erst dann wird das Freischaltbit
; für den unteren Endtaster gesetzt und das für den oberen Endtaster
; rückgesetzt. Dasselbe gilt für das Fenster.

rollo_r equ freischalt.0
rollo_h equ freischalt.1
fenster_z equ freischalt.2
fenster_a equ freischalt.3
client_aktiv equ freischalt.4
alarm equ freischalt.5
reset equ freischalt.6




;*************************************************************************
;*************************************************************************
;**                                                                     **
;**                           Initialisierungen                         **
;**                                                                     **
;*************************************************************************
;*************************************************************************

                ajmp 0100h
org 0100h

                mov r_timer_stepL, #00h
                mov r_timer_stepH, #00h
                mov f_timer_step, #00h
```

```
                mov zeitzL, #entprell_zl
                mov zeitzH, #entprell_zh

;*********************************
;Initialisieren der verschiedenen
;Zähler
                mov char_count, #0
                mov gerzstd, #0
                mov tasten, #0
                mov flzahl, #0

;*********************************
;Initialisieren der Bits endRolloO -
;P3.5 als Eingänge (Endtaster)

                orl P3, #00111100b
                clr client_aktiv
                mov freischalt, #0Ah
                mov status_RoFe, #0

;*********************************
;Einstellen der verschiedenen
;Schalter im SFR
                anl TMOD, #0Fh          ;Einstellen der Timer Modi
                orl TMOD, #11h          ;für die Timer 0 und 1
                mov TH2, #0FFh          ;Initialisieren des 2ten Bytes von Timer2
                mov TL2, #0D9h          ;Initialisieren des 1sten Bytes von Timer2
                mov RCAP2L, #0D9h       ;Nachladewert für Timer2 definieren
                mov RCAP2H, #0FFh       ;Nachladewert für Timer2 definieren
                setb TCLK               ;setzen des TransmitClockbits
                setb RCLK               ;setzen des EmpfangClockbits
                mov TH1, #00h           ;Initialisieren des 2ten Bytes von Timer1 –> Zeitüberwachung
                mov TL1, #05Bh          ;Initialisieren des 1ten Bytes von Timer1
                mov TL0, #000h          ;Initialisieren des 1ten Bytes von Timer0 –> AD-Wandler
                mov TH0, #0FFh          ;Initialisieren des 2ten Bytes von Timer0
                mov SCON, #052h         ;Einstellen des Modus für V24
                anl PCON, #000h
                setb ET1                ;Timer1 Interruptfreigabe
                setb EA                 ;Allgemeine Interruptfreigabe
                setb ET0                ;Timer0 Interruptfreigabe
                setb TR2                ;Timer2 Aktivierung
                setb PT0                ;Timer0 Prioritätsbit
                clr alarm               ;initialisieren des ‘alarm’-Bit
                setb reset              ;initialisieren des ‘reset’-Bit
                mov R7, #255


;*************************************************************************
;*************************************************************************
;**                                                                     **
;**                              Programme                              **
;**                                                                     **
;*************************************************************************
;*************************************************************************

;**************************** Hauptprogramm ****************************

; Die Unterprogramme ‘backdoor’ und ‘grundpos’ werden nur ein einziges Mal
; durchlaufen. ‘backdoor’ ermöglicht einen Einstieg ins Programm, um die
; Position des Fensters und Rollos zu ändern.

; Das Bit ‘alarm’ verhindert ein Durchlaufen der Routinen ‘client_anmeld’
; und ‘incomingkey’ bei einem Feueralarm. So ist es dann nicht mehr mög-
; lich die Controllereinheit per Tastenzugriff oder Befehlsempfang über
; serielle Schnittstelle. Nur noch die Routine ‘motorstop’ wird
; abgearbeitet.

; Das Bit ‘reset’ wird durch die Routine ‘grundpos’ gesetzt, falls sich die
; Komponenten Rollo und Fenster beim Start nicht in der Grundposition befinden.
; In der Routine ‘motorstop’ wird das Bit dann zurück gesetzt, wenn
; Rollo und Fenster in Grundposition gefahren sind.


                acall backdoor
                acall grundpos

start:          acall motorstop
                jb alarm, start
                acall client_anmeld
                jb reset, start
                acall incomingkey
                ajmp start



; Das ‘client_aktiv’-Bit unterbindet die Abarbeitung der Befehlempfangsroutine,
```

```
; solange sich der Server (im PAP und im nachfolgenden Client genannt!) nicht
; bei der Controllereinheit angemeldet hat.
; Hat sich der Client angemeldet, so wird sie Routine ‘client_anmeld’ nicht mehr
; komplett durchlaufen. bis sich der Client wieder abmeldet.

client_anmeld:   jb client_aktiv, receive
cl_an2:          jb reset, cl_anm_ret
                 acall anmeld_temp
                 jnb RI, cl_anm_ret
                 clr RI
                 mov a, sbuf
                 cjne a, #’a’, cl_anm_ret
                 setb client_aktiv
                 acall anmeldsuc
                 sjmp cl_anm_ret
receive:         acall received
cl_anm_ret:      ret



;*********** Unterprogramm Temperaturabfrage ohne Anmeldung ***********

; Solange sich kein Client an die Controllereinheit angemeldet hat, muß
; die Temperatur “selbständig” abgefragt werden, da die Temperatur-
; abfrage vom Client übernommen wird.


anmeld_temp:     djnz R7, anmeld_temp_ret
                 acall temp
                 mov R7, #255
anmeld_temp_ret: ret



;******* Unterprogramm Rückfahren in Grundposition nach Neustart *******

; Überprüft nacheinander den oberen bzw den vorderen Endtaster von Rollo bzw.
; Fenster und fährt die dementsprechene Komponente in Grundstellung, falls die
; Endtaster nicht betätigt sind.


grundpos:        jb endRolloO, grundposf
                 setb rollo_H
                 clr rollo_R
                 orl status_RoFe, #3
                 acall rolloh
grundposf:       jb endFensterZ, status_reset
                 clr fenster_a
                 setb fenster_z
                 orl status_RoFe, #0Ch
                 acall fensterz
status_reset:    acall back
                 acall restart
                 jnb endRolloO, grundpos_ret
                 jnb endFensterZ, grundpos_ret
                 clr reset
grundpos_ret:    ret



;********************** Unterprogramm Tastenabfrage **********************
; Nachdem eine Taste gedrückt wurde, wird der Inhalt eines Zählregisters
; abgefragt. Ist es während dem Drücken der Taste auf ‘0’ runter gezählt
; worden, gilt die Taste als entprellt und es kann mit der Auswertung
; fortgefahren werden. Bei den vier Lichtern wird jetzt noch darauf
; geachtet, welche Flanke soeben erzeugt wurde. Dies wird durch Abfrage
; des alten Pegels und Abspeicherung des neuen Pegels erreicht.
; Zur Erkennung, ob mehrere Tasten gedrückt wurden, wird das Tastenab-
; bild Bit für Bit überprüft und die Anzahl der gesetzten Bit festgehalten.
; Beträgt die Anzahl der Bits größer als eins, so wird die Routine wieder
; beendet und es folgt keine Auswertung.
;
; Da eine Fallabfrage unter diesen Umständen sehr unübersichtlich wäre,
; die Wiederholung eines einzigen Befehls mit unterschiedlichen
; Argumenten zur Folge hätte und zeitlich gesehen länger dauern würde,
; wurde hier eine wesentlich elegantere Methode eingesetzt:
; Abhängig von der Taste, die gedrückt wird, wird direkt eine bestimmte
; Stelle im Speicher angesprungen, bei der sich die zur der Taste dazu
; gehörenden Befehle befinden.
;
incomingkey:     mov b, #8
                 mov tasten, P0
                 mov a, tasten
                 cpl a
                 cjne a, #0, check_mulitkey        ;Ist eine Taste gedrückt?
```

```
                mov a, flzahl                   ;Nein: Alte Zustände durch ‘0’ ersetzen
                swap a
                anl a, #0Fh
                mov flzahl, a
                ajmp incom_ret

check_mulitkey: cjne char_count, #0, a_cont
                inc R0
a_cont:         jnb acc.0, next_turn
                inc char_count
next_turn:      rr a
                djnz b, check_mulitkey
                cjne char_count, #1, incom_ret  ;Wurde nur eine Taste gedrückt?
                djnz zeitzL, incom_ret          ; Ja: Entprellzeit 1 erreicht?
                djnz zeitzH, incom_ret          ;    Ja: Entprellzeit 2 erreicht?
                mov a, R0                       ;      Ja: Wert der gedrückten
                subb a, #1                      ;          Taste für Sprung vor-
                                                ;          bereiten.
                mov b, #4                       ;Schrittweite miteinberechnen
                mul ab                          ;
                mov dptr, #k1                   ;Adresse der ersten Sprungmarke ‘k1’
                mov b, #1                       ;in Datenzeiger kopieren
                jmp @a+dptr                     ;Springe zur der Adresse, die durch
                                                ;Adresse + vorbereiteter Offset
                                                ;angezeigt wird

back:           mov a, gerzstd
                cpl a
                mov P2, a
                acall reloadtaste
incom_ret:      mov char_count, #0
                mov R0, #0
                ret

k1:             acall key1                      ;Adresse ‘k1’ + Offset 0h
                ajmp back
k2:             acall key2                      ;Adresse ‘k1’ + Offset 4h
                ajmp back
k3:             acall key3                      ;Adresse ‘k1’ + Offset 8h
                ajmp back
k4:             acall key4                      ;Adresse ‘k1’ + Offset 12h
                ajmp back
k5:             acall rollor                    ;Adresse ‘k1’ + Offset 16h
                ajmp back
k6:             acall rolloh                    ;Adresse ‘k1’ + Offset 20h
                ajmp back
k7:             acall fenstera                  ;Adresse ‘k1’ + Offset 24h
                ajmp back
k8:             acall fensterz                  ;Adresse ‘k1’ + Offset 28h
                ajmp back

key1:           setb lt1neu
                jb lt1alt, fl1
                cpl licht1
fl1:            mov c, lt1neu
                mov lt1alt, c
                ret

key2:           setb lt2neu
                jb lt2alt, fl2
                cpl licht2
fl2:            mov c, lt2neu
                mov lt2alt, c
                ret

key3:           setb lt3neu
                jb lt3alt, fl3
                cpl licht3
fl3:            mov c, lt3neu
                mov lt3alt, c
                ret

key4:           setb lt4neu
                jb lt4alt, fl4
                cpl licht4
fl4:            mov c, lt4neu
                mov lt4alt, c
                ret



; ********************** Unterprogramm Endabschalter ********************
motorstop:      jnb rollo_h, runter             ;Freigabebit für oberen Endtaster gesetzt?
                jnb endRolloO, runter           ;Ja: Ist diese Taste betätigt?
                anl status_RoFe, #0Ch
                setb P2.4                       ;Ja: Motor aus
```

```
                jb alarm, fensterstop
                clr rollo_h                     ;Freigabebits umsetzen
                setb rollo_r
                ajmp rolloreset
runter:         jnb rollo_r, fensterstop        ;dto.
                jnb endRolloU, fensterstop
                orl status_RoFe, #3
                setb P2.5
                setb rollo_h
                clr rollo_r
rolloreset:     anl gerzstd, #11001111b
                acall rtimerend
                jnb P2.6, fensterstop
                jnb P2.7, fensterstop
                clr TR1
                clr reset
                acall reload_rf

fensterstop:    jnb fenster_z, auf
                jnb endFensterZ, auf
                anl status_RoFe, #03h
                setb P2.7
                setb fenster_a
                clr fenster_z
                ajmp fenstereset
auf:            jnb fenster_a, motorstop_end
                jnb endFensterA, motorstop_end
                orl status_RoFe, #0ACh
                setb P2.6
                jb alarm, motorstop_end
                setb fenster_z
                clr fenster_a
fenstereset:    anl gerzstd, #00111111b
                acall ftimerend
                jnb P2.4, motorstop_end
                jnb P2.5, motorstop_end
                clr TR1
                clr reset
                acall reload_rf
motorstop_end:  jnb reset, motorend_end
                acall temp
motorend_end:   ret


;***************** Unterprogramm Befehlsempfang über V24 ***************
; Die Routine ‘received’ funktioniert ähnlich, wie die Routine
; ‘incomingkey’. Allerdings werden hier keine Tastenverriegelung und
; -entprellung verwendet. Weiterhin wird die Bereichsmaskierung für das
; empfangene Byte auf 2 Grenzen angewandt, da der niedriegste HEX-Wert der
; Bytes nicht bei 0h beginnt, sondern bei 30h. Ebenso verhält es sich
; mit der oberen Grenze. Der größtmögliche HEX-Wert entspricht 3Fh.
;

received:       jnb RI, received_end
                clr RI
                mov R7, sbuf            ;Zeichen einlesen
                mov a, R7               ;Zeichen zwischenspeichern
                add a, #0C0h            ;Bereichmaskierung obere Grenze
                jc send_error           ;Ausserhalb gewünschtem Bereich?
                mov a, R7               ;Nein: Zeichen aus Zwischenspecher lesen
                subb a, #30h            ;Bereichmaskierung untere Grenze
                jc send_error           ;Ausserhalb gewünschtem Bereich?
                mov a, R7               ;Nein: Zeichen aus Zw.speicher lesen
                mov b, #04              ;Schrittweite miteinbeziehen
                subb a, #30h            ;Pointer vorbereiten
                mul ab                  ;Adressschritte vorbereiten
                mov dptr, #order1       ;Pointer festlegen
                jmp @a+dptr             ;Auf durch Pointer zeigende Adresse springen


order1:         setb licht1             ;L1 an  ‘0’ -> #30h
                ajmp ord_selected
order2:         clr licht1              ;L1 aus ‘1’
                ajmp ord_selected
order3:         setb licht2             ;L2 an  ‘2’
                ajmp ord_selected
order4:         clr licht2              ;L2 aus ‘3’
                ajmp ord_selected
order5:         setb licht3             ;L3 an  ‘4’
                ajmp ord_selected
order6:         clr licht3              ;L3 aus ‘5’
                ajmp ord_selected
order7:         setb licht4             ;L4 an  ‘6’
                ajmp ord_selected
order8:         clr licht4              ;L4 aus ‘7’
```

```
                ajmp ord_selected
order9:         acall rollor            ;Rollo runter    '8'
                ajmp ord_selected
order10:        acall rolloh            ;Rollo hoch      '9'
                ajmp ord_selected
order11:        acall fenstera          ;Fenster auf     ':'
                ajmp ord_selected
order12:        acall fensterz          ;Fenster zu      ';'
                ajmp ord_selected
order13:        acall abmeldung         ;Client abmelden  -> '<'
                ajmp received_end
order14:        acall status            ;Status abfragen -> '='
                ajmp received_end
order15:        acall temp              ;Temp einholen -> '>'
                ajmp received_end
order16:        acall swreset           ;Softwarereset -> '?'
                ajmp received_end

ord_selected:   mov a, gerzstd
                cpl a
                mov P2, a
received_end:   ret


;******************** Unterprogramm Software Reset ********************
; Programmierung und Aktivierung des Watchdog-Timers. Nach Aktivierung
; des WDT läuft die Controllereinheit nur noch 16ms und führt dann einen
; Hardware-Reset aus. Danach beginnt sie wieder bei Adresse 0000h die
; Befehle abzuarbeiten.

swreset:        anl WMCON, #01h
                orl WMCON, #01h
                ret


;******************** Unterprogramm Messagesausgabe *********************
; Messageausgaben an den Client, inklusive des Statuswortes.
; Anmeldebestätigung und Temperatur werden an anderen Stellen geschickt.

send_error:     mov a, #'E'                     ;—> 69d
                sjmp send_it

anmeldsuc:      mov a, #'B'                     ;—> 66d
                sjmp send_it

restart:        mov a, #'N'                     ;—> 78d
                sjmp send_it

abmeldung:      mov a, #'A'                     ;—> 65d
                clr client_aktiv
                sjmp send_it

r_overheat:     mov a, #'T'                     ;—>84d
                sjmp send_it

send_it:        acall serout
                ret


; Das endgültige Statuswort setzt sich aus den unteren vier Bits
; der Variable 'gerzstd' und der aus vier Bits bestehenden Variable
; 'status_RoFe' zusammen.

status:         mov char_count, #4
                mov b, gerzstd
                anl b, #0Fh
                mov a, status_RoFe
                anl a, #0Fh
stat_rla:       rl a
                djnz char_count, stat_rla
                orl a, b
                acall serout
                ret


;********************** Unterprogramm Motorsteuerung ******************
; Bei jeder Aktion mit Rollo oder Fenster wird das Befehlswort geändert,
; gemäß dem Anschluß der Relais an Port 2, und am Ende

rollor:         jnb rollo_r, rollorr
                orl status_RoFe, #1
                setb motor1_2
                clr motor1_1
                jb TR1, rollorr
                setb TR1
rollorr:        ret
```

```
rolloh:          jnb rollo_h, rollohr
                 anl status_RoFe, #0Eh
                 setb motor1_1
                 clr motor1_2
                 jb TR1, rollohr
                 setb TR1
rollohr:         ret

fensterz:        jnb fenster_z, fensterar
                 anl status_RoFe, #0Bh
                 setb motor2_2
                 clr motor2_1
                 jb TR1, fensterar
                 setb TR1
fensterar:       ret

fenstera:        jnb fenster_a, freturn
                 orl status_RoFe, #04h
                 setb motor2_1
                 clr motor2_2
                 jb TR1, freturn
                 setb TR1
freturn:         ret



;*************** Unterprogramm Tastenentprellzeit Nachlader ***********
; Hier werden Zentral die beiden Zählregister für die Warteschleife
; zum softwaremäßigen Entprellen nachgeladen.

reloadtaste:     mov zeitzL, #entprell_zl
                 mov zeitzH, #entprell_zh
                 ret



;******************** Unterprogramm Senden über V24 ******************
; Zentrale Senderoutine. Parameterübergabe mittels Abspeicherung des
; zu sendenden Zeichen in Akku durch die aufrufende Routine.

serout:          jnb TI, serout
                 clr TI
                 mov sbuf, a
                 ret



;************** Unterprogramm Starten des A/D-Wandlers ***************
; Routine zum Starten des Umwandlungsprozesses des A/D-Wandlers. Falls
; bereits ein Wandlungsprozess läuft, wird die aktuelle Anfoerderung
; verworfen.

temp:            jb TR0, tempret          ;Läuft bereits eine Wandlung?
                 clr convStart            ;Nein: Starte Wandlung
                 setb TR0                 ;Starte Timer0
tempret:         ret



;*********************** Unterprogramm Backdoor **********************
; Ermöglicht parallel zum normalen Ablauf eine Art Serviceeinstellung.
; Dient zum Anfahren von beliebigen Punkten des Fensters und des Rollos.
; Tasten haben dieselbe Funktion, wie im normaen Betrieb, abgesehen
; von den Lichttasten.
; Nach Abschließen der gewünschten Einstellung Resettaste drücken.

backdoor:        mov a, P0
                 cjne a, #01011111b, back_door_ret

back_secure:     jnb P0.7, back_secure
                 mov zeitzl, #255
                 mov zeitzh, #255
back_step:       djnz zeitzL, back_step
                 djnz zeitzH, back_step

endlos:          mov a, P0
                 cpl a
                 jb P0.4, rh
                 clr P2.5
                 setb P2.4
                 sjmp endlos
rh:              jb P0.5, fa
                 clr P2.4
                 setb P2.5
```

```
                sjmp endlos
fa:             jb P0.6, fz
                clr P2.6
                setb P2.7
                sjmp endlos
fz:             jb P0.7, allstop
                clr P2.7
                setb P2.6
                sjmp endlos
allstop:        orl P2, #0F0h
                sjmp endlos

back_door_ret:  ret


;************************************************************************
;************************************************************************
;**                                                                    **
;**                     Interrupt Serviceroutinen                      **
;**                                                                    **
;************************************************************************
;************************************************************************

;*************** Unterprogramm Zeitüberwachung der Motoren ************
timer1int:      jb motor1_1, checkr                     ;Läuft der Rollomotor?
                jb motor1_2, checkr                     ;
                ajmp checkf
checkr:         acall rollotimer                        ;Ja: Überprüfe Laufzeit des Rollo.
checkf:         jb motor2_1, checkf1                    ;Läuft der Fenstermotor?
                jb motor2_2, checkf1                    ;
                ajmp jmpreload
checkf1:        acall fenstertimer                      ;Ja: Überprüfe Laufzeit des Fensters.
jmpreload:      acall reload_rf                         ;Nachladewerte in die betreffenden Register
                reti                                    ;schreiben.

rollotimer:     inc r_timer_stepL                       ;Zählregister um eins hochzählen
                cjne r_timer_stepL, #00, jmpnorollover  ; und prüfen ob Rollo länger
                inc r_timer_stepH                       ;als voreingestellte Zeit läuft
jmpnorollover:  cjne r_timer_stepL, #ZeitrolloL, retry  ; für 25s -> #7Eh
                cjne r_timer_stepH, #ZeitrolloH, retry  ; für 25s -> #01h
rolloend:       clr motor1_1                            ;Befehlswort aktualisieren.
                clr motor1_2                            ;
                setb P2.5                               ;Motor direkt ansteuern und
                setb P2.4                               ;abschalten.
                jb motor2_1, rtimerend                  ;Läuft der Fenstermotor?
                jb motor2_2, rtimerend                  ;
                clr TR1                                 ;Nein: Dann Timer1 deaktivieren.
                setb reset                              ;Reset-Bit setzen
rtimerend:      mov r_timer_stepL, #0                   ;Zählregister des Rollotimers auf
                mov r_timer_stepH, #0                   ;Null setzen.
retry:          ret

reload_rf:      mov TH1, #00h                           ;Timer1 Nachladewert in Timerregister
                mov TL1, #05Bh                          ;schreiben.
                ret

fenstertimer:   inc f_timer_step                        ;Zählregister um eins hochzählen und prüfen
                cjne f_timer_step, #ZeitFenster, ftimer_return     ;ob Fenster länger als eingestellte Zeit
                                                                   ;läuft.
f_stop_sensor:  setb P2.6                               ;Motor direkt ansteuern und abschalten.
                setb P2.7                               ;
                clr motor2_1                            ;Befehlswort aktualisieren
                clr motor2_2                            ;
                jb motor1_1, ftimerend                  ;Läuft der Rollomotor?
                jb motor1_2,ftimerend                   ;
                clr TR1                                 ;Nein: Dann deaktiviere Timer1.
                setb reset                              ;Setze Alarm-Bit.
ftimerend:      mov f_timer_step, #0                    ;Zählregister des Rollotimers auf Null setzen.
ftimer_return:  ret



;************** Unterprogramm für A/D-Wandlung **************
; liest den gewandelten Wert ein und überprüft ihn mittels
; einer Bereichsüberschreitungsroutine. Besteht Feueralarm,
; wird ein 'T' an den Client zur Signalisierung gesendet.
; Weiterhin werden die Komponenten in Grundstellung gefahren.
; Falls kein Alarm besteht, wird der gewandelte Wert an den
; Client übertragen, sofern dieser angemeldet ist.

ADTimeup:       clr readValue                   ;Schaltet die Tri-States des A/D-Wandler niederohmig.
                mov a, P1                       ;Einlesen des am Port1 befindlichen Bitwortes in den Akku.
                orl P3, #11000000b              ;Beide Steuerleitungen des A/D-Wandlers auf High-Pegel
setzen.
                mov b, a                        ;Akku in Register B zwischenspeichern.
```

```
alarmcheck:     add a, #225                     ;Bei Überlauf der Addition herrscht Feueralarm.
                jnc sendtemp                    ;Überlauf des Akku?
                setb rollo_h                    ;Ja: Freigabebits für Grundposition setzen.
                setb fenster_z                  ;
                clr rollo_r                     ;
                clr fenster_a                   ;
                orl P2, #0Fh                    ;Lichter ausschalten.
                clr p2.4                        ;Motoren direkt ansteuern und in
                setb P2.5                       ;Grundpositionen fahren.
                clr P2.7                        ;
                setb P2.6                       ;
                setb alarm                      ;Alarm-Bit setzen
                jnb client_aktiv, adreti        ;Falls Client angemeldet,
                acall r_overheat                ;das Zeichen 'T' senden
                sjmp adreti
sendtemp:       jnb client_aktiv, adreti        ;Falls Client angemeldet,
                mov a, b                        ;gewandelten Wert als Temperatur
                acall serout                    ;an Client schicken.
ADreti:         clr TR0                         ;Deaktivieren des Timers0.
                mov TH0, #0FFh                  ;Nachladewerte in die Timer-
                mov TL0, #000h                  ;zählregister schreiben.
                reti



;************************************************************************
;************************************************************************
;**                                                                    **
;**                    Belegung der Einsprungadressen                  **
;**                                                                    **
;************************************************************************
;************************************************************************

org 001Bh
                ajmp timer1int

org 000Bh
                ajmp ADTimeup
                end
```

12. Source Codes des Clients

Die Routine Show

```
Public Sub Show(incomingData As String)
    Dim dispInstruction As String
    Dim dispData As String

    dispInstruction = Mid(incomingData, 1, InStr(1, incomingData, "]"))
    dispData = Mid(incomingData, InStr(1, incomingData, "]") + 1, Len(incomingData))
    frmClient.lcdMessage.Speed = 150

    Select Case dispInstruction
        Case "[WELCOME]"
            frmClient.lcdMessage.ActiveColor = &HFF00&
            frmClient.lcdMessage.InactiveColor = &H4000&
            frmClient.lcdMessage.AutoScroll = True
            frmClient.lcdMessage.Caption = "Welcome to your Room Control Center"
        Case "[MESSAGE]"
            frmClient.lcdMessage.ActiveColor = &HFF00&
            frmClient.lcdMessage.InactiveColor = &H4000&
            frmClient.lcdMessage.AutoScroll = False
            frmClient.lcdMessage.Caption = dispData
        Case "[CONNECT]"
            frmClient.lcdMessage.ActiveColor = &HFF00&
            frmClient.lcdMessage.InactiveColor = &H4000&
            frmClient.lcdMessage.AutoScroll = False
            frmClient.lcdMessage.Caption = "connecting..."
        Case "[WAIT]"
            frmClient.lcdMessage.ActiveColor = &HFF00&
            frmClient.lcdMessage.InactiveColor = &H4000&
            frmClient.lcdMessage.AutoScroll = False
            frmClient.lcdMessage.Caption = "waiting..."
        Case "[RESET]"
            frmClient.lcdMessage.ActiveColor = &HFF00&
            frmClient.lcdMessage.InactiveColor = &H4000&
            frmClient.lcdMessage.AutoScroll = False
            frmClient.lcdMessage.Caption = "Unit reseted"
        Case "[FAILED]"
            frmClient.lcdMessage.ActiveColor = &HFF00&
            frmClient.lcdMessage.InactiveColor = &H4000&
            frmClient.lcdMessage.AutoScroll = False
            frmClient.lcdMessage.Caption = "connection failed"
        Case "[DISCONNECT]"
            frmClient.lcdMessage.ActiveColor = &HFF00&
            frmClient.lcdMessage.InactiveColor = &H4000&
            If Len(dispData) > 12 Then
                frmClient.lcdMessage.AutoScroll = False
            Else
                frmClient.lcdMessage.AutoScroll = True
            End If
            frmClient.lcdMessage.Caption = "disconnected" & describe
        Case "[TEMP]"
            frmClient.lcdMessage.ActiveColor = &HFF00&
            frmClient.lcdMessage.InactiveColor = &H4000&
            frmClient.lcdMessage.AutoScroll = False
            frmClient.lcdMessage.Caption = "Temperature:            " & dispData & "°C"
        Case "[TIME]"
            frmClient.lcdMessage.ActiveColor = &HFF00&
            frmClient.lcdMessage.InactiveColor = &H4000&
            frmClient.lcdMessage.AutoScroll = False
            frmClient.lcdMessage.Caption = "Server Time:          " & dispData
        Case "[FIRE]"
            frmClient.lcdMessage.AutoScroll = False
            frmClient.lcdMessage.ActiveColor = &HFF&
            frmClient.lcdMessage.InactiveColor = &H40&
            frmClient.lcdMessage.Caption = "Fire Alert!!!"
    End Select
End Sub
```

Die Routine `Process` aus dem Modul `Empfang.bas`

Deklaration globaler Variablen

```
Public dispTT As String
Public Fire As Boolean
```

Routine `Process`

```
Public Sub Process(inData As String)
Dim dispCommand As String
Dim dispInstruction As String
Dim dispData As String

Do While InStr(1, inData, vbCrLf)
  sCommand = Mid(inData, 1, InStr(1, inData, vbCrLf) - 1)
  sInstruction = Mid(sCommand, 1, InStr(1, sCommand, "]"))
  sData = Mid(sCommand, InStr(1, sCommand, "]") + 1, Len(sCommand))

  Select Case UCase(sInstruction)

    Case "[AUTHORIZE]"
      frmClient.cmdConn.Picture = _
          frmClient.ilsConnect.ListImages(2).Picture
      frmClient.cmdConn.ToolTipText = "Disconnect"
      frmClient.ledStatusServer.Status = True
      frmClient.frmConnect.Caption = "Connect to Server:  " & frmClient.sckClient.RemoteHostIP _

      Display.Show ("[WAIT]")
      INISchreiben "Server", frmClient.txtRemoteServer.Text, "Config"
      INISchreiben "Login", frmClient.txtUsername.Text, "Config"

    Case "[MAILER]"
      Display.Show ("[MESSAGE]") & sData

    Case "[STATE]"
      Select Case sData
        Case "UnitOn"
          frmClient.ledStatusUnit.Status = True
          frmClient.Panel_Enable
        Case "UnitOff"
          frmClient.Panel_Disable
          frmClient.ledStatusUnit.Status = False
        Case "Reset"
          Display.Show ("[RESET]")
      End Select

    Case "[LABEL_PORT1]": frmClient.cmdSwitch1.Caption = sData
    Case "[LABEL_PORT2]": frmClient.cmdSwitch2.Caption = sData
    Case "[LABEL_PORT3]": frmClient.cmdSwitch3.Caption = sData
    Case "[LABEL_PORT4]": frmClient.cmdSwitch4.Caption = sData
    Case "[LABEL_PORT5]": frmClient.cmdMotor2.Caption = sData
    Case "[LABEL_PORT6]": frmClient.cmdMotor1.Caption = sData

    Case "[ERROR]"
        Select Case sData
            Case "Unknown User or Password"
                frmClient.sckClient.Close
                MsgBox "Wrong username or password", vbExclamation, "Error Message"
        End Select

    Case "[PORT5]"
            Select Case sData
                Case "left"
                    frmClient.cmdMotor1.Enabled = True
                    frmClient.ledMotor1.blink = False
                    frmClient.ledMotor1.LedColor = Green
                    frmClient.ledMotor1.Status = True

                Case "right"
                    frmClient.cmdMotor1.Enabled = True
                    frmClient.ledMotor1.blink = False
                    frmClient.ledMotor1.LedColor = Red
                    frmClient.ledMotor1.Status = True

                Case "movingright"
                    frmClient.cmdMotor1.Enabled = False
                    frmClient.ledMotor1.LedColor = Red
                    frmClient.ledMotor1.blink = True

                Case "movingleft"
                    frmClient.cmdMotor1.Enabled = False
                    frmClient.ledMotor1.LedColor = Green
```

```
                frmClient.ledMotor1.blink = True

        End Select

Case "[PORT6]"
  Select Case sData
    Case "left"
      frmClient.cmdMotor2.Enabled = True
      frmClient.ledMotor2.blink = False
      frmClient.ledMotor2.LedColor = Green
      frmClient.ledMotor2.Status = True

    Case "right"
      frmClient.cmdMotor2.Enabled = True
      frmClient.ledMotor2.blink = False
      frmClient.ledMotor2.LedColor = Red
      frmClient.ledMotor2.Status = True

    Case "movingright"
      frmClient.cmdMotor2.Enabled = False
      frmClient.ledMotor2.LedColor = Red
      frmClient.ledMotor2.blink = True

    Case "movingleft"
      frmClient.cmdMotor2.Enabled = False
      frmClient.ledMotor2.LedColor = Green
      frmClient.ledMotor2.blink = True
  End Select

Case "[PORT1]"
  If frmClient.ledSwitch1.Status = True Then
    If sData = "off" Then
      frmClient.ledSwitch1.Status = False
    End If
  Else
    If sData = "on" Then
      frmClient.ledSwitch1.Status = True
    End If
  End If

Case "[PORT2]"
  If frmClient.ledSwitch2.Status = True Then
    If sData = "off" Then
      frmClient.ledSwitch2.Status = False
    End If
  Else
    If sData = "on" Then
      frmClient.ledSwitch2.Status = True
    End If
  End If

Case "[PORT3]"
  If frmClient.ledSwitch3.Status = True Then
    If sData = "off" Then
      frmClient.ledSwitch3.Status = False
    End If
  Else
    If sData = "on" Then
      frmClient.ledSwitch3.Status = True
    End If
  End If

Case "[PORT4]"
  If frmClient.ledSwitch4.Status = True Then
    If sData = "off" Then
      frmClient.ledSwitch4.Status = False
    End If
  Else
    If sData = "on" Then
      frmClient.ledSwitch4.Status = True
    End If
  End If

Case "[TIME]"
  If dispTT = "" Or dispTT = "Time" Then
    Display.Show ("[TIME]") & sData
    dispTT = "Time"
  End If

Case "[TEMP]"
  If dispTT = "Temp" Then
    If sData = "Fire" Then
      frmClient.Panel_Disable
      Display.Show ("[FIRE]")
    Else
      Display.Show ("[TEMP]") & sData
```

```
          End If
          dispTT = "Time"
        Else
          dispTT = "Temp"
        End If

      Case "[DISCONNECT]"
        frmClient.sckClient.Close
        frmClient.Panel_Disable
        frmClient.cmdConn.Picture = _
            frmClient.ilsConnect.ListImages(1).Picture
        frmClient.cmdConn.ToolTipText = "Connect"
        frmClient.frmConnect.Caption = "Connect to Server: disconnected"
        frmClient.ledStatusServer.Status = False
        frmClient.ledStatusUnit.Status = False
        Display.Show ("[DISCONNECT]") & sData

      Case Else
        frmClient.sendToServer ("[ERROR]unknown command: ") & UCase(sInstruction) & sData

    End Select
    inData = Mid(inData, InStr(1, inData, vbCrLf) + 2, Len(inData))
    Loop
End Sub
```

Die Routinen der Formulars `client.frm`

Deklaration globaler Variablen

```
Public Connect As Integer
```

Routine `cmdInfo_Click`

```
Private Sub cmdInfo_Click()          'Button für Info Message Box
    Info
End Sub
```

Routine `Form_Load`

```
Private Sub Form_Load()
    swfTAR.Base = App.Path & "\tar.swf"
    swfTAR.Movie = App.Path & "\tar.swf"
    Panel_Disable
    INIFileName = App.Path & "\Config.ini"
    txtRemoteServer.Text = INILesen("Server", "Config")
    txtUsername.Text = INILesen("Login", "Config")
    Display.Show ("[WELCOME]")
End Sub
```

Routine `cmdConnect_Click`

```
Private Sub cmdConn_Click()
    If (sckClient.State = 7) Then
        On Error GoTo errCmdExit
        Display.Show ("[DISCONNECT]")
        sendToServer ("[DISCONNECT]")
        cmdConn.Picture = _
            ilsConnect.ListImages(1).Picture
        cmdConn.ToolTipText = "Connect"
        Exit Sub

errCmdExit:
    ElseIf (sckClient.State = 0) Then
        If Fire = False Then
            Display.Show ("[FAILED]")
        End If
        On Error GoTo errorCmdConn

        If txtRemoteServer.Text = "" Then
            MsgBox "Please enter a server", vbExclamation, "Error Message"
            txtRemoteServer.SetFocus
            Exit Sub
        End If
        If txtUsername.Text = "" Then
            MsgBox "Please enter a username", vbExclamation, "Error Message"
            txtRemoteServer.SetFocus
            Exit Sub
        End If

        sckClient.Close
        sckClient.Connect txtRemoteServer.Text, 35400
        Exit Sub
errorCmdConn:

    End If
End Sub
```

Routine `sckCLient_Connect`

```
Private Sub sckClient_Connect()
    sendToServer ("[AUTHORIZE]" & txtUsername.Text & "," & txtPassword.Text & ",")
    Display.Show ("[CONNECT]")
End Sub
```

Routine sckClient_DataArrival

```
Private Sub sckClient_DataArrival(ByVal bytesTotal As Long)
    Dim sString As String
    If sckClient.State = 7 Then
        sckClient.GetData sString, vbString
        Empfang.Process sString
    End If
End Sub
```

Routine sendToServer

```
Public Sub sendToServer(toSend As String)
    If sckClient.State = 7 Then
        sckClient.SendData (toSend & vbCrLf)
    End If
End Sub
```

Routine Form_Terminate

```
Private Sub Form_Terminate()
    On Error GoTo Termin
        sendToServer ("[DISCONNECT]")
Termin:
End Sub
```

Routine Form_Unload

```
Private Sub Form_Unload(Cancel As Integer)
    On Error GoTo Unload
    sendToServer ("[DISCONNECT]")
Unload:
End Sub
```

Routine sckClient_Close

```
Private Sub sckClient_Close()
    sckClient.Close
    cmdConn.Picture = _
            ilsConnect.ListImages(1).Picture
    cmdConn.ToolTipText = "Connect"
    Panel_Disable
    ledStatusServer.Status = False
    ledStatusUnit.Status = False
    Display.Show ("[DISCONNECT]")
End Sub
```

Routine sckClient_Error

```
Private Sub sckClient_Error(ByVal Number As Integer, Description As String, ByVal Scode As Long, ByVal
Source As String, ByVal HelpFile As String, ByVal HelpContext As Long, CancelDisplay As Boolean)
    cmdConn.Enabled = True
    sckClient.Close
End Sub
```

Routine cmdMotor2_CLick

```
Private Sub cmdMotor2_Click()
    If ledMotor2.LedColor = Red Then
        sendToServer ("[MOTOR2]left")
        cmdMotor2.Enabled = False
    End If
    If ledMotor2.LedColor = Green Then
        sendToServer ("[MOTOR2]right")
        cmdMotor2.Enabled = False
    End If
End Sub
```

Routine `cdmMotor1_Click`

```
Private Sub cmdMotor1_Click()
    If ledMotor1.LedColor = Green Then
        sendToServer ("[MOTOR1]right")
        cmdMotor1.Enabled = False
    End If
    If ledMotor1.LedColor = Red Then
        sendToServer ("[MOTOR1]left")
        cmdMotor1.Enabled = False
    End If
End Sub
```

Routine `cmdSwitch1_Click`

```
Private Sub cmdSwitch1_Click()
    If ledSwitch1.Status = True Then
        sendToServer ("[SWITCH1]off")
    End If
    If ledSwitch1.Status = False Then
        sendToServer ("[SWITCH1]on")
    End If
End Sub
```

Routine `cmdSwitch2_Click`

```
Private Sub cmdSwitch2_Click()
    If ledSwitch2.Status = True Then
        sendToServer ("[SWITCH2]off")
    End If
    If ledSwitch2.Status = False Then
        sendToServer ("[SWITCH2]on")
    End If
End Sub
```

Routine `cmdSwitch3_Click`

```
Private Sub cmdSwitch3_Click()
    If ledSwitch3.Status = True Then
        sendToServer ("[SWITCH3]off")
    End If
    If ledSwitch3.Status = False Then
        sendToServer ("[SWITCH3]on")
    End If
End Sub
```

Routine `cmdSwitch_Click`

```
Private Sub cmdSwitch4_Click()
    If ledSwitch4.Status = True Then
        sendToServer ("[SWITCH4]off")
    End If
    If ledSwitch4.Status = False Then
        sendToServer ("[SWITCH4]on")
    End If
End Sub
```

Routine `cmdReset_Click`

```
Private Sub cmdReset_Click()
    sendToServer ("[RESET]")
    Panel_Disable
End Sub
```

Routine `cmdExit_Click`

```
Private Sub cmdExit_Click()
    If Connect = 1 Then
        sendToServer ("[DISCONNECT]")
    End If
    End
End Sub
```

Routine Panel_Disable

```
Public Function Panel_Disable()
    cmdMotor1.Enabled = False
    cmdMotor2.Enabled = False
    cmdSwitch1.Enabled = False
    cmdSwitch2.Enabled = False
    cmdSwitch3.Enabled = False
    cmdSwitch4.Enabled = False
    cmdReset.Enabled = False
End Function
```

Routine Panel_Enable

```
Public Function Panel_Enable()
    cmdMotor1.Enabled = True
    cmdMotor2.Enabled = True
    cmdSwitch1.Enabled = True
    cmdSwitch2.Enabled = True
    cmdSwitch3.Enabled = True
    cmdSwitch4.Enabled = True
    cmdReset.Enabled = True
End Function
```

Routine Info

```
Private Function Info()
    Dim Msg
    Dim CR
    CR = Chr(13) + Chr(10)
    Msg = "Raumkomponentensteuerung über TCP/IP " + CR + CR
    Msg = Msg + "Client Modul " + CR + CR
    Msg = Msg + "© by Götz Mayer & Oliver Barkhofen"
    MsgBox Msg, vbInformation, "Info"
End Function
```

Routine txtPassword_KeyPress

```
Public Sub txtPassword_KeyPress(KeyAscii As Integer)
    cmdConn.Default = True
End Sub
```

13. Source Codes des Servers

Die Routinen des Formulars `server.frm`

Deklaration globaler Variablen

```
Dim gUnitConnected As Boolean, gHosting As Boolean, gFireAlert As Boolean, gConnectionTries%
```

Routine `Form_Load`

```
Private Sub Form_Load()
' Laden des gespeicherten COM-Ports, Feststellung der dynamischen IP und gegebenenfalls Admin-
' Registrierung

    Dim ComConfig As String

    INIFileName = App.Path & "\Config.ini"
    ComConfig = INILesen("COM", "Config")
    If ComConfig = "" Then
        MSComm1.CommPort = 1
        INISchreiben "COM", "1", "Config"
    Else
        MSComm1.CommPort = CInt(ComConfig)
    End If
    GetDynIP
    StatusBar1.Panels(3) = "Com" & MSComm1.CommPort
    StatusBar1.Panels(4) = Left(Time(), 5)
    If GetSetting("Server", "Admin", "Name") = "" Then
         frmAdminRegi.Show
    End If
    Protocol "Server", "online"
    If Not StatusBar1.Panels(2) = "Offline" Then
        If INILesen("StartUp", "SendMail") = "SendMail" Then frmSendMail.Show
    Else
        mnuMailSettings.Enabled = False
        mnuSend.Visible = False
        Label1.Visible = True
    End If
End Sub
```

Routine `cmdConnectUnit_Click`

```
Private Sub cmdConnectUnit_Click()
' Verbindungsaufnahme und -trennung zur Controller Unit. Falls die Unit nicht connected war,
' wird sie abgemeldet und gleich wieder angemeldet. Beim Anmelden läuft ein Timer, der
' nach fünf Anmeldeversuchen eine Fehlermeldung ausgibt.


Dim t As Single

    If gUnitConnected = False Then
        On Error GoTo errCmdExit
        mnuPortSettings.Enabled = False
        cmdConnectUnit.Enabled = False
        UpdateStatus "Trying to connect to Unit ...", 1
        MSComm1.Settings = "9600,N,8,1"
        MSComm1.InputLen = 1
        MSComm1.PortOpen = True
        Timer1.Enabled = True

      Else
        cmdConnectUnit.Picture = ILSUnit.ListImages(1).Picture
        UpdateStatus "Unit disconnected!!!", 1
        Sende 60
        MSComm1.PortOpen = False
        gUnitConnected = False
        Led2.LedColor = Red
        Protocol "Unit", "offline"
        mnuPortSettings.Enabled = True
    End If
    Exit Sub

errCmdExit:
    If MSComm1.PortOpen = True Then MSComm1.PortOpen = False
    Protocol "Server", "Attempted to connect to unit: " + Err.Description
    UpdateStatus "Int: " + ReplaceLetters(Err.Description), 1
    cmdConnectUnit.Enabled = True
End Sub
```

Routine cmdHost_Click

```
Private Sub cmdHost_Click()
' Setzt den Server in Empfangsbereitschaft. Bei einer erfolgreichen Verbindung trennt der Button
' die Verbindung und setzt die Bereitschaft zurück. Befand sich der Server zuvor in
' Bereitschaft so wird auch nur die Bereitschaft deaktiviert

    If gHosting = False Then
        Hosting
        cmdHost.Picture = ILSClient.ListImages(2).Picture
    Else
        If gUserConnected = False Then
            UpdateStatus "Host-Mode disabled.", 2
            Protocol "Server", "Host-Mode disabled"
        Else
            gUserConnected = False
            SendToClient "[DISCONNECT]"
            Protocol "Server", "Connection closed"
        End If
        SendToClient "[DISCONNECT]"
        cmdHost.Picture = ILSClient.ListImages(1).Picture
        gHosting = False
        Led1.LedColor = Red
    End If
End Sub
```

Routine Hosting

```
Private Sub Hosting()
' Setzt den Server in Bereitschaft

    sckConnect.Close
    sckConnect.LocalPort = 35400
    sckConnect.Listen
    UpdateStatus ReplaceLetters("hosting..."), 2
    Protocol "Server", "hosting"
    gHosting = True
End Sub
```

Routine mnuMailSettings_Click

```
Private Sub mnuMailSettings_Click()
' Lädt das Maileinstellungsfenster
    frmMailSettings.Show
End Sub
```

Routine mnuPortSettings_Click

```
Private Sub mnuPortSettings_Click()
' Lädt das ComPorteinstellungsfenster, falls keine offene Verbindung besteht. Ansonsten anzeige
' einer Fehlermeldung

    If MSComm1.PortOpen = False Then
        frmPortSettings.Show
    Else
        MsgBox "Close Connection to Unit first, please."
    End If
End Sub
```

Routine mnuRommSettings_Click

```
Private Sub mnuRoomSettings_Click()
' Lädt das Raumkonfigurationsfenster

    frmRoomSettings.Show
End Sub
```

Routine mnuUser_Click

```
Private Sub mnuUser_Click(Index As Integer)
' Lädt das Adminanmeldefenster bevor das Userfenster angezeigt wird.

    frmAdmin.Show
End Sub
```

Routine mnuSend_Click

```
Private Sub mnuSend_Click()
' Falls der Server sich im Internet befindet, werden den Usern die IP zugesandt. Je nachdem
' welche Einstellung in der Config.ini eingestellt ist, wird das Mailversandfenster gezeigt
' oder nicht

    If Not StatusBar1.Panels(2) = "Offline" Then
        If INILesen("ShowWindow", "SendMail") = True Then
            frmSendMail.Show
        Else
            frmSendMail.Hide
        End If
    End If
End Sub
```

Routine sckConnect_Close

```
Private Sub sckConnect_Close()
' Falls der Client die Verbindung abgebrochen hat, wird der Server wieder in
' Empfangsbereitschaft gesetzt. Andernfalls deaktiviert er die Bereitschaft.

    If gUserConnected = True Then
        Protocol "Client", "User " & gGetUser & " logged off"
        Hosting
    Else
        UpdateStatus "Disconnected", 2
        cmdHost.Picture = ILSClient.ListImages(1).Picture
        Protocol "Server", "Disconnect"
    End If
    frmClient.Caption = "Client Messages "
    gUserConnected = False
    Led1.LedColor = Red
End Sub
```

Routine mnuInfo_Click

```
Private Sub mnuInfo_Click()
    Dim Msg

    Msg = "Raumkomponentensteuerung über TCP/IP " + vbCrLf + vbCrLf
    Msg = Msg + "Server Modul " + vbCrLf + vbCrLf
    Msg = Msg + "© by Götz Mayer & Oliver Barkhofen"
    MsgBox Msg, vbInformation, "Info"
End Sub
```

Routine sckConnect_DataArrival

```
Private Sub sckConnect_DataArrival(ByVal bytesTotal As Long)
' Liest die angekommenden Daten aus und gibt sie an die Datenverarbeitungsroutine weiter.

    Dim sString As String

    sckConnect.GetData sString, vbString
    ProcessData sString, -1
End Sub
```

Routine sckConnect_ConnectionRequest

```
Private Sub sckConnect_ConnectionRequest(ByVal requestID As Long)
' Akzeptiert ankommende Verbindungsanforderungen.

    If sckConnect.State <> sckClosed Then sckConnect.Close
        sckConnect.Accept requestID
        Protocol "Server", "Connection Request by Client with IP: " & sckConnect.RemoteHostIP
End Sub
```

Routine Timer2_Timer

```
Private Sub Timer2_Timer()
    StatusBar1.Panels(4) = Left(Time(), 5)
End Sub
```

Routine Timer1_Timer

```
Private Sub Timer1_Timer()
' Nachdem der Timer aktiviert wurde, wird fünfmal versucht, den Server mit der
' Controller Unit zu verbinden. Schlägt dies fehl, dann erscheint im Display für die Unit
' eine Fehlermeldung.
    If gConnectionTries < 5 Then
        Sende 60
        t = Timer()
        While Timer() < t + 0.05
        Wend
        Sende 97

        Status = Empfang()

        If Status = 66 Then
            cmdConnectUnit.Enabled = True
            cmdConnectUnit.Picture = ILSUnit.ListImages(2).Picture
            UpdateStatus "Connection success!!!", 1
            gUnitConnected = True
            tmrStatus.Enabled = True
            Timer1.Enabled = False
            Led2.LedColor = Green
            Protocol "Unit", "online"

        ElseIf Status = 69 Then
            cmdConnectUnit.Enabled = True
            UpdateStatus "Connection failure!!!", 1
            MSComm1.PortOpen = False
            mnuPortSettings.Enabled = True
        End If

        gConnectionTries = gConnectionTries + 1
    Else
        gConnectionTries = 0
        Timer1.Enabled = False
        MSComm1.PortOpen = False
        mnuPortSettings.Enabled = True
        UpdateStatus "Unit not available!", 1
        cmdConnectUnit.Enabled = True
    End If
End Sub
```

Routine tmrStatus_Timer

```
Private Sub tmrStatus_Timer()
' Der StatusTimer frägt bei angemeldeter Controller Unit den Status der einzelnen Anschlüsse
' ab. Ebenso die Temperatur.
    StatusTimer
End Sub
```

Routine Form_Unload

```
Private Sub Form_Unload(cancel As Integer)
    cmdExit_Click
End Sub
```

Routine cdmExit_Click

```
Private Sub cmdExit_Click()
    SendToClient "[DISCONNECT]due to service work"
    Protocol "Server", "Exit" & vbCrLf, False
    sckConnect.Close
    If MSComm1.PortOpen = True Then
        Sende 60
        MSComm1.PortOpen = False
    End If
    End
End Sub
```

Die Routinen des Formulars `User.frm`

Deklaration der globalen Variablen

```
Dim gUser As Integer
Dim pw1Chg, pw2Chg, emailChg As Boolean
```

Routine `cmdDelUser_Click`

```
' Löscht einen markierten User, indem die nachfolgenden User um eine Position nach oben
' kopiert werden. Der letzte User, der durch diese Aktion zweimal auftauchen würde, wird
' mit "leer" beschrieben.

Private Sub cmdDelUser_Click()
    Dim count%, maxUser%, i%

    maxUser = lstUser.ListCount
    count = lstUser.ListIndex
    If count > -1 Then
        For i = count + 1 To maxUser
            SaveProfile GetProfile(0, i + 1), GetProfile(1, i + 1), GetProfile(2, i + 1), i
        Next
        lstUser.RemoveItem (count)
        SaveProfile "leer", "leer", "leer", maxUser
    Else
        MsgBox "Please mark a user!", vbExclamation
    End If
    frmUser.Caption = "User: " & lstUser.ListCount
End Sub
```

Routine `cmdEditUser_Click`

```
' Prüft, ob ein User ausgewählt wurde und bereitet Eingabefelder und Schaltflächen
' für Eingaben und Useraktionen vor.

Private Sub cmdEditUser_Click()
    If lstUser.ListIndex > -1 Then
        ButtonDeActivate ("useredit")
        txtUser.Item(3).text = GetProfile(2, lstUser.ListIndex + 1)
    Else
        MsgBox "Please mark a user!", vbExclamation
    End If
End Sub
```

Routine `cmdNewUser_Click`

```
' Prüft, ob bereits die maximale Anzahl an User besteht. Falls nicht werden Schaltflächen
' und Eingabefelder vorbereitet.

Private Sub cmdNewUser_Click()
    If lstUser.ListCount = 16 Then
        MsgBox "No more new users.", vbExclamation
    Else
        ButtonDeActivate ("newuser")
        lstUser.ListIndex = -1
    End If
End Sub
```

Routine `cmdOK2_Click`

```
' Es wird nur der Inhalt der Textbox ausgelesen und als Namen des neuen Users gespeichert.
' Anschließend werden die Eingabefelder und Schaltflächen für die Passwort- und eMaileingaben
' vorbereitet.
Private Sub cmdOK2_Click()
    If Not txtUser.Item(1) = "Please enter a User's name." Then
        lstUser.AddItem txtUser.Item(1)
        frmUser.Caption = "User: " & lstUser.ListCount
        ButtonDeActivate ("useredit")
    Else
        ButtonDeActivate ("normal")
    End If
End Sub
```

Routine cmdOK3_Click

```
' Stellt fest, ob es sich bei dieser Aktion um einen neuen oder bereits bestehenden User handelt.
' Bei bestehenden Usern müssen die Felder nicht ausgefüllt werden, da die zuvor gespeicherten
' Daten beim Drücken des 'cmdOK3'-Buttons wieder zurück gespeichert werden.
' Handelt es sich um einen neuen User müssen alle Felder ausgefüllt werden. Bei der eMail-Adresse
' ist zusätzlich auf die richtige Syntax zu achten.

Private Sub cmdOK3_Click()
    Dim usercount%, i%
    Dim pwold$, pwnew$, email$

    If lstUser.ListIndex = -1 Then
        usercount = lstUser.ListCount - 1
    Else
        usercount = lstUser.ListIndex
    End If


   pwold = PasswordDecode(GetProfile(1, usercount + 1))
   email = GetProfile(2, usercount + 1)
   cmdOKUser.SetFocus

    If pw1Chg = True And pw2Chg = True Then
        If txtUser.Item(1).text = txtUser.Item(2).text Then
            pwnew = txtUser.Item(1).text
            ButtonDeActivate ("normal")
        Else
            MsgBox "The passwords you typed in don't match!", vbExclamation
        End If
    ElseIf pwold = "leer" Then
        MsgBox "Please fill in the password fields."
        ButtonDeActivate "useredit"
        Exit Sub
    Else
        pwnew = pwold
    End If

    If emailChg = False Then
        If email = "leer" Then
            MsgBox "Please fill in the eMail field."
            ButtonDeActivate "useredit"
        End If
    Else
        If Not InStr(txtUser.Item(3).text, "@") > 0 Then
            MsgBox "Type in a correct eMail address, please."
            ButtonDeActivate "useredit"
            txtUser.Item(3).SetFocus
            Exit Sub
        Else
            email = txtUser.Item(3).text
            ButtonDeActivate ("normal")
        End If
    End If

    If (pw1Chg And pw2Chg Or emailChg) = False And Not email = "leer" Then
        ButtonDeActivate "normal"
        Exit Sub
    Else
        pw1Chg = False
        pw2Chg = False
        emailChg = False
        SaveProfile lstUser.List(usercount), PasswordEncode(pwnew), email, usercount + 1
    End If

End Sub
```

Routine mnuChgAdmin_Click

```
'Lädt das Fenster für die Speicherung der Logindaten des Administrators

Private Sub mnuChgAdmin_Click()
    frmAdminRegi.Show
End Sub
```

Routine cmdOKUser_Click

```
' Versteckt das Fenster 'User' und entlädt es aus dem Speicher

Private Sub cmdOKUser_Click()
    frmUser.Hide
    Set frmUser = Nothing
End Sub
```

Routine `Form_Load`

```
'Beim Laden des Fensters werden die Merker für die Eingabefeldänderungen zurückgesetzt und
' geprüft ob bereits User registriert wurden

Private Sub Form_Load()
    Dim user As String

    pw1Chg = False
    pw2Chg = False
    emailChg = False
    user = GetProfile(0, 1)
    cmdOKUser.Default = True

    Select Case user
        Case "":     CreateUser_ini
        Case "leer":
        Case Else:   LoadUser_ini
    End Select
End Sub
```

Routine `ButtonDeActivate`

```
' Stellt für die einzelnen Aktionen die Schaltflächen, Variablen und Eingabefelder ein.

Private Sub ButtonDeActivate(button As String)
    Select Case button
        Case "newuser":     txtUser.Item(1).PasswordChar = ""
                            txtUser.Item(1).Visible = True
                            txtUser.Item(1).text = "Please enter a User's name."
                            txtUser.Item(1).SetFocus
                            txtUser.Item(2).Visible = False
                            txtUser.Item(3).Visible = False
                            cmdOK2.Visible = True
                            cmdOK3.Visible = False
                            lstUser.Enabled = False

        Case "useredit":    txtUser.Item(1).PasswordChar = ""
                            txtUser.Item(1).Visible = True
                            txtUser.Item(1).text = "Please enter a password."
                            txtUser.Item(1).SetFocus
                            txtUser.Item(2).Visible = True
                            txtUser.Item(2).PasswordChar = ""
                            txtUser.Item(2).text = "Please repeat the password."
                            txtUser.Item(3).text = "Please enter email address."
                            txtUser.Item(3).Visible = True
                            cmdOK2.Visible = False
                            cmdOK3.Visible = True
                            lstUser.Enabled = False
                            cmdNewUser.Locked = True

        Case "normal":      txtUser.Item(1).Visible = False
                            txtUser.Item(2).Visible = False
                            txtUser.Item(3).Visible = False
                            cmdOK2.Visible = False
                            cmdOK3.Visible = False
                            cmdNewUser.Locked = False
                            lstUser.Enabled = True
    End Select

End Sub
```

Routine `txtUser_KeyPress`

```
'Je nachdem welche Taste gefrückt wurde, wird der Vorgabetext entfernt und stattdessen das
' eingegeben Zeichen dargestellt. Beim Eingabefeld für den Namen bzw. das erste Passwort
' wird noch geprüft um welche Aktion es sich genau handelt.

Private Sub txtUser_KeyPress(Index As Integer, KeyAscii As Integer)
    Select Case Index
        Case 1
            If txtUser.Item(1) = "Please enter a password." Then
                txtUser.Item(1).text = ""
                cmdOK3.Default = True
                txtUser.Item(1).PasswordChar = "*"
            ElseIf txtUser.Item(1) = "Please enter a User's name." Then
                txtUser.Item(1).text = ""
                cmdOK2.Default = True
            End If
            pw1Chg = True

        Case 2
            If txtUser.Item(2).text = "Please repeat the password." Then
                txtUser.Item(2).text = ""
                txtUser.Item(2).PasswordChar = "*"
            End If
            cmdOK3.Default = True
            pw2Chg = True

        Case 3
            If txtUser.Item(3).text = "Please enter email address." Then
                txtUser.Item(3).text = ""
            End If
            cmdOK3.Default = True
            emailChg = True
    End Select
End Sub
```

Die Routinen des Formulars `SendMail.frm`

Routine `CommandButton1_Click`

```
Private Sub CommandButton1_Click()
' Setzt Fortschrittsaneige zurück und löscht Ereignisliste
' Versteckt und entlädt Fenster 'SendMail'
    ProgressBar1.Value = 0
    flex1.Clear
    Hide
    Set frmSendMail = Nothing
End Sub
```

Routine `cmdDefault_Click`

```
Private Sub cmdDefault_Click()
' Erweitert das Fenster um die Liste und läßt es auch wieder schrumpfen
    If frmSendMail.Height = 2805 Then
        frmSendMail.Height = 6000
        cmdDefault.Picture = ils1.ListImages(2).Picture
    ElseIf frmSendMail.Height = 6000 Then
        frmSendMail.Height = 2805
        cmdDefault.Picture = ils1.ListImages(1).Picture
    End If
End Sub
```

Routine `Form_Load`

```
Public Sub Form_Load()
'Initialisierung der Ereignisliste und der Fortschrittsanzeige
    flex1.ColWidth(0) = flex1.Width
    flex1.ColAlignment(0) = Left
    Timer1.Enabled = True
    Protocol "Server", "Sending Mail..."
    ProgressBar1.Min = 0
    ProgressBar1.Value = 0
    ProgressBar1.Max = 5
End Sub
```

Routine `Timer1_Timer`

```
Private Sub Timer1_Timer()
' Ruft die eigentliche Routine zum Mail versenden zeitverzögert zur Fensterdarstellung auf
' da bei einem Aufruf aus der Routine Form_Load das Fenster unsichtbar bleibt
    If Not gSendToUser Then SendIt (1)
    Timer1.Enabled = False
End Sub
```

Routine `SendMailSock_DataArrival`

```
Public Sub SendMailSock_DataArrival(ByVal bytesTotal As Long)
' Liest Daten aus dem TCP-Eingangspuffer
  If Not SendMailSock.State <> 7 Then SendMailSock.GetData Result
End Sub
```

Routine `Timer2_Timer`

```
Private Sub Timer2_Timer()
' TimeOut-Timer für Mailversand
sec = sec + 1
End Sub
```

Routine `SendMailSock_Close`

```
Private Sub SendMailSock_Close()
' Falls Verbindung durchGegenstelle getrennt wird, erfolgt hier ebenfalls eine Trennung um einen
' definierten Zustand zu erreichen
    SendMailSock.Close
End Sub
```

Die Routinen des Formulars `MailSetting.frm`

Routine `chkShow_Click`

```
Private Sub chkShow_Click()
' Überprüft den Zustand der Optionsbox 'chkShow' und sperrt bzw. gibt die Optionsbox 'chkHide'
' frei.
    If chkShow.Value = 0 Then
        chkHide.Enabled = False
    Else
        chkHide.Enabled = True
    End If
End Sub
```

Routine `cmdCancel_Click`

```
Private Sub cmdCancel_Click()
' Versteckt und entlädt das Fenster aus dem Speicher.
    frmMailSettings.Hide
    Set frmMailSettings = Nothing
End Sub
```

Routine `cmdOK_Click`

```
Private Sub cmdOK_Click()
' Speichert die eingebenen und eingestellten Werte in die Config.ini. Anschließend wird
' das Fenster versteckt und aus dem Speicher entladen.

    If txtSMTP.text = "" Then
        MsgBox "Please enter a SMTP-Server.", vbExclamation
        txtSMTP.SetFocus
    Else
        INISchreiben "Server", txtSMTP.text, "SendMail"
    End If

    If chkShow.Value = 0 Then
        INISchreiben "ShowWindow", "False", "SendMail"
    Else
        INISchreiben "ShowWindow", "True", "SendMail"
    End If

    If chkHide.Value = 0 Then
        INISchreiben "HideWindow", "False", "SendMail"
    Else
        INISchreiben "HideWindow", "True", "SendMail"
    End If

    If chkSendIP.Value = 0 Then
        INISchreiben "StartUp", "DontMail", "SendMail"
    Else
        INISchreiben "StartUp", "SendMail", "SendMail"
    End If

    frmMailSettings.Hide
    Set frmMailSettings = Nothing
End Sub
```

Routine `Form_Load`

```
Private Sub Form_Load()
' Lädt die Einstellungen aus der Config.ini und stellt die Optionsboxen und Eingabefelder
' dem entsprechend ein.

    INIFileName = App.Path & "\Config.ini"
    Load_MailSettings
    cmdOK.Default = True
End Sub
```

Die Routinen des Formulars `frmAdminRegi.frm`

Routine `cmdExit_Click`

```
Private Sub cmdExit_Click()

        If Text1.text = "" And Text2.text = "" And Text3.text = "" Then
            frmAdminRegi.Hide
            frmMain.Show
            Set frmAdminRegi = Nothing
        ElseIf Text2.text = Text3.text And Not Text1.text = "" Then
            SaveSetting "Server", "Admin", "Name", LCase(Text1.text)
            SaveSetting "Server", "Admin", "Password", PasswordEncode(Text2.text)
            frmAdminRegi.Hide
            frmMain.Show
            Set frmAdminRegi = Nothing
        Else
            MsgBox "Repeat entries!", vbExclamation
            Text3.text = ""
            Text2.text = ""
        End If
End Sub
```

Routine `Form_Load`

```
Private Sub Form_Load()
    cmdExit.Default = True
    frmMain.Hide
    frmUser.Hide
End Sub
```

Routine `Text3_KeyPress`

```
Private Sub Text3_KeyPress(KeyAscii As Integer)
    cmdExit.Default = True
End Sub
```

Die Routine des Formulars `frmAdmin.frm`

Routine `cmdExit_Click`

```
Private Sub cmdExit_Click()
' Überprüft die Eingabe in den Eingabefeldern und speichert bei Übereinstimmung der Passwörter
' den Namen und das verschlüsselte Passwort in der Registry ab

    If Text1.Item(1) = "" And Text1.Item(2) = "" Then
        frmAdmin.Hide
        Set frmAdmin = Nothing
    ElseIf LCase(GetSetting("Server", "Admin", "Name")) = LCase(Text1.Item(1)) And _
        UCase(GetSetting("Server", "Admin", "Password")) = PasswordEncode(Text1.Item(2)) Then
        frmUser.Show
        frmAdmin.Hide
        Set frmAdmin = Nothing
    Else
        MsgBox "Either Password or User is unknown.", vbExclamation
        Text1.Item(1) = ""
        Text1.Item(2) = ""
        Text1.Item(1).SetFocus
    End If
End Sub
```

Routine `Form_Load`

```
Private Sub Form_Load()
' Legt den 'cmdExit'-Button als Standardschaltfläche fest.
    cmdExit.Default = True
End Sub
```

Die Routinen des Formulars `frmPortSettings.frm`

Routine `cmbPortSettings`

```
Private Sub cmbPortSettings_Click()
' Speichert den eingestelleten Port und weist den gewählten Port dem MSComm1 Steuerelement zu.

        frmMain.MSComm1.CommPort = cmbPortSettings.ListIndex + 1
        INISchreiben "COM", CStr(cmbPortSettings.ListIndex + 1), "Config"
End Sub
```

Routine `cmdExit_Click`

```
Private Sub cmdExit_Click()
' Zeigt den gewählten Port in der Statuszeile von Fenster 'Main' an.
' Versteckt und entlädt das Fenster 'PortSettings'

    frmMain.StatusBar1.Panels(3) = "Com" & frmMain.MSComm1.CommPort
    frmPortSettings.Hide
    Set frmPortSettings = Nothing
End Sub
```

Routine `Form_Load`

```
Private Sub Form_Load()
' Listet in der KomboBox vier Ports auf und zeigt auf den Port, der in der Config.ini
' gespeichert ist.

    INIFileName = App.Path & "\Config.ini"
    cmbPortSettings.AddItem "Com1", 0
    cmbPortSettings.AddItem "Com2", 1
    cmbPortSettings.AddItem "Com3", 2
    cmbPortSettings.AddItem "Com4", 3
    cmbPortSettings.ListIndex = (frmMain.MSComm1.CommPort - 1)
End Sub
```

Die Routinen des Formulars `frmRoomSettings.frm`

Routine `cmdDefault_Click`

```
Private Sub cmdDefault_Click()
' Stellt die Standardbeschriftung der Buttons im Client her.

    Dim i As Integer
    For i = 1 To 4
        txtPort.Item(i) = "Light" & i
    Next
    txtPort.Item(6) = "Window"
    txtPort.Item(5) = "Shutter"
End Sub
```

Routine `cmdExit_Click`

```
Private Sub cmdExit_Click()
' Überprüft, ob alle Eingabefelder ausgefüllt wurden und gibt gegebenenfalls eine Fehlermeldung
' aus. Bei korrekter Eingabe werden die Inhalte der Felder in die Config.ini gespeichert
' und bei bestehender Verbindung zum Client an diesen gesendet. Anschließend wird das
' Fenster 'RoomSettings' versteckt und entladen.
    Dim i As Integer
    If txtPort.Item(1).text = "" Or txtPort.Item(2).text = "" _
    Or txtPort.Item(3).text = "" Or txtPort.Item(4).text = "" _
    Or txtPort.Item(5).text = "" Or txtPort.Item(6).text = "" Then
        MsgBox "Please fill in every 'Port'-field!", vbExclamation
    Else
        For i = 1 To 6
            If Len(txtPort.Item(i)) > 13 Then txtPort.Item(i) = Left(txtPort.Item(i), 12) & "."
            INISchreiben "Port" & i, (txtPort.Item(i).text), "RoomConfiguration"
        Next
        LoadRoomConfig
        frmRoomSettings.Hide
        Set frmRoomSettings = Nothing
    End If
    cmdExit.Default = True
End Sub
```

Routine `CommandButton1_Click`

```
Private Sub CommandButton1_Click()
' Versteckt und entlädt das Fenster 'RoomSettings'

    frmRoomSettings.Hide
    Set frmRoomSettings = Nothing
End Sub
```

Routine `Form_Load`

```
Private Sub Form_Load()
' Überprüft, ob in der Config.ini bereits Einträge für die Buttonbeschriftung vorhanden sind.
' Falls kein Eintrag vorhanden ist, werden die Standardeinträge hergestellt und gespeichert.
' Ansonsten werden die bereits eingegeben Einträge in die Eingabefelder eingetragen.

    Dim port$, i%
    INIFileName = App.Path & "\Config.ini"
    port = INILesen("Port1", "RoomConfiguration")
    If port = "" Then
        cmdDefault_Click
        For i = 1 To 6
            INISchreiben "Port" & i, txtPort.Item(i), "RoomConfiguration"
        Next
    Else
        For i = 1 To 6
            txtPort.Item(i) = INILesen("Port" & i, "RoomConfiguration")
        Next
    End If
End Sub
```

Routine txtPort_GotFocus

```
Private Sub txtPort_GotFocus(Index As Integer)
' Markiert den Text auf den entweder mit der Maud geklickt oder mit der
' Tab-Taste angewählt wurde.

    txtPort.Item(Index).SelStart = 0
    txtPort.Item(Index).SelLength = Len(txtPort.Item(Index).text)
    txtPort.Item(Index).SetFocus
End Sub
```

Die Routinen des Moduls RegistryModul.bas

Routine GetProfile

```
Public Function GetProfile(what As Integer, place As Integer) As String
' Liest ein durch 'place' und 'what' adressiertes Benutzerdatum aus der Registry aus.
    Dim BackString As String

    Select Case what
        Case 0: BackString = "Name"
        Case 1: BackString = "WindowType"
        Case 2: BackString = "eMail"
    End Select

    GetProfile = GetSetting("Server", "User" & place, BackString)
End Function
```

Routine SaveProfile

```
Public Sub SaveProfile(user As String, pw As String, email As String, place As Integer)
' Schreibt alle Daten eines Benutzers in das durch 'place' zeigegende Konto
    SaveSetting "Server", "User" & place, "Name", user
    SaveSetting "Server", "User" & place, "WindowType", pw
    SaveSetting "Server", "User" & place, "eMail", email
End Sub
```

Die Routine des Moduls DisplayModul.bas

```
Public Sub UpdateStatus(vsStatus As String, LCD As Integer)
' Stellt den zu schreibenden Text in das jeweilige Display.
' Falls die Zeichenkette größer als 32 Zeichen ist, wird die Funktion
' 'Autoscroll' eingeschaltet, die ein durchlaufen des überlangen
' Textes veranlasst

    If LCD = 1 Then
        If Len(vsStatus) > 32 Then
            frmMain.LCD1.AutoScroll = True
        Else
            frmMain.LCD1.AutoScroll = False
        End If
        frmMain.LCD1.Caption = vsStatus
    ElseIf LCD = 2 Then
        If Len(vsStatus) > 32 Then
            frmMain.LCD2.AutoScroll = True
        Else
            frmMain.LCD2.AutoScroll = False
        End If
        frmMain.LCD2.Caption = vsStatus
    End If
End Sub
```

Die Routinen des Moduls `CommModul.bas`

Deklaration globaler Variablen

```
Public gUserConnected As Boolean
Public gReset As Boolean
Public gGetUser$
Const CR = vbCr + vbLf
Const CM = "Client Messages "
```

Routine `ProcessData`

```
Public Sub ProcessData(vsString As String, viConnection As Integer)
' Auswertung des empfangenen Datenpakets vom Client kommend. Ist bereits ein
' Benutzer angemeldet, so wird die Anmelderoutine übersprungen, solange bis
' die Verbindung abgebaut wird.
' Bei einem Anmeldeversuch werden Benutzernamen und -passwort mit denen in der
' Registry vorhandenen verglichen und bei einer Übereinstimmung wird der
' betreffende Benutzer angemeldet. Gibt es keine Übereinstimmung, wird eine
' Fehlermeldung an den Client geschickt.
Dim i As Integer
Dim sCommand As String
Dim sInstruction As String
Dim sData As String
Dim bTemp As Boolean
Dim Login() As String
Dim guseritem As Integer
If gUserConnected = False Then
      If Left(vsString, 11) = "[AUTHORIZE]" Then
         vsString = Mid(vsString, 12, Len(vsString))
         Login = Split(vsString, ",", -1)
         For guseritem = 1 To 16
             DoEvents
             gGetUser = GetProfile(0, guseritem)
             If Login(0) = GetProfile(2, guseritem) Then
                 gUserConnected = True
                 SendToClient "[MAILER]Sending mail...."
                 UpdateStatus "Sending mail....", 2
                 SendIt 2, GetProfile(2, guseritem), gGetUser, _
                 PasswordDecode(GetProfile(1, guseritem))
                 SendToClient "[DISCONNECT]Mail sent"
                 Exit For
             End If
             If LCase(Login(0)) = LCase(gGetUser) And _
             Login(1) = PasswordDecode(GetProfile(1, guseritem)) Then
                 gUserConnected = True
                 SendToClient "[AUTHORIZE]"
                 frmMain.Led1.LedColor = Green
                 frmMain.frmClient.Caption = CM + "- Connected to " + frmMain.sckConnect.RemoteHostIP _
                     & "(User: " & gGetUser & ")"
                 UpdateStatus "Connected to " + frmMain.sckConnect.RemoteHostIP, 2
                 LoadRoomConfig
                 Protocol "Server", "User " & gGetUser & " logged in"
                 Exit For
              End If
         Next
         If gUserConnected = False Then
             SendToClient "[ERROR]Unknown User or Password"
             gUserConnected = True
         End If
     Else
         SendToClient ("[ERROR]Unknown Command! Login first!")
     End If
Else
     Do While InStr(1, vsString, vbCrLf)
         sCommand = Mid(vsString, 1, InStr(1, vsString, vbCrLf) - 1)
         sInstruction = Mid(sCommand, 1, InStr(1, sCommand, "]"))
         sData = Mid(sCommand, InStr(1, sCommand, "]") + 1, Len(sCommand))
         i = 1
          Select Case sInstruction

             Case "[DISCONNECT]"
                 SendToClient "[DISCONNECT]"
                 gUserConnected = True

             Case "[RESET]"
                 Sende 63
                 gReset = True

             Case "[MOTOR1]"
                 If LCase(sData) = "left" Then Sende 58
                 If LCase(sData) = "right" Then Sende 59
```

```
            Case "[MOTOR2]"
                If LCase(sData) = "left" Then Sende 57
                If LCase(sData) = "right" Then Sende 56

            Case "[SWITCH1]"
                If LCase(sData) = "on" Then Sende 48
                If LCase(sData) = "off" Then Sende 49

            Case "[SWITCH2]"
                If LCase(sData) = "on" Then Sende 50
                If LCase(sData) = "off" Then Sende 51

            Case "[SWITCH3]"
                If LCase(sData) = "on" Then Sende 52
                If LCase(sData) = "off" Then Sende 53

            Case "[SWITCH4]"
                If LCase(sData) = "on" Then Sende 54
                If LCase(sData) = "off" Then Sende 55

            Case "[ERROR]"
                i = 2
                vsString = Left(vsString, 7)

            Case Else
            i = 0
            MsgBox "Unknown message: " & vbCrLf & vsStatus
            SendToClient "Unknown Command."
          End Select
          Protocol "Client", sInstruction & sData
          If Not Left(vsString, 12) = "[DISCONNECT]" Then UpdateStatus ReplaceLetters(vsString), i
          vsString = Mid(vsString, InStr(1, vsString, vbCrLf) + 2, Len(vsString))
    Loop
End If
End Sub
```

Routine SendToClient

```
Public Sub SendToClient(vsData As String)
' Versendung eines Datenpakets an den Client
        If frmMain.sckConnect.State = 7 Then
            frmMain.sckConnect.SendData (vsData & vbCrLf)
        End If
End Sub
```

Routine Sende

```
Public Sub Sende(Zeichen As Integer)
' Versendung eines Bytewortes an die Controllereinheit
    If frmMain.MSComm1.PortOpen = True Then
        frmMain.MSComm1.Output = Chr$(Zeichen)
    End If
End Sub
```

Routine Din

```
Public Function Din() As String
' Schickt das Zeichen '=' zur Controllereinheit, um den Status der
' einzelnen Komponenten zu erhalten
    Sende 61
    t = Timer
    Do
    Loop Until (t + 0.02) < Timer
    Din = Empfang()
End Function
```

Routine Temp

```
Public Function Temp() As String
' Schickt das Zeichen '>' zur Controllereinheit, um die Temperatur
' zu erhalten    Sende 62
    t = Timer
    Do
    Loop Until (t + 0.02) < Timer
    Temp = Empfang()
End Function
```

Routine `Empfang`

```
Public Function Empfang() As String
' Überprüft, ob sich Zeichen im Empfangspuffer des COM-Ports befinden
' liest und übergibt sie gegebenenfalls an die aufrufende Funktion
    t = Timer
    Do
    Loop Until ((t + 0.05) < Timer) Or (frmMain.MSComm1.InBufferCount > 0)
    If frmMain.MSComm1.InBufferCount > 0 Then
        Empfang = Asc(frmMain.MSComm1.Input)
    End If
End Function
```

Die Routinen des Moduls `FileHandlingModul.bas`

Deklaration globaler Variablen

```
Dim file$, F, fso
Const ForReading = 1, ForWriting = 2, ForAppending = 8
```

Routine `Protocol`

```
Public Sub Protocol(user As String, action As String, Optional firstRun As Boolean)
' Erstellt gegebenfalls eine neue Protokolldatei (abhängig vom Datum) mit neuem Header.
' Zahlt für unterschiedlich lange Meldungen die Abstände der Spalten aus, um
' ansehnlich formatierten Text zu erhalten
    Dim eimer1 As String
    Dim i As Integer

    Set fso = CreateObject("Scripting.FileSystemObject")
    file = App.Path & "\" & Date & "  - Protocol.log"

    If fso.FileExists(file) = False Then
        Set F = fso.OpenTextFile(file, ForAppending, True)
        F.Write "Date:        Time:       User:           Action:                  " + vbCrLf
        For i = 1 To 80
            eimer1 = eimer1 + "-"
        Next
        F.Write eimer1 + vbCrLf
        eimer1 = ""
        WriteProtocol user, action
    Else
        Set F = fso.OpenTextFile(file, ForAppending, True)
        WriteProtocol user, action
    End If
    F.Close
End Sub
```

Routine `WriteProtocol`

```
Private Sub WriteProtocol(user$, action$)
' Die eigentliche Schreibroutine
    Dim i%, eimer$(2)

    For i = Len(user) To 14
        eimer(1) = eimer(1) + " "
    Next
    For i = Len(action) To 41
        eimer(2) = eimer(2) + " "
    Next
    F.Write Date & "  " & Mid(Time(), 1, 5) & "       " & user & eimer(1) & _
    action & eimer(2) & vbCrLf
End Sub
```

Die Routinen des Moduls `ConfigModul.bas`

Deklaration globaler Variablen

```
Public contCount As Integer
```

Routine `LoadRoomConfig`

```
Public Sub LoadRoomConfig()
' Laden mit folgendem Versenden der Buttonbeschriftungen aus der 'Config.ini'
    Dim i As Integer
    Dim t As Single

    INIFileName = App.Path & "\Config.ini"
    For i = 1 To 6
        SendToClient ("[LABEL_PORT" & i & "]" & INILesen("Port" & i, "RoomConfiguration"))
        Protocol "Server", CStr("[LABEL_PORT" & i & "]" & INILesen("Port" & i, "RoomConfiguration"))
    Next
End Sub
```

Routine `CreateUser`

```
Public Sub CreateUser_ini()
' Erstellen des 16 Benutzerkonten in der Registry mit den Einträgen 'leer'
    Dim i As Integer
    For i = 1 To 16
       SaveProfile "leer", "leer", "leer", i
    Next
End Sub
```

Routine `LoadUser_ini`

```
Public Sub LoadUser_ini()
' Laden der 16 Benutzerkonten, falls der Menüpunkt 'User...' angeklickt
' wurde
    Dim contCount As Integer
    Dim user As String

    For contCount = 1 To 16
        user = GetProfile(0, contCount)
        If user = "leer" Then Exit For
        frmUser.lstUser.AddItem user
    Next

    gUser = frmUser.lstUser.ListCount
    frmUser.Caption = "User: " & frmUser.lstUser.ListCount
End Sub
```

Routine `Load_MailSettings`

```
Public Sub Load_MailSettings()
' Laden der Einstellungen für das 'Maileinstellungs'- Fenster
    If INILesen("Server", "SendMail") = "" Then Create_MailSettings
    frmMailSettings.txtSMTP.text = INILesen("Server", "SendMail")
    If INILesen("ShowWindow", "SendMail") = "True" Then
        frmMailSettings.chkShow.Value = 1
        frmMailSettings.chkHide.Enabled = True
    Else
        frmMailSettings.chkShow.Value = 0
        frmMailSettings.chkHide.Enabled = False
    End If

    If INILesen("HideWindow", "SendMail") = "True" Then
        frmMailSettings.chkHide.Value = 1
    Else
        frmMailSettings.chkHide.Value = 0
    End If

    If INILesen("StartUp", "SendMail") = "DontMail" Then
        frmMailSettings.chkSendIP.Value = 0
    Else
        frmMailSettings.chkSendIP.Value = 1
    End If
End Sub
```

Routine `Create_MailSettings`

```
Private Sub Create_MailSettings()
    INISchreiben "Server", "smtprelay.t-online.de", "SendMail"
    INISchreiben "ShowWindow", "True", "SendMail"
    INISchreiben "HideWindow", "False", "SendMail"
    INISchreiben "StartUp", "DontMail", "SendMail"
End Sub
```

Die Routinen des Moduls `GetDynamicIP.bas`

Deklaration globaler Variablen, Konstanten und Funktionen

```
' Da dieses Modul von jemand anderem stammt, wird hier nur kurz die Funktion beschrieben. Es
' wurde so abgeändert, daß es in unsere Programme eingebunden werden kann.
'
' Das Modul "GetDynamicIPModule" hat die Aufgabe bei einer bestehenden Verbindung zum Internet
' die durch den Provider vergebenen IP zu ermitteln. Falls eine dynamische IP vorhanden ist, so
' wird sie in die Statuszeile eingetragen. Andernfalls erscheint an dieser Stelle ein "offline"
' und ein Hinweistext.
'
'
' Quelle : http://www.reinecke-krueger.de, Active-VB Forum

Option Explicit
Private Declare Function WSAGetLastError Lib "WSOCK32.DLL" () _
As Long

Private Declare Function WSAStartup Lib "WSOCK32.DLL" (ByVal _
wVersionRequired&, lpWSAData As WinSocketDataType) _
As Long

Private Declare Function WSACleanup Lib "WSOCK32.DLL" () _
As Long

Private Declare Function gethostname Lib "WSOCK32.DLL" (ByVal _
HostName$, ByVal HostLen%) As Long

Private Declare Function gethostbyname Lib "WSOCK32.DLL" _
(ByVal HostName$) As Long

Private Declare Function gethostbyaddr Lib "WSOCK32.DLL" _
(ByVal addr$, ByVal laenge%, ByVal typ%) As Long

Private Declare Sub RtlMoveMemory Lib "kernel32" (hpvDest As _
Any, ByVal hpvSource&, ByVal cbCopy&)

Const WS_VERSION_REQD = &H101
Const WS_VERSION_MAJOR = WS_VERSION_REQD \ &H100 And &HFF&
Const WS_VERSION_MINOR = WS_VERSION_REQD And &HFF&
Const MIN_SOCKETS_REQD = 1
Const SOCKET_ERROR = -1
Const WSADescription_Len = 256
Const WSASYS_Status_Len = 128

Private Type HostDeType
    hName As Long
    hAliases As Long
    hAddrType As Integer
    hLength As Integer
    hAddrList As Long
End Type

Private Type WinSocketDataType
    wversion As Integer
    wHighVersion As Integer
    szDescription(0 To WSADescription_Len) As Byte
    szSystemStatus(0 To WSASYS_Status_Len) As Byte
    iMaxSockets As Integer
    iMaxUdpDg As Integer
    lpszVendorInfo As Long
End Type

Private Declare Function RasEnumConnections Lib "RasApi32.DLL" _
Alias "RasEnumConnectionsA" (lpRasCon As Any, lpcb As _
Long, lpcConnections As Long) As Long

Private Declare Function RasGetConnectStatus Lib "RasApi32.DLL" _
Alias "RasGetConnectStatusA" (ByVal hRasCon As Long, _
```

```
lpStatus As Any) As Long

Const RAS_MaxEntryName = 256
Const RAS_MaxDeviceType = 16
Const RAS_MaxDeviceName = 32

Private Type RASType
    dwSize As Long
    hRasCon As Long
    szEntryName(RAS_MaxEntryName) As Byte
    szDeviceType(RAS_MaxDeviceType) As Byte
    szDeviceName(RAS_MaxDeviceName) As Byte
End Type

Private Type RASStatusType
    dwSize As Long
    RasConnState As Long
    dwError As Long
    szDeviceType(RAS_MaxDeviceType) As Byte
    szDeviceName(RAS_MaxDeviceName) As Byte
End Type

Dim OnlineFlag As Boolean
```

Routine DFÜStatus

```
Private Function DFÜStatus() As Boolean
    Dim RAS(255) As RASType, RASStatus As RASStatusType
    Dim lg&, lpcon&, Result&

    RAS(0).dwSize = 412
    lg = 256 * RAS(0).dwSize
    Result = RasEnumConnections(RAS(0), lg, lpcon)

    If lpcon = 0 Then
        DFÜStatus = False
    Else
        RASStatus.dwSize = 160
        Result = RasGetConnectStatus(RAS(0).hRasCon, RASStatus)

        If RASStatus.RasConnState = &H2000 Then
            DFÜStatus = True
        Else
            DFÜStatus = False
        End If
    End If
End Function
```

Routine Online

```
Private Function Online() As Boolean
    Dim Test As Boolean

    Test = DFÜStatus
    If Test = False Then MsgBox ("Keine Online Verbindung vorhanden ! Bitte einwählen !")
    Online = Test
End Function
```

Routine GetDynIP

```
Public Sub GetDynIP()
    Static LastOnline As Boolean
    OnlineFlag = DFÜStatus
    If OnlineFlag And Not LastOnline Then
        Call GetIt
    ElseIf Not OnlineFlag Then
        frmMain.StatusBar1.Panels(2) = "Offline"
    End If
    LastOnline = OnlineFlag
End Sub
```

Routine CleanSockets

```
Private Sub CleanSockets()
    Dim Result&
    Result = WSACleanup()
    If Result <> 0 Then
        MsgBox ("Socket Error " & Trim$(Str$(Result)) & _
        " in Prozedur 'CleanSockets' aufgetreten !")
        End
    End If
End Sub
```

Routine MyHostName

```
Private Function MyHostName() As String
    Dim HostName As String * 256

    If gethostname(HostName, 256) = SOCKET_ERROR Then
        MsgBox "Windows Sockets error " & Str(WSAGetLastError())
        Exit Function
    Else
        MyHostName = NextChar(Trim$(HostName), Chr$(0))
    End If
End Function
```

Routine GetIt

```
Private Sub GetIt()
    Dim X%
    Dim IP$, DNS$, HOST$

    If Not Online Then Exit Sub
    InitSockets
    HOST = MyHostName$()

    Do
        IP = HostByName$(HOST, X)
        If Len(IP) = 0 Then Exit Do
        If Left$(IP, 4) <> "192." Then
            frmMain.StatusBar1.Panels(2) = "Own IP: " & IP
        End If
        X = X + 1
    Loop

    CleanSockets
End Sub
```

Routine HostByName

```
Private Function HostByName(Name$, Optional X% = 0) As String
    Dim MemIp() As Byte
    Dim Y%
    Dim HostDeAddress&, HostIp&
    Dim IpAddress$
    Dim HOST As HostDeType

    HostDeAddress = gethostbyname(Name)
    If HostDeAddress = 0 Then
        HostByName = ""
        Exit Function
    End If
    Call RtlMoveMemory(HOST, HostDeAddress, LenB(HOST))
    For Y = 0 To X
        Call RtlMoveMemory(HostIp, HOST.hAddrList + 4 * Y, 4)
        If HostIp = 0 Then
            HostByName = ""
            Exit Function
        End If
    Next Y
    ReDim MemIp(1 To HOST.hLength)
    Call RtlMoveMemory(MemIp(1), HostIp, HOST.hLength)
    IpAddress = ""
    For Y = 1 To HOST.hLength
        IpAddress = IpAddress & MemIp(Y) & "."
    Next Y
    IpAddress = Left$(IpAddress, Len(IpAddress) - 1)
    HostByName = IpAddress
End Function
```

Routine InitSocket

```
Private Sub InitSockets()
    Dim Result%
    Dim LoBy%, HiBy%
    Dim SocketData As WinSocketDataType
    Result = WSAStartup(WS_VERSION_REQD, SocketData)
    If Result <> 0 Then
        MsgBox ("'winsock.dll' antwortet nicht !")
        End
    End If
End Sub
```

Routine NextChar

```
Private Function NextChar(text$, Char$) As String
    Dim POS%
    POS = InStr(1, text, Char)
    If POS = 0 Then
        NextChar = text
        text = ""
    Else
        NextChar = Left$(text, POS - 1)
        text = Mid$(text, POS + Len(Char))
    End If
End Function
```

Die Routinen des Moduls Key.bas

Deklaration globaler Variablen

```
Const key$ = "Fischers Fritze fischt frische Fische."
```

Routine PasswordEncode

```
Public Function PasswordEncode(plainPass$) As String
' Verschlüsselt das Passwort mit dem oben angegebenen Schlüssel.
' Schlüssel und Passwort werden zeichenweise mit der XOR-Methode verknüpft
    Dim shortkey$, Pass$, hexPass$, passLength%, point16%, point1%, i%

    passLength = Len(plainPass)
    shortkey = Left(key, passLength)
    For i = 1 To passLength
        Pass = Chr(Asc(Mid(plainPass, i, 1)) Xor Asc(Mid(shortkey, i, 1)))
        point16 = Cut(Asc(Pass) / 16)
        point1 = Asc(Pass) Mod 16
        hexPass = hexPass & ConvertCharToHex(point16) & ConvertCharToHex(point1)
    Next
    PasswordEncode = hexPass
End Function
```

Routine PasswordDecode

```
Public Function PasswordDecode(encPass$) As String
' Entschlüsselt das Passwort mit dem oben angegebenen Schlüssel,
' ebenfalls zeichenweise.
    Dim shortkey$, plainPass$, passLength%, i%, hexPass$, Pass%
    If Not encPass = "leer" Then
        passLength = Len(encPass) / 2
        shortkey = Left(key, passLength)
        For i = 1 To passLength
            hexPass = Mid(encPass, i * 2 - 1, 2)
            Pass = ConvertHexToChar(hexPass)
            plainPass = plainPass & Chr(Pass Xor Asc(Mid(shortkey, i, 1)))
        Next
        PasswordDecode = plainPass
    End If
End Function
```

Routine ConvertHexToChar

```
Private Function ConvertHexToChar(hex$) As Integer
' Bei der Entschlüsselung wird die Zeichenkette aus der Registry
' gelesen. Die Zeichenkette besteht hier aus mehrere zweistelligen Hexzahlen.
' Hier werden diese Hexzahlen in ihre zugehörigen ASCII-Zeichen
' umgewandelt
    Dim i%, point(4)
    point(1) = Left(hex, 1)
    point(2) = Right(hex, 1)
    For i = 1 To 2
        If Asc(point(i)) > 64 Then
            point(i + 2) = CStr(Asc(point(i)) - 55)
        Else
            point(i + 2) = point(i)
        End If
    Next

    ConvertHexToChar = CInt((CDbl(point(3)) + CDbl(point(4) / 16)) * 16)
End Function
```

Routine ConvertCharToHex

```
Private Function ConvertCharToHex(Char%) As String
' Beim Verschlüsseln des Passwort entstehen durch die XOR-Methode
' wirre Zeichenketten mit Zeichen aus dem ASCII-Zeichensatz.
' Die ASCII-Hexzahlen jedes einzelnen Zeichens werden hier
' gewonnen und anschliessend abgespeichert
    If Char > 9 Then
        ConvertCharToHex = Chr(Char + 55)
    Else
        ConvertCharToHex = CStr(Char)
    End If
End Function
```

Routine Cut

```
Private Function Cut(figure As String) As Integer
' Schneidet bei der Umwandlung von ASCII-Zeichen in HEX-Code, die durch die Division
' entstehenden Dezimalstellen ab
    Dim feld
    feld = Split(figure, ",")
    Cut = feld(0)
End Function
```

Routine ReplaceLetters

```
Public Function ReplaceLetters(word As String) As String
' Ersetzt 'Ä, Ö, Ü, ä, ö, ü und ß' durch Zeichen, die im amerikanischen 7-Bit-ASCII-Code
' enthalten sind.
    Dim replacedword As String
    Dim i As Integer
    Dim letter As String

    For i = 1 To Len(word)
        letter = Mid(word, i, 1)
        Select Case (letter)
            Case "ä":  replacedword = replacedword + "ae"
            Case "ö":  replacedword = replacedword + "oe"
            Case "ü":  replacedword = replacedword + "ue"
            Case "Ä":  replacedword = replacedword + "Ae"
            Case "Ü":  replacedword = replacedword + "Ue"
            Case "Ö":  replacedword = replacedword + "Oe"
            Case "ß":  replacedword = replacedword + "ss"
            Case Else: replacedword = replacedword + letter
        End Select
    Next
    ReplaceLetters = replacedword
End Function
```

Die Routinen des Moduls `SendMailModul.bas`

Deklaration globaler Variablen

```
' Da dieses Modul bis auf die Routine 'FListProtocol' von jemand anderem stammt, wird hier nur kurz
' die Funktion beschrieben. Es wurde so abgeändert, daß es in unsere Programme eingebunden werden kann.
'
' Das Modul "SendMailModul" versendet entweder die dynmaische IP an alle registriertem Benutzer,
' oder die Logindaten einens einzelnen Benutzers an seine eMail-Adresse.
'
'
' Quelle : http://www.reinecke-krueger.de, Active-VB Forum
Dim body$, Mail$, server$, j%
Public sec%, Result$, gSendToUser As Boolean
```

Routine `SendIt`

```
Public Sub SendIt(mode%, Optional email$, Optional Login$, Optional pw$)
    INIFileName = App.Path & "\Config.ini"
    server = INILesen("Server", "SendMail")
    j = 0

    If mode = 1 Then
      If SendMailtoAll Then
          frmSendMail.List1.AddItem ("Email erfolgreich verschickt")
      Else
        frmSendMail.List1.AddItem ("Fehler beim Versenden aufgetreten")
      End If
    ElseIf mode = 2 Then
       gSendToUser = True
       If SendMail(email, Login, pw) Then Protocol "Server", "Mail to User " & _
       Login & " (" & email & ") was sent."
       UpdateStatus "Mail sent", 2
    End If

    j = 0
End Sub
```

Routine `Response`

```
Private Function Response(RCode$) As Boolean
  frmSendMail.Timer2.Enabled = True
  TimeOut = 1
  sec = 0
  Response = True

  Do While Left$(Result, 3) <> RCode
    DoEvents
    If sec > TimeOut * 5 Then
      If Len(Result) Then
        frmSendMail.List1.AddItem ("SMTP Error! Falscher Rückgabewert")
        FListProtocol "Receiving: SMTP Error! Wrong type of returned data" & vbCrLf, 3
      Else
        frmSendMail.List1.AddItem ("SMTP Error! Time out")
        FListProtocol "Failure: SMTP Error! Time out" & vbCrLf, 3
      End If
      Response = False
      Exit Function
    End If
  Loop
  FListProtocol "Receiving: " & Result, 1


sec = 0
  Result = ""
End Function
```

Routine `SendMailtoAll`

```
Private Function SendMailtoAll() As Boolean
  Dim outTO$, outFR$, i%, email$

    If frmSendMail.SendMailSock.State = sckClosed Then
      On Error GoTo ERRORMail
        body = "This eMail was created by HomeControl Server and was sent at" & vbCrLf
        body = body & Time & " on " & Date & vbCrLf
        body = body & "The Server currently has the IP: " & Mid(frmMain.StatusBar1.Panels(2).text, 9)
        Mail = "From: HomeControl Server Mailer" & " <olbob@t-online.de>"
        Mail = Mail & vbCrLf & "Date: " & Format(Date, "Ddd")
        Mail = Mail & ", " & Format(Date, "dd Mmm YYYY") & " "
        Mail = Mail & Format(Time, "hh:mm:ss") & " +0100" & vbCrLf
```

```
          Mail = Mail & "X-Mailer: HomeControl Server Mailer"
          Mail = Mail & vbCrLf & "To: All User registered at HomeControl <allUsers@HomeControl.de>"
          Mail = Mail & vbCrLf & "Subject: Current IP" & vbCrLf
          Mail = Mail & vbCrLf & body & vbCrLf & "." & vbCrLf


        frmSendMail.SendMailSock.LocalPort = 0
        outFR = "mail from: olbob@t-online.de" & vbCrLf

        frmSendMail.SendMailSock.Connect server, 25
        If Not Response("220") Then GoTo ERRORMail

        frmSendMail.ProgressBar1.Value = frmSendMail.ProgressBar1.Value + 1
        FListProtocol "Sending: HELO homecontrolserver.de" & vbCrLf, 2
        frmSendMail.SendMailSock.SendData ("HELO homecontrolserver.de" & vbCrLf)
        If Not Response("250") Then GoTo ERRORMail

        frmSendMail.ProgressBar1.Value = frmSendMail.ProgressBar1.Value + 1
        FListProtocol "Sending: " & outFR, 2
        frmSendMail.SendMailSock.SendData (outFR)
        If Not Response("250") Then GoTo ERRORMail

        For i = 1 To 16
          email$ = GetProfile(2, i)
          If email = "leer" Then Exit For
          outTO = "rcpt to: " & email & vbCrLf
          frmSendMail.SendMailSock.SendData (outTO)
          FListProtocol "Sending: " & outTO, 2
          If Not Response("250") Then GoTo ERRORMail
        Next

        frmSendMail.ProgressBar1.Value = frmSendMail.ProgressBar1.Value + 1
        FListProtocol "Sending: data" & vbCrLf, 2
        frmSendMail.SendMailSock.SendData "data" & vbCrLf
        If Not Response("354") Then GoTo ERRORMail

        frmSendMail.ProgressBar1.Value = frmSendMail.ProgressBar1.Value + 1
        frmSendMail.SendMailSock.SendData (Mail)
        FListProtocol Mail, 2
        If Not Response("250") Then GoTo ERRORMail

        frmSendMail.ProgressBar1.Value = frmSendMail.ProgressBar1.Value + 1
        FListProtocol "Sending: quit" & vbCrLf, 2
        frmSendMail.SendMailSock.SendData ("quit" & vbCrLf)
        If Not Response("221") Then GoTo ERRORMail
        SendMailtoAll = True
      End If

      If INILesen("HideWindow", "SendMail") = "True" Then
          frmSendMail.Hide
          frmSendMail.ProgressBar1.Value = 0
          frmSendMail.List1.Clear
          Set frmSendMail = Nothing
      End If

ERRORMail:
      frmSendMail.SendMailSock.Close
      If Err.Number > 0 Then
          FListProtocol "Failure: " & Err.Description & vbCrLf, 3
          Protocol "SendMail", "An error occured while sending mail to all"
      End If
      frmSendMail.Timer2.Enabled = False
End Function
```

Routine `FListProtocol`

```
Private Sub FListProtocol(text$, Optional mode%)
' Schreibst die Fehler, Befehle und Bestätigungen in das 'SendMail'- Fenster.
' Je nachdem um welche Nachricht es sich handelt, wird diese mit einer anderen
' Hintergrundfarbe in das Fenster geschrieben.
    Dim h%, eimer$
    h = 0
    eimer = ""

     While Len(text)
        h = InStr(text, vbCrLf)
        eimer = Left(text, h - 1)
        frmSendMail.List1.AddItem eimer
        frmSendMail.flex1.AddItem eimer
        frmSendMail.flex1.Row = j
        If mode = 1 Then
            frmSendMail.flex1.CellBackColor = &HEEEEFF
        ElseIf mode = 2 Then
            frmSendMail.flex1.CellBackColor = &HEAFFEA
        ElseIf mode = 3 Then
```

```
            frmSendMail.flex1.CellBackColor = &HFFFFC0
        End If
        j = j + 1
        text = Right(text, Len(text) - (h + 1))
    Wend
End Sub
```

Routine `SendMail`

```
Private Function SendMail(email$, Login$, pw$) As Boolean
    Dim outTO$, outFR$, i%

    If frmSendMail.SendMailSock.State = sckClosed Then
      On Error GoTo ERRORMail

        body = "This eMail was created by HomeControl Server and was sent at" & vbCrLf
        body = body & Time & " on " & Date & vbCrLf
        body = body & "You have recently requested your login data. Here they are:"
        body = body & vbCrLf & vbCrLf & "Your login name: " & Login & vbCrLf
        body = body & "Your password: " & pw & vbCrLf & "Have fun."

        Mail = "From: HomeControl Server Mailer" & " <olbob@t-online.de>"
        Mail = Mail & vbCrLf & "Date: " & Format(Date, "Ddd")
        Mail = Mail & ", " & Format(Date, "dd Mmm YYYY") & " "
        Mail = Mail & Format(Time, "hh:mm:ss") & " +0100" & vbCrLf
        Mail = Mail & "X-Mailer: HomeControl Server Mailer"
        Mail = Mail & vbCrLf & "To: " & Login & "<" & email & ">"
        Mail = Mail & vbCrLf & "Subject: Your login data" & vbCrLf
        Mail = Mail & vbCrLf & body & vbCrLf & "." & vbCrLf



      frmSendMail.SendMailSock.LocalPort = 0
      outFR = "mail from: olbob@t-online.de" & vbCrLf

      frmSendMail.SendMailSock.Connect server, 25
      If Not Response("220") Then GoTo ERRORMail

      frmSendMail.SendMailSock.SendData ("HELO homecontrolserver.de" & vbCrLf)
      If Not Response("250") Then GoTo ERRORMail

      frmSendMail.SendMailSock.SendData (outFR)
      If Not Response("250") Then GoTo ERRORMail

      outTO = "rcpt to: " & email & vbCrLf
      frmSendMail.SendMailSock.SendData (outTO)
      If Not Response("250") Then GoTo ERRORMail

      frmSendMail.SendMailSock.SendData "data" & vbCrLf
      If Not Response("354") Then GoTo ERRORMail

      frmSendMail.SendMailSock.SendData (Mail)
      If Not Response("250") Then GoTo ERRORMail

      frmSendMail.SendMailSock.SendData ("quit" & vbCrLf)
      If Not Response("221") Then GoTo ERRORMail
      SendMail = True
      gSendToUser = False

    End If

ERRORMail:
    frmSendMail.SendMailSock.Close
    If Err.Number > 0 Then
        FListProtocol "Failure: " & Err.Description & vbCrLf, 3
        SendToClient "[MAILER]An error occured while sending mail"
        Protocol "SendMail", "An error occured while sending mail to " & Login & "."
    Else
        SendToClient "[MAILER]Mail sent"
    End If
    frmSendMail.flex1.Clear
    frmSendMail.ProgressBar1.Value = 0
    Set frmSendMail = Nothing
    Exit Function
End Function
```

Routine `StatusTimer` des Moduls `StatusTimerModul.bas`

```
Public gtimercount As Integer

Public Sub StatusTimer()
' Die Routine wird, bei Verbindung zur Controllereinheit, nach jeden 250ms durchlaufen und
' erfträgt den Status der Komponenten. Diese werden dann zu dem Client, falls angemeldet,
' übertragen. Bei jedem 8 Durchlauf wird die Temperatur abgefragt, ausgwertet, ob Feueralarm
' besteht und anschliessend zu dem angemeldeten Client übertragen.
    Dim D$, t$
    Dim S%, W%

    If Empfang = "78" Then
        SendToClient ("[STATE]Reset")
        Sende 97
    End If

    If gReset = False Or gUserConnected = True Then

       If frmMain.MSComm1.PortOpen = True Then
            SendToClient ("[STATE]UnitOn")
            D = Din()

            If D = "" Then Exit Sub


            If Not D = "69" Then

                    For n = 0 To 3
                        If (D And (2 ^ n)) = 0 Then
                            SendToClient ("[PORT" & n + 1 & "]off")
                        Else
                            SendToClient ("[PORT" & n + 1 & "]on")
                        End If
                    Next

                    S = D And 48
                    Select Case S
                        Case "0":  SendToClient ("[PORT6]left")
                        Case "16": SendToClient ("[PORT6]movingright")
                        Case "32": SendToClient ("[PORT6]movingleft")
                        Case "48": SendToClient ("[PORT6]right")
                    End Select

                    W = D And 192
                    Select Case W
                        Case "0":   SendToClient ("[PORT5]right")
                        Case "64":  SendToClient ("[PORT5]movingleft")
                        Case "128": SendToClient ("[PORT5]movingright")
                        Case "192": SendToClient ("[PORT5]left")
                    End Select

            Else
                SendToClient ("[ERROR]Error404")
            End If
        Else
            SendToClient ("[STATE]UnitOff")
        End If

        If gtimercount < 4 * 2 Then
            gtimercount = gtimercount + 1
        Else
            SendToClient ("[TIME]" & Left(Time(), 5))
            gtimercount = 0

            t = Temp()
            If t = "84" Then
                SendToClient ("[TEMP]Fire")
                frmMain.tmrStatus.Enabled = False
                frmMain.cmdConnectUnit.Enabled = False
                UpdateStatus "Fire alert", 1
                gFireAlert = True
                Protocol "Unit", "Fire alert"
            Else
                SendToClient ("[TEMP]" & t)
            End If
        End If
    End If
End Sub
```

14. Das von Server und Client gemeinsam benutzte Modul IOatINI.bas

Deklaration globaler Variablen

```
' Da dieses Modul von jemand anderem stammt, wird hier nur kurz die Funktion beschrieben. Es
' wurde so abgeändert, daß es in unsere Programme eingebunden werden kann.
'
' Das Modul "INIMODULE" ist für das Auslesen und das Verändern von INI-Dateien zuständig. In diesem Fall
' wird nur die "Config.ini" im Installationspfad des Servers durch dieses Modul bearbeitet.
'
'
' Quelle : http://www.vbwelt.de

Option Explicit
Private Declare Function GetPrivateProfileString Lib "kernel32" Alias "GetPrivateProfileStringA" (ByVal _
lpApplicationName As String, ByVal lpKeyName As Any, ByVal lpDefault As String, ByVal lpReturnedString _
As String, ByVal nSize As Long, ByVal lpFileName As String) As Long
Private Declare Function WritePrivateProfileString Lib "kernel32" Alias "WritePrivateProfileStringA" _
(ByVal lpApplicationName As String, ByVal lpKeyName As Any, ByVal lpString As Any, ByVal lpFileName As _
String) As Long

Public INIFileName As String
```

Routine `INILesen`

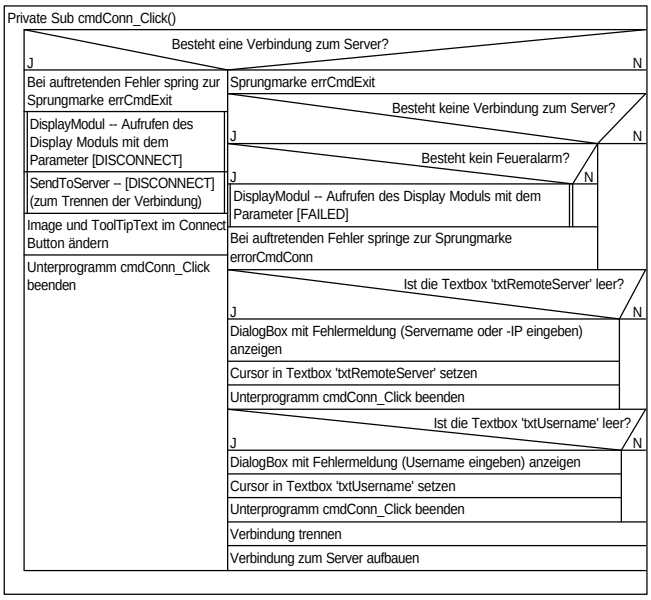
```
Public Function INILesen(key As String, Optional Paragraph As String = "Allgemein") As String
  If INIFileName = "" Then INIFileName = App.Path & "\" & App.Title & ".INI"
  INILesen = String(255, 0)
  If key <> "" Then Call GetPrivateProfileString(Paragraph, key, "", INILesen, 255&, INIFileName)
INILesen = Replace(INILesen, Chr(0), "")
End Function
```

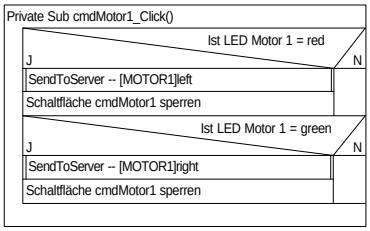
Routine `INISchreiben`

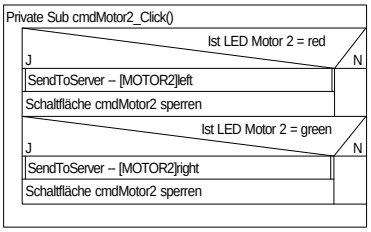
```
Public Sub INISchreiben(key As String, Optional Value As String = "", Optional Paragraph As String = _
"Allgemein")
  If INIFileName = "" Then INIFileName = App.Path & "\" & App.Title & ".INI"
  If key <> "" Then WritePrivateProfileString Paragraph, key, Value, INIFileName
End Sub
```

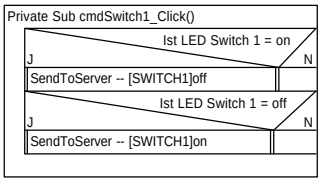
15. Struktogramme des Clients

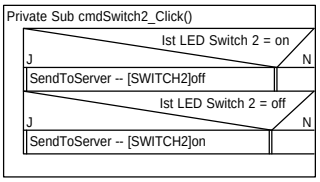
Die Routinen des Form `client.frm`

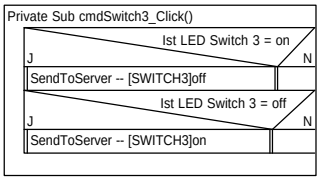


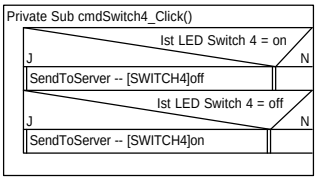


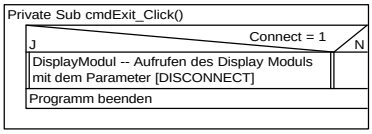


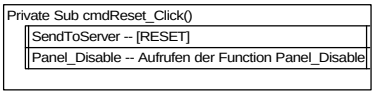


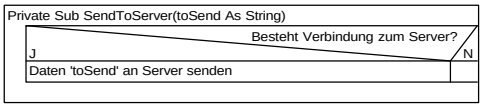


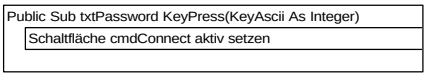


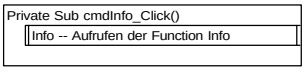


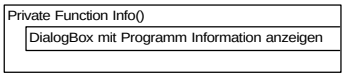


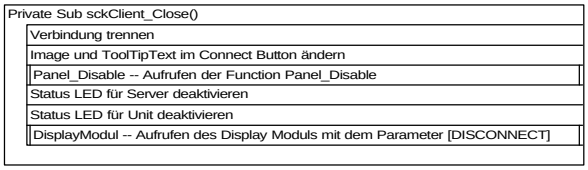


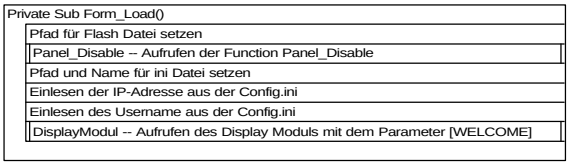


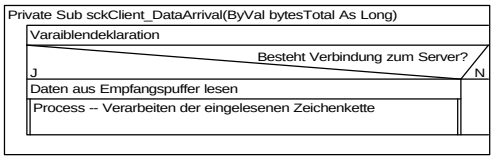


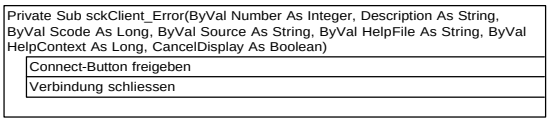


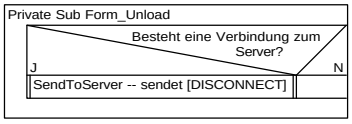


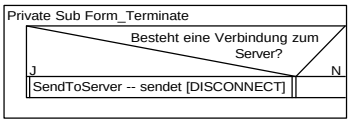


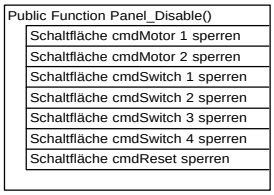


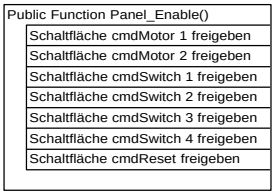




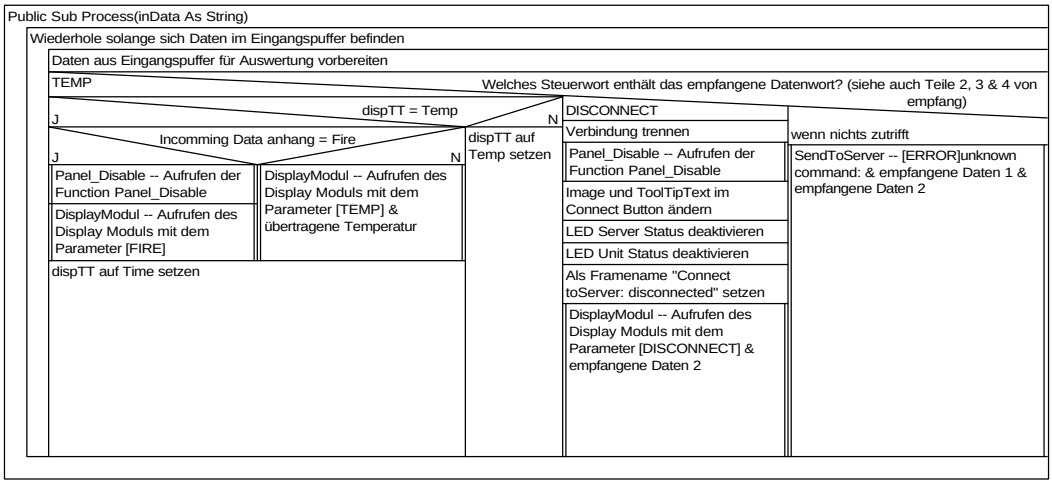


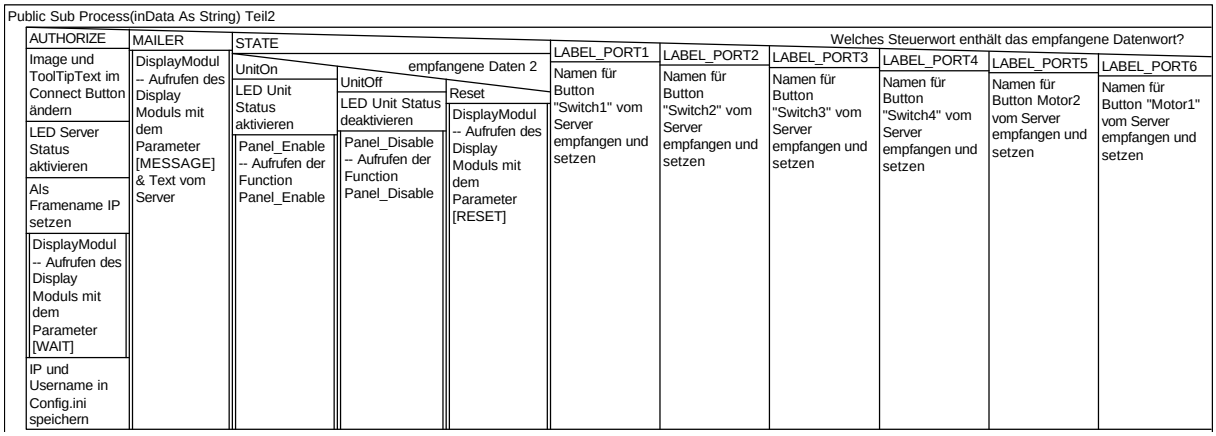


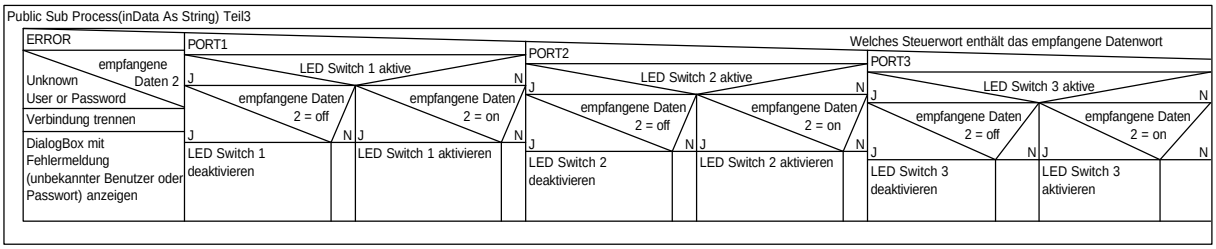


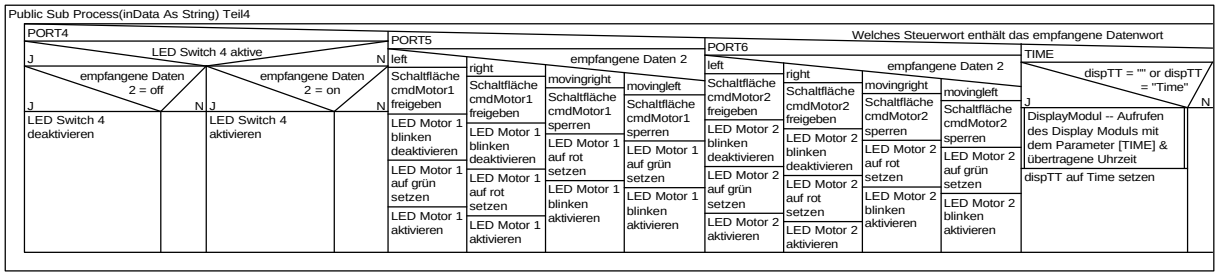


Die Routine `Process` des Moduls `Empfang.bas`

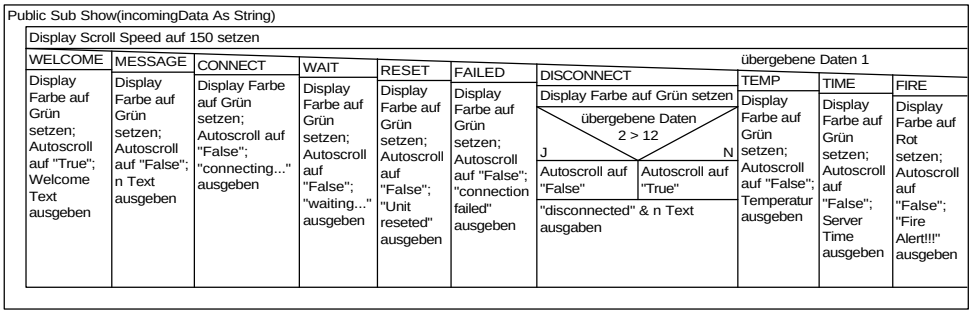






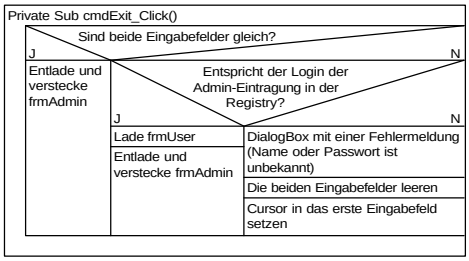


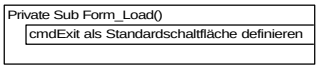
Die Routine `Show` des Moduls `Display.bas`



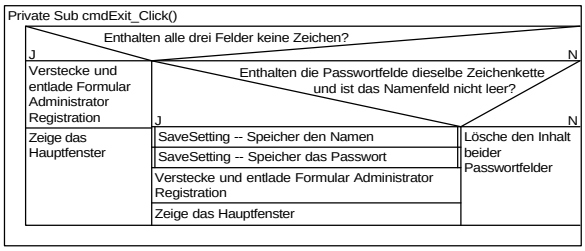
16. Struktogramme des Server

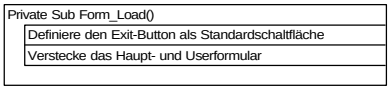
Die Routinen des Formulars `Admin.frm`

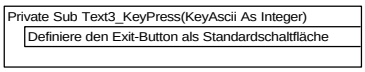




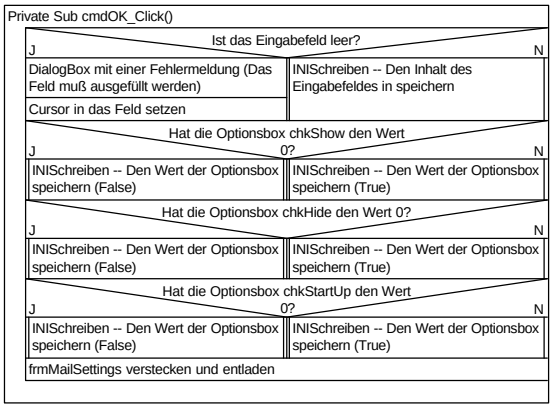
Die Routinen des Formulars `AdminRegi.frm`

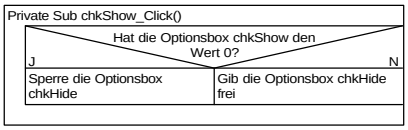


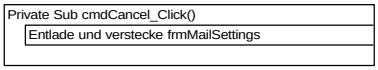


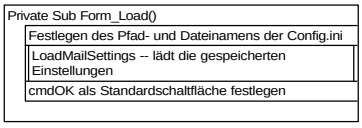


Die Routinen des Formulars `MailSettings.frm`

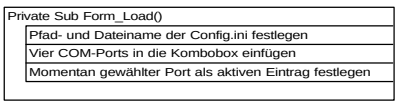


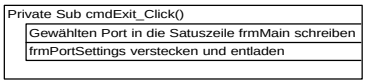


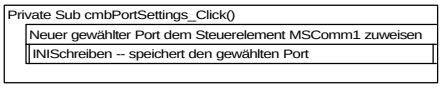




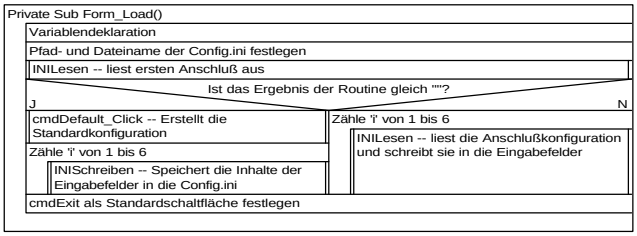
Die Routinen des Formulars `PortSettings.frm`

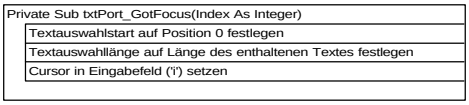


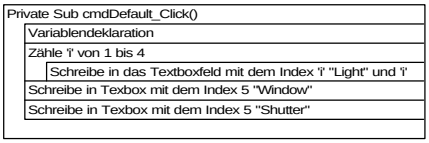


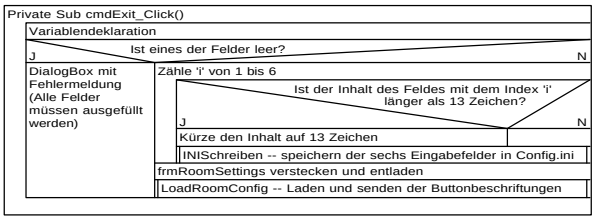


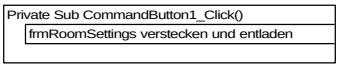
Die Routinen des Formulars `RoomSettings.frm`



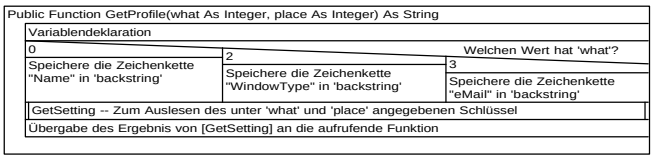


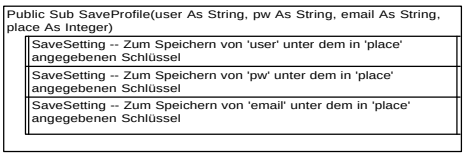


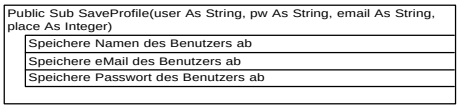




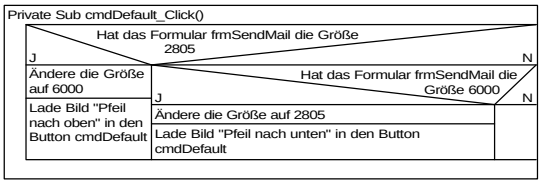
Die Routinen des Moduls `Registry.bas`

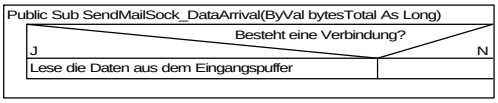


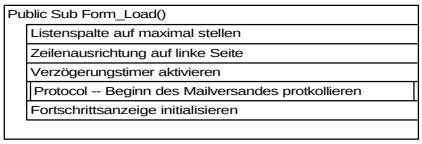


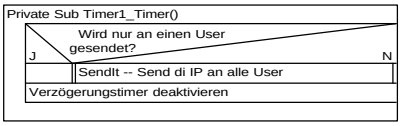


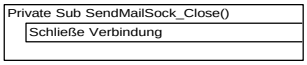
Die Routinen des Formulars `SendMail.frm`

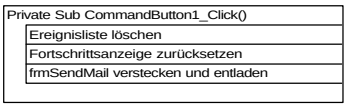


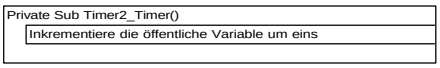




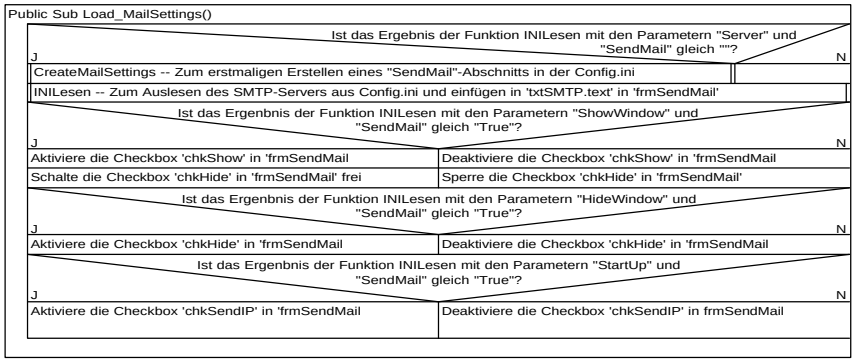


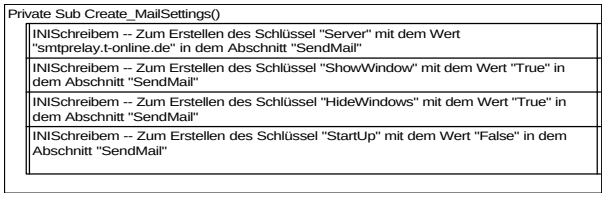


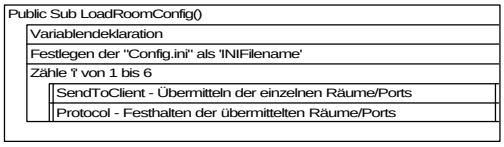


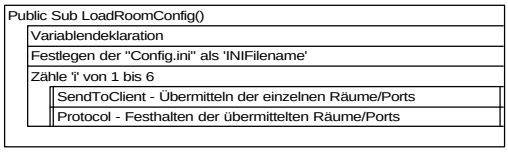


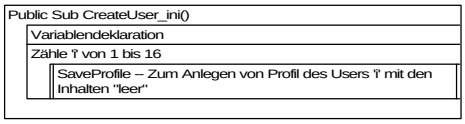
Die Routinen des Moduls `Config.bas`



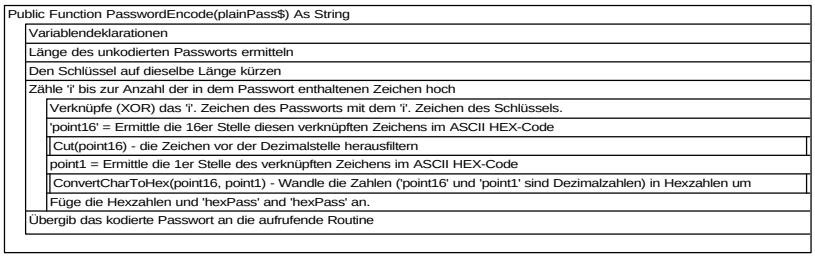


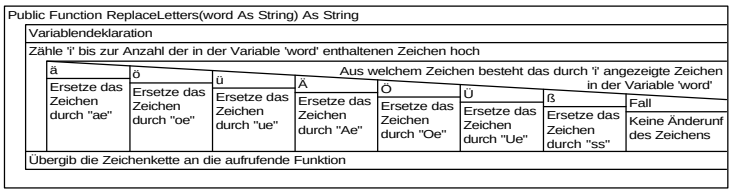


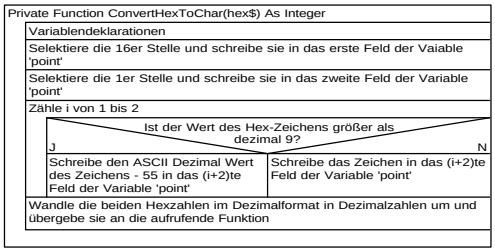


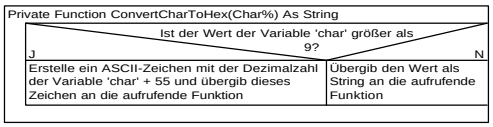


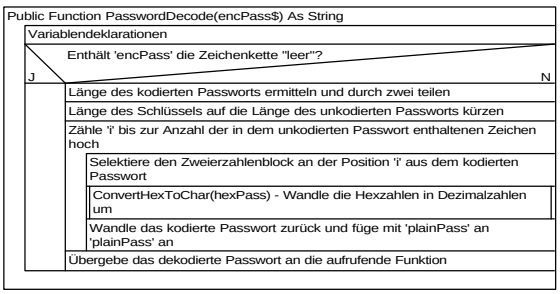
Die Routinen des Moduls `Registry.bas`

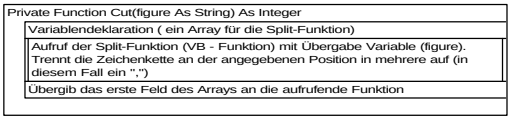




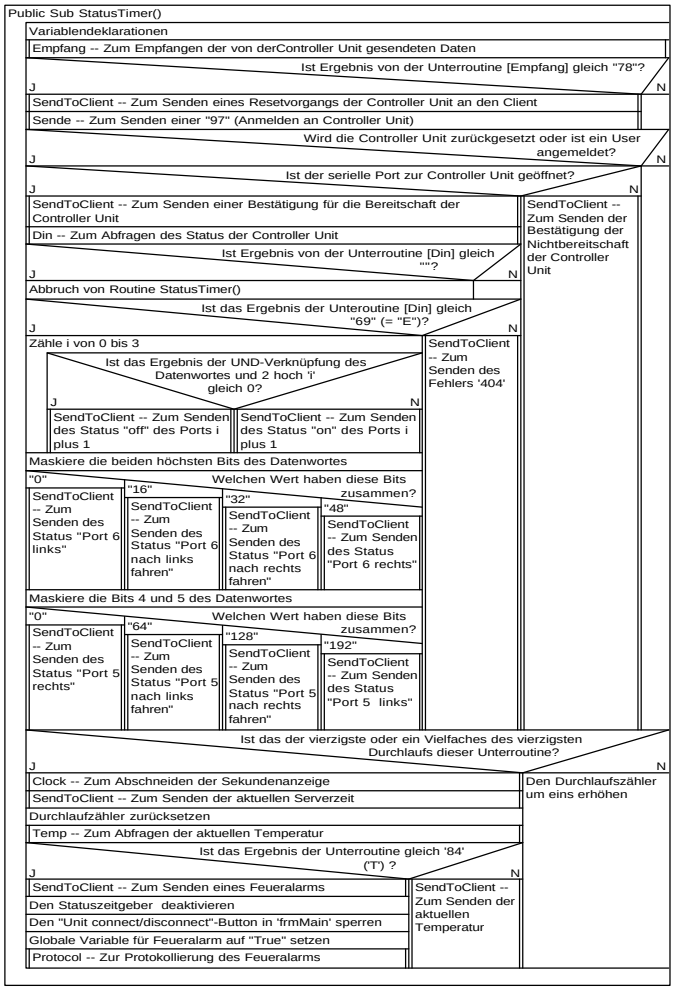




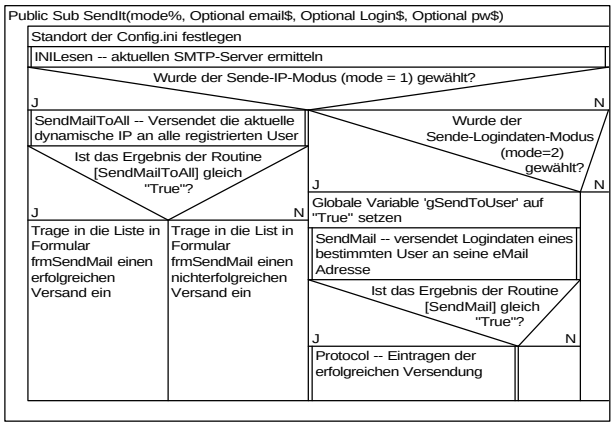


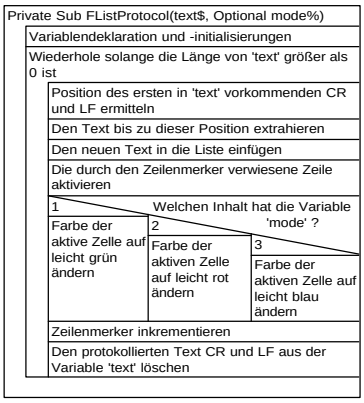


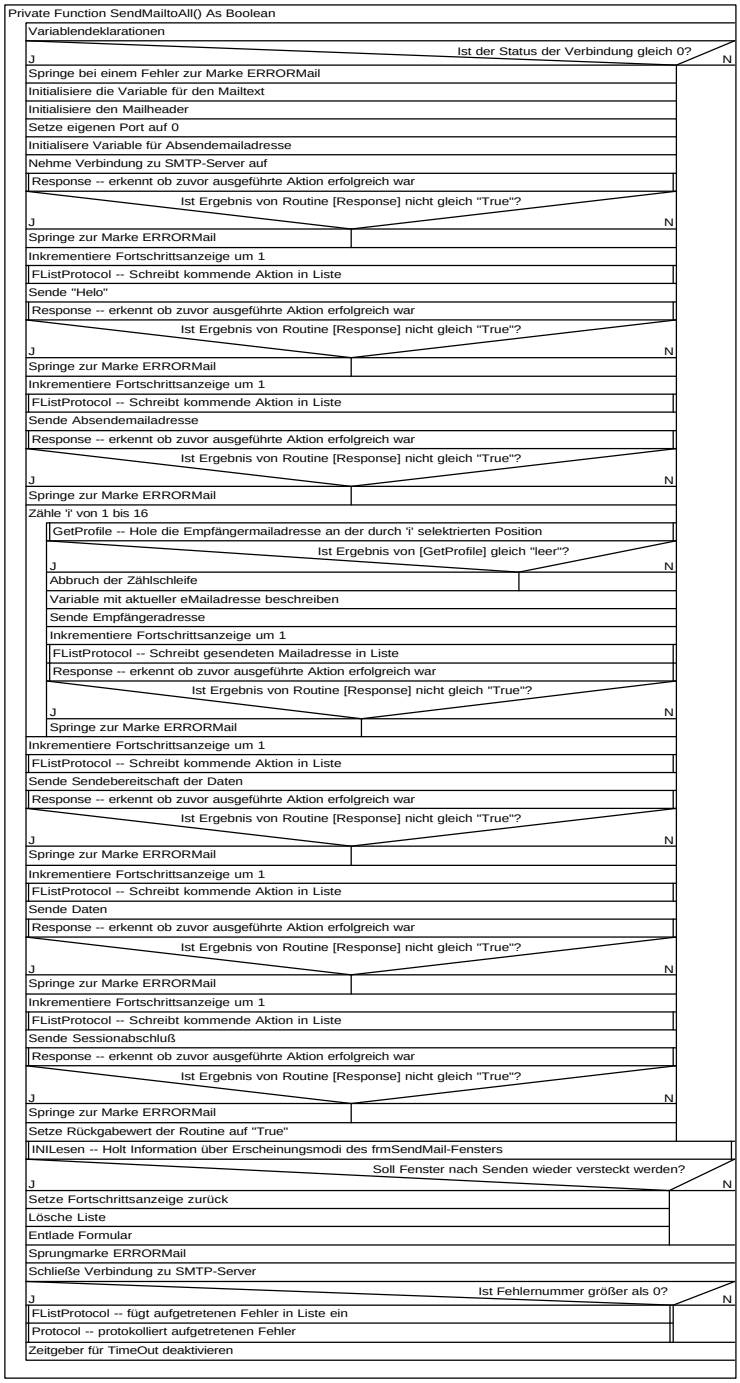
Die Routine `StatusTimer` des Moduls `StatusTimer.bas`

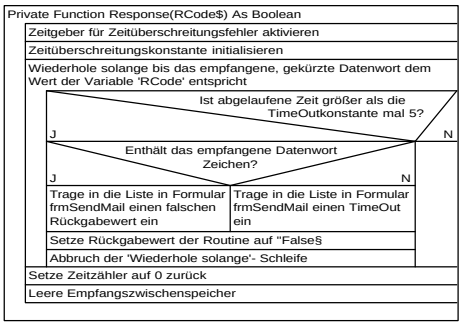


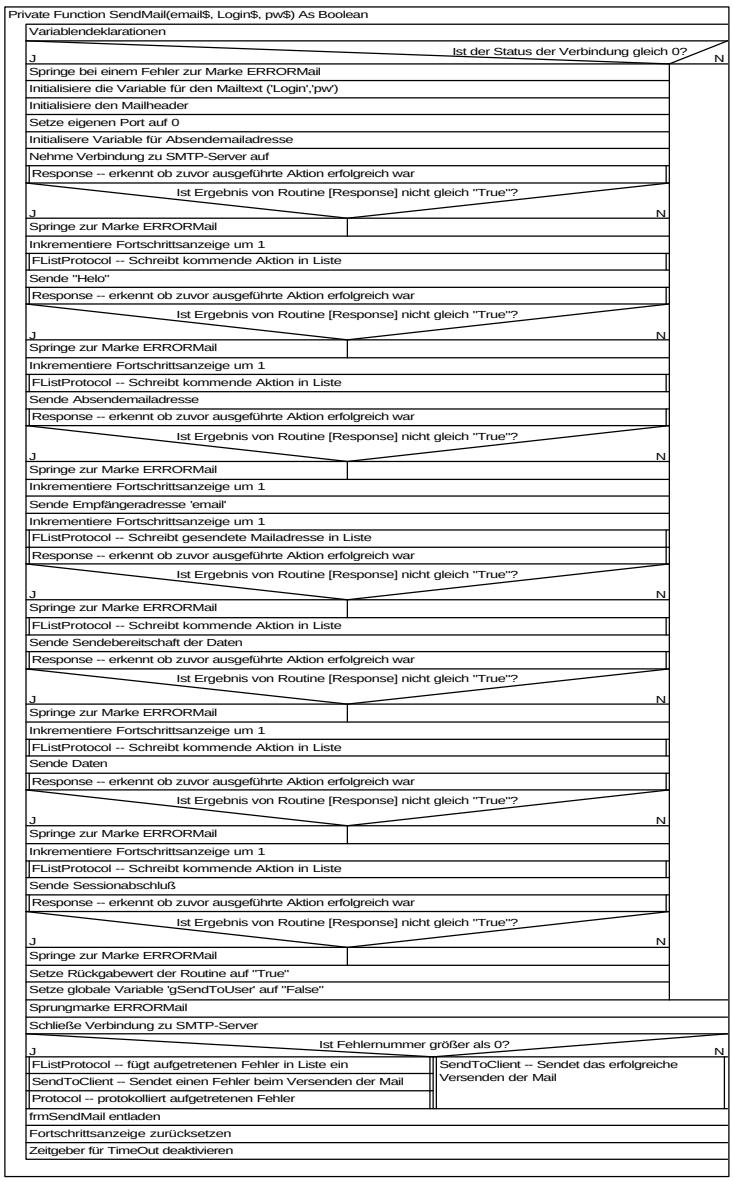
Die Routinen des Moduls `SendMail.bas`



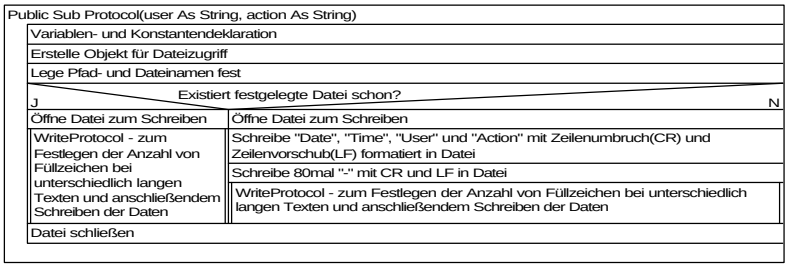


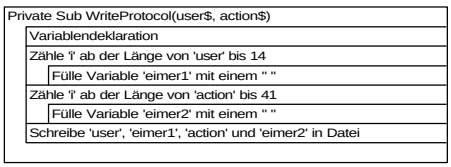

Private Function SendMailtoAll() As Boolean
Variablendeklarationen
Ist der Status der Verbindung gleich 0?
J
N
Springe bei einem Fehler zur Marke ERRORMail
Initialisiere die Variable für den Mailtext
Initialisiere den Mailheader
Setze eigenen Port auf 0
Initialisere Variable für Absendemailadresse
Nehme Verbindung zu SMTP-Server auf
Response -- erkennt ob zuvor ausgeführte Aktion erfolgreich war
Ist Ergebnis von Routine [Response] nicht gleich "True"?
J
N
Springe zur Marke ERRORMail
Inkrementiere Fortschrittsanzeige um 1
FListProtocol -- Schreibt kommende Aktion in Liste
Sende "Helo"
Response -- erkennt ob zuvor ausgeführte Aktion erfolgreich war
Ist Ergebnis von Routine [Response] nicht gleich "True"?
J
N
Springe zur Marke ERRORMail
Inkrementiere Fortschrittsanzeige um 1
FListProtocol -- Schreibt kommende Aktion in Liste
Sende Absendemailadresse
Response -- erkennt ob zuvor ausgeführte Aktion erfolgreich war
Ist Ergebnis von Routine [Response] nicht gleich "True"?
J
N
Springe zur Marke ERRORMail
Zähle 'i' von 1 bis 16
GetProfile -- Hole die Empfängermailadresse an der durch 'i' selektrierten Position
Ist Ergebnis von [GetProfile] gleich "leer"?
J
N
Abbruch der Zählschleife
Variable mit aktueller eMailadresse beschreiben
Sende Empfängeradresse
Inkrementiere Fortschrittsanzeige um 1
FListProtocol -- Schreibt gesendeten Mailadresse in Liste
Response -- erkennt ob zuvor ausgeführte Aktion erfolgreich war
Ist Ergebnis von Routine [Response] nicht gleich "True"?
J
N
Springe zur Marke ERRORMail
Inkrementiere Fortschrittsanzeige um 1
FListProtocol -- Schreibt kommende Aktion in Liste
Sende Sendebereitschaft der Daten
Response -- erkennt ob zuvor ausgeführte Aktion erfolgreich war
Ist Ergebnis von Routine [Response] nicht gleich "True"?
J
N
Springe zur Marke ERRORMail
Inkrementiere Fortschrittsanzeige um 1
FListProtocol -- Schreibt kommende Aktion in Liste
Sende Daten
Response -- erkennt ob zuvor ausgeführte Aktion erfolgreich war
Ist Ergebnis von Routine [Response] nicht gleich "True"?
J
N
Springe zur Marke ERRORMail
Inkrementiere Fortschrittsanzeige um 1
FListProtocol -- Schreibt kommende Aktion in Liste
Sende Sessionabschluß
Response -- erkennt ob zuvor ausgeführte Aktion erfolgreich war
Ist Ergebnis von Routine [Response] nicht gleich "True"?
J
N
Springe zur Marke ERRORMail
Setze Rückgabewert der Routine auf "True"
INILesen -- Holt Information über Erscheinungsmodi des frmSendMail-Fensters
Soll Fenster nach Senden wieder versteckt werden?
J
N
Setze Fortschrittsanzeige zurück
Lösche Liste
Entlade Formular
Sprungmarke ERRORMail
Schließe Verbindung zu SMTP-Server
Ist Fehlernummer größer als 0?
J
N
FListProtocol -- fügt aufgetretenen Fehler in Liste ein
Protocol -- protokolliert aufgetretenen Fehler
Zeitgeber für TimeOut deaktivieren


Private Function Response(RCode$) As Boolean
Zeitgeber für Zeitüberschreitungsfehler aktivieren
Zeitüberschreitungskonstante initialisieren
Wiederhole solange bis das empfangene, gekürzte Datenwort dem Wert der Variable 'RCode' entspricht
Ist abgelaufene Zeit größer als die TimeOutkonstante mal 5?
J
N
Enthält das empfangene Datenwort Zeichen?
J
N
Trage in die Liste in Formular frmSendMail einen falschen Rückgabewert ein
Trage in die Liste in Formular frmSendMail einen TimeOut ein
Setze Rückgabewert der Routine auf "False§
Abbruch der 'Wiederhole solange'- Schleife
Setze Zeitzähler auf 0 zurück
Leere Empfangszwischenspeicher

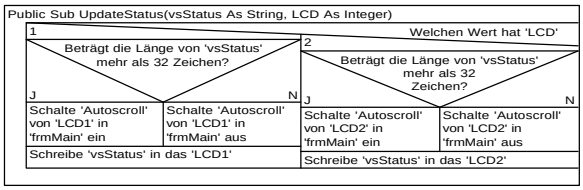


Die Routinen des Moduls `Protocol.bas`

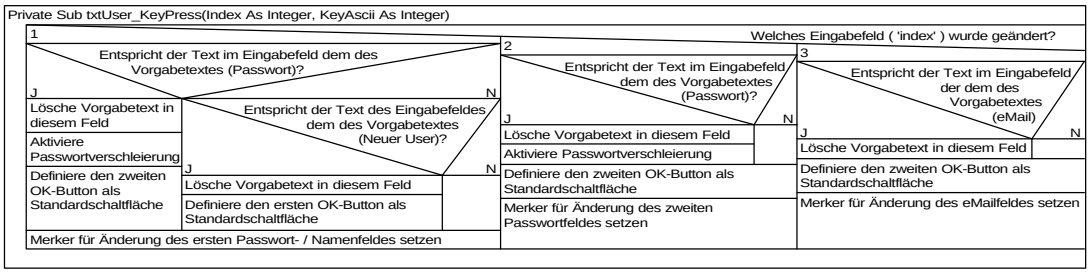


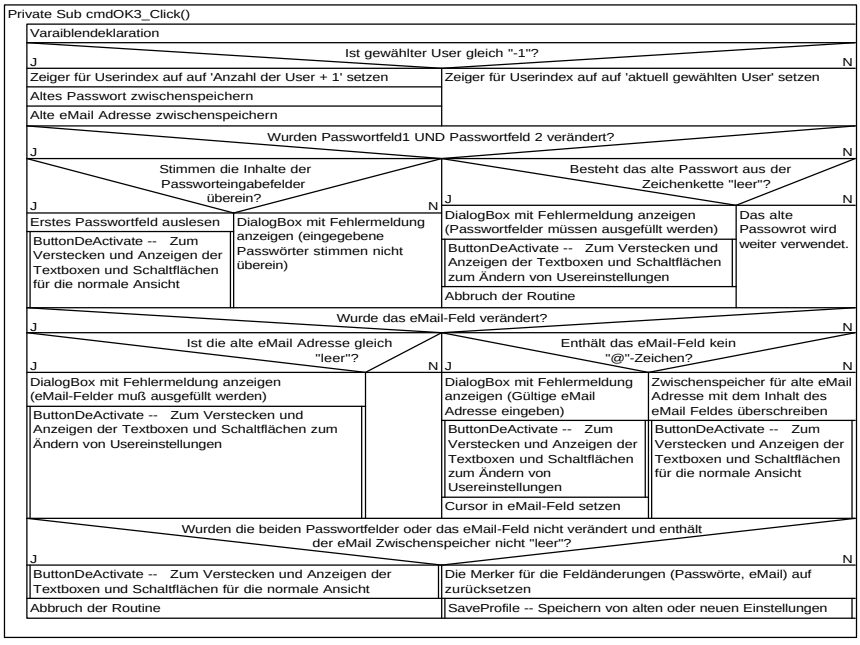


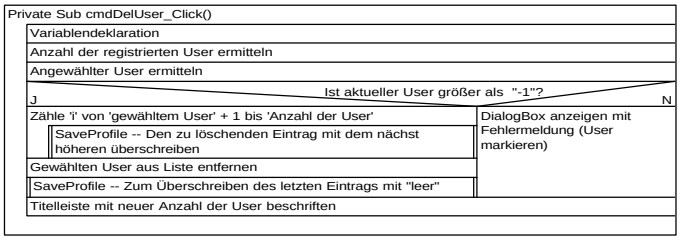
Die Routine `UpdateStatus` des Moduls `Display.bas`

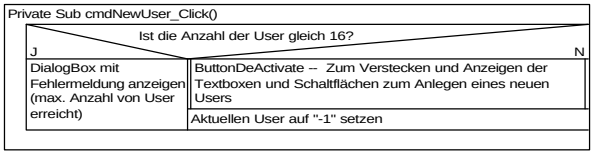


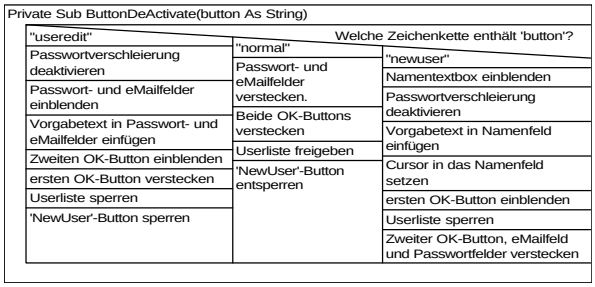
Die Routinen des Formulars `User.frm`

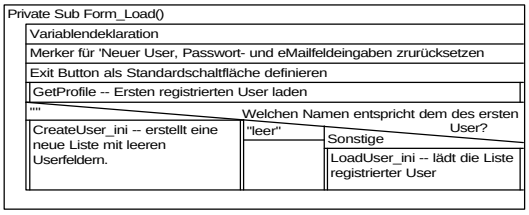


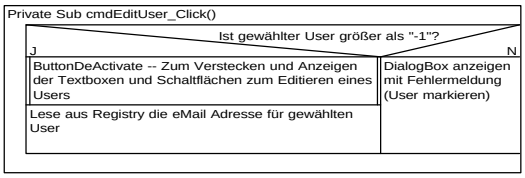


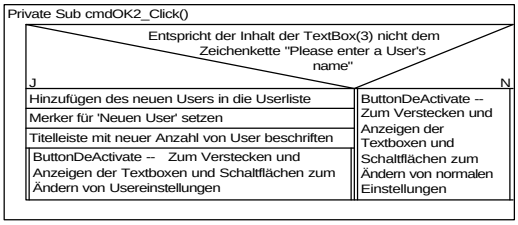


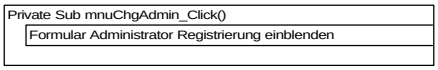


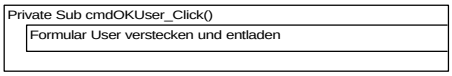




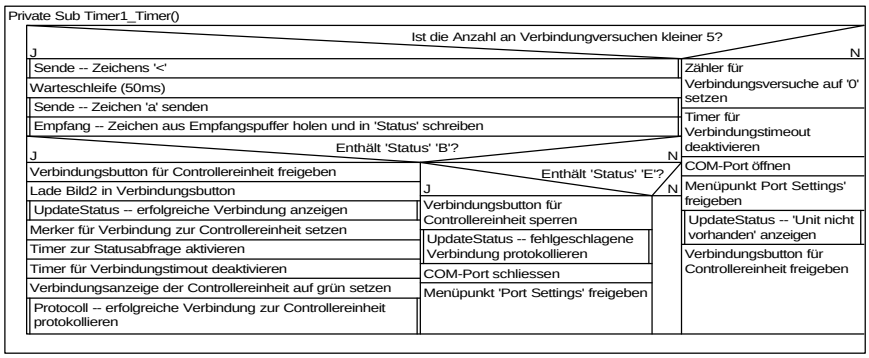


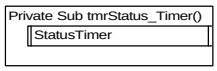


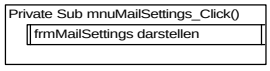


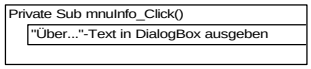


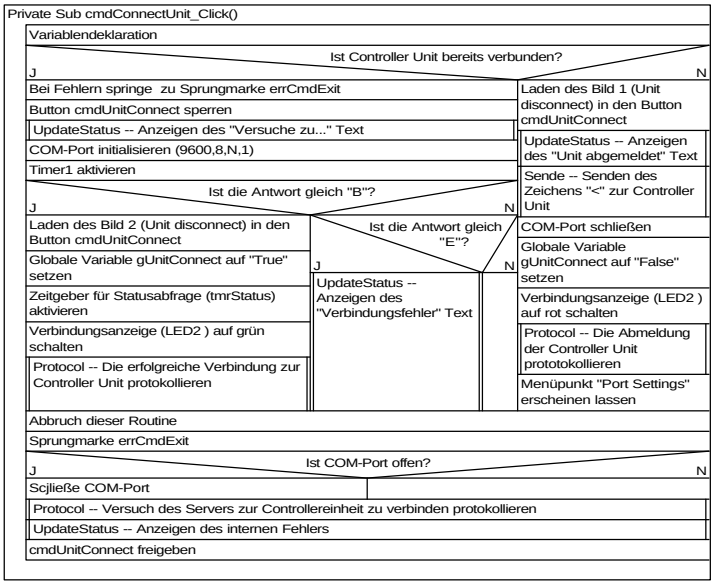
Die Routinen des Formulars `Server.frm`

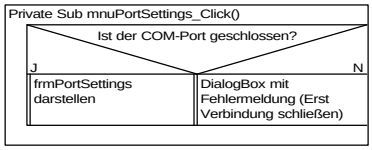


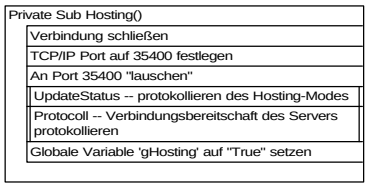


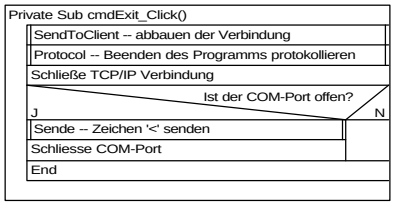


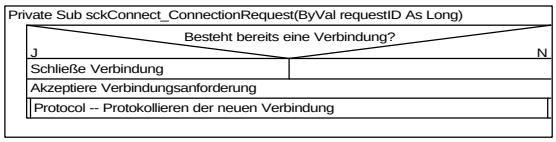


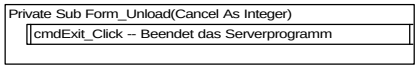


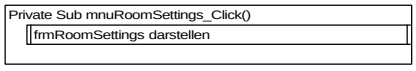


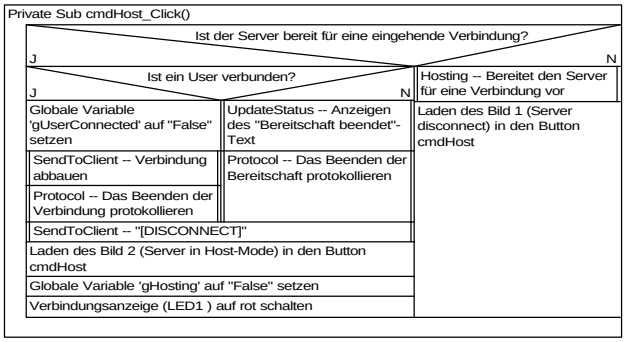


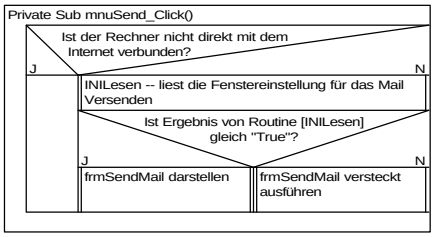


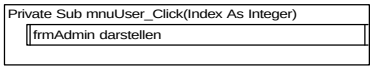


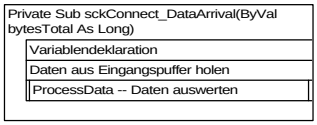


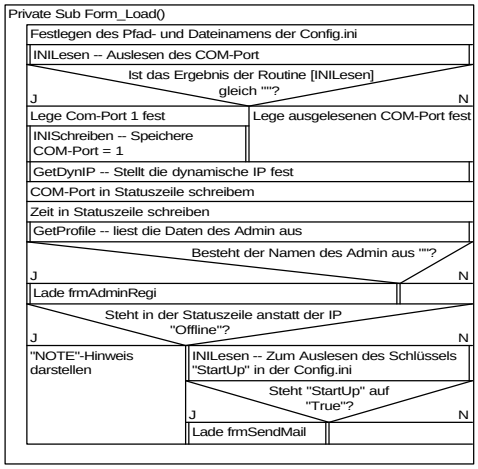


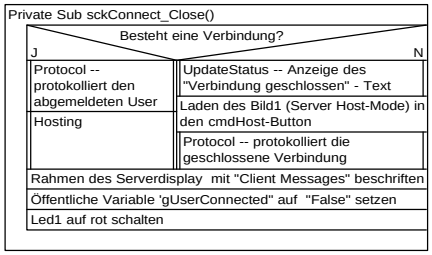


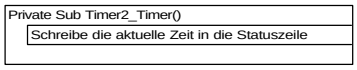




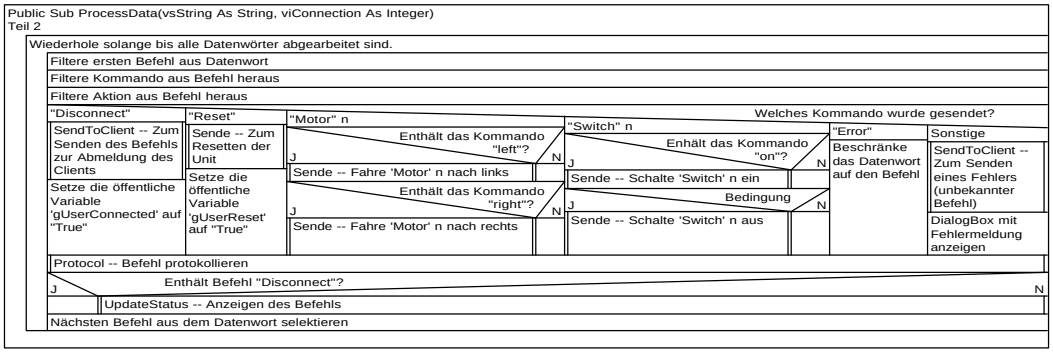


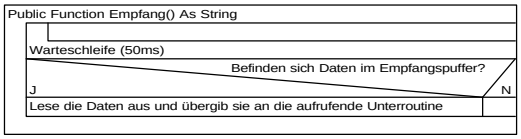


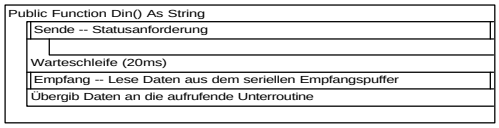


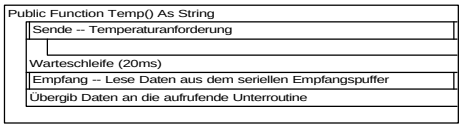


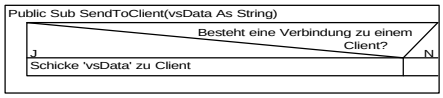
Die Routinen des Moduls `Comm.bas`

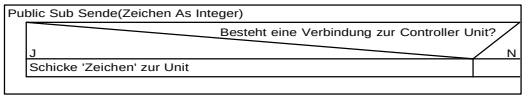


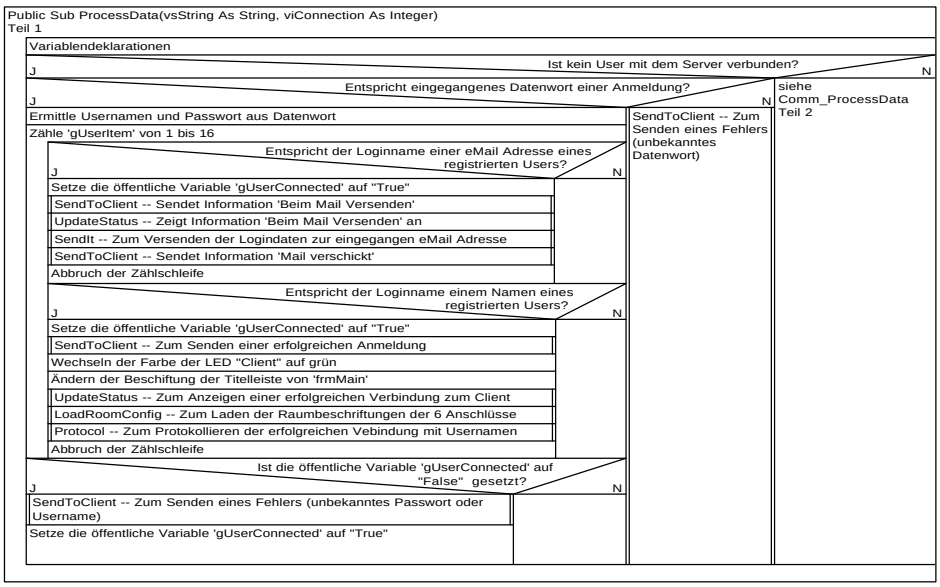






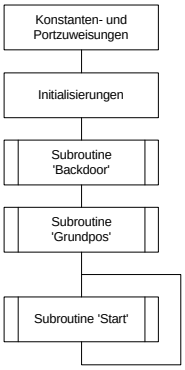




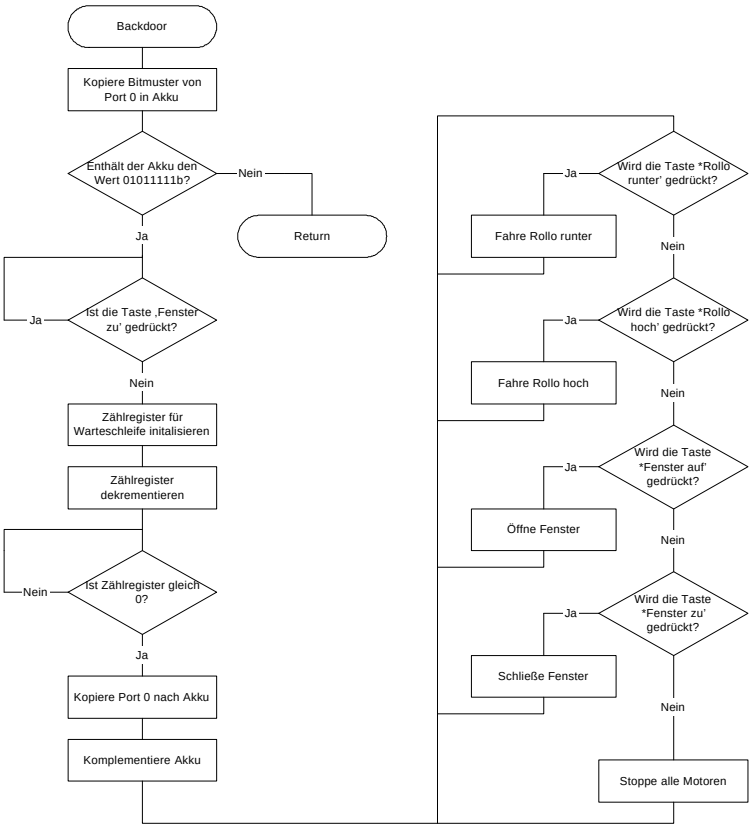

Public Sub ProcessData(vsString As String, viConnection As Integer)
Teil 1
Variablendeklarationen
Ist kein User mit dem Server verbunden?
J
N
Entspricht eingegangenes Datenwort einer Anmeldung?
J
N
siehe Comm_ProcessData Teil 2
Ermittle Usernamen und Passwort aus Datenwort
SendToClient -- Zum Senden eines Fehlers (unbekanntes Datenwort)
Zähle 'gUserItem' von 1 bis 16
Entspricht der Loginname einer eMail Adresse eines registrierten Users?
J
N
Setze die öffentliche Variable 'gUserConnected' auf "True"
SendToClient -- Sendet Information 'Beim Mail Versenden'
UpdateStatus -- Zeigt Information 'Beim Mail Versenden' an
SendIt -- Zum Versenden der Logindaten zur eingegangen eMail Adresse
SendToClient -- Sendet Information 'Mail verschickt'
Abbruch der Zählschleife
Entspricht der Loginname einem Namen eines registrierten Users?
J
N
Setze die öffentliche Variable 'gUserConnected' auf "True"
SendToClient -- Zum Senden einer erfolgreichen Anmeldung
Wechseln der Farbe der LED "Client" auf grün
Ändern der Beschiftung der Titelleiste von 'frmMain'
UpdateStatus -- Zum Anzeigen einer erfolgreichen Verbindung zum Client
LoadRoomConfig -- Zum Laden der Raumbeschriftungen der 6 Anschlüsse
Protocol -- Zum Protokollieren der erfolgreichen Vebindung mit Usernamen
Abbruch der Zählschleife
Ist die öffentliche Variable 'gUserConnected' auf "False" gesetzt?
J
N
SendToClient -- Zum Senden eines Fehlers (unbekanntes Passwort oder Username)
Setze die öffentliche Variable 'gUserConnected' auf "True"

17. Programmablaufplan des Assemblerprogramms des Controllers

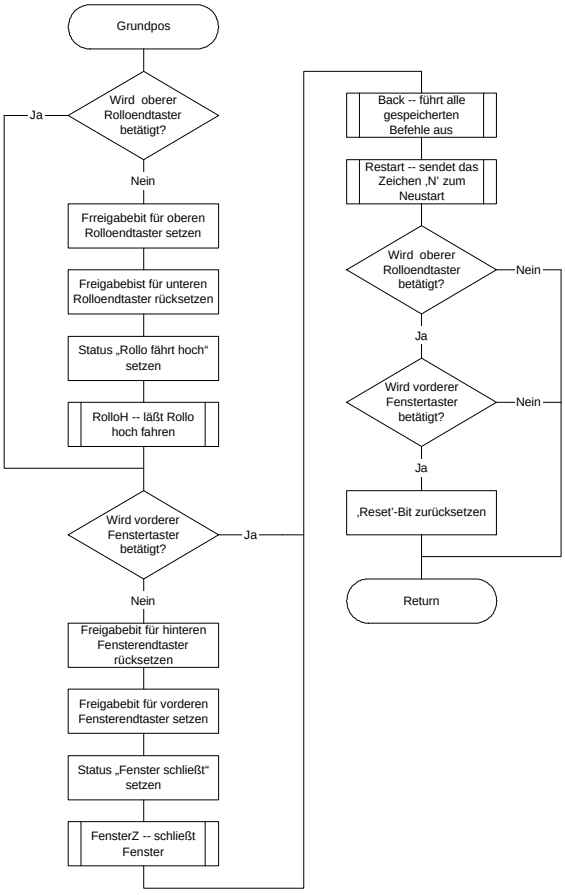
Routine `Hauptprogramm`



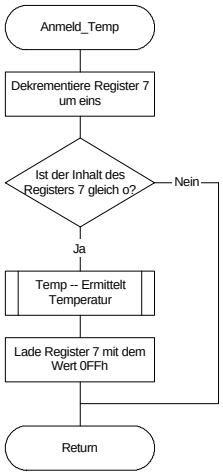
Routine `Backdoor`



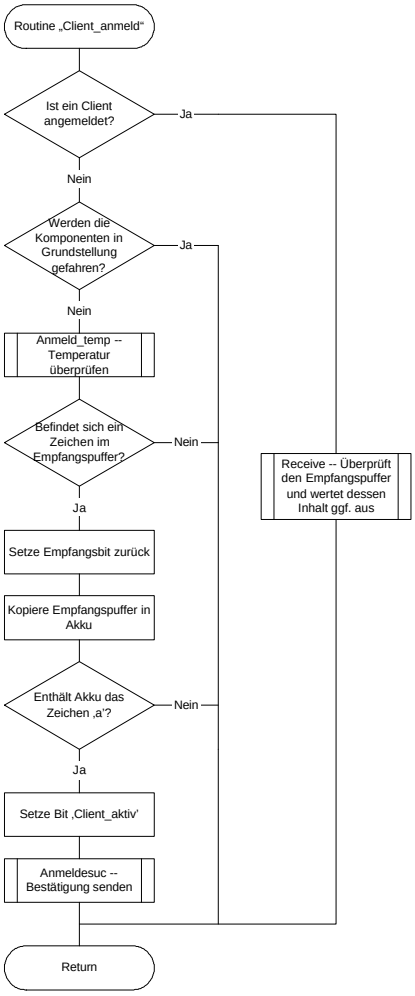
Routine `Grundpos`



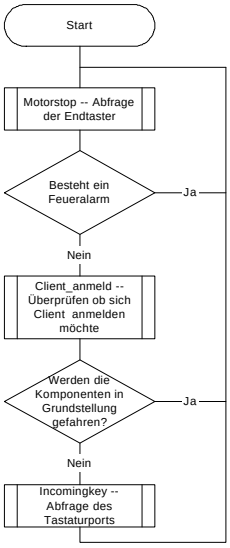
Routine `Anmeld_Temp`



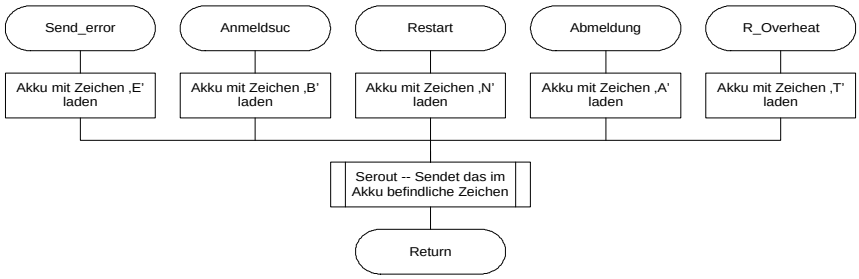
Routine `Client_Anmeld`

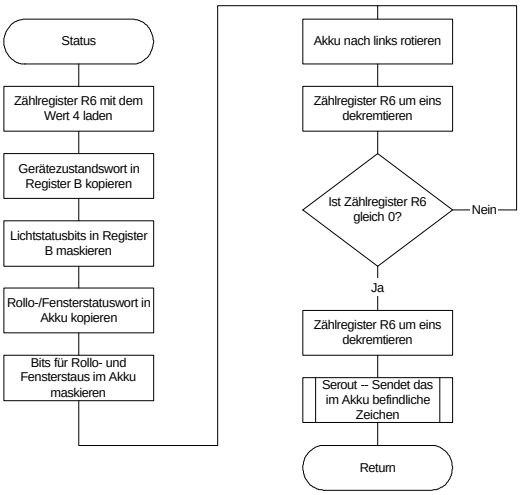


Routine `Start`

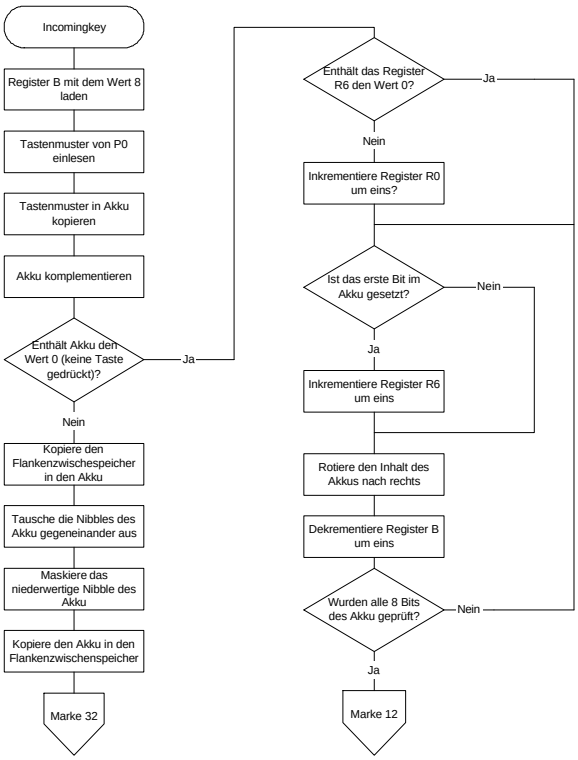


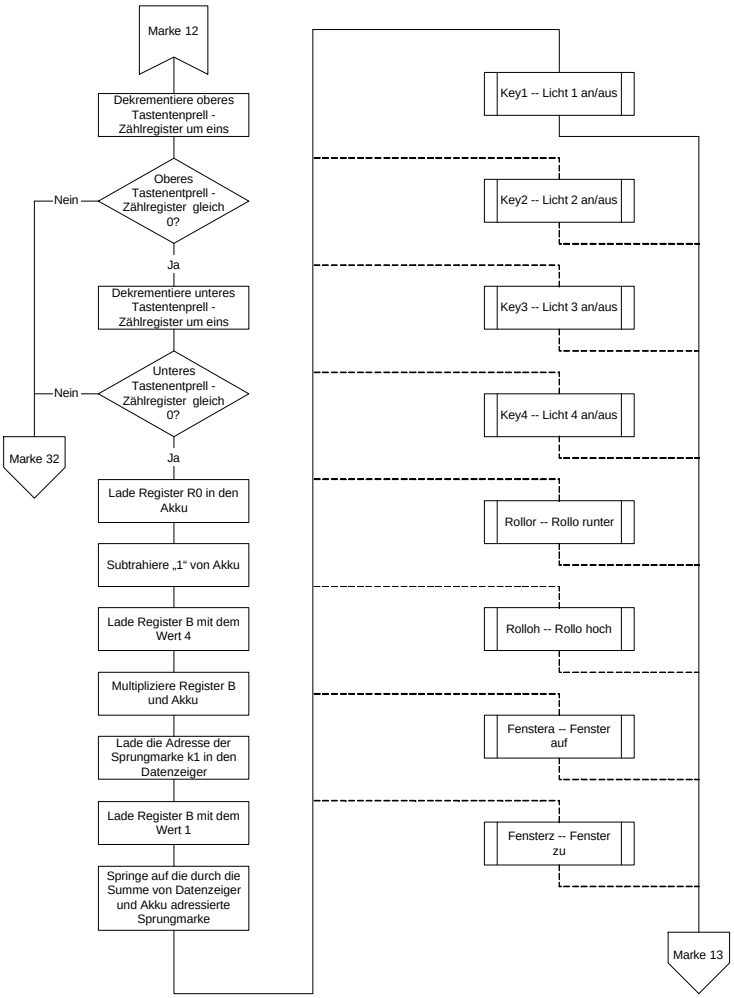
Bereich `Messageausgabe` mit den Routinen `Send_error, Anmeldsuc, Restart, Abmeldung, R_Overheat` und `Status`

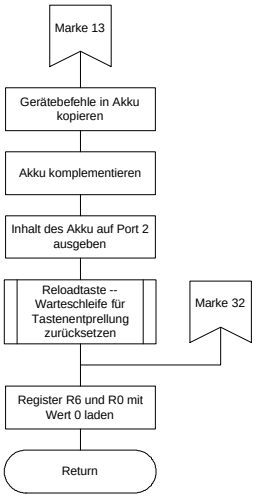


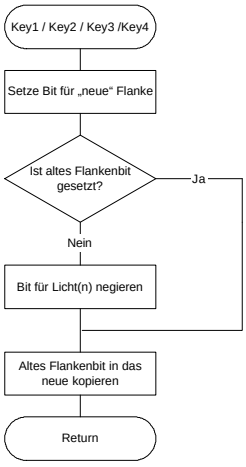


Bereich `Tastenabfrage` mit den Routinen `Incomingkey, Key1, Key2, Key3` und `Key4`

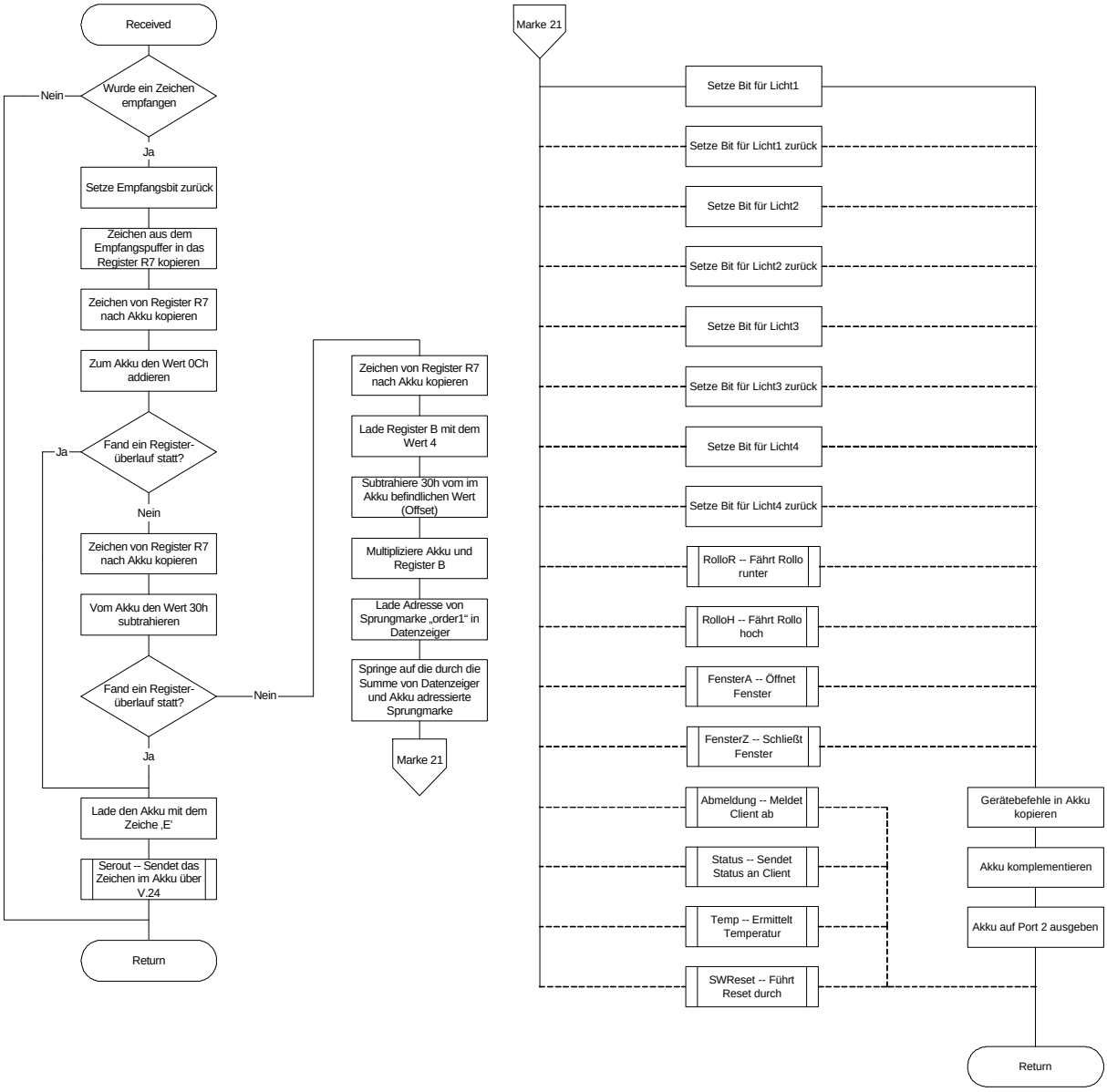



Marke 12
Dekrementiere oberes Tastenentprell - Zählregister um eins
Oberes Tastenentprell - Zählregister gleich 0?
Nein
Ja
Dekrementiere unteres Tastenentprell - Zählregister um eins
Unteres Tastenentprell - Zählregister gleich 0?
Nein
Ja
Marke 32
Lade Register R0 in den Akku
Subtrahiere „1“ von Akku
Lade Register B mit dem Wert 4
Multipliziere Register B und Akku
Lade die Adresse der Sprungmarke k1 in den Datenzeiger
Lade Register B mit dem Wert 1
Springe auf die durch die Summe von Datenzeiger und Akku adressierte Sprungmarke
Key1 -- Licht 1 an/aus
Key2 -- Licht 2 an/aus
Key3 -- Licht 3 an/aus
Key4 -- Licht 4 an/aus
Rollor -- Rollo runter
Rolloh -- Rollo hoch
Fenstera -- Fenster auf
Fensterz -- Fenster zu
Marke 13

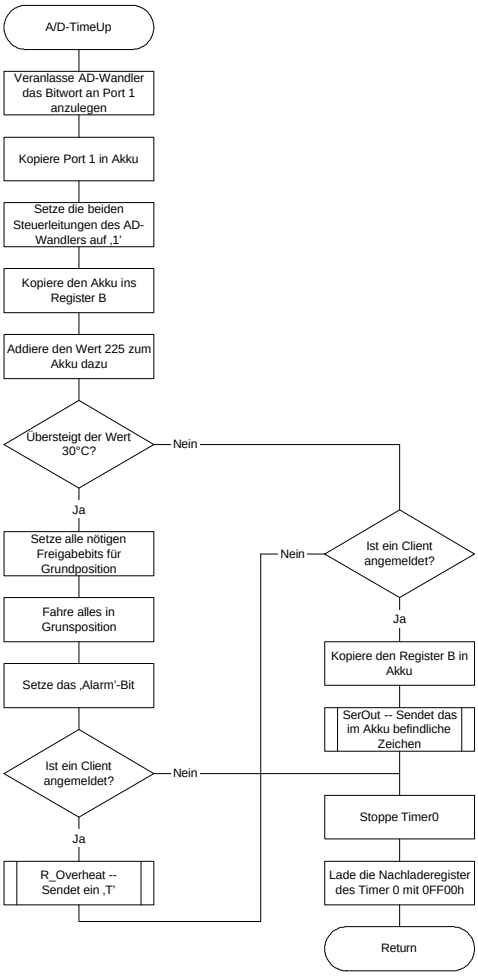

Marke 13
Gerätebefehle in Akku kopieren
Akku komplementieren
Inhalt des Akku auf Port 2 ausgeben
Reloadtaste -- Warteschleife für Tastenentprellung zurücksetzen
Marke 32
Register R6 und R0 mit Wert 0 laden
Return


Key1 / Key2 / Key3 /Key4
Setze Bit für „neue“ Flanke
Ist altes Flankenbit gesetzt?
Ja
Nein
Bit für Licht(n) negieren
Altes Flankenbit in das neue kopieren
Return

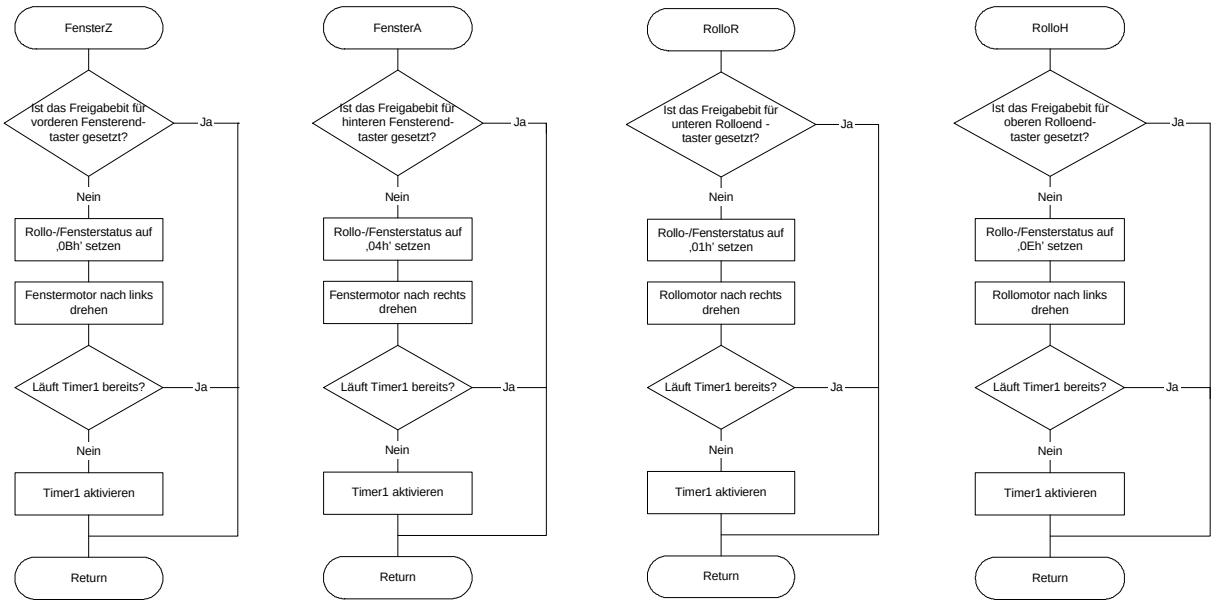
Die Routine `Received`



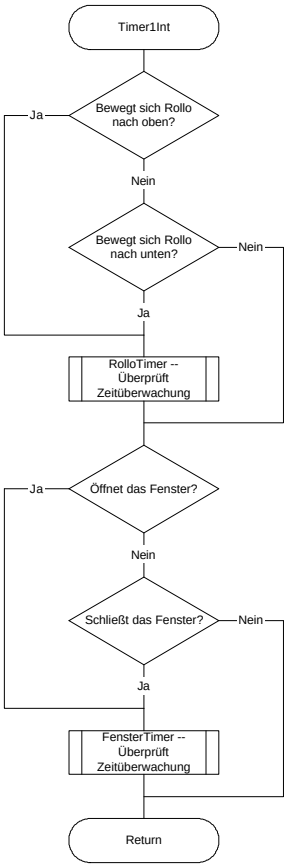
Die Routine `AD_TimeUp`

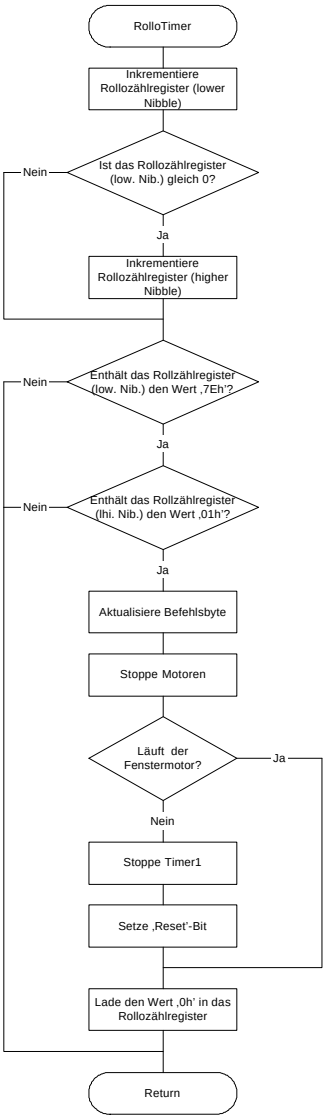


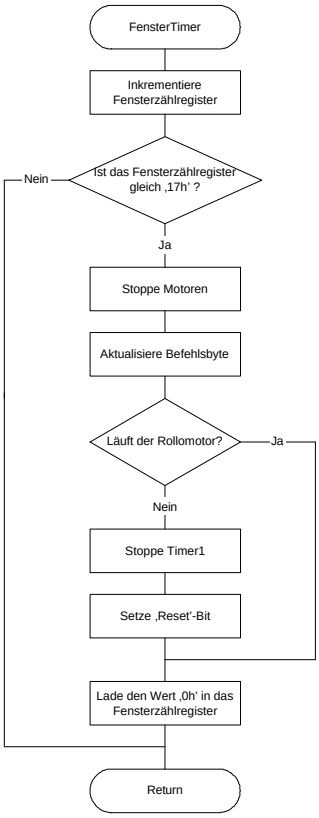
Bereich `Motorsteuerung` mit den Routinen `FensterZ`, `FensterA`, `RolloR` und `RolloH`



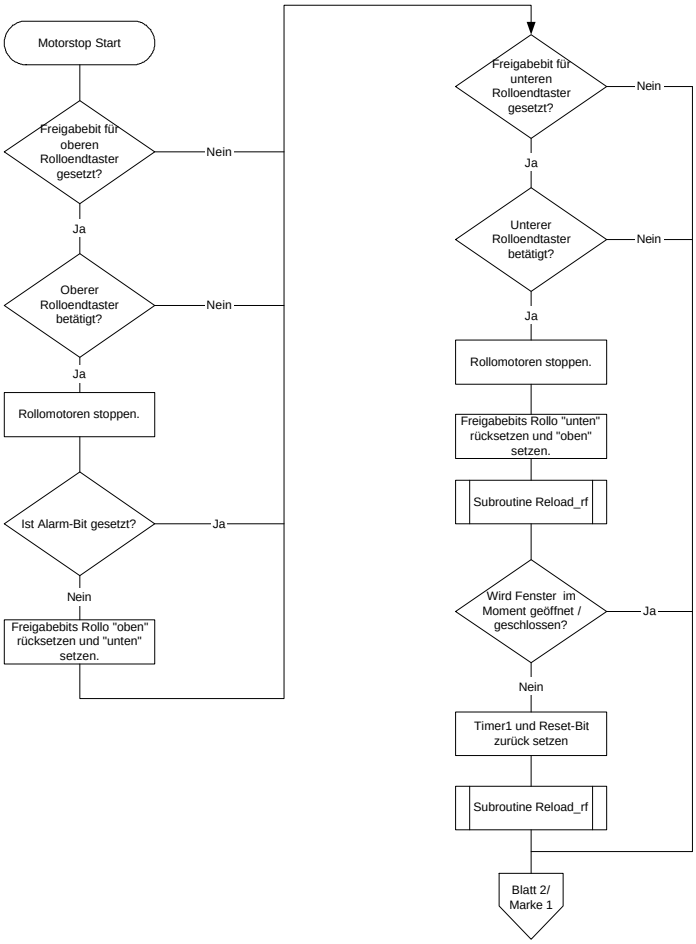
Der Bereich `Zeitüberwachung der Motoren` mit den Routinen `Timer1Int, RolloTimer, Fenstertimer`

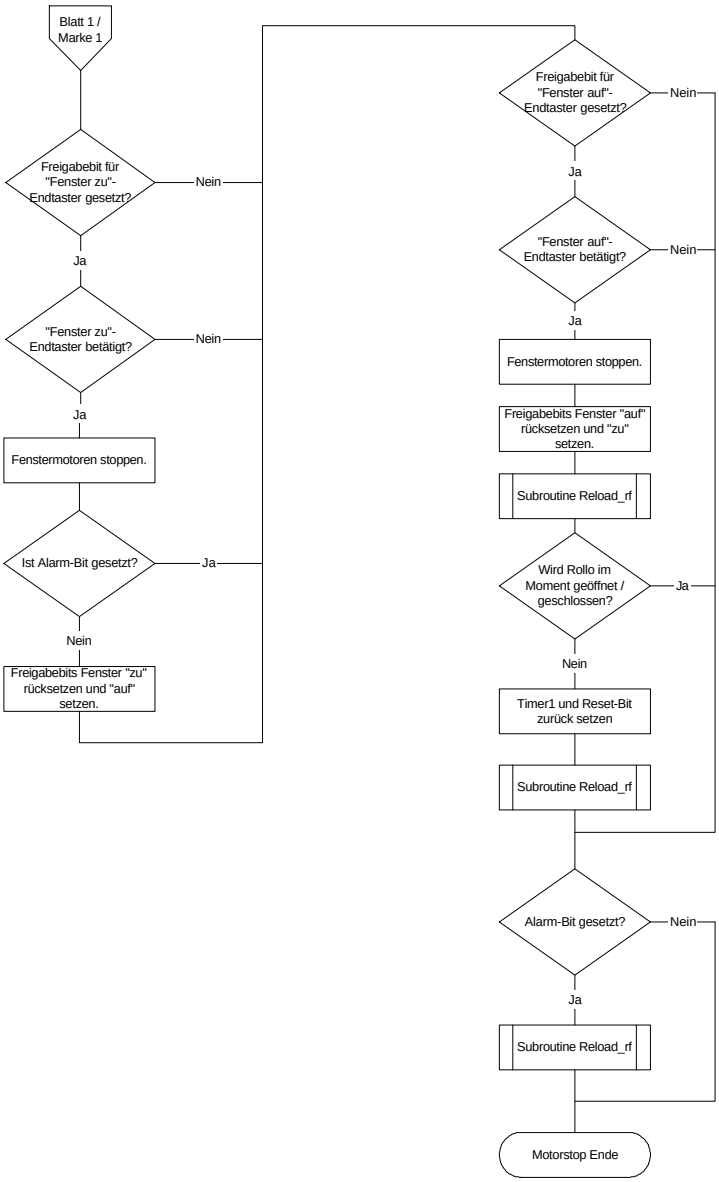






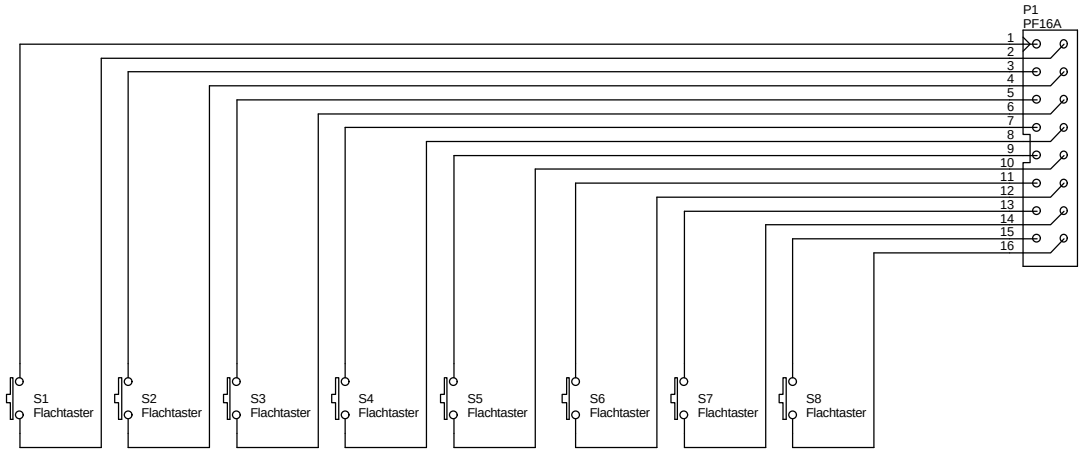
Die Routine `Motorstop`



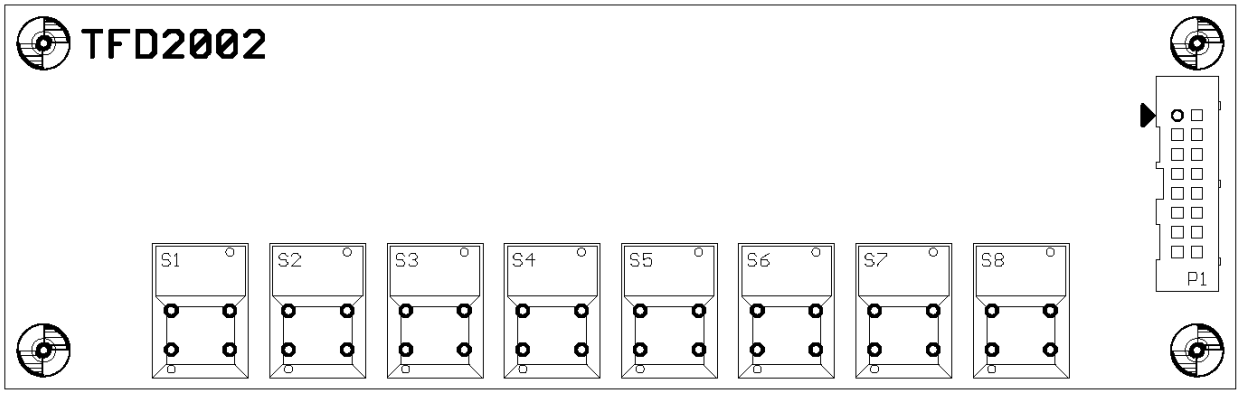

Blatt 1 / Marke 1
Freigabebit für "Fenster zu"-Endtaster gesetzt?
Nein
Ja
"Fenster zu"-Endtaster betätigt?
Nein
Ja
Fenstermotoren stoppen.
Ist Alarm-Bit gesetzt?
Ja
Nein
Freigabebits Fenster "zu" rücksetzen und "auf" setzen.
Freigabebit für "Fenster auf"-Endtaster gesetzt?
Nein
Ja
"Fenster auf"-Endtaster betätigt?
Nein
Ja
Fenstermotoren stoppen.
Freigabebits Fenster "auf" rücksetzen und "zu" setzen.
Subroutine Reload_rf
Wird Rollo im Moment geöffnet / geschlossen?
Ja
Nein
Timer1 und Reset-Bit zurück setzen
Subroutine Reload_rf
Alarm-Bit gesetzt?
Nein
Ja
Subroutine Reload_rf
Motorstop Ende

18.Schaltpläne, Stücklisten und Layouts

18.1 Das Tastenfeld



Title	Tastenfeld für Raum - Komponenten - Steuerung	
Size	Document Number **TFD2002**	Rev **1.0**
Date: Wednesday, April 17, 2002	Sheet 1 of 1	



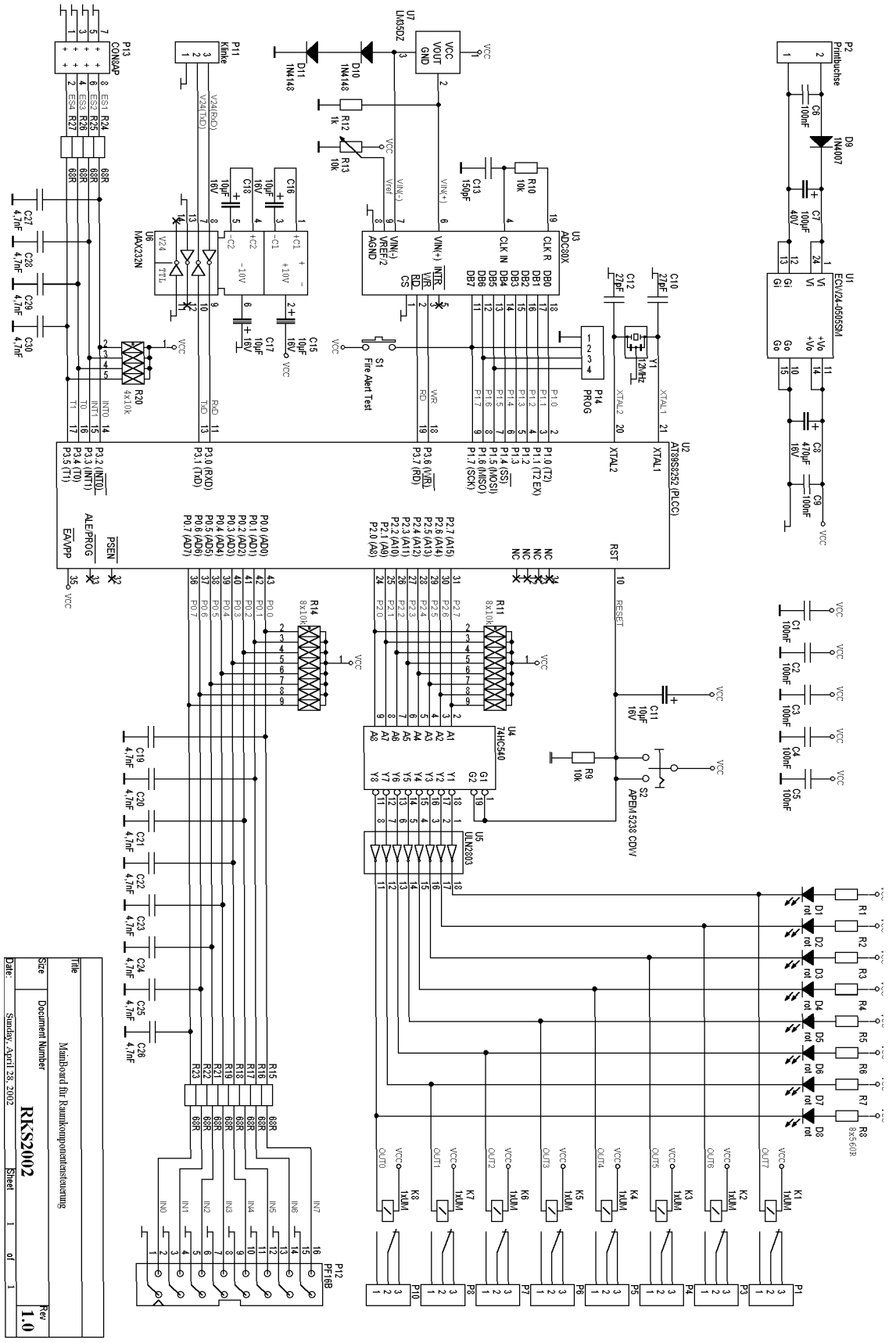
TFD2002

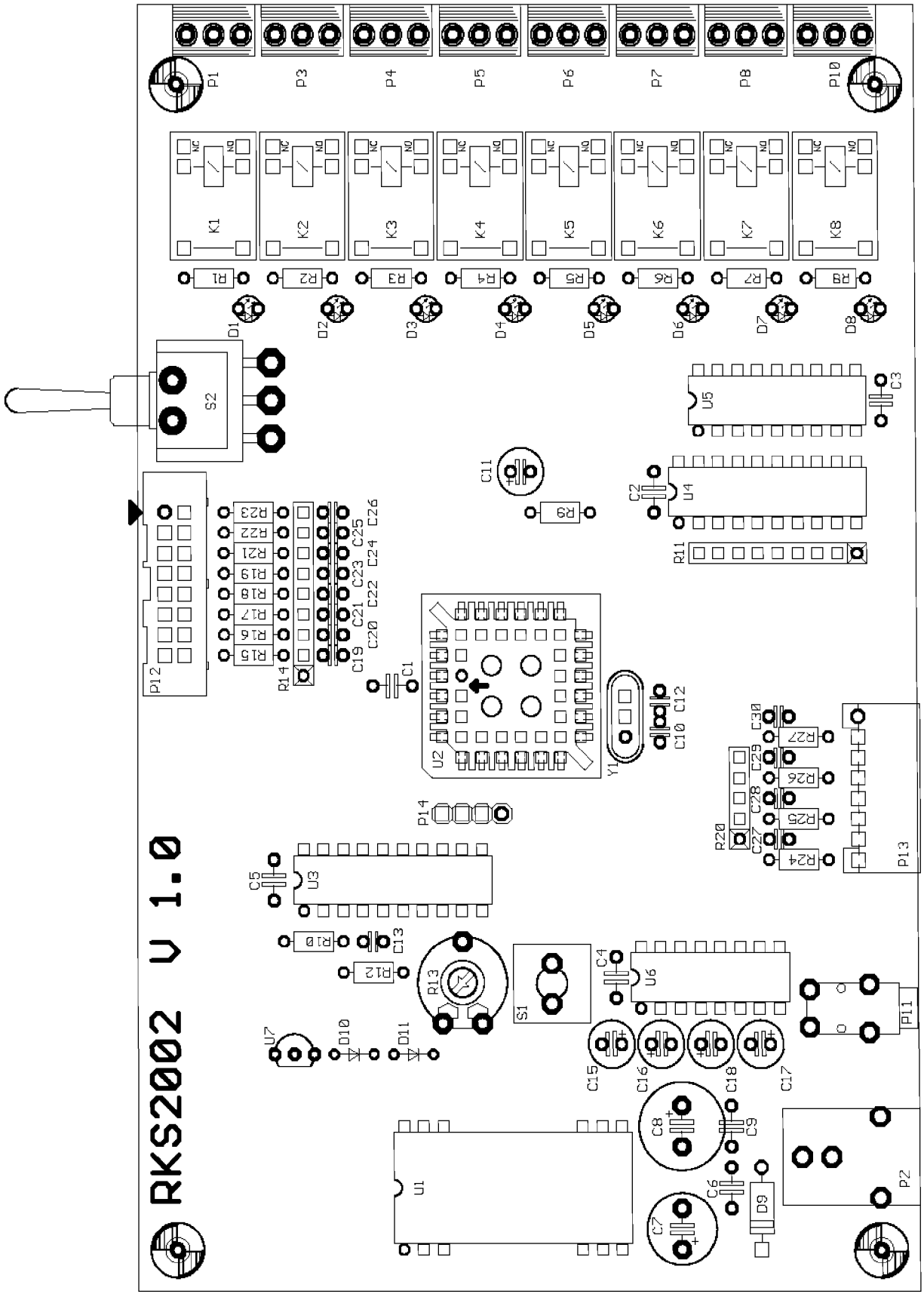
Tastenfeld für Raum - Komponenten - Steuerung

Page 1
Revision 1.0
17.04.2002

Q.	Reference	Part	Dist.	Best. -Nr.	Preis
8	S1,S2,S3,S4,S5,S6, S7,S8	Flachtaster	Reichelt	DIT 1 Ge	1,75 €
1	P1	Steckerleistenwanne	Conrad	74 24 57 - 88	1,35 €

18.2 Die Controllereinheit




RKS2002 V 1.0

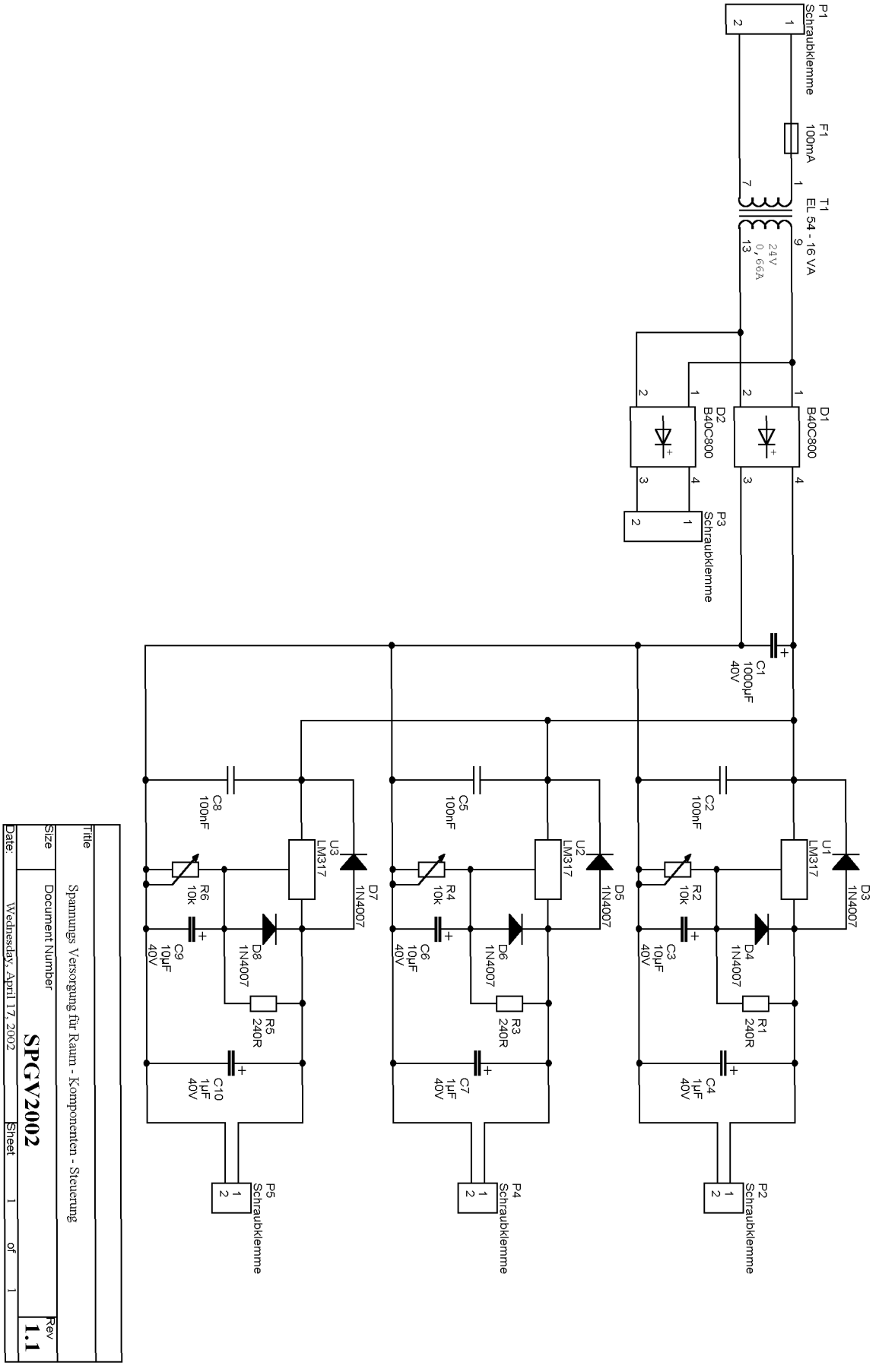
RKS2002

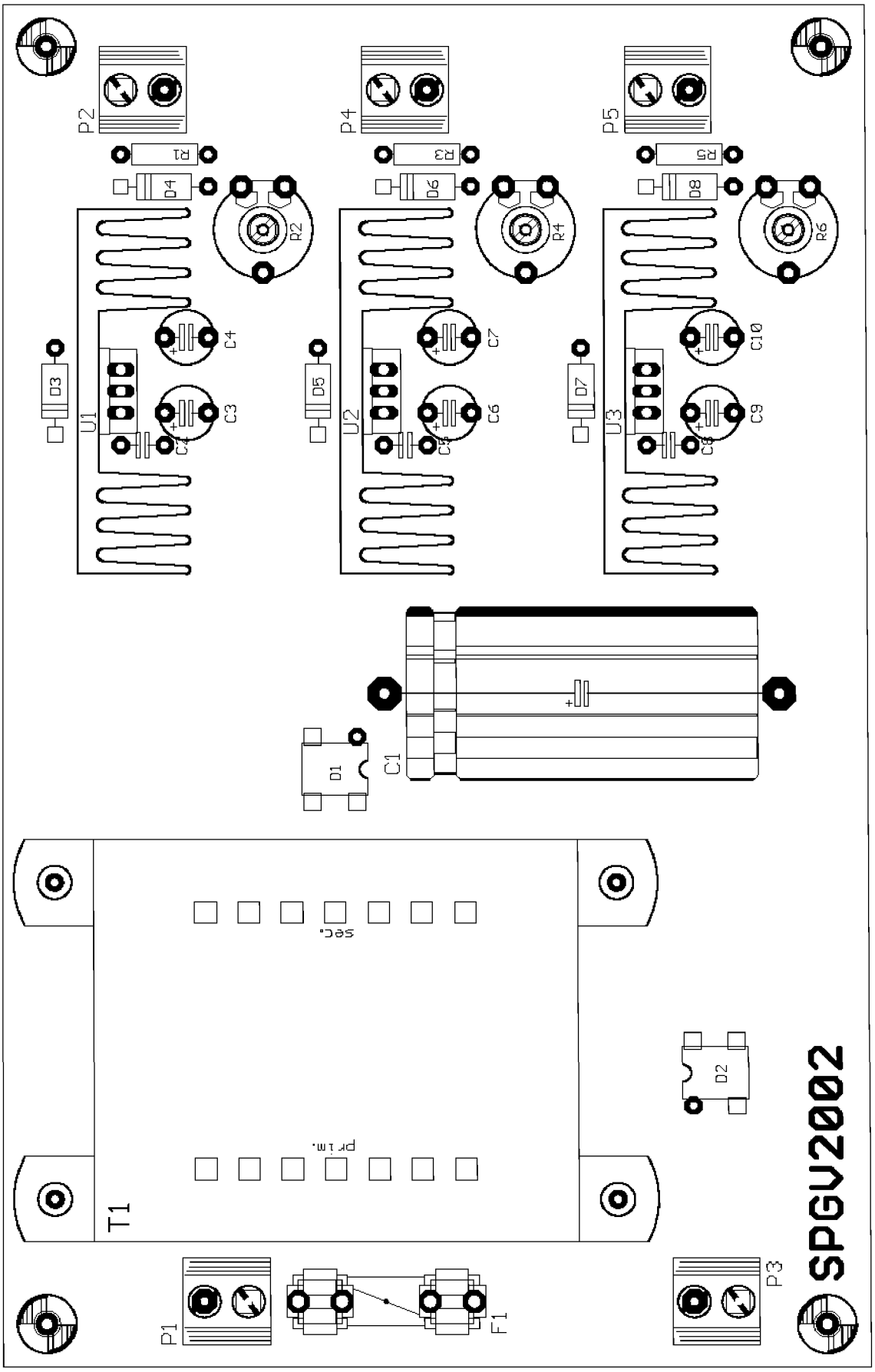
MainBoard für Raum - Komponenten - Steuerung

Q.	Reference	Part	Dist.	Best. -Nr.	Preis
7	C1,C2,C3,C4,C5, C6,C9	100nF	Conrad	45 33 58 - 88	0,42 €
1	C7	100µF / 40V	Reichelt	RAD 100/63	0,11 €
1	C8	470µF/ 16V	Reichelt	RAD 470/16	0,11 €
2	C10,C12	27pF	Reichelt	KERKO 27p	0,02 €
5	C11,C15,C16,C17, C18	10µF/ 16V	Reichelt	RAD 10/35	0,05 €
1	C13	150pF	Reichelt	KERKO 150p	0,04 €
12	C19,C20,C21,C22, C23,C24,C25,C26, C27,C28,C29,C30	4,7nF	Reichelt	KERKO 4,7n	0,06 €
8	D1,D2,D3,D4,D5,D6, D7,D8	LED rot / 2 mA	Reichelt	LED 3mm ST-rt	0,08 €
1	D9	1N4007	Reichelt	1N 4007	0,02 €
2	D10,D11	1N4148	Reichelt	1N 4148	0,02 €
8	K1,K2,K3,K4,K5, K6,K7,K8	1xUM	Conrad	50 51 88 - 88	2,00 €
8	P1,P3,P4,P5,P6,P7, P8,P10	CON3	Conrad	72 99 90 - 88	1,00 €
1	P2	Printbuchse	Conrad	73 79 92 - 88	1,95 €
1	P11	Klinke	Conrad	73 39 62 - 88	0,90 €
1	P12	PF16B	Conrad	74 24 57 - 88	2,65 €
1	P13	CON8AP	Conrad	74 12 56 - 88	3,15 €
1	P14	PROG	Reichelt	SPL 20	0,26 €
8	R1,R2,R3,R4,R5,R6, R7,R8	560R	Reichelt	METALL 560	0,04 €
6	R9,R10,R11,R13, R14,R20	10k	Reichelt	METALL 10,0K	0,04 €
1	R12	1k			0,08 €
12	R15,R16,R17,R18, R19,R21,R22,R23, R24,R25,R26,R27	68R		METALL 68,0	0,04 €
1	S1	APEM 5238 CDW	Buerklin		2,95 €
1	S2	Taster	Conrad	70 77 40 - 88	0,74 €
1	U1	ECW24-0505SM	Fabrimex	ECW24-0505SM	18,00 €
1	U2	AT89S8252 (PLCC)	Reichelt	AT 89S8252 PLCC	7,20 €
1	U2.1	PLCC 44 Fassung	Reichelt	PLCC 44	0,42 €
1	U3	ADC0804	Reichelt	ADC 0804 CN	1,90 €
2	U3.1, U5.1	IC-Sockel DIL20	Reichelt	GS 20 P	0,23 €

Q.	Reference	Part	Dist.	Best. -Nr.	Preis
1	U4	74HC540	Conrad	15 12 89 - 88	2,35 DM
1	U4.1	IC-Sockel DIL18	Reichelt	GS 18 P	0,19 DM
1	U5	ULN2803	Reichelt	ULN 2803 D	0,44 DM
1	U6	MAX232N	Reichelt	MAX 232 CPE	1,00 DM
1	U6.1	IC-Sockel DIL16	Reichelt	GS 16 P	0,18 DM
1	U7	LM35DZ	Reichelt	LM 35 Z	5,00 DM
1	Y1	12MHz	Reichelt	12 HC-18	0,44 DM

18.3 Das Netzteil



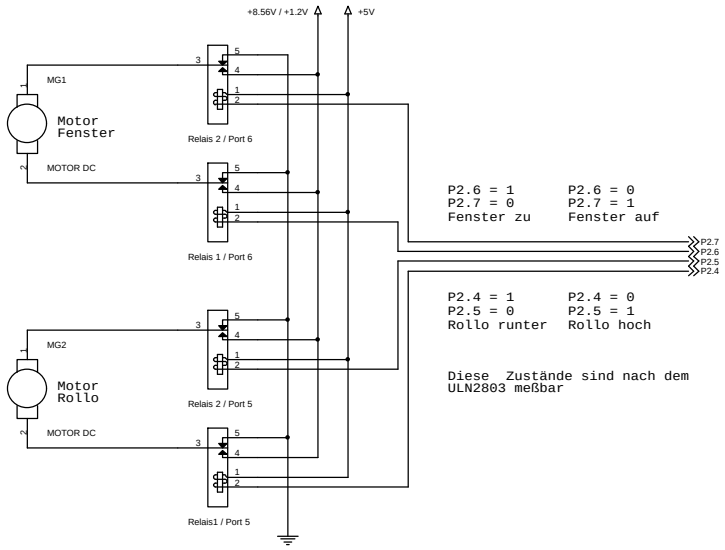

P2
P4
P5
R1
R3
R5
D4
D6
D8
R2
R4
R6
C4
C7
C10
D3
D5
D7
U1
U2
U3
C3
C6
C9
C5
C8
C1
D1
D2
T1
sec.
prim.
P1
P3
F1
SPGV2002

SPGV2002

Spannungsversorgung für Raum - Komponenten - Steuerung

Q.	Reference	Part	Dist.	Best. -Nr.	Preis
1	C1	1000µF / 40V	Reichelt	AX 1000/40	0,66 €
3	C2,C5,C8	100nF	Conrad	45 33 58 - 88	0,42 €
3	C3,C6,C9	10µF / 40V	Conrad	48 17 26 - 88	1,60 €
3	C4,C7,C10	1µF /40V	Conrad	48 16 70 - 88	0,75 €
2	D1,D2	B40C800	Reichelt	B40C800DIP	0,20 €
6	D3,D4,D5,D6,D7,D8	1N4007	Reichelt	1N 4007	0,12 €
1	F1	Sicherung 100mA	Reichelt	FLINK 0,1A	0,12 €
2	F1.1	Sicherungshalter	Reichelt	PL 120000	0,04 €
5	P1,P2,P3,P4,P5	Schraubklemme	Conrad	73 19 86 - 88	2,95 €
3	R1,R3,R5	240R	Reichelt	METALL 240	0,04 €
3	R2,R4,R6	10k / Poti	Reichelt	PIHER 10-L 10k	0,19 €
1	T1	EL 54 - 16 VA	Conrad	55 66 05 - 88	18,95 €
3	U1,U2,U3	LM317	Reichelt	LM317-220	0,36 €

18.4 Schaltplan zum Anschluß der Relais



Quellenverzeichnis

Sämtliche Datenblätter zum Controller AT89S8252
http://www.atmel.com

Datenblatt zum regelbaren Spannungsregler LM317
http://www.st.com

Datenblatt zum Temperatursensor LM35
http://www.national.com

Datenblatt zum Pegelwandlerbaustein MAX232
http://www.national.com

Datenblatt zum Treiberbaustein ULN2803
http://www.national.com

Datenblatt zum Miniatur-Relais FRS1
http://www.maluska.de

Datenblatt zum Transformator der Baureihe 541.
http://www.gerth-trafo.de/firma.htm

Datenblatt zum DC/DC-Wandler ECW12-0505
http://www.fabrimex.ch

Datenblatt zum A/D-Wandler ADC 804
http://www.national.com

Datenblatt zum Tri-State-Baustein 74HC540
http://www.national.com