

Technische Universität Wien

DIPLOMARBEIT

AUTONOM III: Spracherkennung

ausgeführt am

Institut für allgemeine Elektrotechnik und Elektronik
der Technischen Universität Wien,
Abteilung für Angewandte Elektronik,
Leiter Ao. Univ. Prof. Dipl.-Ing. Dr. techn. F. Seifert

unter Anleitung von

Dipl.-Ing. Christian Flachberger
Wiss.Ma. Dipl.-Ing. Paul Panek
Univ. Ass. Dipl.-Ing. Dr.techn. W. Zagler

durch

Gerhard Loidolt
Matr.Nr.: 8626111
Wiener Neustädter Straße 50c/8/3, 2490 Ebenfurth

Ebenfurth, im Mai 1995

Danksagung

Ich möchte meiner Freundin Ingrid Landl für ihre moralische Unterstützung danken.

Mein Dank gilt auch der Fa. AKG für die kostenlose Bereitstellung eines Mikrofons und eines Lautsprechers, die für den praktischen Teil meiner Arbeit nötig waren.

Weiters der Fa. Siemens für die Bereitstellung der NNSR Prototypen.

Herrn Dipl.-Ing. Christian Flachberger, Herrn Dipl.-Ing. Jürgen Demuth, Herrn Dipl.-Ing. Paul Panek, Herrn Dipl.-Ing. Dr. techn. Wolfgang Zagler sowie allen Mitarbeitern der Arbeitsgruppe *fortec* möchte ich für ihre Unterstützung bei Problemen und die gute Zusammenarbeit danken.

Inhaltsverzeichnis

1. Grundlagen	1
1.1 Technische Hilfsmittel	1
1.2 Mögliche Zielgruppen für eine Sprachsteuerung	1
1.3 AUTONOM	2
1.3.1 Wie entstand das Konzept AUTONOM ?	2
1.3.2 Das Baukastensystem	3
1.3.3 Funktionen von AUTONOM	4
1.3.4 Die Benutzeroberfläche	4
1.3.5 Menschen mit Mehrfachbehinderung	5
2. Aufgabenstellung und Ergebnis	6
2.1 Anforderungen an das Spracherkennungsmodul	6
2.2 Auswahl einer Spracherkennung	7
2.3 Ergebnis	8
2.3.1 Realisierung mit dem Prototyp des Spracherkennungsbausteins	8
2.3.2 Realisierung mit dem Serienmodell	8
3. Der Prototyp des Spracherkennungsbausteins von Siemens	10
3.1 Host Interface	10
3.1.1 Pinbelegung	10
3.1.2 Datenformat	
3.2 Befehle	11
3.2.1 Format eines Befehls	11
3.2.2 Befehle	11
3.3 Bedienung des Gerätes	11
3.3.1 Aufnahme von Sprachmustern beginnen	11
3.3.2 Selektion eines Speicherplatzes	12
3.3.3 Sprachmuster lernen	12
3.3.4 Playback	12
3.3.5 Sprachmuster speichern	12
3.3.6 Wort löschen	12
3.3.7 Aufnahme von Sprachmustern beenden	12
3.3.8 Reset	12
3.3.9 Erkennung	13

4. Aufbau einer Spracherkennung mit dem Prototyp von Siemens	14
4.1 Grundkonzept	14
4.2 Verwendete Bausteine	14
4.2.1 Der NNSR Prototyp	14
4.2.2 Der Leitungsempfänger, der Leitungstreiber und der Inverter	14
4.3 Aufbau auf einer Hirschmann-Steckplatte	15
5. Der Spracherkennungsbaustein von Siemens	16
5.1 Architektur	16
5.2 Host Interface	17
5.3 Analog Audio Interface	18
5.4 Host Befehle und NNSR Antworten	18
5.5 Host Befehle	18
5.5.1 Send_status_info	18
5.5.2 Send_configuration	18
5.5.3 Configure_new	18
5.5.4 Initialize_words	19
5.5.5 Send_word_info	19
5.5.6 Clear_word	19
5.5.7 Record	19
5.5.8 Playback_immediate	19
5.5.9 Playback_word	20
5.5.10 Store_pattern	20
5.5.11 Learn	20
5.5.12 Listen	20
5.6 NNSR Antworten	21
5.6.1 Status_info	21
5.6.2 Configuration	21
5.6.3 Ready	21
5.6.4 Word_info	22
5.6.5 Word_recognized	22

6. Aufbau einer Spracherkennung mit dem NNSR	23
6.1 Grundkonzept	23
6.2 Verwendete Bausteine	24
6.2.1 Der Hardware-Kern	24
6.2.2 Der Spracherkennungsbaustein	25
6.2.3 Das LCD-Display	25
6.2.4 Das Mikrophon	26
6.2.5 Der Lautsprecher	26
6.2.6 Der Inverter 74HC540	26
6.2.7 Der Treiber ULN2803A	27
6.2.8 Die Relais	27
7. Software	28
7.1 Betriebsmodus	28
7.2 Programmiermodus	28
7.3 RS 232 Modus	28
7.4 Prozeduren	28
7.4.1 TRAINW	28
7.4.2 CLEARW	29
7.4.3 PLAYBACKW	29
7.4.4 LEARNALL	29
7.4.5 RELSET	29
7.4.6 TO_NNSR	29
7.4.7 FROM_NNSR	29
7.4.8 INIT_NNSR	30
7.4.9 SCHLAFE	30
7.4.10 ERKENNUNG	30
7.4.11 RELAIS_SCHALTEN	30
7.4.12 TIMER	30
7.4.13 TIMER0	30
7.4.14 NOT_TRAINED	30
7.4.15 SPEICHER_VOLL	31
7.4.16 INIT_DISPLAY	31
7.4.17 CLEAR_DISPLAY	31
7.4.18 SEND_DISPLAY	31
7.4.19 LCD_RDY	31
7.4.20 TASTER2_ABF	31
7.4.21 TASTER3_ABF	31
7.4.22 REL_ABF	31
7.4.23 REL_EIN	31
7.4.24 TASTER4_ABF	31
7.4.25 SCHLEIFE	32
7.4.26 RAM_OK	32
7.4.27 Commands HOST -> NNSR	32
7.4.28 ASCII-Tabellen	32

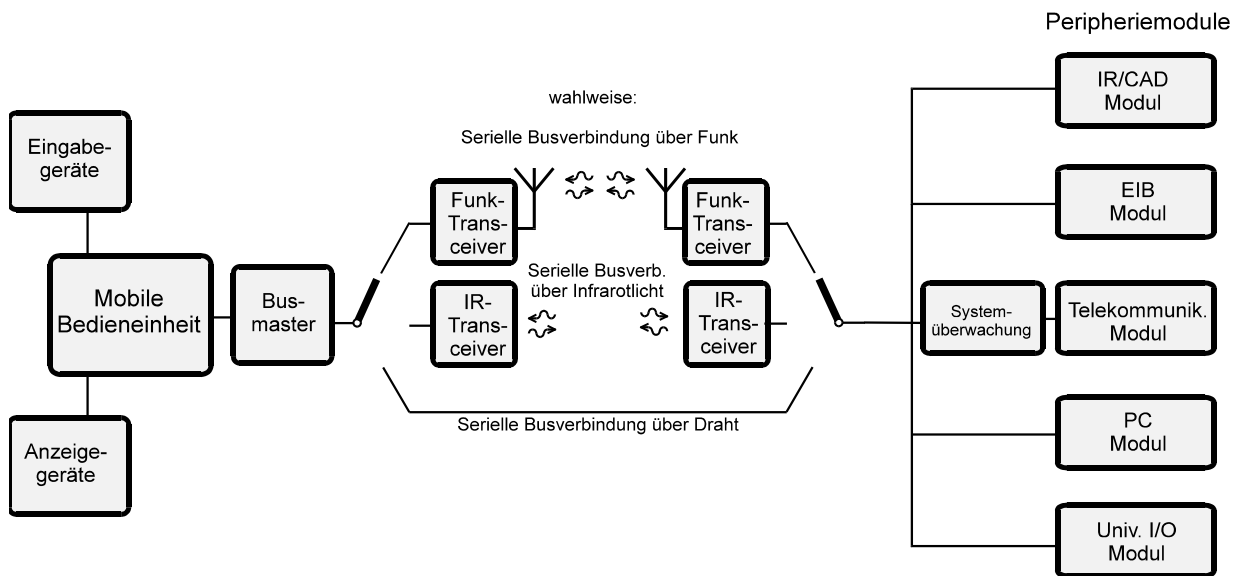
8. Bedienung des Gerätes	33
8.1 Anschlüsse und Bedienungselemente	33
8.1.1 Steuermodul	33
8.1.2 Schaltmodul	34
8.2 Steuermodul	35
8.2.1 Betriebsmodus	35
8.2.1.1 Schaltausgang auswählen	35
8.2.1.2 Befehl Bestätigen	35
8.2.1.3 Grundeinstellung	35
8.2.1.4 Schlüsselwort	35
8.2.2 Programmiermodus	36
8.2.2.1 Wort Lernen	36
8.2.2.2 Wort Löschen	37
8.2.2.3 Playback	37
8.2.2.4 Trainieren	37
8.2.2.5 Schaltausgänge definieren	37
8.2.3 RS 232 Modus	38
8.2.4 Reset	38
8.3 Schaltmodul	38
9. Literaturverzeichnis	39
10. Anhang A	42
10.1 Listing des Assembler Programms	42
10.2 Listing des Turbo Pascal Programms	70
11. Anhang B	72
11.1 Spracherkennungsplatine	72
11.2 Hardware-Kern	79

ZUSAMMENFASSUNG

Es wurde eine Spracherkennung gebaut, die in das AUTONOM Projekt eingebunden werden kann, aber auch als eigenständiges Gerät funktioniert.

Das technische Assistenz-System AUTONOM soll Menschen mit Behinderungen im Alltagsleben zu Hause mehr Handlungsfreiheit, Unabhängigkeit, Selbstbestimmung und Lebensqualität ermöglichen. AUTONOM ist als Baukastensystem konzipiert. Das heißt, man kann aus einem Set genau jene Funktionen auswählen, die den Bedürfnissen des Benutzers entsprechen.

Das AUTONOM - Baukastensystem



Die Spracherkennung ermöglicht es, mit Sprachkommandos elektrische Verbraucher zu steuern. Im Inselbetrieb können bis zu 8 verschiedene Steckdosen geschaltet werden. Eine detaillierte Aufgabenstellung ist in Kapitel 2 nachzulesen. Realisiert wurde die Spracherkennung mit einem Spracherkennungsbaustein von Siemens, der von einem 80C537 Mikrokontroller als Hostprozessor angesteuert wird.

Kapitel 1, Grundlagen,

gibt einen Überblick über mögliche Zielgruppen von AUTONOM III, über technische Hilfsmittel, die bisher zur Verfügung standen, und stellt AUTONOM in groben Zügen vor.

Kapitel 2, Aufgabenstellung und Ergebnis,

beschreibt die Aufgabenstellung, die Auswahl einer Spracherkennung und des Host Prozessors und zieht eine Bilanz über den Prototyp und das fertige Gerät.

Kapitel 3, Der Prototyp des Spracherkennungsbaustein von Siemens,

erläutert den Prototyp des Spracherkennungsbausteins der Fa. Siemens. Es werden das Host Interface, die Befehle und die Bedienung des Prototyps erklärt.

Kapitel 4, Aufbau einer Spracherkennung mit dem Prototyp von Siemens,

Der Prototyp des NNSR wird von einem PC angesteuert. Die Ergebnisse der Erkennung werden wieder an den PC zurückgeliefert und auf dem Bildschirm angezeigt. Das Programm wurde in Turbo Pascal geschrieben. Ein Listing dieses Programms findet sich in Anhang A. Die Verbindung mit dem Versuchsaufbau erfolgte über die serielle Schnittstelle des PC (Com 1). Der Versuchsaufbau wurde auf einem Steckbrett installiert.

Kapitel 5, Der Spracherkennungsbaustein von Siemens,

erläutert die Architektur und die Funktionsweise des Serienbauteils

Kapitel 6, Aufbau einer Spracherkennung mit dem NNSR

Zur Ansteuerung der Peripheriegeräte wurde der Hardware-Kern von AUTONOM verwendet, da diese Platine bereits für andere AUTONOM-Anwendungen benutzt wird und leicht für diesen Zweck adaptierbar ist.

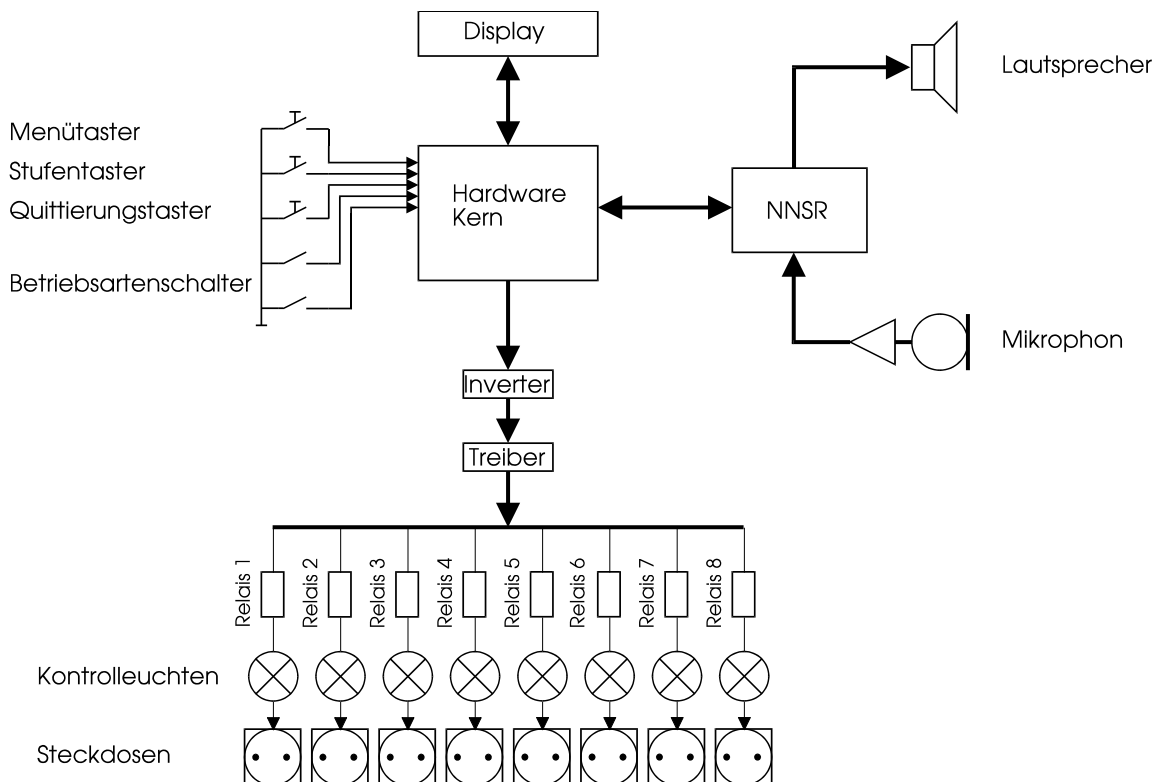
Das Gerät besteht aus 2 Einheiten :

- Steuermodul
- Schaltmodul

Das Steuermodul ist in ein Aluminiumgehäuse in Eurokartengröße eingebaut. Der Lautsprecher befindet sich im Gehäuse des Steuermoduls.

Für das Schaltmodul kam ein Doppelseitiger Kabelkanal der Fa. Legrand zur Verwendung. In diesem Kabelkanal befindet sich ein Netzgerät zur Spannungsversorgung des Steuermoduls und der Relais, weiters die 8 Kontrollleuchten, die 8 Steckdosen, die 8 Relais und der Hauptschalter.

Blockschaltbild der Spracherkennung



Kapitel 7, Software

Hier wird in groben Zügen der Ablauf des Programms erklärt. (Listing siehe Anhang A). Das Programm wurde in der Programmiersprache Assembler geschrieben und auf dem iceMASTER Emulator der Fa. MetaLink getestet.

Es besteht aus 3 Hauptgruppen (siehe Kapitel 8):

- Betriebsmodus (MAIN_PROG)
- Programmiermodus (PROG)
- RS 232 Modus (RS232)

Kapitel 8, Bedienung des Gerätes

Dieses Kapitel ist als Bedienungsanleitung für Benutzer ohne technische Vorkenntnisse gedacht. Sie sollten damit in der Lage sein, das Gerät, in kürzester Zeit selbständig zu bedienen.

Für genauere Informationen stehen die Kapitel 5,6,7,10 und 11 zur Verfügung.

Kapitel 9 Literaturverzeichnis

Kapitel 10, Anhang A,

beinhaltet ein Listing der Programme

Kapitel 11, Anhang B,

enthält die Schaltpläne, das Layout, den Bohrplan und eine Fotografie des Geräts.

1. Grundlagen

1.1 Technische Hilfsmittel

Dieser Absatz gibt einen kurzen Überblick über mögliche technische Hilfsmittel für Menschen mit Bewegungs- und Mehrfachbehinderung. Genaueres findet sich in [9].

Technische Hilfsmittel sind u.a. :

- Fernsteuersysteme
- Spezialtastaturen
- Alternative Eingabegeräte und -verfahren

Beispiele Fernsteuersysteme:

- Infrarot und Trägerfrequenz Fernbedienung (CAD Fa. Legrand)
- James Infrarot Fernsteuerung
- Servo Link für Rollstuhlfahrer

Spezialtastaturen:

- Vergrößerte Tasten
- Tasten mit Lochmasken
- Tasten mit Verzögerung als Schutz gegen Mehrfachanschläge
- Tastaturen, die notwendige gleichzeitige Tastenbetätigungen in Einzelanschläge auflösen
- Spezielle Anordnung der Tasten
- Einhandtastaturen
- Tastaturen mit Sensortasten

Alternative Eingabegeräte und -verfahren:

- Saug-Blas-Schalter
- Zungenschalter
- Lidschlag-Schalter
- optische Pointer Systeme
- Spracheingabe (Gegenstand dieser Diplomarbeit)

1.2 Mögliche Zielgruppen für eine Sprachsteuerung (siehe auch [9] und [23])

Denkbar sind folgende Zielgruppen:

- Immobiler Menschen
- Menschen mit Schwierigkeiten, technische Geräte mit komplexer Bedienung zu handhaben.

Immobil Menschen

Gründe für diese Immobilität können sein:

- Wirbelverletzungen, die nicht operativ stabilisiert sind
- Wirbelverletzungen, die nicht belastet werden dürfen
- andere Verletzungen, die nicht belastet werden dürfen
- schlechter Allgemeinzustand
- interne Erkrankungen
- Schwangerschaften, bei denen absolute Bettruhe gehalten werden muß

Gerade auch bei vorübergehender Immobilität ließe sich AUTONOM III gut einsetzen, da das System ohne großen Installationsaufwand schnell an die jeweilige Situation angepaßt werden kann.

Menschen mit Schwierigkeiten, technische Geräte mit komplexer Bedienung zu handhaben

Gründe für diese Schwierigkeiten können sein:

- Fehlen von Extremitäten
- Fehlende, unzureichende oder unkoordinierte Bewegungsmöglichkeit von Körperteilen
- Bewegungsausfälle oder Bewegungsstörungen infolge neurologischer Ursachen
- fortgeschrittenes Alter
- geistige Behinderung

Es könnten auch komplexere Steuerabläufe, durch ein einziges Sprachkommando abgewickelt werden. Für diese Steueraufgaben müßte die Spracherkennung speziell auf das jeweilige Gerät zugeschnitten werden. Ein einfaches Beispiel wäre das Ersetzen der Fernbedienung eines Fernsehgerätes.

1.3 AUTONOM [28], [29]

In diesem Absatz soll das AUTONOM Projekt kurz vorgestellt werden, da die Spracherkennung nur ein Teilaspekt dieser Umfeldsteuerung ist.

1.3.1 Wie entstand das Konzept AUTONOM ?

Das technische Assistenz-System AUTONOM soll Menschen mit Behinderungen im Alltagsleben zu Hause mehr Handlungsfreiheit, Unabhängigkeit, Selbstbestimmung und Lebensqualität ermöglichen. Um dieses Ziel zu erreichen, ist eine eingehende Betrachtung der Bedürfnisse dieser Menschen erforderlich.

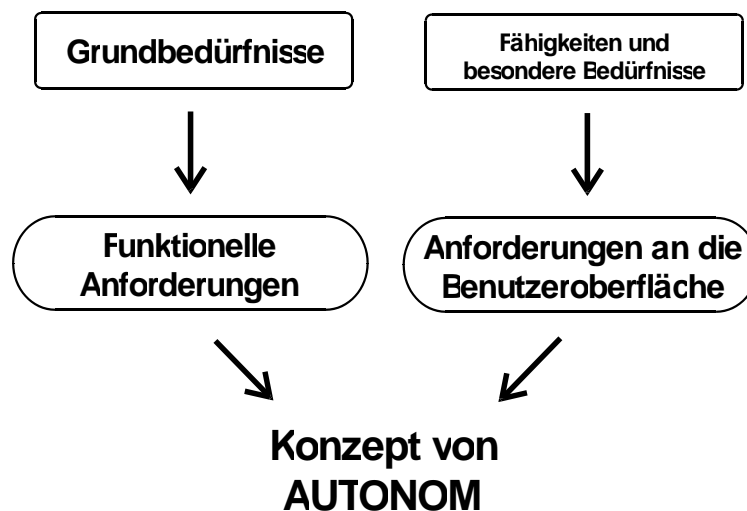


Abbildung 1.1 Das Konzept von AUTONOM

Zwei grundlegende Fragen sind von Bedeutung:

- Was soll so ein technisches Assistenzsystem alles können?
- Wie muß das technische Assistenzsystem gestaltet sein, um von Menschen mit verschiedenen Behinderungen und Kombinationen von Behinderungen optimal verwendet werden zu können.

Die Betrachtung der ersten Frage gibt Aufschluß über die funktionellen Anforderungen an das Assistenzsystem, die zweite Frage über die Anforderungen an die Benutzeroberfläche. Funktionelle Anforderungen und Anforderungen an die Benutzeroberfläche sind gewissermaßen die beiden Standbeine des Konzeptes von AUTONOM (Abbildung 1.1).

1.3.2 Das Baukastensystem

Bevor näher auf die verschiedenen Funktionen von AUTONOM eingegangen wird, soll hier ein kurzer Überblick über das gesamte System gegeben werden.

AUTONOM ist als Baukastensystem konzipiert. Das heißt, man kann aus einem Set genau jene Funktionen und auch genau jene Bedienelemente auswählen, die exakt den Bedürfnissen des Benutzers entsprechen. Das System kann jederzeit erweitert, angepaßt und auch vereinfacht werden. Abbildung 1.2 gibt einen Überblick über das System.

Auf der linken Seite ist die **Bedieneinheit** dargestellt. Sie dient dem Benutzer als „Kommandozentrale“ für das ganze System. Sie ist mobil und kann bei Bedarf auf einem Rollstuhl mitgeführt werden. An die Bedieneinheit können verschiedene Eingabegeräte wie Spezialschalter (vergrößerte Taster, Lidschlagschalter, Saug/Blasschalter etc.), Zeigegeräte (Maus, Trackball, Touchscreen) oder auch Spracheingabe (Thema dieser Arbeit) angeschlossen werden. Zur Ausgabe an den Benutzer können verschiedene Ausgabegeräte wie CRT-Schirme, LCD-Schirme, Klanggeneratoren oder Sprachausgabe verwendet werden.

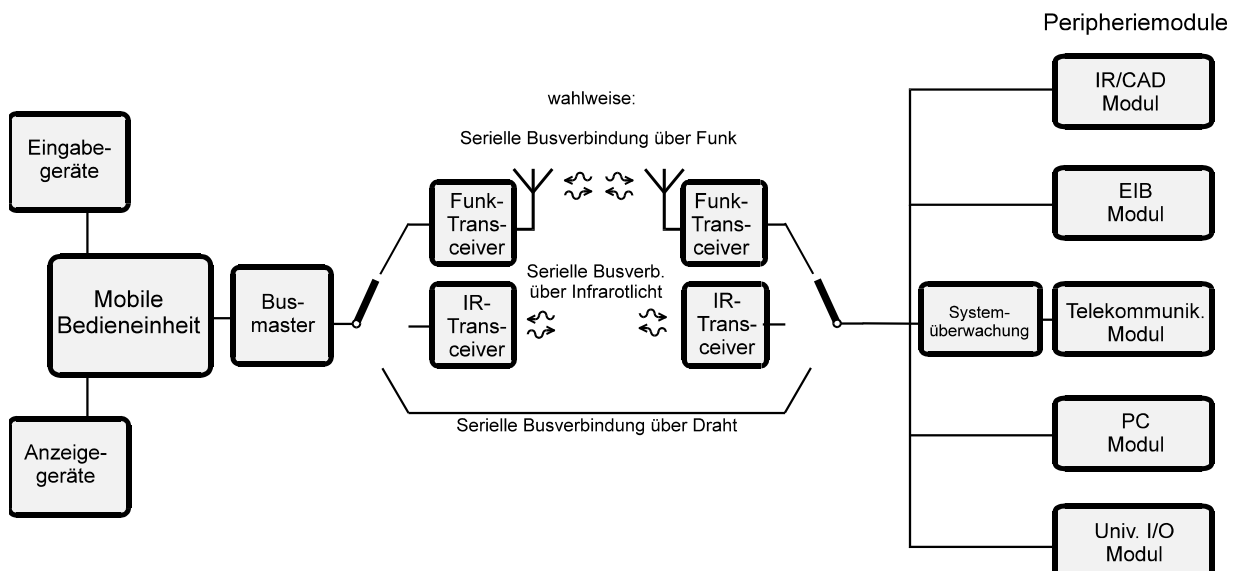


Abbildung 1.2 Das AUTONOM - Baukastensystem

Auf der rechten Seite von Abbildung 1.2 sind verschiedene **Peripheriemodule** dargestellt. Diese Bausteine sind für die Ausführung der Funktionen des Systems zuständig. Beispiele sind: Ein Modul zur Ansteuerung der Gegensprechanlage, ein Modul zur Ansteuerung des Telefons mit integrierter Notruffunktion, ein Modul zur Fernsteuerung verschiedener Geräte mittels Infrarotsignalen, ein Modul zum Anschluß an ein Installationsbussystem (EIB), ein Modul zur Ansteuerung eines Computers durch Emulation von Tastatur und Maus etc.

Die **Verbindung** zwischen der mobilen Bedieneinheit und den Peripheriemodulen kann im einfachsten Fall über ein Kabel erfolgen, ist aber auch über Infrarot- oder Funksignale möglich.

1.3.3 Funktionen von AUTONOM

Die grundlegenden Funktionen umfassen die Bereiche Kommunikation, Steuerung von Geräten, Sicherheit und Computeranwendung. Um einen raschen Überblick zu geben, werden sie im folgenden schlagwortartig erläutert.

Kommunikation

- Bildsymbolkommunikation (z. B. BLISS)
- Sprachsynthese
- Benutzung von Telefon, Gegensprechanlage,...

Steuerung von Geräten

- Fernsteuerung von HiFi/TV/Video, Beleuchtung, Jalousien, Heizung,...
- Rollstuhlsteuerung
- Steuerung von Hilfsmitteln basierend auf Robotertechnologie
- Anbindung an Installationsbussysteme

Sicherheitsfunktionen

- Telefonnotruf, Totmanneinrichtung
- Verwendung von Gegensprechanlage
- Fernsteuerung des Torschlosses

Computeranwendung

Emulation von Tastatur und Maus

- Lesen elektronischer Bücher
- Verwendung von interaktiver Schulsoftware
- Teleshopping, Telebanking, Telebooking
- Korrespondenz, persönliche Notizen, Spiele, ...

1.3.4 Die Benutzeroberfläche

Die Benutzeroberfläche muß individuell auf den jeweiligen Benutzer zugeschnitten werden können. Verschiedene Eingabegeräte und Selektionsverfahren (scanning, zeigen, ...) müssen zu Verfügung stehen.

Grundsätzlich ist die Benutzeroberfläche von AUTONOM ähnlich einer üblichen modernen Oberfläche aufgebaut. Sie besteht ebenfalls aus Menüs, die eine Art von Icons enthalten. Icons sind grafische Symbole mit einer zugeordneten Funktion. (z. B. einem Kommando, das ausgeführt wird).

Damit mehrfachbehinderte Menschen optimal unterstützt werden können, müssen in das System integrierte Geräte, Funktionen und auch Meldungen (wie z. B. „Achtung: das Telefon läutet“) in einer multimodalen Form präsentiert werden. Das heißt sie können dem Benutzer nicht nur über grafische Symbole, sondern auch über Text, Klänge oder gesprochene Sprache angezeigt werden.

Da die Bedürfnisse extrem individuell sind, ist ein Konfigurationsprogramm Bestandteil des AUTONOM-Systems, das den Aufbau völlig individueller Benutzeroberflächen gestattet. Das Konfigurationsprogramm läuft unter MS-Windows, ist leicht zu bedienen und ermöglicht Therapeutinnen, Therapeuten und anderen Personen mit guter Kenntnis des Benutzers die Erstellung und Adaptierung der Benutzeroberfläche. Dies geschieht durch Erzeugen von Menüs, Plazieren von grafischen Symbolen, Auswählen und Zuordnen von Befehlen, Klängen oder Textelementen usw.

Ein leistungsfähiges technisches Assistenzsystem wie AUTONOM erfordert ein gewisses Maß an Training und Schulung des Benutzers. Die unkomplizierte aber effiziente Möglichkeit zur flexiblen Gestaltung der Oberfläche von AUTONOM ermöglicht einen pädagogischen Aufbau des Trainingsprogrammes: Es kann mit einer sehr einfachen Oberfläche begonnen werden, mit zunehmender Erfahrung des Benutzers können laufend weitere Funktionen hinzugefügt werden. Dadurch wird auch die Integration in Therapieprogramme optimal unterstützt.

1.3.5 Menschen mit Mehrfachbehinderung

Das AUTONOM-System ist für Menschen mit Bewegungs- und Mehrfachbehinderung ausgelegt. Wie werden mehrfachbehinderte Menschen unterstützt?

- **Menschen mit zusätzlicher Sehbehinderung**
werden durch die Möglichkeit einer vergrößerten Darstellung der Icons bis hin zur bildschirmfüllenden, kontrastreichen Darstellung eines einzelnen Icons unterstützt. Für blinde Menschen kann eine komplette akustische Oberfläche aufgebaut werden.
- **Menschen mit zusätzlicher geistiger Behinderung**
werden durch die Möglichkeit des Aufbaus vereinfachter Menüs und durch Icons mit kompletten in sich abgeschlossenen Funktionen unterstützt. Spezielle leicht assoziierbare Grafiken und die Verwendung von Touchscreens erleichtern die Benutzung.
- **Menschen, die an Demenz leiden,**
werden durch die Möglichkeit, das System Schritt für Schritt zu vereinfachen und so immer den Fähigkeiten anzupassen, unterstützt.
- **Menschen mit zusätzlicher Sprechbehinderung**
werden durch die eingebaute Sprachausgabe unterstützt
- **Menschen mit zusätzlicher Sprachbehinderung**
werden durch die Möglichkeit, umfangreiche Bildsymbolkommunikationstafeln in das System zu integrieren, unterstützt.

2. Aufgabenstellung und Ergebnis

2.1 Anforderungen an das Spracherkennungsmodul

Das Hauptproblem für bewegungsbeeinträchtigte Personen ist das mechanische „handling“ technischer Geräte. Eine Möglichkeit, diese Geräte steuerbar zu machen, besteht im Einsatz verschiedener Speziialschalter z.B. Saug-Blas-Schlalter, Zungenschalter, Lidschlagschalter,...(siehe [9]). Durch die Benützung der Sprache als Steuerungsmittel kann der Bedienungskomfort jedoch wesentlich erhöht werden.

Bei der Realisierung einer solchen Sprachsteuerung sollen folgende Anforderungen erfüllt werden:

- In einem Lernmodus sollen verschiedene Sprachmuster (= benötigte Befehle in unterschiedlicher Aussprache) gespeichert werden können, damit ein flexibler Einsatz des Systems möglich ist.
- Gerät durch ein Schlüsselwort in den Ruhezustand versetzen.
- Ausführung eines Befehls erst nach einer Bestätigung
- Resetwort, um alle Schaltausgänge in einen vordefinierten Zustand zu bringen
- exakte Unterscheidung zwischen gesprochenem Befehl und Hintergrundgeräusch
- Möglichkeit eines Inselbetriebs ohne PC.
- Möglichkeit einer Einbindung in das AUTONOM Projekt
- Im Inselbetrieb sollten bis zu 8 verschiedene Verbraucher geschaltet werden.
- Schaltausgänge wahlweise konstant schalten, oder gepulst für eine vordefinierte Zeitspanne (Glocke, Fensteröffner, Bett verstellen).
- Kein großer Installationsaufwand erforderlich.
- Möglichst einfache Bedienung, auch durch ungeschulte Personen.

Entwurf einer Architektur:

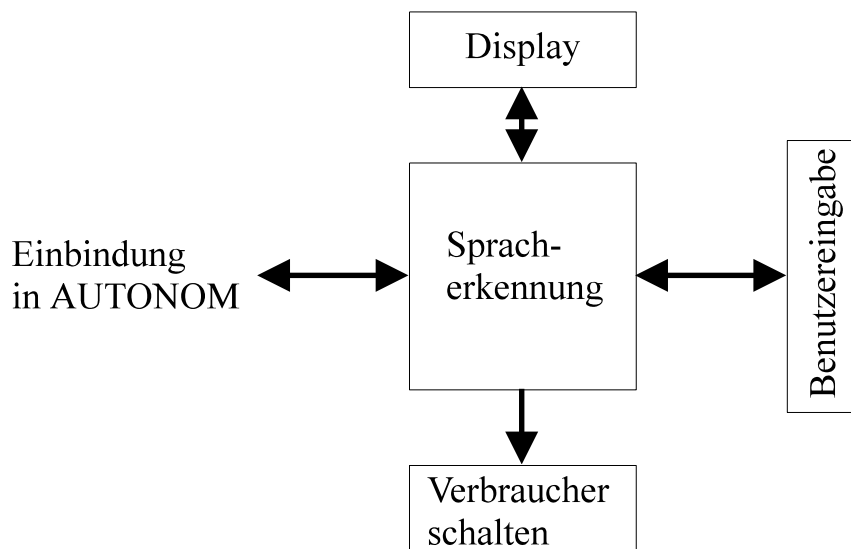


Abbildung 2.1 Blockschaltbild der Spracherkennung

2.2 Auswahl einer Spracherkennung

Es wurden folgende Systeme untersucht (siehe auch [13]):

- IBM Voice Type [24]
Das IBM Voice Type ist für die Textverarbeitung am PC gedacht. Es ist die komplette Eingabe eines Textes durch Sprache möglich. Das Programm ist lernfähig, das heißt die Eigenheiten der Sprache des Benutzers wird registriert und für spätere Erkennungen abgespeichert. Ein über 7000 Begriffe umfassendes Wörterbuch ermöglicht die Erkennung von bis zu 2000 benutzerdefinierter Wörter und den zugehörigen Befehlen. Ein 80 000 Worte umfassendes Wörterbuch ist ebenfalls vorhanden. Das System benötigt einen leistungsfähigen PC.
- Windows Sound System [25]
Das Windows Sound System ist eine Spracherkennung, die unter Windows arbeitet. Das Programmpaket beinhaltet neben der Spracherkennung noch Funktionen, um z.B. das Spektrum eines Wortes bearbeiten zu können. Die eigentliche Spracherkennung ist von der Fa. Dragon entwickelt worden. Das System ist lernfähig. Der Hersteller wäre auch bereit für einen speziellen Anwenderfall eine Runtime Version, die sich nur auf die Spracherkennung beschränkt, zur Verfügung zu stellen.
- Siemens Neural Net Speech Recognizer [2]
Dieser Baustein wurde für das Autotelefon entwickelt und ermöglicht, eine Telefonnummer auszuwählen. Er kann von einem Mikrokontroller angesteuert werden. Es können maximal 64 verschiedene Wörter eingelernt werden. Der Platzbedarf beschränkt sich auf eine Platine von 60 x 70mm. Das System ist im Vergleich zu den vorgenannten äußerst preiswert.

Die Wahl fiel auf den Siemens Neural Net Speech Recognizer wegen der Möglichkeit eines Betriebs ohne PC und des geringen Preises. 64 verschiedene Wörter reichen für AUTONOM III auf jeden Fall. Weiters ausschlaggebend war die geringe Nebengeräuschempfindlichkeit.

Als Hostprozessor wurden folgende Modelle untersucht:

- Siemens 80C501 [26]
- Siemens 80C537 [5]
- Microchip PIC 16C57 [27]

Der PIC 16C57 wäre insofern eine interessante Lösung gewesen, da das PICSTART Development System nur 1300 öS kostet und sich der Baustein mit ca. 55 öS zu Buche schlägt. Jedoch hat der PIC 16C57 zu wenige Ports und ist für ein externes ROM nicht ausgelegt.

Der 80C501 hat zu wenige Ports um Display, Schalter, Relais, Serielle Schnittstelle, RAM und den Spracherkennungsbaustein zu bedienen. So fiel die Wahl auf den 80C537 Microcontroller, zumal ein Emulator am Institut vorhanden ist.

2.3 Ergebnis

2.3.1 Realisierung mit dem Prototyp des Spracherkennungsbausteins

Diese Realisierung zeigte ein gutes Ergebnis in Bezug auf die Spracherkennung:

- Sehr geringe Nebengeräuschempfindlichkeit.
- Einfache Bedienung.
- Gute Erkennensicherheit, wenn es sich tatsächlich um eines der gespeicherten Wörter handelt.

Jedoch traten folgende Schwachpunkte auf:

- Das Gerät ist langsam, da es für die Auswertung eines gesprochenen Wortes ca. 3 s braucht.
- Der Baustein muß vor jeder Erkennung angestossen werden.
- Es steht für die Spracheingabe nur ein bestimmter Zeitschlitz von 2 s zur Verfügung. Ist dieser Zeitschlitz verpaßt muß der Baustein neu angestossen werden.
- Das Gerät mißt den Dynamikumfang des Eingangsignals. Kommt es zu dem Ergebnis, daß es sich um Sprache und nicht um ein Hintergrundgeräusch handelt, wird das Wort einer Speicherplatznummer zugeordnet ohne die Wahrscheinlichkeit einer solchen Zuordnung zu berücksichtigen.
- Anzahl verschiedener Wörter auf 15 beschränkt.
- Der Platzbedarf ist relativ groß, was aber nicht so entscheidend ist.

2.3.2 Realisierung mit dem Serienmodell

Das Serienmodell zeigt einen anderen Aufbau als der Prototyp. Die in Kapitel 2.3.1 genannten Schwachpunkte wurden zum Teil behoben:

- Die Schnelligkeit der Auswertung ist erhöht worden. Sie liegt jetzt bei ca. 0,5 s
- Der Baustein gibt als Antwort nicht nur die Speicherplatznummer, sondern auch die Wahrscheinlichkeit der Erkennung, zurück.
- Die maximale Anzahl verschiedener Wörter liegt jetzt bei 64, was für diesen Anwendungsfall ausreichend ist.
- Der Platzbedarf wurde auf eine Platine von 60 x 70 mm reduziert.

Da der Baustein für das Autotelefon konzipiert wurde, blieb jedoch als wesentlicher Schwachpunkt vorerst die Tatsache, daß der Baustein angestossen werden muß und dann nur ein bestimmter Zeitschlitz zum Sprechen des Wortes zur Verfügung steht.

Die Fa. Siemens änderte jedoch 1994 die Software des Bausteins, die in einem PROM abgelegt ist und nachgerüstet werden konnte. Nun kam der wesentliche Durchbruch für die AUTONOM Spracherkennung mit zwei Neuerungen:

- Ständiges Mithören ist nun möglich, da das Anstossen nun von der Software des Host Prozessors erfolgt und das Sprachkommando zu einem beliebigen Zeitpunkt kommen kann.
- Die zurückkommenden Parameter wurden noch durch eine Correlation und ein spektrales Moment ergänzt, was die Sicherheit einer Bewertung weiter erhöht.

Erste Selbstversuche brachten folgendes Ergebnis:

- Das Gerät wurde auf der ifabo (Internationale Fachmesse für Büro, Computer, Telecom) vom 25.-28. April 1995 in Wien ausgestellt. Dabei zeigte sich, daß das Gerät auch bei dem sehr starkem Hintergrundlärm einwandfrei funktionierte und es zu keinen Fehlschaltungen kam.
- Das Gerät wurde im privaten Wohnzimmer neben dem auf normaler Lautstärke laufenden Fernseher betrieben. Die Sprache, die aus dem Fernseher kam, beeinflusste das Gerät nicht. Auf die Kommandos des Benutzers reagierte es jedoch einwandfrei.
- Die Akustik des Raumes spielt eine große Rolle. Daraus ergibt sich, daß das Gerät in dem Raum, in dem es benutzt wird, auch trainiert werden soll.
- Ähnlich klingende Stimmen werden durchaus auch akzeptiert, allerdings bei einem großen Unterschied der Stimmen (z.B. auf Männerstimme trainiert und Sprachkommando von Frauenstimme gesprochen) ist keine Funktion mehr möglich.

Bei den Tests wurden folgende Wörter, entsprechend einer fiktiven Anwendung trainiert:

Wort 1	Licht
Wort 2	Lampe
Wort 3	Fenster
Wort 4	Jalousie
Wort 5	Ventilator
Wort 6	Fernseher
Wort 7	Video
Wort 8	Alarm
Resetwort	Reset
positive Bestätigung	Ja
negative Bestätigung	Nein
Schlüsselwort	Nomi

Der nächste Schritt ist sicherlich der, das Gerät einem Praxistest zu unterziehen und die sich daraus ergebenden Verbesserungswünsche einfließen zu lassen.

3. Der Prototyp des Spracherkennungsbausteins von Siemens [1]

Der Spracherkennungsbaustein basiert auf einem neuronalen Netz. Der Begriff Neuron stammt aus dem Griechischen und ist die Bezeichnung für eine funktionelle Einheit des Nervensystems. Ein Neuron besteht aus der Nervenzelle, der Nervenfasern und Verästelungen (Synapsen), die zur Weiterleitung der Reize dienen. Nach diesem Vorbild wurden „digitale Nervensysteme“ für Computer entwickelt, die man neuronale Netze nennt. Mit ihrer Hilfe können Computer lernen, Entscheidungen zu treffen. Gewisse Parameter dieser Elemente können sich in einer Lernphase automatisch so einstellen, daß bestimmte am Eingang anliegende Merkmalsvektoren ein bestimmtes Ergebnis am Ausgang liefern.

Neuronale Netze sind sehr gut für die Spracherkennung geeignet. Sie erweisen sich vor allem dann als besonders erfolgreich, wenn die Testmuster durch Störungen (etwa Umgebungsgeräusche) verfälscht sind.

Zur Steigerung der Leistungsfähigkeit wertet man außer der akustisch-phonetischen Information noch weitere Informationsquellen aus. Eine sehr wichtige ist das aufgabenbezogene Wissen. Bei den meisten Anwendungen ist der Einsatzbereich inhaltlich begrenzt, so daß es nur wenige zulässige Wörter gibt (siehe [30]).

Die Fa. Siemens stellte für erste Versuche einen Prototyp des Spracherkennungsbausteins zur Verfügung (Prototyp des NNSR (Neural Net Speech Recognizer)). Das Interface zum Host Prozessor ist allerdings gänzlich anders aufgebaut wie beim späteren Serienbaustein (siehe Kapitel 5 dieser Arbeit). Auch die Funktionsweise ist etwas anders als beim Serienbaustein.

Es können 12 verschiedene Wörter, mit maximal 15 Sprachmustern pro Wort, eingelernt werden. Jedes Wort ist einer Speicherplatznummer von 1 bis 99 zuzuordnen.

3.1 Host Interface

3.1.1 Pinbelegung

Pin 2	+12 V
Pin 7	Masse
Pin 8	serieller Dateneingang
Pin 15	Masse
Pin 16	serieller Datenausgang

3.1.2 Datenformat

Übertragungsart:	asynchron	
Datenrate:	2400 baud	
Format eines Byte:	1 Startbit	(5V ⇒ logisch 0)
	8 Datenbits	(zuerst LSB)
	1 Bit parity even	
	1 Stopbit	(0V ⇒ logisch 1)

3.2 Befehle

3.2.1 Format eines Befehls

1 Startbyte	N x Informationsbyte	1 Stopbyte
02h	Code	0Dh

Das Informationsbyte muß innerhalb einer Meldung einige Male wiederholt werden.

Es wird zwischen kurzen (< 0,5 s) und langen Meldungen (>0,5 s) unterschieden. Für kurze Meldungen hat N den Defaultwert 3, für lange Meldungen ist N = 220.

In Folge handelt es sich um kurze Meldungen soweit nicht anders angegeben.

3.2.2 Befehle

Befehl	Informationsbyte	Bemerkungen
Select	2A h	
Speicherplatznummer folgt	23 h	
0	30 h	
1	31 h	
2	32 h	
3	33 h	
4	34 h	
5	35 h	
6	36 h	
7	37 h	
8	38 h	
9	39 h	
Ende der Speicherplatznummr	3C h	
Löschen	3A h	lange Meldung
Playback	3B h	
Speichern	3E h	
Aufzeichnen	60 h	

3.3 Bedienung des Gerätes

Nach Anlegen der Versorgungsspannung befindet sich das Gerät im Betriebsmodus (siehe 3.3.8 und 3.3.9).

3.3.1 Aufnahme von Sprachmustern beginnen

In den Programmiermodus gelangt man mit folgender Befehlsfolge (siehe 3.2 Befehle):

Select, Speicherplatznummer folgt, x, x, Ende der Speicherplatznummer
x... Zahl von 0-9

3.3.2 Sprachmuster lernen

Die Aufnahme eines Sprachmusters wird durch den Befehl **Aufzeichnen** erreicht.

Nach der Aufnahmezeit von 1 s erfolgt die automatische Wiedergabe dieses Sprachmusters auf dem Lautsprecher.

Für eine Kontrollwiedergabe siehe 3.3.3 Playback

3.3.3 Playback

Mit dem Befehl **Playback** können soeben aufgesprochene Muster nochmals angehört werden.

3.3.4 Sprachmuster speichern

Der Befehl **Speichern** speichert das Sprachmuster, das sich im Aufnahmespeicher befindet, ab. Die erfolgreiche Ausführung wird mit einem kurzen Ton (Frequenz 1kHz; Dauer 0,1 s) quittiert.

3.3.5 Selektion eines Speicherplatzes

Sind alle gewünschten Sprachmuster zu diesem Wort abgespeichert wird mit der Befehlsfolge

Select, Speicherplatznummer folgt, x, x, Ende der Speicherplatznummer
x... Zahl von 0-9

ein neues Wort selektiert, und mit den vorstehenden Befehlen bearbeitet.

3.3.6 Wort löschen

Der Befehl **Löschen** entfernt alle Sprachmuster, die zu dem selektierten Wort gehören. Der Programmiermodus wird mit diesem Befehl wieder verlassen.

3.3.7 Aufnahme von Sprachmustern beenden

Mit der Befehlsfolge

Select, Speicherplatznummer folgt, 0, Ende der Speicherplatznummer

wird der Programmiermodus wieder verlassen.

Zuvor erfolgt ein Lernvorgang, dessen Ende mit einem kurzen Quittierungston (Frequenz 1kHz; Dauer 1 s) angezeigt wird. Danach befindet sich das Gerät wieder im Betriebsmodus.

3.3.8 Reset

Die Befehlsfolge **Select, Select, Select** löscht alle vorhandenen Sprachmuster.

Der Resetbefehl kann im Betriebsmodus gegeben werden.

3.3.9 Erkennung

Eine Erkennung wird mit dem Befehl **Aufzeichnen** im Betriebsmodus gestartet.

Der NNSR Prototyp nimmt dann ein Sprachmuster mit der Dauer von 1 s über das angeschlossene Mikrofon auf.

Das erkannte Wort wird als Platznummer am seriellen Ausgang ausgegeben:

Clear, Playback, Playback, x, x, Ende der Speicherplatznummer
x... Zahl von 0-9

Es erfolgt auch die akustische Wiedergabe des zugehörigen Sprachkommandos über den Lautsprecher.

Sind jedoch keine gelernten Wörter vorhanden (mindestens 2 Wörter mit je einem Sprachmuster) wird ein Fehlbedienungsston (Frequenz 2kHz; Dauer 0,5 s) ausgegeben.

4. Aufbau einer Spracherkennung mit dem Prototyp von Siemens

Im folgenden Kapitel werden der Versuchsaufbau und die verwendeten Bauelemente näher beschrieben.

4.1 Grundkonzept

Der Prototyp des NNSR wird von einem PC angesteuert. Die Ergebnisse der Erkennung werden wieder an den PC zurückgeliefert und auf dem Bildschirm angezeigt. Die Handshakeleitungen der RS 232 Schnittstelle werden nicht verwendet (siehe Abbildung 4.2). Das Programm wurde in Turbo Pascal geschrieben. Ein Listing dieses Programms findet sich in Anhang A. Die Verbindung mit dem Versuchsaufbau erfolgte über die serielle Schnittstelle des PC (Com 1). Der Versuchsaufbau wurde auf einer Hirschmann-Steckplatte installiert (siehe Abbildung 4.2).

Blockschaltbild der Spracherkennung

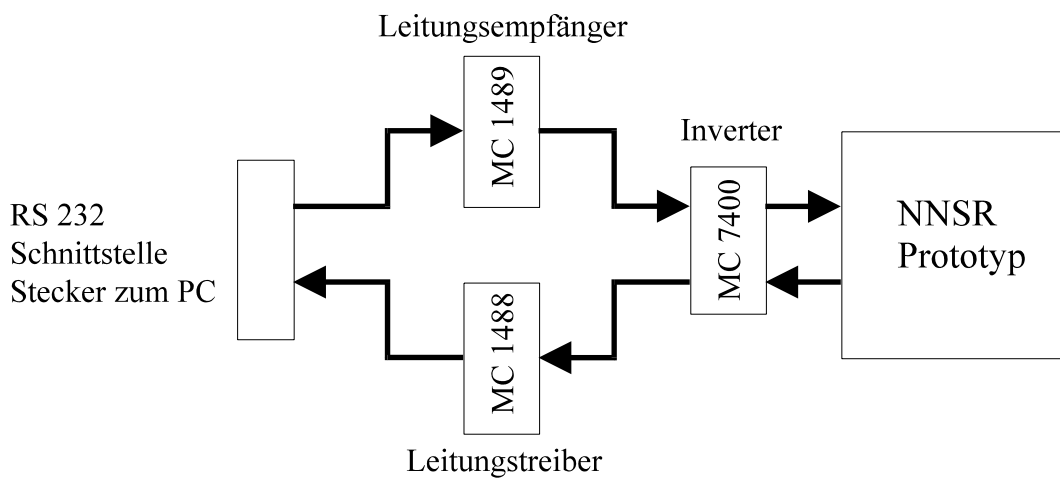


Abbildung 4.1 Blockschaltbild der Spracherkennung

4.2 Verwendete Bausteine

4.2.1 Der NNSR Prototyp

Der Prototyp des Spracherkennungsbausteines wurde bereits im Kapitel 3 dieser Arbeit ausführlich behandelt.

4.2.2 Der Leitungsempfänger, der Leitungstreiber und der Inverter

Die Signale vom und zum NNSR Prototyp müssen einen Pegel von 0V und 5V haben. Die Daten auf der seriellen Schnittstelle werden mit einem Pegel von +12V und -12V übertragen. Der Grund für diese höheren Spannungen liegt in der erhöhten Datensicherheit auch bei langen Übertragungstrecken.

Der Leitungsempfänger

Der 4-fache Leitungsempfänger bringt die Signale auf den benötigten 5V Pegel zurück. Zur Verwendung kam ein MC 1489 Baustein.

Der Leitungstreiber

Der 4-fache Leitungstreiber invertiert die logischen Signale vom NNSR-Prototyp und bringt sie auf den höheren Pegel.

Zur Verwendung kam ein MC 1488 Baustein.

Der Inverter

Der Inverter ist notwendig um die Signale von und zum NNSR Prototyp logisch richtig aufzubereiten.

5 V \Rightarrow logisch 0

0 V \Rightarrow logisch 1

Zur Verwendung kam ein NAND Gatter MC 7400. Die Beschaltung ist aus Abbildung 4.2 zu sehen.

4.3 Aufbau auf einer Hirschmann-Steckplatte

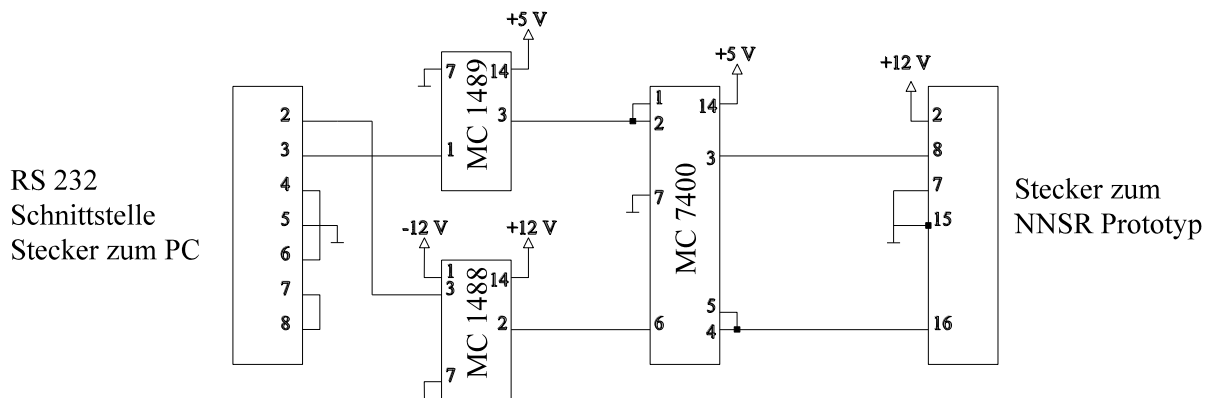


Abbildung 4.2 Versuchsaufbau

5. Der Spracherkennungsbaustein von Siemens [2]

5.1 Architektur

Für die Spracherkennung wurde ein NNSR (Neural Net Speech Recognizer) der Fa. Siemens herangezogen. Der NNSR basiert auf einen DSP (Digital Signal Processor) mit Host Interface Port der Fa. ANALOG DEVICES. Er wurde für die Erkennung einzelner Wörter, wie Namen, Befehle, Zahlen gebaut. Auf der Platine befinden sich folgende Einheiten:

DSP (Digital Signal Processor)
A/D Converter
D/A Converter
Anti Aliasing Filter
Sample & Hold
RAM für Sprachmuster und Neural Net Weights
RAM Batterie
Batterie Controller

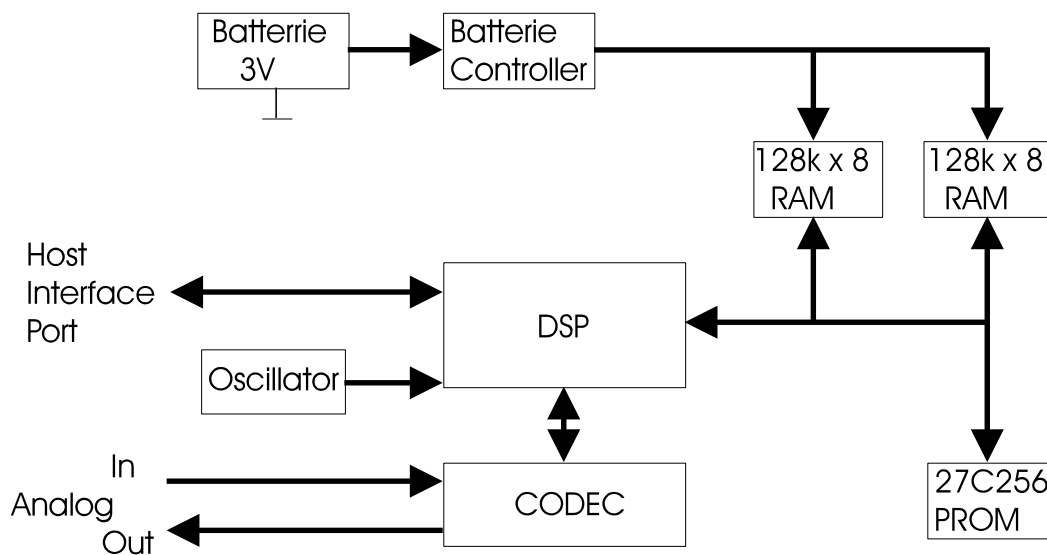


Abbildung 5.1 Blockdiagramm des NNSR

Es können 320 Sprachmuster gespeichert werden. Die Anzahl der Sprecher wird nur durch die Kapazität des NNSR begrenzt. Die 320 Sprachmuster können z.B. auf 40 Wörter, mit je 4 Sprachmuster von 2 Sprechern, aufgeteilt werden. Der Spracherkennungsbaustein ist sehr unempfindlich für Nebengeräusche. Die minimale Anzahl verschiedener Wörter ist 2. Die maximale Anzahl 32 wenn acoustic_feedback gesetzt ist, 64 wenn acoustic_feedback nicht gesetzt ist.

5.2 Host Interface

Das HIP (Host Interface Port) des NNSR ist ein paralleles I/O Port (siehe [3]). Durch das HIP kann der NNSR vom Host Prozessor memory mapped als Peripheriebaustein angesprochen werden.

Die Kommunikation mit dem Host Prozessor wird über die HMSG (Host message) und HREADY Leitungen abgewickelt.

150ms nach einem Reset sendet der NNSR ein „Ready“-Signal. Ein Befehl vom Host Prozessor mit der HIP Adresse 0 (opcode Byte) erzeugt im NNSR einen Interrupt, der ihn zwingt den Befehl (opcode und Parameter) zu lesen. Deswegen muß bei Kommandos, die mehrere Bytes lang sind, das Byte 0 als letztes gesendet werden.

Nachdem der NNSR einen Befehl erhalten hat, erzeugt er die Antwort und wartet bis der Host Prozessor bereit ist, eine Nachricht vom NNSR zu empfangen und die HREADY- Leitung setzt. Nachdem der Host Prozessor die Nachricht aus dem NNSR ausgelesen hat, ist der NNSR bereit, den nächsten Befehl zu empfangen.

Zwei Pins konfigurieren den NNSR für verschiedene Typen von Host Prozessoren.

HMD0 konfiguriert den Bus Strobe:

HMD0 = 0 separater Read und Write Strobe
HMD0 = 1 Read/ Write Strobe und Data Strobe

HMD1 konfiguriert die Adress/ Daten-Leitungen:

HMD1 = 0 separate Daten und Adress-Leitungen
HMD1 = 1 gemultiplexte Daten/ Adress-Leitung

Pin Belegung des Host Interface Port:

1	QRESET	Reset input
2	HD0	Daten/ Adressleitung 0
3	HD1	Daten/ Adressleitung 1
4	HD2	Daten/ Adressleitung 2
5	HD3	Daten/ Adressleitung 3
6	HD4	Daten/ Adressleitung 4
7	HD5	Daten/ Adressleitung 5
8	HD6	Daten/ Adressleitung 6
9	HD7	Daten/ Adressleitung 7
10	ALE	Address latch enable/ Host address 2
11	HA0	Host address 0
12	HA1	Host address 1
13	QHWR	write strobe
14	QHRD	read strobe
15	HMD1	address/data mode
16	HMD0	bus strobe select
17	QSEL	HIP select input
18	QHACK	HIP acknowledge
19	HMSG	Message to host available
20	HREADY	Host ready to receive
21	ANOUTA	Analog audio output a
22	ANOUTB	Analog audio output b
23	ANGND	Analog audio ground
24	ANIN	Analog audio input
25	GND	Ground
26	VCC	Power supply

5.3 Analog Audio Interface

Für optimale Ergebnisse sollte der Audio Eingang an einen Mikrofon Verstärker angeschlossen werden. Der Mikrofon Verstärker sollte im Bereich von 0,2 bis 3,4 kHz eine gleichmäßige Verstärkung aufweisen. Der Audio Ausgang kann an einen Ausgangsverstärker, oder direkt an einen 300 Ohm (zw. a und b) bzw. 600 Ohm (a oder b gegen Masse) Lautsprecher angeschlossen werden.

5.4 Host Befehle und NNSR Antworten

Host Befehle	NNSR Antwort
Send_status_info	Status_info
Send_configuration	Configuration
Configure_new	Ready
Initialize_words	Ready
Send_word_info	Word_info
Clear_word	Ready
Record	Ready
Playback_immediate	Ready
Playback_word	Ready
Store_pattern	Ready
Learn	Ready
Listen	Word_recognize

5.5 Host Befehle

Die Byte Nummern korrespondieren mit den Host Interface Port Adressen.

5.5.1 Send_status_info

Byte 0 opcode 00h

Überprüft den aktuellen Status des NNSR.

5.5.2 Send_configuration

Byte 0 opcode 01h

Überprüft die aktuelle Konfiguration des NNSR

5.5.3 Configure_new

Byte 0 opcode 02h

Byte 1 acoustic_feedback 00h acoustic_feedback set
anders acoustic_feedback reset

Byte 2 number_of_words xxh Anzahl verschiedener Wörter die erkannt werden
sollen. (02h bis 20h mit acoustic_feedback
02h bis 40h ohne acoustic feedback)

Konfiguriert den NNSR. Wenn acoustic_feedback gesetzt ist, wird das erste trainierte Sprachmuster jedes Wortes gespeichert, um es nach einer Erkennung wiederzugeben. Wenn die gewünschte Konfiguration ungültig ist, bleibt die alte Konfiguration und der Status unverändert. Wenn die Konfiguration geändert wird, werden alle trainierten Sprachmuster gelöscht, und der NNSR befindet sich im untrainierten Zustand.

Ausnahmen hiervon:

- die neue Konfiguration entspricht der alten
- nur `acoustic_feedback` wird zurückgesetzt

Beeinflusste Flags:	<code>acoustic_feedback</code>	abhängig von der Konfiguration
	<code>nnsr_trained</code>	unverändert, wenn die Konfiguration kompatibel gelöscht, wenn Konfiguration gültig aber inkompatibel
	<code>recording_available</code>	gelöscht, bei gültiger Konfiguration

5.5.4 Initialize_words

Byte 0 opcode 03h

Löscht die Wort Liste, alle trainierten Sprachmuster und bringt den NNSR in den untrainierten Status.
Die Konfiguration bleibt erhalten.

5.5.5 Send_word_info

Byte 0	opcode	10h	
Byte 1	word_index	xxh	Index des gewünschten Wortes (01..number_of_words)h

Verlangt Information über das gewünschte Wort.

5.5.6 Clear_word

Byte 0	opcode	11h	
Byte 1	word_index	xxh	Index des Wortes das gelöscht werden soll (01..number_of_words)h

Entfernt das gewünschte Wort von der Wort Liste und löscht alle dazugehörigen Sprachmuster.

Beeinflusste Flags:	<code>nnsr_trained</code>	gelöscht, wenn die Wort Liste nach der Ausführung leer ist
---------------------	---------------------------	---

5.5.7 Record

Byte 0 opcode 18h

Startet eine Sprachaufnahme. Das Sprachaufnahmefenster ist 1,792 s geöffnet. Die maximale Energie der aufgezeichneten Sprache wird normiert und die Aufnahme im Aufnahme Speicher abgelegt.

Beeinflusste Flags:	<code>recording_available</code>	gesetzt
---------------------	----------------------------------	---------

5.5.8 Playback_immediate

Byte 0 opcode 19h

Startet ein Playback des Inhaltes des Aufnahme Speichers.

5.5.9 Playback_word

Byte 0	opcode	1Ah	
Byte 1	word_index	xxh	Index des gewünschten Wortes (01..number_of_words)h

Startet ein Playback des ausgewählten Wortes (wenn acoustic feedback gesetzt ist). Der Inhalt des Aufnahme Speichers wird mit dem Sprachmuster des ausgewählten Wortes überschrieben.

Beeinflusste Flags: recording_available gelöscht, wenn erfolgreich ausgeführt

5.5.10 Store_pattern

Byte 0	opcode	1Bh	
Byte 1	word_index	xxh	Index des Wortes für das der Inhalt des Aufnahme Speichers gespeichert werden soll

Speichert den Inhalt des Aufnahme Speichers auf die angesprochene Wortnummer. Wenn acoustic_feedback gesetzt, und noch kein Sprachmuster für dieses Wort gespeichert ist, wird eine 16kbit/s Version für Playback erzeugt. Zusätzlich wird, wenn das Sprachmuster für dieses Wort das erste ist, die Wort Nummer zur Wort Liste hinzugefügt.

Beinflusste Flags: recording_available gelöscht, wenn erfolgreich ausgeführt
 nnsr_trained gelöscht, wenn erfolgreich ausgeführt

5.5.11 Learn

Byte 0	opcode	1Ch	
--------	--------	-----	--

Startet die Lern Procedure. Sie muß ausgeführt werden, wenn das NNSR gelöscht ist und ein Listen Befehl kommen soll. Diese Procedure kann bis zu 5 Minuten dauern.

Beeinflusste Flags: nnsr_trained gesetzt, wenn erfolgreich ausgeführt
 gelöscht, sonst

5.5.12 Listen

Byte 0	opcode	1Eh	
--------	--------	-----	--

Startet eine Sprachaufzeichnung, wenn ein Pegel am Analog Eingang anliegt, gefolgt von einer Auswertung. Verarbeitete Länge der Sprache = 1,820 s

Beeinflusste Flags: recording_available gelöscht

5.6 NNSR Antworten

Der execution_success_code (Byte 1) enthält die Fehlermeldung. Wenn mehr als ein Fehler bei einem Befehl auftritt, wird nur der mit der höchsten Priorität (niedrigster Code) angezeigt.

5.6.1 Status_info

Byte 0	opcode	20h	
Byte 1	execution_success_code	00h	erfolgreich
Byte 2	number_of_free_words	xxh	Anzahl der Wörter, für die keine Sprachmuster gespeichert sind.
Byte 3	first_free_word_index	xxh	Index des ersten Wortes, für das keine Sprachmuster gespeichert sind.
Byte 4	nnsr_trained	00h	NNSR trainiert
		01h	NNSR nicht trainiert

Antwortet mit dem aktuellen Status des NNSR.

5.6.2 Configuration

Byte 0	opcode	21h	
Byte 1	execution_success_code	00h	erfolgreich
Byte 2	acoustic_feedback	00h	acoustic_feedback gesetzt
		anders	acoustic_feedback nicht gesetzt
Byte 3	number_of_words	xxh	maximale Anzahl verschiedener Wörter, die erkannt werden können
Byte 4	pattern_per_word	xxh	maximale Anzahl speicherbarer Sprachmuster pro Wort. (Diesen Wert errechnet der NNSR in Abhängigkeit von number_of_words und dem acoustic_feedback Flag)

Antwortet mit der aktuellen Konfiguration des NNSR.

5.6.3 Ready

Byte 0	opcode	28h	
Byte 1	execution_success_code	00h	erfolgreich
		01h	ungültige Konfiguration (in Verbindung mit Configure_new)
		02h	ungültiger Wort Index (in Verbindung mit Clear_word, Playback_word und Store_pattern)
		04h	acoustic_feedback ist nicht gesetzt (in Verbindung mit Playback_word)
		08h	Aufnahme Speicher leer (in Verbindung mit Playback_immediate, Playback_word, Store_pattern)
		10h	maximale Anzahl der Sprachmuster für dieses Wort gespeichert (in Verbindung mit Store_pattern)
		20h	keine Sprachmuster gespeichert (in Verbindung mit Learn)
		80h	keine Sprachaktivität erkannt (in Verbindung mit Store pattern)

Zeigt an, das der aktuelle Befehl ausgeführt wurde (Nur in Verbindung mit Befehlen, die keine Parameter zurückerwarten).

5.6.4 Word_info

Byte 0	opcode	30h	
Byte 1	execution_success_code	00h	erfolgreich
		02h	ungültiger Wort Index
Byte 2	word_index	xxh	Index des gewünschten Wortes
Byte 3	store_patterns	xxh	Anzahl der gespeicherten Sprachmuster

Zeigt wieviele Sprachmuster für ein selektiertes Wort gespeichert sind.

5.6.5 Word_recognized

Byte 0	opcode	3Fh	
Byte 1	execution_success_code	00h	erfolgreich
		40h	NNSR nicht trainiert
		80h	keine Sprache erkannt
Byte 2	word_index	xxh	Index des erkannten Wortes
Byte 3	probability_code	xxh	Code für die Wahrscheinlichkeit einer korrekten Erkennung
Byte 4	correlation_code	xxh	Code für die Correlation des Wortes
Byte 5	spectral._mom._code	xxh	Code für das spektrale Moment des Wortes

Antwortet mit der Nummer des erkannten Wortes.

probability_code:	00h	bestmögliche Erkennung
	0Fh	Erkennung zu 50% (schlechtester Fall)
		Werte < 0Ch stehen für zuverlässige Erkennung
correlation_code:	00h	schlechtester Fall
	bis 7Fh	bester Fall
		dabei ist zu beachten, daß kurze Worte eine höhere Correlation haben als lange.
spectral._mom._code:	00h bis 7Fh	
		Kurze Worte haben ein kleines Moment, lange ein großes.

Die Fa. Siemens gibt folgenden Algorithmus für eine praxisgerechte Auswertung einer Erkennung an:

$$\text{correlation_code} + (\text{spectral._mom._code} / 2) > 60h \quad \text{Schwelle für positive Erkennung}$$

6. Aufbau einer Spracherkennung mit dem NNSR

Im folgenden Kapitel werden, die einzelnen Bauteile, die für den Aufbau der Spracherkennung verwendet wurden, näher beschreiben.

6.1 Grundkonzept

Das Gerät besteht aus 2 Einheiten :

- Steuermodul (siehe Abbildung 8.1)
- Schaltmodul (siehe Abbildung 8.2)

Das Steuermodul ist in ein Aluminiumgehäuse ,in Eurokartengröße, eingebaut. Der Lautsprecher befindet sich im Gehäuse des Steuermoduls. Der Schaltplan, der Spracherkennungsplatine, ist Abbildung 11.4, in Kapitel 11, zu entnehmen.

Zur Ansteuerung der Peripheriegeräte wurde der sogenannte Hardware-Kern (siehe 6.2.1) von AUTONOM verwendet, da diese Platine bereits für andere AUTONOM-Anwendungen benutzt wird und leicht für diesen Zweck adaptierbar ist.

Für das Schaltmodul kam ein doppelseitiger Kabelkanal der Fa. Legrand zur Verwendung (siehe [18]). In diesem Kabelkanal befindet sich ein Netzgerät, zur Spannungsversorgung des Steuermoduls und der Relais, weiters die 8 Kontrollleuchten, die 8 Steckdosen, die 8 Relais und der Hauptschalter. Der Schaltplan des Schaltmoduls ist aus Abbildung 11.5 in Kapitel 11 zu ersehen.

Blockschaltbild der Spracherkennung

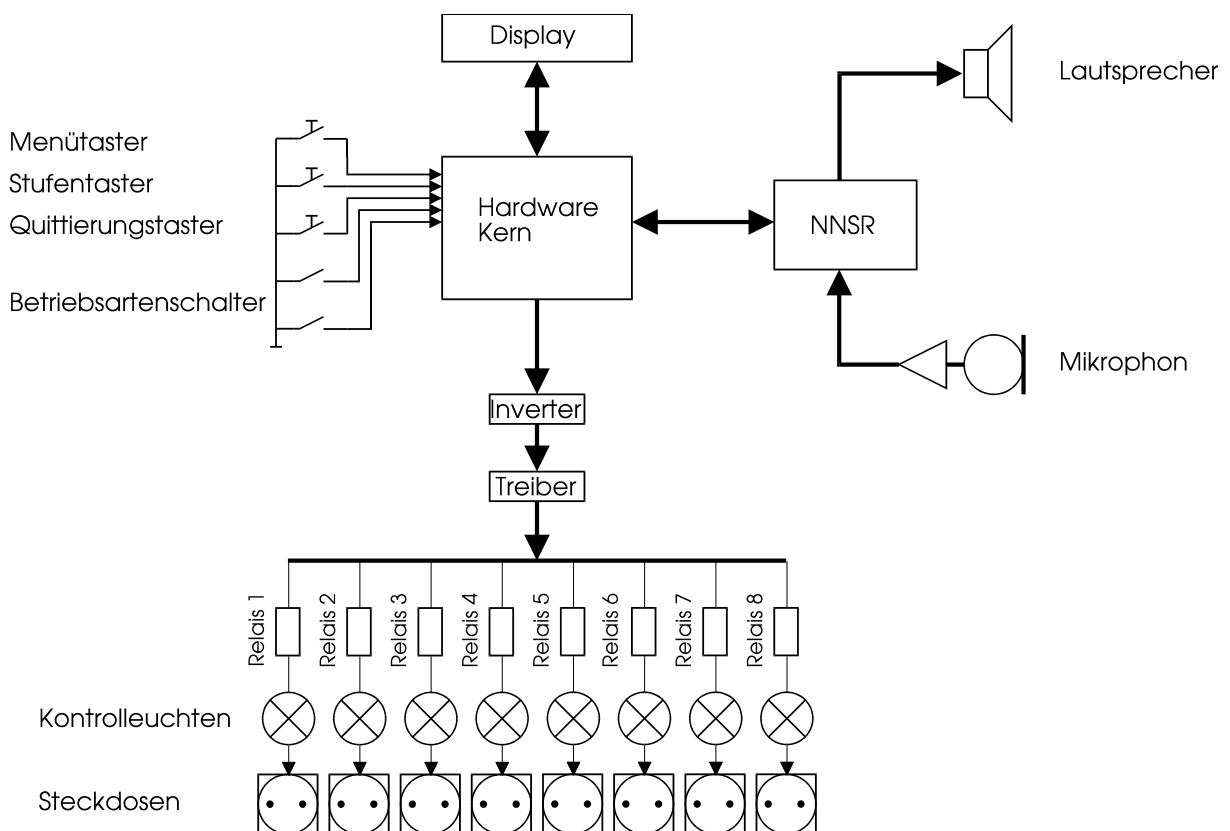


Abbildung 6.1 Blockschaltbild der Spracherkennung

6.2 Verwendete Bausteine

6.2.1 Der Hardware-Kern

Der Hardware-Kern wurde von Hrn. Jürgen Demuth für das AUTONOM-Projekt entwickelt.

Er beinhaltet folgende Einheiten:

- 80C537 Mikrokontroller der Fa.Siemens (siehe [5])
- EPROM mit dem Programm
- Jumper zur Auswahl des EPROM's
- gepufferte SRAM (siehe [17])
- Schwingquarz
- Spannungswandler
- 96-polige Federleiste als Verbindung zur Spracherkennungsplatine
- Stecker für Betriebsartschalter, Menütaster und Stufentaster
- MAX 232 und Stecker für RS 232
- Stecker für Resettaster

Die Daten- und Adressleitungen des NNSR, auf der Spracherkennungsplatine, können über die externen Speicheradressen FF00h, FF01h, FF02h, FF03h, FF04h und FF05h angesprochen werden. Die Handshakeleitungen hängen auf Port 1 und 3 des Mikrokontrollers. Der Betriebsartenschalter, der Menütaster und der Stufentaster hängen auf Port 6 des Mikrokontrollers. Das Display ist an Port 3 und 5 angeschlossen und der Inverter befindet sich an Port 4. Die Schwingfrequenz des Quarzes liegt bei 14,7456 MHz.

Die Stellung der Jumper für das EPROM ist der Abbildung 11.7 in Anhang B zu entnehmen. Der Schaltplan des Hardware-Kern befindet sich in Anhang B Abbildung 11.6.

Das gepufferte SRAM:

Die Daten zur Relaisstellung (siehe 8.2.2.5 Schaltausgänge definieren) müssen in ein gepuffertes RAM abgelegt werden, damit sie auch nach einer zeitweiligen Abschaltung des Gerätes noch vorhanden sind. Zur Verwendung kam ein DS1225AD 64 K Nonvolatile SRAM der Fa. DALLAS Semiconductor.

Anschließend noch eine Auflistung der verwendeten Pins, auf der 96-poligen Federleiste:

Pin	Hardware-Kern	Spracherkennungsplatine
A14	P3.2	Enable für Display
A15	P3.3	R / \overline{W} für Display
A16	P3.4	Register Select für Display
A17	P3.5	QRESET für NNSR
A18	P3.6	QHWR für NNSR
A19	P3.7	QHRD für NNSR
A23	$\overline{SEL1}$	QHSEL für NNSR
A26	ALE	ALE für NNSR
A30	VCC	5V Spannungsversorgung für NNSR und Display
A31	GND	Masse
A32	U++	12 V Spannungsversorgung für Hardware-Kern

Pin	Hardware-Kern	Spracherkennungsplatine
B22	AD0	HD0 für NNSR
B23	AD1	HD1 für NNSR
B24	AD2	HD2 für NNSR
B25	AD3	HD3 für NNSR
B26	AD4	HD4 für NNSR
B27	AD5	HD5 für NNSR
B28	AD6	HD6 für NNSR
B29	AD7	HD7 für NNSR
B30	VCC	5V Spannungsversorgung für NNSR und Display
B31	GND	Masse
B32	U++	12 V Spannungsversorgung für Hardware-Kern
C11	GND	GND für Displaykontrast
C12	P5.0	DB0 für Display
C13	P5.1	DB1 für Display
C14	P5.2	DB2 für Display
C15	P5.3	DB3 für Display
C16	P5.4	DB4 für Display
C17	P5.5	DB5 für Display
C18	P5.6	DB6 für Display
C19	P5.7	DB7 für Display
C26	P1.4	HMSG des NNSR
C28	P1.6	HREADY für NNSR
C29	P1.7	Pin für Taster 4
C30	VCC	5V Spannungsversorgung für NNSR und Display
C31	GND	Masse
C32	U++	12 V Spannungsversorgung für Hardware-Kern

6.2.2 Der Spracherkennungsbaustein

Der Spracherkennungsbaustein wurde bereits in Kapitel 5 dieser Arbeit ausführlich behandelt.

6.2.3 Das LCD-Display

Zur Verwendung kommt ein DMC16106A der Fa. Optrex (siehe [4]).

Es handelt sich dabei um ein Display mit 16 Zeichen in einer Zeile. Jedes Zeichen besteht aus 8x11 Punkten.

Blockschaltbild des Display

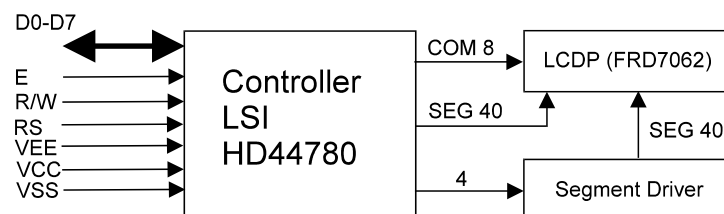


Abbildung 6.2 Blockschaltbild des Display

Das Display wird über die E, R/W und RS Leitung vom Mikrokontroller angesprochen. Die Zeichen werden im ASCII-Code über die Datenleitungen an das Display geschickt.

Genauere Informationen über die nötige Ansteuerung bietet [4].

6.2.4 Das Mikrophon

Der Spracherkennungsbaustein braucht an seinem Analog Eingang ein Mikrophon mit folgenden Daten:

Eingangswiderstand 20kOhm
max. Eingangsspannung 4,5 Vss
Frequenzbereich 0,2-3,4 kHz

Zur Auswahl standen 2 Mikrophone mit einem eingebauten Vorverstärker:

- Das ME 2 der Fa. Peiker
- Die AKG Maus Q 400 MK II T (siehe [14])

Die Fa. Siemens gibt für diese Version des NNSR jedoch an, daß das ME 2 die bessere Qualität liefert.

Notwendige Beschaltung des ME 2

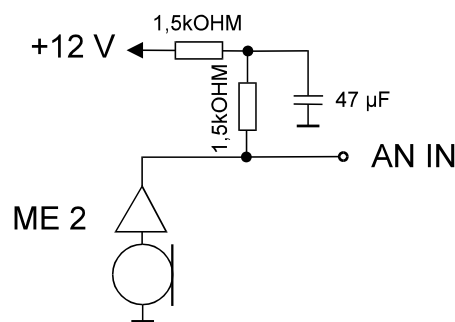


Abbildung 6.3 Beschaltung des Mikrophon

6.2.5 Der Lautsprecher

Der Spracherkennungsbaustein hat einen Analog Ausgang mit folgenden Daten (bei 1kHz):

maximale Ausgangsspannung:

a oder b gegen Masse 4 Vss
a gegen b 8 Vss

Ausgangswiderstand:

a oder b gegen Masse 1 Ohm/10µF
a gegen b 1 Ohm/ 5µF

Eingangswiderstand des Lautsprechers :

a oder b gegen Masse 600 Ohm
a gegen b 300 Ohm

Die Fa. AKG Acoustics Ges.m.b.H., Wien stellte dankenswerter weise den Prototyp eines 300 Ohm Lautsprechers zu Verfügung. Das Serienmodell wird im Sommer 1995 im Handel erhältlich sein. Dieser 300 Ohm Lautsprecher kann direkt, ohne Vorverstärker, an die Klemmen a und b des NNSR angeschlossen werden.

6.2.6 Der Inverter 74HC540 [15]

Der 74HC540 ist ein 8-fach Inverter. Der Baustein ist zwischen Mikroprozessor und Relais geschaltet, damit bei einem Reset des Mikroprozessors die Relais und damit die Ausgänge nicht auf „Ein“ gehen. Er wurde ausgewählt, da sich bei diesem Baustein alle Eingänge auf einer, und die Ausgänge auf der gegenüberliegenden Seite befinden. Das kommt den beschränkten Platzverhältnissen auf der Spracherkennungsplatine entgegen.

6.2.7 Der Treiber ULN2803A [16]

Der ULN2803A ist ein 8-fach Treiber mit Darlington-Transistoren:

- NPN Darlington-Transistoren
- Open-Collector Ausgänge
- Mit integrierter Supressdiode

Er befindet sich zwischen Mikroprozessor und den Relais.

6.2.8 Die Relais

Die Relais befinden sich im Schaltmodul und schalten die 8 Steckdosen und dazugehörigen Kontrollampen.

Die maximale Schaltleistung eines Relais liegt bei 250 W / 220V \approx .

Die Relaispule braucht eine Schaltspannung von 12 V-.

Zur Verwendung kamen JW1FSN Relais der Fa. Matsushita / Thailand.

7. Software

Hier wird in groben Zügen der Ablauf des Programms erklärt werden. (Listing siehe Anhang A).

Das Programm wurde in der Programmiersprache Assembler geschrieben und auf dem iceMASTER Emulator der Fa. MetaLink getestet. Ergänzende und weiterführende Literatur findet sich in [2], [4], [5], [6], [7], [19], [20], [21] und [22].

Es besteht aus 3 Hauptgruppen (siehe Kapitel 8):

- Betriebsmodus (MAIN_PROG)
- Programmiermodus (PROG)
- RS 232 Modus (RS232)

7.1 Betriebsmodus

Zuerst wird der Betriebsartenschalter abgefragt, dann wird an den Spracherkennungsbaustein der Befehl Listen (siehe 5.5.12) geschickt und die Antwort abgewartet. Die Antwort wird auf eine Fehlermeldung, positive Erkennung, Schlüsselwort, Bestätigungsworte, Resetwort und Selektion eines Ausgangs hin überprüft. Danach wird auf das Display die Nummer des erkannten Wortes ausgegeben. War die Fehlermeldung positiv wird die Prozedur NOT TRAINED aufgerufen.

7.2 Programmiermodus

Zuerst wird der Betriebsartenschalter abgefragt, dann werden die Prozeduren TRAINW, CLEARW, PLAYBACKW, LEARNALL, RELSET (siehe 7.4) durchlaufen.

7.3 RS 232 Modus

Zuerst wird der Betriebsartenschalter abgefragt, dann wird auf das Display „RS 232“ geschrieben. Nach der Initialisierung der Schnittstelle wird auf ankommende Daten vom PC gewartet. Diese 6 Byte werden in den OUT_BUF geschrieben und an den Spracherkennungsbaustein geschickt. Die 6 Byte lange Antwort des Spracherkennungsbaustein wird in den IN_BUF geschrieben und der Inhalt des IN_BUF an den PC zurückgeschickt.

7.4 Prozeduren

7.4.1 TRAINW

Zuerst wird der Betriebsartenschalter abgefragt, dann wird auf das Display „LERNEN WORT x“ geschrieben. Es werden ständig der Menütaster, der Stufentaster und der Quittierungstaster abgefragt.

Wird der Menütaster betätigt, wird in das CLEARW Menü gesprungen.

Wird der Stufentaster betätigt, wird die Ausgangsnummer erhöht.

Wenn die Quittierungstaste gedrückt wird, wird an den Spracherkennungsbaustein der Befehl REC_WORD geschickt und die Antwort abgewartet. Danach durch den Befehl PLAYBIMM der Inhalt des Recording Buffer an den Lautsprecher ausgegeben. Mit dem Befehl STORWORD wird der Inhalt des Recording Buffer abgespeichert.

7.4.2 CLEARW

Zuerst wird der Betriebsartenschalter abgefragt, dann wird auf das Display „LÖSCHEN WORT x“ geschrieben. Es werden ständig der Menütaster, der Stufentaster und der Quittierungstaster abgefragt. Wird der Menütaster betätigt, wird in das PLAYBACKW Menü gesprungen. Wird der Stufentaster betätigt, wird die Ausgangsnummer erhöht. Wenn die Quittierungstaste gedrückt wird, wird an den Spracherkennungsbaustein der Befehl CLEAWORD geschickt.

7.4.3 PLAYBACKW

Zuerst wird der Betriebsartenschalter abgefragt, dann wird auf das Display „PLAYBACK WORT x“ geschrieben. Es werden ständig der Menütaster, der Stufentaster und der Quittierungstaster abgefragt. Wird der Menütaster betätigt, wird in das LEARNALL Menü gesprungen. Wird der Stufentaster betätigt, wird die Ausgangsnummer erhöht. Wenn die Quittierungstaste gedrückt wird, wird an den Spracherkennungsbaustein der Befehl PLAYBWORD geschickt. Dieser legt das gewünschte Sprachmuster an den Lautsprecher.

7.4.4 LEARNALL

Zuerst wird der Betriebsartenschalter abgefragt, dann wird auf das Display „TRAINIEREN“ geschrieben. Es werden ständig der Menütaster und der Quittierungstaster abgefragt. Wird der Menütaster betätigt, wird in das RESET Menü gesprungen. Wenn die Quittierungstaste gedrückt wird, wird an den Spracherkennungsbaustein der Befehl LEARN geschickt. Der Spracherkennungsbaustein verarbeitet dann alle gespeicherten Sprachmuster.

7.4.5 RESET

Zuerst wird eine Initialisierung des Hilfsakkumulators und der Ausgangsnummer vorgenommen, dann der Betriebsartenschalter abgefragt. Die abgespeicherte Einstellung des Ausgangs wird aus dem SRAM geholt und auf dem Display angezeigt. Es werden ständig der Menütaster, der Stufentaster und der Quittierungstaster abgefragt. Wird der Menütaster betätigt, wird in das TRAINW Menü gesprungen. Wird der Stufentaster betätigt, wird die nächste Ausgangsbelegung angezeigt. Sind alle Möglichkeiten durchgespielt wird die Ausgangsnummer erhöht (siehe 8.2.2.5 Schaltausgänge definieren). Bei betätigen der Quittierungstaste wird der aktuelle Wert im SRAM abgespeichert und die Ausgangsnummer erhöht.

7.4.6 TO_NNSR

Hier werden die Befehle, die im OUT_BUF stehen, an den Spracherkennungsbaustein geschickt. Das Byte 0 muß als letztes gesendet werden, da es im NNSR einen Interrupt auslöst. Der Spracherkennungsbaustein wird über die externe Speicheradresse FF00h angesprochen.

7.4.7 FROM_NNSR

Der Mikroprozessor sendet Host Ready an den NNSR und wartet, bis der NNSR mit Host Message antwortet. Diese Antwort löst einen Interrupt im Mikroprozessor aus, in dessen Abarbeitung das NNSR_MSG Flag gesetzt wird. Ist dies geschehen, werden die Antworten des NNSR von seinem Host Interface Port ausgelesen und in den IN_BUF geschrieben.

7.4.8 INIT_NNSR

In der Initialisierung des NNSR wird der Befehl CONFINEW mit den Optionen acoustic_feedback set und 11 verschiedene Wörter an den Spracherkennungsbaustein geschickt. Da diese Configuration von der vorherigen nicht abweicht, bleiben die gespeicherten Sprachmuster erhalten.

7.4.9 SCHLAFE

Wird im Betriebsmodus das Schlüsselwort erkannt, springt das Programm in diese Prozedur. Auf das Display wird „ICH SCHLAFE“ geschrieben, dann wird an den NNSR der Befehl LISTEN geschickt und die Antwort, auf eine positive Erkennung hin, ausgewertet. Wenn das erkannte Wort das Schlüsselwort war, wird die Prozedur wieder verlassen.

7.4.10 ERKENNUNG

Hier wird untersucht ob ein Wort als erkannt akzeptiert werden kann.

Die Schwelle für eine positive Erkennung wird wie folgt angesetzt:

$$\text{correlation_code} + (\text{spectral_mom_code} / 2) > 60h$$

Der correlation_code steht im IN_BUF+4, der spectral_mom_code im IN_BUF+5

Ergibt die Auswertung, daß ein Wort als erkannt akzeptiert ist, wird das Flag POSITIV gesetzt.

7.4.11 RELAIS_SCHALTEN

Zuerst wird überprüft, ob das Flag RELAIS gesetzt ist, dann ob das Resetwort gesprochen wurde. Wurde das Resetwort gesprochen, werden die jeweiligen Relaiseinstellungen aus dem REL_BUF ausgelesen und die dazugehörigen Zeitwerte in den REL_TIMER geschrieben. Wurde ein einzelnes Relais selektiert, wird überprüft, ob es ein oder ausgeschaltet werden soll.

- Relais soll eingeschaltet werden: Aus dem zugehörigen REL_BUF wird ausgelesen, ob das Relais konstant oder gepulst geschaltet wird. Der entsprechende Zeitwert wird in den REL_TIMER geschrieben.
- Relais soll ausgeschaltet werden: Der entsprechende Zeitwert wird in den REL_TIMER geschrieben.

7.4.12 TIMER

Diese Prozedur enthält die Zählregister für den Zeitschlitz und die Relais. Im Hintergrund des Programms läuft der Timer 0 des Mikroprozessors, dieser löst bei einem Timer overflow einen Interrupt aus. Jede Sekunde wird die Timerprozedur aufgerufen und wenn nötig die Zählregister decrementiert. Steht im ZEITSCHLITZ eine Zahl grösser 0, ist die Zeit für die Bestätigung noch nicht abgelaufen.

In den Zählregistern für die Relais (REL_TIMER) können folgende Werte stehen:

- FFh Relais ist auf Dauer ein, das Zählregister wird nicht decrementiert
- 01-064h Relais ist ein, bei jedem Durchlauf der Prozedur wird das Zählregister decrementiert
- 00h Relais ist auf Dauer ausgeschaltet, das Zählregister wird bei einem Durchlauf nicht beeinflusst.

7.4.13 TIMER0

Diese Prozedur ist der Zähler für den Timer 0 Interrupt. Sie wird benötigt da ca. alle 62.5 ms ein Interrupt ausgelöst wird, die Prozedur Timer aber nur 1 mal pro Sekunde aufgerufen werden soll.

7.4.14 NOT_TRAINED

Diese Fehlermeldung wird vom Betriebsmodus aus aufgerufen, wenn der NNSR nicht trainiert war. Auf das Display wird dann „NICHT TRAINIERT“ ausgegeben.

7.4.15 SPEICHER_VOLL

Diese Fehlermeldung wird von der Prozedur TRAINW aufgerufen, wenn die maximale Anzahl speicherbarer Sprachmuster pro Wort überschritten wurde. Auf das Display wird „SPEICHER VOLL“ ausgegeben.

7.4.16 INIT_DISPLAY

In dieser Prozedur wird die Initialisierung des Displays vorgenommen. Die Bedeutung der Befehle kann aus [4] entnommen werden.

7.4.17 CLEAR_DISPLAY

Hier wird die Anzeige des Display gelöscht.

7.4.18 SEND_DISPLAY

Hier wird der Inhalt des DISP_BUF an das Display geschickt. Nach jedem Bit wird abgewartet, bis das Display wieder bereit ist.

7.4.19 LCD_RDY

In dieser Prozedur wird abgewartet bis das Display für das nächste Zeichen ist bereit.

7.4.20 TASTER2_ABF

Diese Prozedur fragt den Taster 2 ab. Es ist eine Tastenentprellung eingebaut. Wenn er gedrückt und wieder losgelassen wurde wird die Menünummer erhöht.

7.4.21 TASTER3_ABF

Diese Prozedur fragt den Taster 3 ab. Es ist eine Tastenentprellung eingebaut. Wenn er gedrückt und wieder losgelassen wurde wird die Ausgangnummer erhöht. Weiters wird aus der ASCII-Code Tabelle der ASCII-Code der Ausgangnummer für das Display geholt.

7.4.22 REL_ABF

Diese Prozedur wird aus dem Menü RELSET aufgerufen und fragt den Taster 3 ab. Es ist eine Tastenentprellung eingebaut. Wenn er gedrückt und wieder losgelassen wurde, wird die nächst mögliche Ausgangbelegung erzeugt, oder wenn alle Möglichkeiten durchgespielt wurden, die Ausgangnummer erhöht.

7.4.23 REL_EIN

In dieser Prozedur werden die Relaiseinstellung aus dem SRAM ausgelesen und in den REL_BUF übertragen.

7.4.24 TASTER4_ABF

Diese Prozedur fragt den Taster 4 ab. Es ist eine Tastenentprellung eingebaut. Wurde die Taste gedrückt und wieder losgelassen, wird das Bit QUITT gesetzt.

7.4.25 SCHLEIFE

Diese Prozedur beinhaltet die Warteschleife für die Tastenentprellung. Sie wird aufgerufen wenn Taster 2, 3 oder 4 gedrückt und wieder losgelassen werden.

7.4.26 RAM_OK

Diese Prozedur stellt sicher, daß bei der allerersten Inbetriebnahme des SRAM keine ungültigen Werte für die Relaiseinstellungen eingeschrieben sind.

7.4.27 Commands HOST -> NNSR

Hier stehen die Hexadezimal-Codes für die Befehle, die an den Spracherkennungsbaustein geschickt werden.

7.4.28 ASCII-Tabellen

In den ASCII-Tabellen stehen die ASCII-Codes für die Ausgangsnummer und die Relaiseinstellungen, die am Display angezeigt werden.

8. Bedienung des Gerätes

Dieses Kapitel ist als Bedienungsanleitung, für Benutzer ohne technische Vorkenntnisse, gedacht. Sie sollten damit in der Lage sein das Gerät, in kürzester Zeit, selbständig zu bedienen.
Für genauere Informationen stehen die Kapitel 5,6,7,10 und 11 zur Verfügung.

8.1 Anschlüsse und Bedienungselemente

8.1.1 Steuermodul

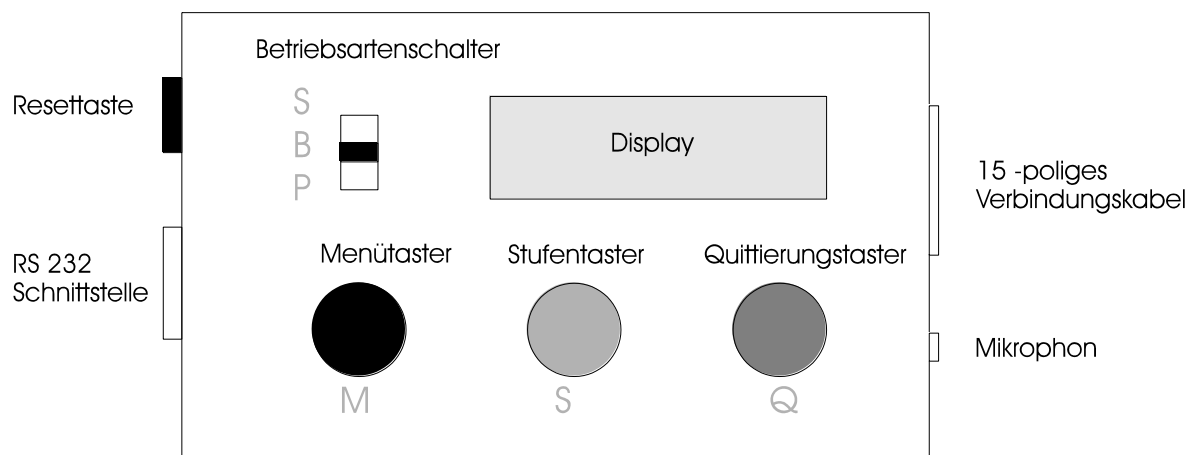


Abbildung 8.1 Gehäuse des Steuermoduls

Das Steuermodul wird mit Hilfe des 15-poligen Verbindungskabels an das Schaltmodul angeschlossen. Da auch die Energieversorgung über dieses Kabel erfolgt, ist ohne Schaltmodul keine Funktion möglich. Die RS 232 Schnittstelle wird nur benötigt, wenn die Steuerung der Spracherkennung über einen PC erfolgt (siehe 8.2.3 RS 232 Modus). An die Mikrofon-Buchse wird das ME 2 (Fa. Peiker) angeschlossen.

8.1.2 Schaltmodul

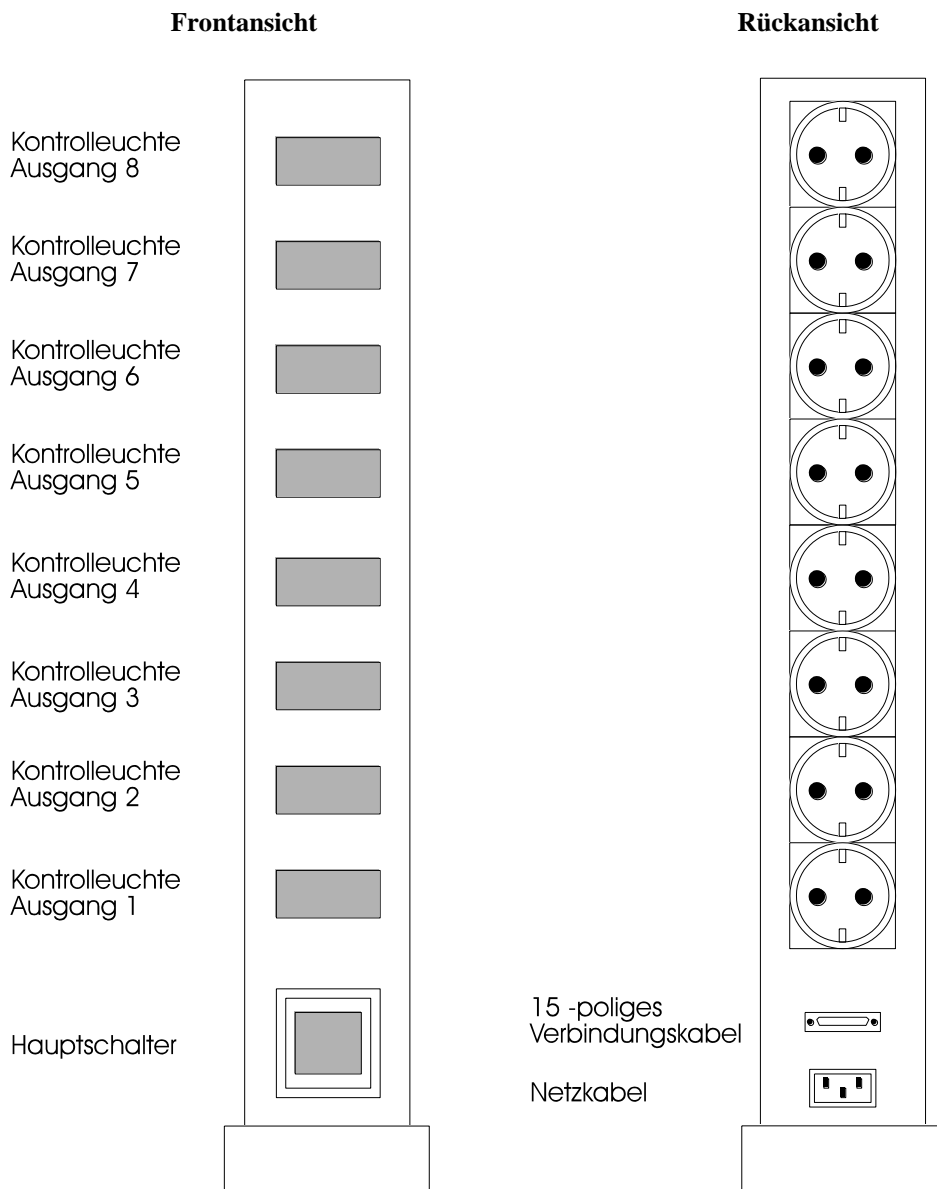


Abbildung 8.2 Gehäuse des Schaltmoduls

Das Schaltmodul wird mit Hilfe des Netzkabels an die Spannungsversorgung (220 V) angeschlossen. Dann das Schaltmodul mit dem Steuermodul verbinden und den Hauptschalter einschalten. An die 8 Steckdosen können die Verbraucher angeschlossen werden. Dabei ist zu beachten, daß die einzelnen Verbraucher eine Leistung von 250 W nicht überschreiten dürfen und die Versorgung für maximal 2000 Watt ausgelegt ist. Das Gerät ist jetzt betriebsbereit.

8.2 Steuermodul

8.2.1 Betriebsmodus

Der Betriebsartenschalter wird in Stellung B gebracht (siehe Abbildung 8.1). War das Gerät vorher im RS 232 Modus muß die Resettaste gedrückt werden. Die Anzeige am Display erlischt nun und das Gerät ist im Betriebsmodus. Zeigt das Display statt dessen die Meldung „NICHT TRAINIERT“, muß zuvor in den Programmiermodus gegangen werden und die Trainerprozedur ausgeführt werden. Sind keine Worte abgespeichert, muß auch das zuerst vorgenommen werden (siehe 8.2.2 Programmiermodus). Im Betriebsmodus kann das Gerät über das Mikrophon angesprochen werden. Es genügt leise zu sprechen. Der Abstand Mikrophon - Mund des Sprechers sollte im Idealfall ca. 20 - 50 cm betragen, da es sich um ein Nahbesprechungsmikrophon handelt. Ist dies nicht möglich, sollten die Sprachmuster zumindest im selben Abstand gelernt werden, wie er im Betrieb vorkommt (siehe 8.2.2.1 Wort Lernen). Die Wörter sollten mit einem Abstand von ca. 2 s gesprochen werden, damit eine fehlerfreie Bearbeitung gewährleistet ist.

8.2.1.1 Schaltausgang auswählen

Mit einem der Wörter 1-8 kann ein Ausgang selektiert werden. Die korrekte Erkennung wird am Display angezeigt. Es wird noch kein Schaltvorgang eingeleitet, da erst die Bestätigung abgewartet werden muß. Kommt diese nicht innerhalb von 10 s, oder wird inzwischen ein anderer Befehl gesprochen und auch erkannt, erlischt die Selektion des Wortes wieder.

8.2.1.2 Befehl Bestätigen

Ist ein Ausgang selektiert, und kommt die Bestätigung zeitgerecht, wird ein Schaltbefehl durchgeführt.

Die Form desselben hängt von folgendem ab (siehe 8.2.2.5 Schaltausgänge definieren):

- Ausgang konstant geschaltet; Bestätigung positiv ==> Ausgang wird eingeschaltet
War er bereits eingeschaltet, gibt es keine Änderung.
- Ausgang konstant geschaltet; Bestätigung negativ ==> Ausgang wird ausgeschaltet
War er bereits ausgeschaltet, gibt es keine Änderung.
- Ausgang gepulst geschaltet; Bestätigung positiv ==> Ausgang wird eingeschaltet . Die Zeituhr beginnt zu laufen. Nach Ablauf der eingestellten Zeit schaltet sich der Ausgang von selbst ab.

War er bereits eingeschaltet, bleibt er eingeschaltet, die Zeituhr wird zurückgesetzt und läuft nochmals ab.

- Ausgang gepulst geschaltet; Bestätigung negativ ==> Die Zeituhr wird unterbrochen, der Ausgang sofort abgeschaltet.
War er bereits ausgeschaltet, bleibt er ausgeschaltet.

8.2.1.3 Grundeinstellung

Wird das Resetwort (Wort R für Reset) gesprochen und positiv bestätigt, werden alle Ausgänge in einen vordefinierten Zustand gebracht (siehe 8.2.2.5 Schaltausgänge definieren).

8.2.1.4 Schlüsselwort

Wird das Schlüsselwort gesprochen, geht die Spracherkennung in einen Ruhezustand. Auf das Display wird „ICH SCHLAFE“ geschrieben, und die Spracherkennung reagiert auf keinen Befehl mehr. Wird das Schlüsselwort nochmals gesprochen geht das Gerät wieder in den normalen Betriebszustand über. Die ein- oder ausgeschalteten Ausgänge werden von diesem Befehl nicht beeinflusst.

8.2.2 Programmiermodus

Um in den Programmiermodus zu gelangen, wird der Betriebsartenschalter in Stellung P gebracht (siehe Abbildung 8.1). Befand sich das Gerät zuvor im Betriebsmodus, muß ein Wort gesprochen oder die Resettaste betätigt werden. Das Display zeigt dann „LERNEN WORT 1“. Hiermit befindet sich das Gerät im Programmiermodus.

Es stehen nun 3 Tasten zur Verfügung, um das Gerät zu bedienen:

- Menütaster
Mit dieser Taste kann im Menü weitergeschaltet werden

LERNEN WORT x
LÖSCHEN WORT x
PLAYBACK WORT x
TRAINIEREN
AUSGANG x K0

- Stufentaster
Mit dieser Taste kann die Wortnummer bzw. die Ausgangsnummer erhöht werden

Wort Nummern 1 - 8 korrespondieren mit den zugehörigen Ausgängen
Wort R bezeichnet das Resetwort
Wort E bezeichnet die positive Bestätigung
Wort A bezeichnet die negative Bestätigung
Wort S bezeichnet das Schlüsselwort

Ausgangsnummer 1 - 8 K0, K1, P1 - P 99 (siehe 8.2.2.5 Schaltausgänge definieren)

- Quittierungstaster
Mit dieser Taste kann eine Aktion ausgeführt werden.

Diese 3 Tasten sind nur im Programmiermodus aktiv, im Betriebsmodus und im RS 232 Modus sind sie ohne Bedeutung.

8.2.2.1 Wort Lernen

Vor der Aufzeichnung sollte man sich überzeugen, ob schon Sprachmuster zu diesem Wort gespeichert sind (siehe 8.2.2.3 Playback). Ist ein unerwünschtes Muster abgespeichert, muß das Wort zuvor gelöscht werden (siehe 8.2.2.2 Wort Löschen).

Mit dem Menütaster wird „LERNEN WORT x“ aufgerufen, dann die gewünschte Wortnummer mit dem Stufentaster eingestellt. Nach Drücken und wieder Loslassen der Quittierungstaste muß unmittelbar gesprochen werden.

Zwischen den einzelnen Aufzeichnungen sollte man einige Sekunden vergehen lassen, da das Steuermodul etwas Zeit zur Verarbeitung braucht. Das aufgezeichnete Wort wird auf dem Lautsprecher wiedergegeben. Hört man nur ein Rauschen, hat man den Zeitschlitz, der zur Aufnahme zur Verfügung steht, verpaßt. Es müssen dann alle Sprachmuster, die zu diesem Wort gehören, gelöscht werden (siehe 8.2.2.2 Wort Löschen) und die Aufzeichnung für dieses Wort neu begonnen werden. Ist nichts im Lautsprecher zu hören, war nur die Zeit zwischen den einzelnen Aufzeichnungen zu kurz, die Aufzeichnung wurde nicht vorgenommen, es kann aber normal fortgefahren werden (ohne Wort Löschen und Neubeginn).

Bis zu 26 Sprachmuster pro Wort können aufgezeichnet werden. Ist diese maximale Anzahl erreicht, erscheint am Display die Meldung „SPEICHER VOLL“. Diese Sprachmuster müssen nicht auf einmal eingelesen werden, sondern können nach und nach ergänzt werden. Wird das Steuermodul vom Netz getrennt, bleiben die gelernten Sprachmuster erhalten.

Anschließend noch einige Regeln:

- Keine Worte, die ähnlich klingen z.B. „Ein“ und „Nein“
- Keine sehr langen Worte
- Sprachmuster eventuell mit verschiedenen Hintergrundgeräuschen aufnehmen
- Worte, die nicht gut erkannt werden, nachlernen oder durch andere ersetzen
- Für das Schlüsselwort ein ungewöhnliches, im täglichen Sprachgebrauch selten vorkommendes Wort verwenden
- Einzelne Sprachmuster mit verschiedener Betonung sprechen, eventuell zu verschiedenen Tageszeiten
- Abstand und Winkel zum Mikrofon variieren

8.2.2.2 Wort Löschen

Mit dem Menütafter wird „LÖSCHEN WORT x“ aufgerufen, dann die gewünschte Wortnummer mit dem Stufentaster eingestellt. Durch Drücken der Quittierungstaste werden sämtliche zu diesem Wort gehörigen Sprachmuster gelöscht. Mit Playback Wort kann überprüft werden, ob dies auch geschehen ist (siehe 8.2.2.3 Playback).

8.2.2.3 Playback

Mit dem Menütafter wird „PLAYBACK WORT x“ aufgerufen, dann die gewünschte Wortnummer mit dem Stufentaster eingestellt. Durch drücken der Quittierungstaste kann das erste abgespeicherte Sprachmuster eines Wortes angehört werden. Ist nichts zu hören, sind keine Sprachmuster zu diesem Wort gespeichert.

8.2.2.4 Trainieren

Mit dem Menütafter wird „TRAINIEREN“ aufgerufen. Durch Drücken der Quittierungstaste wird die Prozedur gestartet. Danach kann der Betriebsartenschalter in Stellung B gebracht werden, erlischt die Anzeige, ist das Gerät betriebsbereit. Die Ausführung dieser Prozedur kann einige Minuten dauern.

Wann muß trainiert werden:

- Nachdem verschiedene Sprachmuster zu verschiedenen Wörtern gelernt wurden (siehe 8.2.2.1 Wort Lernen) und wieder in den Betriebsmodus umgeschaltet wird.
- Wenn zu einem bereits vorhandenen Wort neue Muster hinzugelehrt werden.

Wann muß nicht trainiert werden:

- Wenn ein oder mehrere Wörter nur gelöscht werden.
- Wenn die Schaltausgänge neu definiert werden.
- Wenn mit Playback ein Wort angehört wird.

8.2.2.5 Schaltausgänge definieren

Mit dem Menütafter wird „AUSGANG x xxx“ aufgerufen. Der zum entsprechenden Ausgang abgespeicherte Wert wird angezeigt: z.B.: AUSGANG 3 P 5

Bedeutung der Abkürzungen:

- K 0 Der Ausgang wird konstant geschaltet wenn ein Schaltbefehl kommt (siehe 8.2.1.1 Schaltausgang auswählen). Wird das Resetwort gesprochen (siehe 8.2.1.3 Grundeinstellung), ist dieser Ausgang stromlos.
- K 1 Der Ausgang wird konstant geschaltet wenn ein Schaltbefehl kommt (siehe 8.2.1.1 Schaltausgang auswählen). Wird das Resetwort gesprochen (siehe 8.2.1.3 Grundeinstellung), ist dieser Ausgang stromführend.
- P1 - P99 Der Ausgang wird für 1 s - 99 s eingeschaltet wenn ein Schaltbefehl kommt (siehe 8.2.1.1 Schaltausgang auswählen). Wird das Resetwort gesprochen (siehe 8.2.1.3 Grundeinstellung), ist dieser Ausgang stromlos.

Mit dem Stufentaster können jetzt alle Möglichkeiten ausgehend vom abgespeicherten Wert durchgegangen werden. Der nächste Ausgang erscheint, wenn alle Werte für diesen Ausgang durchlaufen wurden, oder mit der Quittierungstaste ein Wert abgespeichert wird. Die Anzeige für diesen Ausgang beginnt wieder mit dem aktuell gespeicherten Wert.

Wird das Steuermodul vom Netz getrennt, bleibt die Ausgangsbelegung erhalten.

8.2.3 RS 232 Modus

Das Gerät wird mit einem RS 232 Kabel an einen PC angeschlossen. Um in den RS 232 Modus zu gelangen, wird der Betriebsartenschalter in Stellung S gebracht (siehe Abbildung 8.1). Befand sich das Gerät zuvor im Betriebsmodus, muß ein Wort gesprochen, oder die Resettaste betätigt werden. Das Display zeigt dann „RS 232“. Hiermit befindet sich das Gerät im RS 232 Modus. Jetzt kann die Spracherkennung über einen PC gesteuert werden. Die möglichen Befehle und die Antworten finden sich im Kapitel 5 dieser Arbeit.

Die serielle Schnittstelle arbeitet mit 9600 baud , 1 Start-, 8 Daten-,und 1 Stoppbit.

Befehle, die vom PC kommen, werden an den Spracherkennungsbaustein weitergegeben, und die Antworten an den PC zurückgeliefert.

Wurde der RS 232 Modus ohne angeschlossenen PC aktiviert, kann er durch Umstellen des Betriebsartenschalters auf B oder P und anschließendem Drücken der Resettaste wieder verlassen werden.

8.2.4 Reset

Wird die Resettaste betätigt, werden alle Ausgänge spannungslos geschaltet, und in den am Betriebsartenschalter eingestellten Modus gegangen.

8.3 Schaltmodul

Mit dem Hauptschalter werden sowohl das Schaltmodul als auch das Steuermodul eingeschaltet.

Die Kontrolleuchten zeigen an, welcher Ausgang gerade ein- oder ausgeschaltet ist.

9. Literaturverzeichnis

- [1] NNSR - Hardware - Schnittstelle, Prototypbeschreibung
Siemens AG, Wien, 1992

- [2] NNSR Neural Net Speech Recognizer
Siemens AG, Wien, 1993

- [3] DSP Microcomputer mit Host Interface Port
ADSP - 2111
Analog Devices, Norwood, 1993

- [4] OPTREX Liquid Crystal Display
OPTREX CORPORATION, Tokyo, 1993

- [5] Microcomputer Components SAB 80C517 / 80C537
User's Manual
Siemens AG, München, 1994

- [6] Microcontroller Family SAB 8051
Pocket Guide
Siemens AG, München, 1994

- [7] 8051 Cross Assembler User's Manual
MetaLink Corporation, Arizona, 1989

- [8] Friedrich, Tabellenbuch Elektrotechnik / Elektronik
Dümmler Bonn, Bonn, 1982

- [9] Wolfgang L. Zagler, Franz Peter Seiler
„Elektronische Hilfsmittel für Behinderte, Rehabilitationstechnik“
Arbeitsunterlage zur Vorlesung Rehabilitationstechnik, TU Wien, 4/1992

- [10] Ditmar Schultschik
„Autonom I: Grundlagen für AUTONOM und Fernbedienungsmodul“, Diplomarbeit
TU Wien, 11/1992

- [11] Jürgen Demuth
„Autonom II: Software AUTOSOFT für ein Steuerungs- und Kommunikationssystem für behinderte
Menschen“, Diplomarbeit
TU Wien, 2/1993

- [13] Karl Watz
„Autonom IV: Telefonsteuermodul“, Diplomarbeit
TU Wien, 11/1994

- [14] Q 400 Mk II (T)

- AKG Acoustics Ges.m.b.H., Wien, 1992
- [15] High-speed CMOS 74HC/HCT/HCU Logic family
PHILLIPS Semiconductors, Eindhoven, 1993
- [16] Distrelec Katalog
Distrelec GmbH, Wien, 1995
- [17] Dallas Semiconductor Product Data Book
Dallas Semiconductor, Dallas, 1990
- [18] Legrand Produktkatalog 1994/95
Legrand Österreich GmbH, Wernberg, 1994
- [19] Mikrocomputerarchitektur
Prof. Dr. R. Eier
Skriptum zur Vorlesung, TU Wien, 1992
- [20] Digitale Systeme
Prof. Dr. R. Eier
Skriptum zur Vorlesung, TU Wien, 1987
- [21] iceMASTER User's Manual
MetaLink Corporation, Arizona, 1991
- [22] Host Demo NNSR Diskette
Siemens AG, Wien, 1993
- [23] QUERSCHNITT GELÄHMT
ein Rehabilitationsteam präsentiert seine Arbeitsmethoden
Allgemeine Unfallversicherungsanstalt, Wien
- [24] IBM Voice Type (Personal Dictation System), Broschüre
IBM Österreich Personal Software Marketing, Wien, 1994
- [25] Windows Sound System, Benutzerhandbuch
Microsoft Corporation, 1992
- [26] SAB-C501 8-Bit Single Chip Microcontroller
User's Manual
Siemens AG, München, 1993
- [27] Embedded Control Handbook
Microchip Technology Inc., Chandler, 1992

- [28] AUTONOM- Ein technisches Assistenzsystem für Menschen mit Bewegungs- und Mehrfachbehinderung
Christian Flachberger
Vortrag im Rahmen der integra, Fachmesse für Rehabilitation und Integration
Altenhof am Hausruck, September 1994
- [29] Compose Autonomy-An Adaptable User Interface for Assistive Technology Systems
Christian Flachberger, Paul Panek, Wolfgang L. Zagler
fortec Institut für Allgemeine Elektrotechnik und Elektronik,
Papier, präsentiert beim 2. TIDE Congress (The European Context for Assistive Technology),
Paris, 1995
- [30] Spracherkennung durch Computersysteme, Seminararbeit
Wolfgang Wangel
Institut für Wirtschaftsinformatik, Wien, 1995

10. Anhang A

10.1 Listing des Assembler Programms

Das Listing des in Kapitel 7 beschriebenen Programms

```
;      Betrieb des NNSR
;
;      by Gerhard Loidolt
;
;
; Primary controls
$MOD517
$TITLE(BETR.NNSR)
$PAGEWIDTH(120)
$DEBUG
$OBJECT
$NOPAGING
;
;
;      Variable declarations
;
BSEG          AT      20h          ; bitadressable segment
QUITT:        DBIT    1           ; Bit für Quittierungs-Taster
POSITIV:      DBIT    1           ; Bit für positive Erkennung
RELAIS:       DBIT    1           ; Bit für Relaisbestätigung
NNSR_MSG:     DBIT    1           ; Flag message from NNSR
; received
DISP_RS       BIT     P3.4        ; Register Select for display
DISP_RW       BIT     P3.3        ; Display read/write
DISP_E        BIT     P3.2        ; RS232
RTS_I         BIT     P1.1        ; RS232
TASTER4       BIT     P1.7        ; QUITTIERUNG_TASTER
ASCII_NUM     DATA   2Ch        ; ASCII-Code der
; AUSGANGNUMMER
AUSGANGNUMMER DATA   2Dh
MENUENUMMER   DATA   2Eh
ZAEHLER       DATA   2Fh        ; Zähler für SCHLEIFE
;
;
DSEG          AT      30h        ; internal data memory
OUT_BUF:      DS      6          ; buffer for commands to NNSR
IN_BUF:       DS      6          ; buffer for messages from NNSR
DISP_BUF:    DS      16         ; buffer for display messages
REL_BUF:      DS      9          ; buffer für Relaisstellungen
REL_NUM:      DS      1          ; buffer für selectiertes Relais
REL_TIMER:    DS      9          ; buffer für Relais-timer
ZEITSCHLITZ: DS      1          ; für Bestätigung
ZAEHLERT0:    DS      1          ; Zähler für TF0
;
;
;      ORG      60h          ; stack area
STACK:        DS      0
```

```

XSEG          AT    0000h          ; gepuffertes RAM
RELAIS1:     DS    1              ; Relais 1 konstant oder gepulst
RELAIS2:     DS    1              ; Relais 2 konstant oder gepulst
RELAIS3:     DS    1              ; Relais 3 konstant oder gepulst
RELAIS4:     DS    1              ; Relais 4 konstant oder gepulst
RELAIS5:     DS    1              ; Relais 5 konstant oder gepulst
RELAIS6:     DS    1              ; Relais 6 konstant oder gepulst
RELAIS7:     DS    1              ; Relais 7 konstant oder gepulst
RELAIS8:     DS    1              ; Relais 8 konstant oder gepulst
;
XSEG          AT    0FF00h        ; NNSR über externe
; Speicheradresse FF00h
; angesprochen
HDR0:        DS    1              ; external memory mapped
HDR1:        DS    1              ; HOST DATA REGISTERS
; (HDR0 bis HDR5)

```

*****Interrupt Vector Tabelle*****

```

CSEG          AT    0000h          ; after RESET
INT_RES:     JMP   INIT           ; jump to initialisation

```

; Timer 0 wird zum gepulsten schalten der Relais verwendet.

; Der Timer 0 Interrupt incrementiert den Zähler für T0

```

CSEG          AT    000Bh          ; TF0
INT_TFO:     CALL  TIMER0
             RETI

```

; Der externe Interrupt 2 wird ausgelöst wenn der NNSR eine Antwort in seinem Register hat

```

CSEG          AT    004Bh          ; external interrupt 2
INT_EX2:     SETB  NNSR_MSG        ; set flag message from NNSR
             RETI

```

*****Initialisierung*****

```

INIT:        SETB  IEN0.7          ; ermöglicht Interrupts
             SETB  IEN0.1          ; ermöglicht TF0
             CLR   T2CON.5         ; INT 2 auf fallende Flanke
             ; getriggert
             MOV   TMOD,#1h        ; T0 MOD 1
             MOV   P4,0FFh         ; alle Relais ausschalten
             MOV   REL_TIMER+1,#00h ; alle Relais ausschalten
             MOV   REL_TIMER+2,#00h ; alle Relais ausschalten
             MOV   REL_TIMER+3,#00h ; alle Relais ausschalten
             MOV   REL_TIMER+4,#00h ; alle Relais ausschalten
             MOV   REL_TIMER+5,#00h ; alle Relais ausschalten
             MOV   REL_TIMER+6,#00h ; alle Relais ausschalten
             MOV   REL_TIMER+7,#00h ; alle Relais ausschalten
             MOV   REL_TIMER+8,#00h ; alle Relais ausschalten
             CLR   P3.5            ; Reset des NNSR
             SETB  P3.5
             SETB  P1.4            ; Eingang auf high
             SETB  P1.6            ; setze HREADY
             CLR   NNSR_MSG
             SETB  IEN1.1          ; ermöglicht INT 2
             CLR   NNSR_MSG

```

```

INIT_WAIT:      JNB   NNSR_MSG,INIT_WAIT      ; Warte auf Ready vom NNSR
                CLR   IEN1.1          ; dreht INT 2 ab
                CLR   P1.6            ; lösche HREADY
                SETB  TASTER4         ; Eingang auf high
                MOV   P6,0FFh        ; Eingang auf high
                CALL  INIT_DISPLAY    ; Initialisierung des Display
                CALL  CLEAR_DISPLAY   ; löscht das Display
                CALL  INIT_NNSR       ; Initialisierung des NNSR
                CALL  RAM_OK          ; SRAM gültig?
                MOV   ZAEHLERT0,#0h   ; T0 Zähler zurückgesetzt
                SETB  TCON.4          ; startet Timer 0
                JMP   MAIN_PROG       ; goto mainprogram

;***** Mainprogramm Betrieb des NNSR *****

MAIN_PROG:      MOV   A,P6
                JB    ACC.4,MAIN4      ; liest Betriebsart Schalter 1 ein
                CALL  SCHLEIFE        ; zur Schalterentprellung
                MOV   A,P6
                JB    ACC.4,MAIN4
                CALL  PROG             ; in den Programmiermodus
                CALL  CLEAR_DISPLAY
MAIN4:          MOV   A,P6
                JB    ACC.5,MAIN1      ; liest Betriebsart Schalter 5 ein
                CALL  SCHLEIFE        ; zur Schalterentprellung
                MOV   A,P6
                JB    ACC.5,MAIN1
                CALL  RS232           ; Modus RS232
                CALL  CLEAR_DISPLAY
MAIN1:          MOV   OUT_BUF+0,#LISTEN ; Befehl LISTEN in den
                ; Ausg.-Buffer
                CALL  TO_NNSR         ; sendet Befehl zum NNSR
                CALL  FROM_NNSR       ; wartet auf Antwort
                MOV   A,IN_BUF+1      ; execution success code
                CJNE  A,#40h,MAIN7     ; executed sucessfull?
                CALL  NOT_TRAINED
                JMP   MAIN_PROG
MAIN7:          CALL  ERKENNUNG
                JB    POSITIV,MAIN2    ; Wort erkannt?
                CALL  CLEAR_DISPLAY
                JMP   MAIN_PROG        ; kein Wort erkannt
MAIN2:          MOV   A,IN_BUF+2      ; Nummer des erkannten Wortes
                CJNE  A,#0Ch,MAIN5     ; vergleicht mit Schlüsselwort
                CALL  SCHLAFE
                JMP   MAIN3
MAIN5:          MOV   A,IN_BUF+2      ; Nummer des erkannten Wortes
                CJNE  A,#0Ah,MAIN8     ; vergleicht mit Bestätigung EIN
                CALL  RELAIS_SCHALTEN
MAIN8:          MOV   A,IN_BUF+2
                CJNE  A,#0Bh,MAIN6     ; vergleicht mit Bestätigung AUS
                CALL  RELAIS_SCHALTEN
                JMP   MAIN3

```

```

MAIN6:      MOV    A,IN_BUF+2
            CLR    C
            SUBB   A,#0Ah                ; Relais selectiert ?
                                                ; Relais Nr.1-8 und Reset-Wort
                                                ; < Ah

            JNC    MAIN3
            SETB   RELAIS                ; wird gesetzt wenn ein Relais
                                                ; oder Reset-Wort selectiert

            MOV    ZEITSCHLITZ,#Ah      ; Zeitschlitz für Bestätigung öffnen
            MOV    REL_NUM,IN_BUF+2     ; in IN_BUF+2 steht die erkannte
                                                ; Wortnummer

MAIN3:      MOV    DISP_BUF+0,#20h      ; LEERZEICHEN
            MOV    DISP_BUF+1,#20h      ; LEERZEICHEN
            MOV    DISP_BUF+2,#20h      ; LEERZEICHEN
            MOV    DISP_BUF+3,#20h      ; LEERZEICHEN
            MOV    DISP_BUF+4,#20h      ; LEERZEICHEN
            MOV    DISP_BUF+5,#4Eh      ; N
            MOV    DISP_BUF+6,#52h      ; R
            MOV    DISP_BUF+7,#2Eh      ; .
            MOV    A,IN_BUF+2           ; erkannte Wortnummer
            MOV    DPTR,#ASCII_NUMM     ; DPTR steht auf ASCII_NUMM
                                                ; Tabelle
            MOVC   A,@A+DPTR            ; ASCII-Code des erkannten
                                                ; Wortes wird geladen
            MOV    ASCII_NUM,A          ; ASCII_NUM enthält ASCII-
                                                ; Code des erkannten Wortes

            MOV    DISP_BUF+8,ASCII_NUM  ; Speicherplatz-Nummer
            MOV    DISP_BUF+9,#20h      ; LEERZEICHEN
            MOV    DISP_BUF+10,#20h     ; LEERZEICHEN
            MOV    DISP_BUF+11,#20h     ; LEERZEICHEN
            MOV    DISP_BUF+12,#20h     ; LEERZEICHEN
            MOV    DISP_BUF+13,#20h     ; LEERZEICHEN
            MOV    DISP_BUF+14,#20h     ; LEERZEICHEN
            MOV    DISP_BUF+15,#20h     ; LEERZEICHEN
            CALL   CLEAR_DISPLAY
            CALL   SEND_DISPLAY
            LJMP   MAIN_PROG

```

*****Programmiermodus*****

```

PROG:      MOV    AUSGANGNUMMER,#01h    ; AUSGANGNUMMER wird auf
                                                ; 1 gesetzt
            MOV    MENUENUMMER,#01h    ; MENUENUMMER wird auf
                                                ; 1 gesetzt

            MOV    ASCII_NUM,#31h

TRAINW:    ;hier wird ein Wort zuerst im recording buffer aufgenommen, dann der Inhalt
            ; des recording buffer auf den Lautsprecher ausgegeben und dann das Wort
            ; unter der angegebenen Speicherplatznummer im NNSR abgespeichert.
            MOV    A,P6
            JNB    ACC.4,WEITER1A       ; liest Betriebsart Schalter 1 ein
            CALL   SCHLEIFE             ; zur Schalterentprellung
            MOV    A,P6
            JNB    ACC.4,WEITER1A
            RET

```

```

WEITER1A:    MOV    DISP_BUF+0,#20h           ; LEERZEICHEN
             MOV    DISP_BUF+1,#4Ch       ; L
             MOV    DISP_BUF+2,#45h       ; E
             MOV    DISP_BUF+3,#52h       ; R
             MOV    DISP_BUF+4,#4Eh       ; N
             MOV    DISP_BUF+5,#45h       ; E
             MOV    DISP_BUF+6,#4Eh       ; N
             MOV    DISP_BUF+7,#20h       ;LEERZEICHEN
             MOV    DISP_BUF+8,#57h       ; W
             MOV    DISP_BUF+9,#4Fh       ; O
             MOV    DISP_BUF+10,#52h      ; R
             MOV    DISP_BUF+11,#54h      ; T
             MOV    DISP_BUF+12,#20h      ; LEERZEICHEN
             MOV    DISP_BUF+13,ASCII_NUM ; AUSGANGSNUMMER
             MOV    DISP_BUF+14,#20h      ; LEERZEICHEN
             MOV    DISP_BUF+15,#20h      ; LEERZEICHEN
             CALL   SEND_DISPLAY
             CALL   TASTER2_ABF           ; MENÜ-TASTER
             CALL   TASTER3_ABF           ; STUFEN-TASTER
             CALL   TASTER4_ABF           ; QUITTIERUNG
             JNB    QUITT,WEITER1C        ; Keine Quittierung
                                           ; ==> überspringe AUSGABE
WEITER1B:    MOV    OUT_BUF+0,#REC_WORD   ; Wort im recording buffer
                                           ; aufzeichnen
             CALL   TO_NNSR
             CALL   FROM_NNSR
             MOV    OUT_BUF+0,#PLAYBIMM    ; Playback recording buffer
             CALL   TO_NNSR
             CALL   FROM_NNSR
             MOV    OUT_BUF+0,#STORWORD    ; Wort speichern
             MOV    OUT_BUF+1,AUSGANGNUMMER
             CALL   TO_NNSR
             CALL   FROM_NNSR
             MOV    A,IN_BUF+1             ; execution success code
             CJNE  A,#10h,WEITER1C        ; Speicher voll?
             CALL   SPEICHER_VOLL         ; „Speicher voll“ auf Display
WEITER1D:    CALL   TASTER2_ABF
             MOV    A,MENUENUMMER
             CJNE  A,#1,CLEARW            ; springt ins nächste Menü wenn
                                           ; Taste 2 betätigt
             MOV    A,AUSGANGNUMMER
             CALL   TASTER3_ABF
             CJNE  A,AUSGANGNUMMER,WEITER1C
             JMP    WEITER1D
WEITER1C:    MOV    A,MENUENUMMER
             CJNE  A,#1,CLEARW            ; springt ins nächste Menü wenn
                                           ; Taste 2 betätigt
             LJMP  TRAINW
CLEARW:      ; hier können alle abgespeicherten Sprachmuster eines Wortes gelöscht
             ; werden
             MOV    A,P6
             JNB   ACC.4,WEITER2A         ; liest Betriebsart Schalter 1 ein
             CALL  SCHLEIFE               ; zur Schalterentprellung
             MOV    A,P6
             JNB   ACC.4,WEITER2A
             RET

```

```

WEITER2A:      MOV  DISP_BUF+0,#20h          ; LEERZEICHEN
                MOV  DISP_BUF+1,#4Ch          ; L
                MOV  DISP_BUF+2,#0EFh        ; Ö
                MOV  DISP_BUF+3,#53h          ; S
                MOV  DISP_BUF+4,#43h          ; C
                MOV  DISP_BUF+5,#48h          ; H
                MOV  DISP_BUF+6,#45h          ; E
                MOV  DISP_BUF+7,#4Eh          ; N
                MOV  DISP_BUF+8,#20h          ; LEERZEICHEN
                MOV  DISP_BUF+9,#57h          ; W
                MOV  DISP_BUF+10,#4Fh         ; O
                MOV  DISP_BUF+11,#52h         ; R
                MOV  DISP_BUF+12,#54h         ; T
                MOV  DISP_BUF+13,#20h         ; LEERZEICHEN
                MOV  DISP_BUF+14,ASCII_NUM    ; AUSGANGSNUMMER
                MOV  DISP_BUF+15,#20h         ; LEERZEICHEN
                CALL SEND_DISPLAY
                CALL TASTER2_ABF             ; MENÜ-TASTER
                CALL TASTER3_ABF             ; STUFEN-TASTER
                CALL TASTER4_ABF             ; QUITTIERUNG
                JNB  QUITT,WEITER2C           ; Keine Quittierung
                                                    ; ==> überspringe AUSGABE
WEITER2B:      MOV  OUT_BUF+0,#CLEAWORD      ; Wort aus Wortspeicher löschen
                MOV  OUT_BUF+1,AUSGANGNUMMER
                CALL TO_NNSR
                CALL FROM_NNSR
WEITER2C:      MOV  A,MENUENUMMER
                CJNE A,#2,PLAYBACKW          ; Springt ins nächste Menü
                                                    ; wenn Taste 2 betätigt
                LJMP CLEARW

PLAYBACKW:     ; hier kann das erste abgespeicherte Sprachmuster eines Wortes auf den
                ; Lautsprecher ausgegeben werden
                ; nur bei „acoustic_feedback“ in Funktion
                MOV  A,P6
                JNB  ACC.4,WEITER3A          ; liest Betriebsart Schalter 1 ein
                CALL SCHLEIFE                ; zur Schalterentprellung
                MOV  A,P6
                JNB  ACC.4,WEITER3A
                RET
WEITER3A:      MOV  DISP_BUF+0,#50h          ; P
                MOV  DISP_BUF+1,#4Ch          ; L
                MOV  DISP_BUF+2,#41h          ; A
                MOV  DISP_BUF+3,#59h          ; Y
                MOV  DISP_BUF+4,#42h          ; B
                MOV  DISP_BUF+5,#41h          ; A
                MOV  DISP_BUF+6,#43h          ; C
                MOV  DISP_BUF+7,#4Bh          ; K
                MOV  DISP_BUF+8,#20h          ; LEERZEICHEN
                MOV  DISP_BUF+9,#57h          ; W
                MOV  DISP_BUF+10,#4Fh         ; O
                MOV  DISP_BUF+11,#52h         ; R
                MOV  DISP_BUF+12,#54h         ; T
                MOV  DISP_BUF+13,#20h         ; LEERZEICHEN
                MOV  DISP_BUF+14,ASCII_NUM    ; AUSGANGSNUMMER
                MOV  DISP_BUF+15,#20h         ; LEERZEICHEN
                CALL SEND_DISPLAY
                CALL TASTER2_ABF             ; MENÜ-TASTER
                CALL TASTER3_ABF             ; STUFEN-TASTER

```



```

CALL TASTER4_ABF ; QUITTIERUNG
JNB QUITT,WEITER3C ; Keine Quittierung
; ==> überspringe AUSGABE
WEITER3B: MOV OUT_BUF+0,#PLAYBWOR ; Wort auf Lautsprecher
MOV OUT_BUF+1,AUSGANGNUMMER
CALL TO_NNSR
CALL FROM_NNSR
WEITER3C: MOV A,MENUENUMMER
CJNE A,#3,LEARNALL ; springt ins nächste Menü
; wenn Taste 2 betätigt
LJMP PLAYBACKW

LEARNALL: ; nachdem verschiedene Wortmuster abgespeichert wurden, müssen sie
; mit dem Learn - Befehl gelernt werden
MOV A,P6
JNB ACC.4,WEITER4A ; liest Betriebsart Schalter 1 ein
CALL SCHLEIFE
MOV A,P6
JNB ACC.4,WEITER4A
RET
WEITER4A: MOV DISP_BUF+0,#20h ; LEERZEICHEN
MOV DISP_BUF+1,#20h ; LEERZEICHEN
MOV DISP_BUF+2,#20h ; LEERZEICHEN
MOV DISP_BUF+3,#54h ; T
MOV DISP_BUF+4,#52h ; R
MOV DISP_BUF+5,#41h ; A
MOV DISP_BUF+6,#49h ; I
MOV DISP_BUF+7,#4Eh ; N
MOV DISP_BUF+8,#49h ; I
MOV DISP_BUF+9,#45h ; E
MOV DISP_BUF+10,#52h ; R
MOV DISP_BUF+11,#45h ; E
MOV DISP_BUF+12,#4Eh ; N
MOV DISP_BUF+13,#20h ; LEERZEICHEN
MOV DISP_BUF+14,#20h ; LEERZEICHEN
MOV DISP_BUF+15,#20h ; LEERZEICHEN
CALL SEND_DISPLAY
CALL TASTER2_ABF ; MENÜ-TASTER
CALL TASTER4_ABF ; QUITTIERUNG
JNB QUITT,WEITER4C ; keine Quittierung
; ==> überspringe AUSGABE
WEITER4B: MOV OUT_BUF+0,#LEARN
CALL TO_NNSR
CALL FROM_NNSR
LJMP RELSET ; springt nach der Learn Procedur
; ins nächste Menü
WEITER4C: MOV A,MENUENUMMER
CJNE A,#4,RELSET ; springt ins nächste Menü
; wenn Taste 2 betätigt
LJMP LEARNALL

```

```

RELSET:          ; hier kann die Einstellung der Relais vorgenommen werden
                 ; K0:          Relais konstant, bei Resetwort wird es ausgeschaltet
                 ; K1:          Relais konstant, bei Resetwort wird es eingeschaltet
                 ; P1-P99:      Relais gepulst 1-99s möglich, bei Resetwort wird es
                 ;              ausgeschaltet
                 ; beim anwählen eines Relais ist die zuerst angezeigte immer die im SRAM
                 ; abgespeicherte
MOV   B,#00h          ; Hilfsacc. wird als Zähler
                           ; verwendet
CALL  REL_EIN         ; von SRAM in REL_BUF
MOV   A,AUSGANGNUMMER ; wenn AUSGANGNUMMER
CLR   C              ; grösser als 8 ist
SUBB  A,#09h         ; auf 1 zurücksetzen
JNC   WEITER5E
JMP   WEITER5A
WEITER5E:        MOV   AUSGANGNUMMER,#01h
WEITER5A:        MOV   ASCII_NUM,#31h          ; ASCII-Code für 1
                 MOV   A,P6
                 JNB  ACC.4,WEITER5B         ; liest Betriebsart Schalter 1 ein
                 CALL  SCHLEIFE
                 MOV   A,P6
                 JNB  ACC.4,WEITER5B
                 RET
WEITER5B:        MOV   A,AUSGANGNUMMER        ; liest aktuelle Einstellung
                           ; des Relais
                 ADD   A,#REL_BUF
                 MOV   R0,A
                 MOV   A,@R0
                 MOV   REL_BUF+0,A          ; aktuelle Einstellung des Relais
                           ; in REL_BUF+0
                 MOV   DISP_BUF+0,#41h      ; A
                 MOV   DISP_BUF+1,#55h      ; U
                 MOV   DISP_BUF+2,#53h      ; S
                 MOV   DISP_BUF+3,#47h      ; G
                 MOV   DISP_BUF+4,#41h      ; A
                 MOV   DISP_BUF+5,#4Eh      ; N
                 MOV   DISP_BUF+6,#47h      ; G
                 MOV   DISP_BUF+7,#20h      ; LEERZEICHEN
                 MOV   DISP_BUF+8,ASCII_NUM  ; AUSGANGSNUMMER
                 MOV   DISP_BUF+9,#20h      ; LEERZEICHEN
                 MOV   DISP_BUF+10,#20h     ; LEERZEICHEN
                 MOV   DISP_BUF+11,#20h     ; LEERZEICHEN
                 MOV   A,REL_BUF+0
                 MOV   DPTR,#ASCII_REL1
                 MOVC  A,@A+DPTR
                 MOV   DISP_BUF+12,A
                 MOV   A,REL_BUF+0
                 MOV   DPTR,#ASCII_REL2
                 MOVC  A,@A+DPTR
                 MOV   DISP_BUF+13,A
                 MOV   A,REL_BUF+0
                 MOV   DPTR,#ASCII_REL3
                 MOVC  A,@A+DPTR
                 MOV   DISP_BUF+14,A
                 MOV   DISP_BUF+15,#20h     ; LEERZEICHEN
                 CALL  SEND_DISPLAY
                 CALL  TASTER2_ABF         ; MENÜ-TASTER
                 CALL  REL_ABF            ; RELAIS-Einstellungen
                 CALL  TASTER4_ABF        ; QUITTIERUNG

```

```

JNB   QUITT,WEITER5C           ; keine Quittierung==>
                                   ; überspringe SRAM einschreiben

INC   AUSGANGNUMMER
MOV   A,AUSGANGNUMMER
CJNE  A,#9,WEITER5D           ; NUR AUSGANG 1-8 möglich
MOV   AUSGANGNUMMER,#1h
WEITER5D:
MOV   A,AUSGANGNUMMER
MOV   DPTR,#ASCII_NUMM       ; DPTR steht auf
                                   ; ASCII_NUMM Tabelle
MOV   A,@A+DPTR              ; ASCII-Code d.
                                   ; AUSGANGSNUMMER wird
                                   ; geladen
MOV   ASCII_NUM,A           ; ASCII_NUM enthält ASCII-
                                   ; Code der aktuellen
                                   ; AUSGANGSNUMMER

MOV   B,#0h
MOV   DPTR,#RELAIS1
MOV   R0,#REL_BUF+1
NEXT5B:
MOV   A,@R0
MOVX  @DPTR,A                ; schreibt Relaisstellungen von
                                   ; REL_BUF in SRAM

INC   DPTR
INC   R0
CJNE  R0,#REL_BUF+9,NEXT5B    ; alle Einstellungen in SRAM?
WEITER5C:
MOV   A,MENUENUMMER
CJNE  A,#5,ZURUECK           ; springt ins nächste Menü
                                   ; wenn Taste 2 betätigt

LJMP  WEITER5A
ZURUECK:
LJMP  TRAINW
RET

```

;***** RS 232 *****

```

RS232:           ; hier werden ankommende Befehle an den NNSR weitergeschickt
                 ; die Antworten wieder auf die RS232 geschrieben
                 ; die serielle Schnittstelle ist 9600 baud, 1Startbit 8 Datenbit 1 Stopbit
                 ; initialisiert
MOV   A,P6
JNB   ACC.5,W_232A           ; liest Betriebsart Schalter 5 ein
CALL  SCHLEIFE
MOV   A,P6
JNB   ACC.5,W_232A
RET

```

```

W_232A:      MOV  DISP_BUF+0,#20h      ; LEERZEICHEN
              MOV  DISP_BUF+1,#20h      ; LEERZEICHEN
              MOV  DISP_BUF+2,#20h      ; LEERZEICHEN
              MOV  DISP_BUF+3,#20h      ; LEERZEICHEN
              MOV  DISP_BUF+4,#20h      ; LEERZEICHEN
              MOV  DISP_BUF+5,#52h      ; R
              MOV  DISP_BUF+6,#53h      ; S
              MOV  DISP_BUF+7,#20h      ; LEERZEICHEN
              MOV  DISP_BUF+8,#32h      ; 2
              MOV  DISP_BUF+9,#33h      ; 3
              MOV  DISP_BUF+10,#32h     ; 2
              MOV  DISP_BUF+11,#20h     ; LEERZEICHEN
              MOV  DISP_BUF+12,#20h     ; LEERZEICHEN
              MOV  DISP_BUF+13,#20h     ; LEERZEICHEN
              MOV  DISP_BUF+14,#20h     ; LEERZEICHEN
              MOV  DISP_BUF+15,#20h     ; LEERZEICHEN
              CALL  SEND_DISPLAY

RS232_DOWN_INIT:      ; Serial Interface
                      ; Mode 1
                      CLR  SM0
                      SETB SM1
                      CLR  SM20
                      SETB REN0        ; Empfangen
                      ANL  PCON,#7Fh   ; SMOD=0
                      CLR  BD          ; Baudratengenerator nicht
                                      ; verwenden
                      SETB CTS_I       ; Eingang
                      SETB RTS_I       ; RTS Rücksetzen
                                      ; Timer 1 (Baudrate)
                      ORL  TMOD,#0F0h   ; TMOD=0010 xxxx B
                      ANL  TMOD,#02Fh   ; (Mode 2)
                      MOV  TH1,#0FCh    ; Reload-Wert (Baudrate)
                      MOV  TL1,#0FCh    ; Vorbelegung
                      SETB TR1          ; Timer starten
                      SETB EAL          ; Interrupts generell enable

RS232_DOWN:      MOV  R0,OUT_BUF+0
NEXT_DOWN:      ORL  IEN2,#08h        ; ES1=1
WARTE_DOWN1:    MOV  A,P6
                JNB  ACC.5,W_232B     ; liest Betriebsart Schalter 5 ein
                CALL SCHLEIFE
                MOV  A,P6
                JNB  ACC.5,W_232B
                RET

W_232B:         JNB  CTS_I,WARTE_DOWN1 ; CTS set?
                ANL  IEN2,#0F7h      ; ES1=0
                CLR  RI0              ; Receive Flag reset
                CLR  TI0              ; Transmit Flag reset
                CLR  RTS_I            ; RTS setzen
                ORL  IEN2,#08h

WARTE_DOWN2:    JNB  RI0,WARTE_DOWN2
                MOV  @R0,S0BUF
                INC  R0

WARTE_DOWN3:    JNB  CTS_I,WARTE_DOWN3
                SETB RTS_I
                CJNE R0,#OUT_BUF+6,NEXT_DOWN

                CALL TO_NNSR
                CALL FROM_NNSR

```

```

RS232_UP_INIT:    CLR    SM0                ; Mode 1
                  SETB   SM1
                  CLR    SM20
                  CLR    RENO                ; Senden
                  ANL    PCON,#7Fh         ; SMOD=0
                  CLR    BD                ; Baudratengenerator nicht
                                          ; verwenden
                  SETB   CTS_I            ; Eingang
                  SETB   RTS_I            ; RTS Rücksetzen
                                          ; Timer 1 (Baudrate)
                  ORL    TMOD,#0F0h       ; TMOD=0010 xxxx B
                  ANL    TMOD,#02Fh       ; (Mode 2)
                  MOV    TH1,#0FCh        ; Reload-Wert (Baudrate)
                  MOV    TL1,#0FCh        ; Vorbelegung
                  SETB   TR1              ; Timer starten
                  SETB   EAL              ; Interrupts generell enable
RS232_UP:        MOV    R0,#IN_BUF+0
NEXT_UP:         CLR    RTS_I            ; RTS setzen
                  ORL    IEN2,#08h        ; ES1=1
WARTE_UP1:      JB     CTS_I,WARTE_UP1    ; CTS set ?
                  ANL    IEN2,#0F7h       ; ES1=0
                  CLR    TI0              ; Transmit Flag reset
                  MOV    S0BUF,@R0
                  CLR    RI0              ; Receive Flag reset
                  ORL    IEN2,#08h
WARTE_UP2:      JNB   TI0,WARTE_UP2        ; fertig Übertragen?
WARTE_UP3:      JNB   CTS_I,WARTE_UP3
                  INC    R0
                  CJNE   R0,#IN_BUF+6,NEXT_UP ; end of buffer?
                  SETB   RTS_I
                  LJMP   RS232
                  RET

;***** Initialisierung des NNSR *****

INIT_NNSR:      ; wird die Configuration geändert werden die Wortliste und die Sprachmuster
                ; gelöscht.
                MOV    OUT_BUF+0,#CONFINEW ; Configuration des NNSR
                MOV    OUT_BUF+1,# 00h     ; acoustic_feedback set
                MOV    OUT_BUF+2,# 0Ch     ; 11 verschiedene Wörter
                CALL   TO_NNSR
                CALL   FROM_NNSR
                RET

```

***** Sende Befehl vom OUT_BUF zum NNSR *****

```
TO_NNSR:           ; hier werden Befehle, die in OUT_BUF+0 bis OUT_BUF+5 stehen an den
                   ; NNSR geschickt. Das Byte 0 muß zuletzt geschickt werden, da es im NNSR
                   ; einen Interrupt auslöst
MOV   DPTR,#HDR1           ; Adresse 2.
                               ; HOST-DATA-Register
NEXTOUT:           MOV   R0,#OUT_BUF+1           ; Adresse 2. Byte in OUT_BUF
                   MOV   A,@R0                 ; lese Byte vom Buffer
                   MOVX  @DPTR,A               ; schreibe in HOST-DATA-Reg.
                   INC   DPTR                   ; inc. Pointer f. HOST-DATA-Reg.
                   INC   R0                     ; inc. Pointer für Buffer
                   CJNE  R0,#OUT_BUF+6,NEXTOUT ; Ende des Buffer?
                   MOV   DPTR,#HDR0           ; JA: address 1.
                               ; HOST-DATA-Reg.
                   MOV   A,OUT_BUF+0           ; lese 1.Byte vom Buffer
                   MOVX  @DPTR,A               ; schreibe in HDR0
                                               ; (Interrupt in NNSR)
                   RET
```

***** Lese Antwort vom NNSR *****

```
FROM_NNSR:        ; Antworten vom NNSR werden in IN_BUF+0 bis IN_BUF+5 geschrieben.
                   ; Der NNSR löst über seine HMSG-Leitung einen Interrupt aus, danach
                   ; werden die 6 Byte eingelesen
HMSG_WT:          CLR   NNSR_MSG
                   SETB  P1.6                   ; signal HOST-READY to NNSR
                   SETB  IEN1.1                 ; ermöglicht INT 2
                   JNB   NNSR_MSG,HMSG_WT       ; warte auf Antwort vom NNSR
                   CLR   IEN1.1                 ; schaltet INT 2 ab
                   CLR   P1.6                   ; löscht HOST_READY
                   MOV   DPTR,#HDR0             ; Adresse HDR0
                   MOV   R0,#IN_BUF            ; Adresse IN_BUF+0
NEXTIN:           MOVX  A,@DPTR                 ; lese Byte vom externen
                                               ; HOST-DATA-Register
                   MOV   @R0,A                 ; und schreibe es in IN_BUF
                   INC   DPTR
                   INC   R0                     ; increment Pointer
                   CJNE  R0,#IN_BUF+6,NEXTIN    ; alles eingeschrieben?
                   CALL  SCHLEIFE
                   RET
```

;***** NNSR schläft *****

SCHLAFE: ; wird das Schlüsselwort erkannt springt der NNSR erst nach nochmaliger
; Erkennung des Schlüsselwortes aus dieser Routine

```
CALL CLEAR_DISPLAY
MOV DISP_BUF+0,#20h ; LEERZEICHEN
MOV DISP_BUF+1,#20h ; LEERZEICHEN
MOV DISP_BUF+2,#49h ; I
MOV DISP_BUF+3,#43h ; C
MOV DISP_BUF+4,#48h ; H
MOV DISP_BUF+5,#20h ; LEERZEICHEN
MOV DISP_BUF+6,#53h ; S
MOV DISP_BUF+7,#43h ; C
MOV DISP_BUF+8,#48h ; H
MOV DISP_BUF+9,#4Ch ; L
MOV DISP_BUF+10,#41h ; A
MOV DISP_BUF+11,#46h ; F
MOV DISP_BUF+12,#45h ; E
MOV DISP_BUF+13,#20h ; LEERZEICHEN
MOV DISP_BUF+14,#20h ; LEERZEICHEN
MOV DISP_BUF+15,#20h ; LEERZEICHEN
CALL SEND_DISPLAY
MOV OUT_BUF+0,#LISTEN
CALL TO_NNSR
CALL FROM_NNSR
CALL ERKENNUNG
JB POSITIV,WEITERSCH
JMP SCHLAFE
```

WEITERSCH: MOV A,IN_BUF+2
CJNE A,#0Ch,SCHLAFE ; vergleicht mit Schlüsselwort
RET

;***** Auswertung des erkannten Wortes *****

ERKENNUNG: ; ein Wort wird als Erkannt akzeptiert wenn
; corell. + (spectr. moment / 2) > 60h erfüllt ist
CLR POSITIV
MOV A,IN_BUF+5 ; spectr. Moment wird in A geladen
MOV B,#2h
DIV AB
ADD A,IN_BUF+4 ; corell. wird in A geladen
CLR C
SUBB A,#60h ; Schwelle für positive Erkennung
JC NEGATIV_ERK ; negative Erkennung wenn
; Carry-Flag gesetzt
SETB POSITIV ; positive Erkennung
NEGATIV_ERK: RET

***** SCHALTET DIE RELAIS *****

```

RELAIS_SCHALTEN:  JB      RELAIS,REL_WEITER      ; wenn kein Relais selectiert
                                                           ; zurück

                                                           RET
REL_WEITER:      CLR      RELAIS                ; Bestätigung löschen
                                                           CALL     REL_EIN                ; liest Relaiseinstellung aus
                                                           ; SRAM
                                                           MOV     A,REL_NUM              ; Relaisnummer
REL_RESET:      CJNE     A,#09h,REL_EINSCHALTEN
                                                           MOV     A,IN_BUF+2            ; Besätigung EIN oder AUS
                                                           CJNE     A,#0Bh,REL_RES
                                                           RET

REL_RES:      ; Reset aller Relais
                                                           ; Einstellung von Relais 1
                                                           MOV     A,REL_BUF+1
REL_RES1B:     CJNE     A,#01h,REL_RES1B
REL_RES2A:     MOV     REL_TIMER+1,#0FFh        ; Relais 1 ein
                                                           JMP     REL_RES2A
REL_RES1B:     MOV     REL_TIMER+1,#00h        ; Relais 1 aus
REL_RES2A:     MOV     A,REL_BUF+2
                                                           CJNE     A,#01h,REL_RES2B
REL_RES2B:     MOV     REL_TIMER+2,#0FFh        ; Relais 2 ein
REL_RES3A:     JMP     REL_RES3A
REL_RES2B:     MOV     REL_TIMER+2,#00h        ; Relais 2 aus
REL_RES3A:     MOV     A,REL_BUF+3
                                                           CJNE     A,#01h,REL_RES3B
REL_RES3B:     MOV     REL_TIMER+3,#0FFh        ; Relais 3 ein
REL_RES4A:     JMP     REL_RES4A
REL_RES3B:     MOV     REL_TIMER+3,#00h        ; Relais 3 aus
REL_RES4A:     MOV     A,REL_BUF+4
                                                           CJNE     A,#01h,REL_RES4B
REL_RES4B:     MOV     REL_TIMER+4,#0FFh        ; Relais 4 ein
REL_RES5A:     JMP     REL_RES5A
REL_RES4B:     MOV     REL_TIMER+4,#00h        ; Relais 4 aus
REL_RES5A:     MOV     A,REL_BUF+5
                                                           CJNE     A,#01h,REL_RES5B
REL_RES5B:     MOV     REL_TIMER+5,#0FFh        ; Relais 5 ein
REL_RES6A:     JMP     REL_RES6A
REL_RES5B:     MOV     REL_TIMER+5,#00h        ; Relais 5 aus
REL_RES6A:     MOV     A,REL_BUF+6
                                                           CJNE     A,#01h,REL_RES6B
REL_RES6B:     MOV     REL_TIMER+6,#0FFh        ; Relais 6 ein
REL_RES7A:     JMP     REL_RES7A
REL_RES6B:     MOV     REL_TIMER+6,#00h        ; Relais 6 aus
REL_RES7A:     MOV     A,REL_BUF+7
                                                           CJNE     A,#01h,REL_RES7B
REL_RES7B:     MOV     REL_TIMER+7,#0FFh        ; Relais 7 ein
REL_RES8A:     JMP     REL_RES8A
REL_RES7B:     MOV     REL_TIMER+7,#00h        ; Relais 7 aus
REL_RES8A:     MOV     A,REL_BUF+8
                                                           CJNE     A,#01h,REL_RES8B
REL_RES8B:     MOV     REL_TIMER+8,#0FFh        ; Relais 8 ein
REL_RES8B:     JMP     REL_EXIT
REL_RES8B:     MOV     REL_TIMER+8,#00h        ; Relais 8 aus
                                                           RET

```



```

REL_EINSCHALTEN:                                ; einzelnes Relais selectiert
MOV A,IN_BUF+2                                  ; Relais EIN oder AUS
CJNE A,#0Ah,REL_AUSSCHALTEN
MOV A,REL_NUM                                    ; Relaisnummer
ADD A,#REL_BUF+0                                ; Adresse des
                                                ; angesprochenen REL_BUF

MOV R0,A
MOV A,@R0                                        ; Einstellung des
                                                ; angesprochenen Relais
                                                ; in REL_BUF+0 abgespeichert

MOV REL_BUF+0,A
CJNE A,#00h,REL_KONSTE                          ; Relais konstant
JMP REL_KONSTE1

REL_KONSTE: CJNE A,#01h,REL_PULS                ; Relais konstant
REL_KONSTE1: MOV A,REL_NUM                      ; Nummer des ausgewählten
                                                ; Relais
ADD A,#REL_TIMER+0                              ; Adresse des Zählers
MOV R0,A
MOV @R0,#0FFh                                   ; angesprochenes Relais
                                                ; einschalten

RET

REL_PULS: MOV A,REL_NUM                        ; Nummer des ausgewählten
                                                ; Relais
ADD A,#REL_TIMER+0                              ; Adresse des Zählers
MOV R0,A
MOV A,REL_BUF+0
DEC A
MOV @R0,A                                       ; schreibt Pulsdauer
                                                ; in selectierten Relais-timer

RET

REL_AUSSCHALTEN: MOV A,REL_NUM                 ; Relaisnummer
ADD A,#REL_TIMER+0                              ; Adresse des Zählers
MOV R0,A
MOV @R0,#000h                                   ; angesprochenes Relais
                                                ; ausschalten

REL_EXIT: RET

```

***** ZÄHLER für die Relais *****

```

TIMER: ; nachdem ein Relais selectiert wurde, muß innerhalb von 10 s eine Bestätigung
; kommen. Ist diese Zeit überschritten, wird die Selection wieder gelöscht
JNB RELAIS,REL_TI1                               ; Relais selectiert ?
DEC ZEITSCHLITZ                                  ; Zeit läuft ab
MOV A,ZEITSCHLITZ
CJNE A,#00h,REL_TI1                             ; Zeit abgelaufen?
CLR RELAIS                                       ; Selection löschen
; in REL_TIMER steht ob das Relais ein-, aus- oder gepulst geschaltet wird
; FFh Relais ein
; 00h Relais aus
; 0-63h Relais ein, Zähler wird im Sekundentakt decremementiert

REL_TI1: MOV A,REL_TIMER+1
CJNE A,#0FFh,REL_TI1A
CLR P4.0                                         ; Relais 1 ein
JMP REL_TI2

REL_TI1A: CJNE A,#000h,REL_TI1B
SETB P4.0                                        ; Relais 1 aus
JMP REL_TI2

REL_TI1B: DEC REL_TIMER+1                       ; Relais 1 gepulst
CLR P4.0

REL_TI2: MOV A,REL_TIMER+2

```

```

CJNE A,#0FFh,REL_TI2A
CLR P4.1
JMP REL_TI3
REL_TI2A: CJNE A,#000h,REL_TI2B
SETB P4.1
JMP REL_TI3
REL_TI2B: DEC REL_TIMER+2
CLR P4.1
REL_TI3: MOV A,REL_TIMER+3
CJNE A,#0FFh,REL_TI3A
CLR P4.2
JMP REL_TI4
REL_TI3A: CJNE A,#000h,REL_TI3B
SETB P4.2
JMP REL_TI4
REL_TI3B: DEC REL_TIMER+3
CLR P4.2
REL_TI4: MOV A,REL_TIMER+4
CJNE A,#0FFh,REL_TI4A
CLR P4.3
JMP REL_TI5
REL_TI4A: CJNE A,#000h,REL_TI4B
SETB P4.3
JMP REL_TI5
REL_TI4B: DEC REL_TIMER+4
CLR P4.3
REL_TI5: MOV A,REL_TIMER+5
CJNE A,#0FFh,REL_TI5A
CLR P4.4
JMP REL_TI6
REL_TI5A: CJNE A,#000h,REL_TI5B
SETB P4.4
JMP REL_TI6
REL_TI5B: DEC REL_TIMER+5
CLR P4.4
REL_TI6: MOV A,REL_TIMER+6
CJNE A,#0FFh,REL_TI6A
CLR P4.5
JMP REL_TI7
REL_TI6A: CJNE A,#000h,REL_TI6B
SETB P4.5
JMP REL_TI7
REL_TI6B: DEC REL_TIMER+6
CLR P4.5
REL_TI7: MOV A,REL_TIMER+7
CJNE A,#0FFh,REL_TI7A
CLR P4.6
JMP REL_TI8
REL_TI7A: CJNE A,#000h,REL_TI7B
SETB P4.6
JMP REL_TI8
REL_TI7B: DEC REL_TIMER+7
CLR P4.6
REL_TI8: MOV A,REL_TIMER+8
CJNE A,#0FFh,REL_TI8A
CLR P4.7
JMP TIMER_END

```

```

REL_TI8A:      CJNE  A,#000h,REL_TI8B
                SETB  P4.7
                JMP   TIMER_END
REL_TI8B:      DEC   REL_TIMER+8
                CLR   P4.7
TIMER_END:    RET

```

;***** Zähler für T0 *****

```

TIMER0:        ; wenn der Timer 0 einen Interrupt auslöst wird der ZAEHLERT0
                ; incrementiert. Wenn er von 0-Fh gezählt hat ist ca. 1 Sekunde vergangen
                ; und die Routine TIMER aufgerufen
                INC   ZAEHLERT0
                MOV   A,ZAEHLERT0
                CJNE  A,#0Fh,TIMER0_W
                MOV   ZAEHLERT0,#0h
                CALL  TIMER
TIMER0_W:      RET

```

;***** Fehlermeldung "nicht trainiert" *****

```

NOT_TRAINED:   MOV   DISP_BUF+0,#20h           ; LEERZEICHEN
                MOV   DISP_BUF+1,#4Eh         ; N
                MOV   DISP_BUF+2,#49h         ; I
                MOV   DISP_BUF+3,#43h         ; C
                MOV   DISP_BUF+4,#48h         ; H
                MOV   DISP_BUF+5,#54h         ; T
                MOV   DISP_BUF+6,#20h         ; LEERZEICHEN
                MOV   DISP_BUF+7,#54h         ; T
                MOV   DISP_BUF+8,#52h         ; R
                MOV   DISP_BUF+9,#41h         ; A
                MOV   DISP_BUF+10,#49h        ; I
                MOV   DISP_BUF+11,#4Eh       ; N
                MOV   DISP_BUF+12,#49h       ; I
                MOV   DISP_BUF+13,#45h       ; E
                MOV   DISP_BUF+14,#52h       ; R
                MOV   DISP_BUF+15,#54h       ; T
                CALL  SEND_DISPLAY
                RET

```

***** Fehlermeldung "Speicher voll" *****

```

SPEICHER_VOLL:    MOV    DISP_BUF+0,#20h    ; LEERZEICHEN
                  MOV    DISP_BUF+1,#20h    ; LEERZEICHEN
                  MOV    DISP_BUF+2,#53h    ; S
                  MOV    DISP_BUF+3,#50h    ; P
                  MOV    DISP_BUF+4,#45h    ; E
                  MOV    DISP_BUF+5,#49h    ; I
                  MOV    DISP_BUF+6,#43h    ; C
                  MOV    DISP_BUF+7,#48h    ; H
                  MOV    DISP_BUF+8,#45h    ; E
                  MOV    DISP_BUF+9,#52h    ; R
                  MOV    DISP_BUF+10,#20h   ; LEERZEICHEN
                  MOV    DISP_BUF+11,#56h   ; V
                  MOV    DISP_BUF+12,#4Fh   ; O
                  MOV    DISP_BUF+13,#4Ch   ; L
                  MOV    DISP_BUF+14,#4Ch   ; L
                  MOV    DISP_BUF+15,#20h   ; LEERZEICHEN
                  CALL   CLEAR_DISPLAY
                  CALL   SEND_DISPLAY
                  RET

```

***** Initialisierung des Display *****

```

INIT_DISPLAY:    MOV    DISP_BUF+0,#110000b    ; Function set
                  MOV    DISP_BUF+1,#001000b    ; Display off
                  MOV    DISP_BUF+2,#000001b    ; Clear Display
                  MOV    DISP_BUF+3,#000010b    ; Cursor at Home
                  MOV    DISP_BUF+4,#001100b    ; Display on
                  MOV    DISP_BUF+5,#000000b    ; schreibe "end of text"
INIT_DIS0:      MOV    R0,#DISP_BUF            ; Adresse von DISP_BUF
                  CALL   LCD_RDY
                  CLR    DISP_RW
                  SETB   DISP_E                ; set Display enable
                  MOV    P5,@R0              ; Befehl ins Display- Register
                  CLR    DISP_E                ; falling edge of enable
                  SETB   DISP_RW
                  INC    R0                    ; increment pointer for buffer
                  CJNE   @R0,#00H,INIT_DIS0    ; Ende ?
                  SETB   DISP_RS              ; Register Select on "data"
                  RET

```

***** Clear Display *****

```

CLEAR_DISPLAY:  CALL   LCD_RDY
                  CLR    DISP_RW
                  SETB   DISP_E                ; set display enable
                  MOV    P5,#01H              ; "clear display"
                  CLR    DISP_E                ; falling edge of enable
                  SETB   DISP_RW
                  SETB   DISP_RS              ; Register Select on "data"
                  RET

```

;***** sende ASCII- Code von DISP_BUF zum Display *****

```
SEND_DISPLAY:    MOV    R0,#DISP_BUF           ; Adresse des DISP_BUF
SEND_NEXT:      CALL   LCD_RDY
                SETB   DISP_RS           ; Register Select on "data"
                CLR    DISP_RW
                SETB   DISP_E           ; set display enable
                MOV    P5,@R0           ; schreibe nächsten ASCII-Code
                                           ; zum Display
                CLR    DISP_E           ; falling edge of enable
                SETB   DISP_RW
                INC    R0                ; increment Pointer
                CJNE   R0,#DISP_BUF+16,SEND_NEXT ; Ende ?
                RET                      ; Ja
```

;***** Abfrage ob LCD bereit *****

```
LCD_RDY:        CLR    DISP_E
                CLR    DISP_RS           ; Register Select on "command"
                SETB   DISP_RW           ; BUSY-Flag lesen
                SETB   DISP_E
                MOV    P5,0FFh          ; P5 muß high sein beim lesen
                JB     P5.7,LCD_RDY      ; warte bis Display not busy
                CLR    DISP_E
                RET
```

;***** Abfrage von Taster 2 *****

```
TASTER2_ABF:    MOV    A,P6
                JB     ACC.6,ZURUECK_T2 ; MENÜ-TASTER wird abgefragt
                CALL   SCHLEIFE         ; Zählschleife zur
                                           ; Tastenentprellung
                MOV    A,P6
                JB     ACC.6,ZURUECK_T2 ; MENÜ-TASTER wird
                                           ; nochmals abgefragt
WARTE_T2:       MOV    A,P6
                JNB   ACC.6,WARTE_T2    ; Wartet auf loslassen der Taste
                CALL   SCHLEIFE         ; Zählschleife zur
                                           ; Tastenentprellung
                MOV    A,P6
                JNB   ACC.6,WARTE_T2    ; MENÜ-TASTER wird
                                           ; nochmals abgefragt
                INC    MENUENUMMER
                MOV    A,MENUENUMMER
                CJNE   A,#6,ZURUECK_T2 ; NUR MENÜ 1-5 möglich
                MOV    MENUENUMMER,#1h
ZURUECK_T2:     RET
```

;***** Abfrage von Taster 3 *****

```
TASTER3_ABF:    MOV    A,P6
                JB     ACC.7,ZURUECK_T3 ; STUFEN-TASTER
                                           ; wird abgefragt
                CALL   SCHLEIFE         ; Zählschleife zur
                                           ; Tastenentprellung
                MOV    A,P6
                JB     ACC.7,ZURUECK_T3 ; STUFEN-TASTER wird
                                           ; nochmals abgefragt
WARTE_T3:       MOV    A,P6
                JNB   ACC.7,WARTE_T3    ; Wartet auf loslassen der Taste
```

```

CALL  SCHLEIFE                ; Zählschleife zur
                                ; Tastenentprellung
MOV   A,P6
JNB   ACC.7,WARTE_T3          ; STUFEN-TASTER wird
                                ; nochmals abgefragt
INC   AUSGANGNUMMER           ; nächster AUSGANG
MOV   A,AUSGANGNUMMER
CJNE  A,#13,WEITER_T3        ; NUR AUSGANG 1-12 möglich
WEITER_T3:
MOV   AUSGANGNUMMER,#1h
MOV   A,AUSGANGNUMMER
MOV   DPTR,#ASCII_NUMM       ; DPTR steht auf
                                ; ASCII_NUMM Tabelle
MOVC  A,@A+DPTR              ; ASCII-Code der
                                ; AUSGANGSNUMMER wird
                                ; geladen
MOV   ASCII_NUM,A            ; ASCII_NUM enthält
                                ; ASCII-Code der aktuellen
                                ; AUSGANGSNUMMER
ZURUECK_T3:
RET

```

;***** Abfrage von Taster 3 und Relaiseinstellung *****

```

REL_ABF:      MOV    A,P6
              JB     ACC.7,ZURUECK_REL      ; STUFEN-TASTER wird
                                              ; abgefragt
              CALL   SCHLEIFE              ; Zählschleife zur
                                              ; Tastenentprellung
              MOV    A,P6
              JB     ACC.7,ZURUECK_REL      ; STUFEN-TASTER wird
                                              ; nochmals abgefragt
WARTE_REL:    MOV    A,P6
              JNB   ACC.7,WARTE_REL        ; Wartet auf loslassen der Taste
              CALL   SCHLEIFE              ; Zählschleife zur
                                              ; Tastenentprellung
              MOV    A,P6
              JNB   ACC.7,WARTE_REL        ; STUFEN-TASTER wird
                                              ; nochmals abgefragt
              MOV    A,REL_BUF+0
              MOV    A,B
              CJNE  A,#64h,WEITER_REL2
              CALL   REL_EIN
              MOV    B,#00h
              INC    AUSGANGNUMMER        ; nächster AUSGANG
              MOV    A,AUSGANGNUMMER
              CJNE  A,#9,WEITER_REL1      ; NUR AUSGANG 1-8 möglich
              MOV    AUSGANGNUMMER,#1h
WEITER_REL1: MOV    A,AUSGANGNUMMER
              MOV    DPTR,#ASCII_NUMM     ; DPTR steht auf
                                              ; ASCII_NUMM Tabelle
              MOVC  A,@A+DPTR             ; ASCII-Code der
                                              ; AUSGANGSNUMMER wird
                                              ; geladen
              MOV    ASCII_NUM,A         ; ASCII_NUM enthält
                                              ; ASCII-Code der aktuellen
                                              ; AUSGANGSNUMMER
              JMP    ZURUECK_REL
WEITER_REL2: INC    B
              INC    REL_BUF+0
              MOV    A,REL_BUF+0
              CJNE  A,#065h,WEITER_REL3
              MOV    REL_BUF+0,#00h
WEITER_REL3: MOV    A,AUSGANGNUMMER
              ADD    A,#REL_BUF
              MOV    R0,A
              MOV    @R0,REL_BUF+0
ZURUECK_REL: RET

```

;***** Liest Relaiseinstellung in REL_BUF ein *****

```

REL_EIN:      MOV    DPTR,#RELAIS1
              MOV    R0,#REL_BUF+1
REL_EINA:     MOVX   A,@DPTR              ; ladet Relaiseinstellungen vom
                                              ; SRAM in REL_BUF
              MOV    @R0,A
              INC    DPTR
              INC    R0
              CJNE  R0,#REL_BUF+9,REL_EINA ; alle Einstellungen
                                              ; geladen?
              RET

```

;***** Abfrage von Taster 4 *****

```
TASTER4_ABF:    CLR    QUITT                ; Quittierungsbit wird gelöscht
                JB     TASTER4,ZURUECK_T4    ; QUITTIERUNG wird abgefragt
                CALL   SCHLEIFE              ; Zählschleife zur
                                                ; Tastenentprellung
                JB     TASTER4,ZURUECK_T4    ; QUITTIERUNG wird nochmals
                                                ; abgefragt
WARTE_T4:       JNB    TASTER4,WARTE_T4      ; Wartet auf loslassen der Taste
                CALL   SCHLEIFE              ; Zählschleife zur
                                                ; Tastenentprellung
                JNB    TASTER4,WARTE_T4      ; QUITTIERUNG wird nochmals
                                                ; abgefragt
ZURUECK_T4:     SETB   QUITT                ; Quittierungsbit wird gesetzt
                RET
```

;***** Zählschleife zur Tastenentprellung*****

```
SCHLEIFE:       MOV    ZAEHLER,#0FFFFh      ; Zähler wird auf FFh gesetzt
WARTESCHLEIFE:  ACALL  SCHLEIFE10            ; Springt zur Schleife
                ACALL  SCHLEIFE10            ; Springt zur Schleife
                ACALL  SCHLEIFE10            ; Springt zur Schleife
                DJNZ   ZAEHLER,WARTESCHLEIFE
                RET
SCHLEIFE10:     NOP
                NOP
                RET
```

;***** ÜBERPRÜFUNG OB RELAISEINSTELLUNGEN GÜLTIG *****

```
RAM_OK:         ;wird nur bei der allerersten Inbetriebnahme des gepufferten SRAM benötigt
                CALL   REL_EIN
                MOV    R0,#REL_BUF+1
                MOV    DPTR,#RELAIS1
RAM_OK1:        MOV    A,@R0
                CLR    C
                SUBB   A,#65h                ; Zahl > 64h ungültige Einstellung
                JNC    RAM_OK2
                JMP    RAM_OK3
RAM_OK2:        MOV    A,#00h                ; setzt Relais auf konstant,
                                                ; bei Reset offen
                MOVX   @DPTR,A
RAM_OK3:        INC    R0
                INC    DPTR
                CJNE   R0,#REL_BUF+9,RAM_OK1
                RET
```


DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	29
DB	50h	; P	30
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	39
DB	50h	; P	40
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	49
DB	50h	; P	50
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	59
DB	50h	; P	60
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	69
DB	50h	; P	70
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	79
DB	50h	; P	80
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	
DB	50h	; P	

	DB	34h	;4	
	DB	34h	;4	
	DB	34h	;4	
	DB	34h	;4	
	DB	34h	;4	
	DB	35h	;5	50
	DB	35h	;5	
	DB	35h	;5	
	DB	35h	;5	
	DB	35h	;5	
	DB	35h	;5	
	DB	35h	;5	
	DB	35h	;5	
	DB	35h	;5	
	DB	35h	;5	
	DB	36h	;6	60
	DB	36h	;6	
	DB	36h	;6	
	DB	36h	;6	
	DB	36h	;6	
	DB	36h	;6	
	DB	36h	;6	
	DB	36h	;6	
	DB	36h	;6	
	DB	36h	;6	
	DB	37h	;7	70
	DB	37h	;7	
	DB	37h	;7	
	DB	37h	;7	
	DB	37h	;7	
	DB	37h	;7	
	DB	37h	;7	
	DB	37h	;7	
	DB	37h	;7	
	DB	37h	;7	
	DB	38h	;8	80
	DB	38h	;8	
	DB	38h	;8	
	DB	38h	;8	
	DB	38h	;8	
	DB	38h	;8	
	DB	38h	;8	
	DB	38h	;8	
	DB	38h	;8	
	DB	38h	;8	
	DB	39h	;9	90
	DB	39h	;9	
	DB	39h	;9	
	DB	39h	;9	
	DB	39h	;9	
	DB	39h	;9	
	DB	39h	;9	
	DB	39h	;9	
	DB	39h	;9	99
ASCII_REL3:	DB	20h	; LEERZEICHEN	
	DB	20h	; LEERZEICHEN	
	DB	20h	; LEERZEICHEN	
	DB	20h	; LEERZEICHEN	

DB	20h	; LEERZEICHEN	
DB	20h	; LEERZEICHEN	
DB	20h	; LEERZEICHEN	
DB	20h	; LEERZEICHEN	
DB	20h	; LEERZEICHEN	
DB	20h	; LEERZEICHEN	
DB	20h	; LEERZEICHEN	
DB	30h	; 0	10
DB	31h	; 1	
DB	32h	; 2	
DB	33h	; 3	
DB	34h	; 4	
DB	35h	; 5	
DB	36h	; 6	
DB	37h	; 7	
DB	38h	; 8	
DB	39h	; 9	
DB	30h	; 0	20
DB	31h	; 1	
DB	32h	; 2	
DB	33h	; 3	
DB	34h	; 4	
DB	35h	; 5	
DB	36h	; 6	
DB	37h	; 7	
DB	38h	; 8	
DB	39h	; 9	
DB	30h	; 0	30
DB	31h	; 1	
DB	32h	; 2	
DB	33h	; 3	
DB	34h	; 4	
DB	35h	; 5	
DB	36h	; 6	
DB	37h	; 7	
DB	38h	; 8	
DB	39h	; 9	
DB	30h	; 0	40
DB	31h	; 1	
DB	32h	; 2	
DB	33h	; 3	
DB	34h	; 4	
DB	35h	; 5	
DB	36h	; 6	
DB	37h	; 7	
DB	38h	; 8	
DB	39h	; 9	
DB	30h	; 0	50
DB	31h	; 1	
DB	32h	; 2	
DB	33h	; 3	
DB	34h	; 4	
DB	35h	; 5	
DB	36h	; 6	
DB	37h	; 7	
DB	38h	; 8	
DB	39h	; 9	
DB	30h	; 0	60
DB	31h	; 1	
DB	32h	; 2	

DB	33h	; 3	
DB	34h	; 4	
DB	35h	; 5	
DB	36h	; 6	
DB	37h	; 7	
DB	38h	; 8	
DB	39h	; 9	
DB	30h	; 0	70
DB	31h	; 1	
DB	32h	; 2	
DB	33h	; 3	
DB	34h	; 4	
DB	35h	; 5	
DB	36h	; 6	
DB	37h	; 7	
DB	38h	; 8	
DB	39h	; 9	
DB	30h	; 0	80
DB	31h	; 1	
DB	32h	; 2	
DB	33h	; 3	
DB	34h	; 4	
DB	35h	; 5	
DB	36h	; 6	
DB	37h	; 7	
DB	38h	; 8	
DB	39h	; 9	
DB	30h	; 0	90
DB	31h	; 1	
DB	32h	; 2	
DB	33h	; 3	
DB	34h	; 4	
DB	35h	; 5	
DB	36h	; 6	
DB	37h	; 7	
DB	38h	; 8	
DB	39h	; 9	99

END

10.2 Listing des Turbo Pascal Programms

Listing des in Kapitel 4 beschriebenen Programms

```
program sr;

uses crt, dos;

{Programm zur Ansteuerung des NNSR Prototyp}

var
    richtig, x, y: char;
    N: longint;
    pfiff: char;
    aux: file of char;

funktion readchar: Char;                                {Lesen eines Zeichens von der seriellen Schnittstelle}

var
    r: Registers;

begin
    r.AH:=2;
    r.DX:=0;
    Intr($14,r);
    readchar:= Char(r.AL);
end;

procedure Ausgabe;

begin
    {$I-}
    write(aux,y);
    if ioreult<> 0 then
        writeln('Plotterfehler');
    {$I+}
end;

procedure init;                                        {Initialisierung der seriellen Schnittstelle}

var
    r: Registers;

begin
    r.AH:= $00;
    r.AL:= $BB;                                        {2400 baud, 8 Datenbits, 1Stoppbit, Parität even}
    r.DX:= $00;
    intr($14,r);
end;
```

```

procedure lesen (var richtig:char);

var
    Wert: char;

begin
    {$ I-}
    for N:=0 to 30 do
        begin
            wert:= readchar;
            if N=24 then
                begin
                    richtig:= wert;
                    case wert of
                        #49: writeln ('Richtig');
                        #50: writeln ('Falsch');
                        #51: writeln ('Telefon');
                        #52: writeln ('Sprechanlage');
                        #53: writeln ('Licht');
                        #54: writeln ('Fernseher');
                        #55: writeln ('Radio');
                        #56: writeln ('Eingabe');
                        #57: writeln ('Ausschalten');
                    end
                end
            end;
        end;
    writeln;
    {$I+}
end;

begin
    {$ Hauptprogramm}
    writeln('Programm starten J')
    readln (pfiff);
    if pfiff = 'J' then
        repeat
            sound (440);
            delay (200);
            nosound;
            repeat
                haupt;
                writeln ('Ist der Befehl richtig ?');
                writeln ('Richtig/Falsch');
                haupt;
                writeln ('Wert: ', richtig ?);
            until richtig = #49;
            writeln ('Beenden J/N');
            readln (pfiff);
            until pfiff = 'J';
        end;
    end.

```