Handbuch Teil 1

R O B O T R O N

Mikrorechnerbausatz Z 1 0 1 3

Handbuch Teil 1

Inhaltsverzeichnis

Maschinenbefehle und ihre Bedeutung 4.3.1. Ladebefehle 4.3.2. Byte- und Doppelbyte-Zaehl-Befehle 4.3.3. Arithmetische Befehle 4.3.4. Vergleichsbefehle 4.3.5. Logische Befehle 4.3.6. Spezielle arithmetische Hilfsoperationen 4.3.7. Befehle zur Bitmanipulation 4.3.8. Verschiebebefehle 4.3.9. Sprungbefehle 4.3.10. Kelleroperationen 4.3.11. Unterprogrammoperationen 4.3.12. Ein- und Ausgabebefehle 4.3.13. Gruppenoperationen fuer Lade-, Vergleichs- und Ein-/ Ausgabe-Befehle 4.3.14. Austauschbefehle 4.3.15. CPU-Steuerbefehle 4.3.16. Bedeutung der Flags 4.4. Unterbrechungsorganisation

Bestandteile des Handbuches:

Handbuch Teil I Handbuch Teil II Anlagenteil

2. Grundbegriffe der Mikrorechentechnik

2.1. Hardware oder Software?

Dieses Kapitel ist vor allem fuer den Leser gedacht, der in der Mikrorechentechnik nicht bewandert ist. Es werden hier einige Grundbegiffe erlaeutert, die das Verstaendnis der nachfolgenden Kapitel erleichtern sollen.

Der erste Begiff, der zu klaeren waere, ist der des Mikrorechners. Ein Mikrorechner ist ein komplexes System verschiedener Funktionseinheiten auf der Basis mikroelektronischer Schalt- kreise, die auf bestimmte Art miteinander in Verbindung treten und durch ihr Gesamtverhalten eine vorgegebene Aufgabe (Programm) loesen. Ein Programm stellt dabei eine Folge von Anweisungen (Befehlen) dar.

Gekennzeichnet wird ein Mikrorechner im wesentlichen durch seine Hard- und Software. Unter Hardware wird dabei sowohl die Gesamtheit der mechanischen und elektronischen Bauelemente, wie integrierte Schaltkreise, Transistoren, Widerstaende usw., als auch die Art und Weise der Verschaltung dieser Bauelemente verstanden.

Als Software eines Rechners werden seine Programme, z. B. Betriebsprogramm und BASIC-Interpreter, bezeichnet. Das Betriebsprogramm (oder auch Betriebssystem) enthaelt die Pro- gramme, die die Zusammenarbeit der einzelnen Systemkomponenten organisieren bzw. ueberhaupt ermeeglichen. Worin unter- scheidet sich aber nun ein Mikrorechner von einer herkoemlichen Schaltung?

Um eine bestimmte Steuerungsaufgabe loesen zu koennen oder immer wiederkehrende Berechnungen zu realisieren, muss nicht immer ein Mikroprozessor verwendet werden. Vielfach ist es einfacher, eine Schaltung mit einfachen Logikschaltkreisen aufzubauen. Eine solche Schaltung haette ueberdies den Vorteil, schneller als ein Mikroprozessor zu arbeiten. Aber be- reits einfache Aenderungen der Aufgabenstellung wuerden einen neuen Schaltungsentwurf erfordern, der mit einem bestimmten Arbeitsaufwand realisiert werden musste. Komplexere Aufgabenstellungen liessen sich auf diese Art ueberhaupt nicht realisieren, da der Aufwand zu hoch werden koennte. Die Loesung einer Aufgabe mit Hilfe eines Mikrorechners ist weitaus einfacher. Der Mikroprozessor ist in der Lage, alle Verknuep- fungsmoeglichkeiten der Logikschaltkreise nachzubilden und damit jedes gewuenschte Verhalten zu realisieren. Die uebrigen Funktionseinheiten des Mikrorechnere enthalten dann in den

Speichereinheiten den Loesungsablauf der Aufgabe in Form von Anweisungen fuer den Mikroprozessor, die Ausgangsdaten sowie konstante Werte. Ueber andere Funktionseinheiten werden Signale aufgenommen sowie Steuersignale wieder abgegeben. Die erreichbare Arbeitsgeschwindigkeit ist kleiner als bei reinen Logikschaltungen. Da aber nicht die maximal erreichbare Arbeitsgeschwindigkeit, sondern die fuer den jeweiligen Prozess oder die Steuerung benoetigte Geschwindigkeit entscheidend ist, ist dieser Nachteil nur in wenigen Faellen von Bedeutung.

Eine Aenderung der Aufgabenstellung fuehrt meist nur zu einer Aenderung der Anweisungen fuer den Mikroprozessor. Diese Aenderung ist schnell realisierbar. Mit dem Mikroprozessor lassen sich ohne technische Veraenderungen vielerlei Aufgabenstellungen besen, es ist meist nur erforderlich, andere Anwei- sungen zu erarbeiten.

2.2. Bestandteile eines Mikrorechners

2.2.1. Zentrale Verarbeitunseinheit

Die Zentrale Verarbeitungseinheit, im Englischen als „Central process unit“ (CPU) bezeichnet, ist der wichtigste Bestandteil eines Mikrorechners. Eine solche CPU liesse sich aus dis- kreten Elementen, d. h. Transistoren, Widerstaenden und Kondensatoren aufbauen, wuerde aber einen sehr grossen Aufwand erfordern. Mit der Entwicklung der Mikroelektronik konnte diese Funktionseinheit in Form einer integrierten Schaltung als sogenannter Mikroprozessor bereitgestellt werden und damit zu einer wesentlichen Vereinfachung im Schaltungsentwurf beitragen. Am Beispiel des Mikroprozessors U880, der im MRB Z1013 Verwendung findet, sollen einige wichtige Bestandteile erlaeutert werden.

Dazu gehoeren:

- **CPU-Steuerung/Befehlsdekodierung**
  Hier werden anhand eines vorgegebenen Befehls bestimmte Signale erzeugt. Bestimmte Zustaende, die von der CPU-Steuerung erkannt werden, sowie der zugefuehrte Takt erzeugen zeitlich festgelegte Signalfolgen, die sowohl den Ablauf innerhalb der CPU steuern, die aber auch als Steuersignale in allenangeschlossenen Funktionseinheiten ausgewertet werden koennen und die gesamten Ablaeufe eines Mikrorechners koordinieren (siehe Zeitdiagramme Anlage 10).
- **Arithmetisch-logische Einheit (ALU)**
  In der ALU koennen Daten entsprechend eines Befehle verknuepft werden. Zu diesen Operationen mit Daten gehoeren: Addition, Subtraktion, UND-Verknuepfung (Konjunktion), ODER- Verknuepfung (Disjunktion) sowie eine Reihe weiterer Operationen wie Verschiebungen und Bitmanipulationen. Eine Veraenderung der Daten ist nur in der ALU moeglich, erforderlichenfalls muessen diese erst in die ALU geholt und danach zuruecktransportiert werden.
- **Registersatz (Zwischenspeicher)**
  In der CPU existieren Zwischenspeicher, die als Register bezeichnet werden. Hier koennen Zwischenergebnisse aufbewahrt und in der ALU miteinander verknuepft werden. Einige Register besitzen spezielle Bedeutung, wie z. B. der sogenannte Kellerzeiger (Stackpointer SP), Befehlszaehler (PC), Refreshregister und Interruptregister (s. auch Abschn. 4.). Bestimmte Register sind doppelt vorhanden und koennen durch einen Befehl umgeschaltet werden. Ein Register wird benutzt, um den Zustand der CPU waehrend der Befehlsabarbeitung zu speichern. Es wird als Flag-Register bezeichnet (die Bezeichnung „Flag“ sollte als Anzeiger verstanden

werden). In einem Register wird der gelesene Befehl zwischengespeichert, bis die durch ihn veranlasste Operation beendet ist. Dieses Register heisst demzufolge Befehlsregister.

Die Arbeit der CPU wird durch eine Reihe von Systemsignalen gekennzeichnet, die als Anschluesse herausgefuehrt wurden und das Zusammenwirken mit den angeschlossenen Funktionseinheiten steuern.

2.2.2. Speicher

In der Mikrorechentechnik haben sich zur Speicherung von Informationen Halbleiterspeicher weitgehend durchgesetzt. Es sind integrierte Schaltungen in unterschiedlichen Gehaeusegroessen, je nach Kapazitaet des Speichers. Speicher werden zu verschiedenen Zwecken benoetigt, z. B. um der CPU die abzuarbeitenden Befehle zur Verfuegung zu stellen. Da die Register der CPU meist nicht ausreichen, alle Zwischenergebnisse aufzubewahren, muessen diese ebenfalls in den Speicher gebracht werden.

Als Modell eines Speichers mag ein langer Schrank mit vielen Faechern dienen. Diese Faecher sind einzeln numeriert. Diese Numerierung soll bei Null beginnen und lueckenlos bis zu einem Endwert erfolgen. Jedes Fach entspricht einem Speicherplatz und kann eine Information enthalten. Die maximale Anzahlnder Faecher bestimmt die Kapazitaet dieses Speichers.

Die Zeit, die vom Anlegen einer Speicherplatzadresse bis zur Bereitstellung der gespeicherten Daten beneetigt wird, wird Zugriffszeit genannt.

Zwischen der Speicherung von Daten und Programmen bestehen einige Unterschiede. Programme werden meist in Speichern aufbewahrt, die auch nach Abschalten der Versorgungsspannung ihren Inhalt behalten. Allerdings koennen diese Speicher nur gelesen werden, zum Beschreiben dieser Speicher sind spezielle Einrichtungen notwendig.

2.2.2.1. Programmspeicher (Nur-Lese-Speicher)

In einem Programmspeicher sind die Anweisungen fuer einen Mikroprozessor enthalten. Diese Anweisungen gehen auch nach Ausschalten des Rechners nicht verloren, sie sind nicht fluechtig. Diese Speicher bezeichnet man als Nur-Lese-Speicher (Read only memory - ROM), die Informationen werden einmal eingegeben und stehen staendig zur Verfuegung. Je nach Eingabe der Information unterscheidet man: ROM's, die bereits waehrend der Herstellung ihre Informationen erhalten und ROM's, die nachtraeglich elektrisch programmiert werden koennen (ein einmaliger Vorgang, da die Struktur des Speichers veraendert wird). Diese letztgenannten Speicher heissen PROM (Prograinmable ROM). Eine weitere Speicherart kann sowohl programmiert als auch wieder geleescht werden. Das Loeschen erfolgt mit ultraviolettem Licht (UV-Licht) und loescht immer den gesamten Speicher. Diese Speicherart nennt man EPROM (Erasable PROM). Das Einschreiben der Programme in den EPROM geschieht mit speziellen Funktionseinheiten, sogenannten EPROM- Programmiergeraeten. In den EPROM wird die zu speichernde Information mittels einer Programmierspannung als Ladungsmenge eingegeben. Nach Erreichen einer vorgegebenen Ladung ist der Baustein programmiert. Die Bestrahlung mit UV-Licht hat zur Folge, dass die gespeicherte Ladungemenge wieder abgebaut wird. Nach dem Loeschen ist der EPROM wieder programmierbar.

2.2.2.2. Datenspeicher (Schreib-Lese-Speicher)

Zur Aufbewahrung von Zwischenergebnissen oder anderen veraenderbaren Informationen werden Schreib-Lese-Speicher verwendet. Da diese Speicher wahlweise gelesen oder beschrieben werden koennen, nennt man sie Speicher mit wahlfreiem Zugriff (Random access memory - RAM). Mit Abschalten der Stromversorgung verlieren RAM's ihren Inhalt, sie sind also nicht zur Aufbewah- rung von Informationen verwendbar, die immer verfuegbar sein muessen. Es werden zwei grundsaetzliche Typen unterschieden: statische und dynamische RAM's.

In den statischen RAM's werden Transistorkombinationen zur Aufbewahrung der Informationen verwendet. Eine solche Transistorkombination kann zwei verschiedene Zustaende annehmen und behaelt eine somit eingetragene Information bis zum Abschalten oder Ueberschreiben mit einer neuen Information.

Dynamische RAM's speichern die, Information als Ladung eines kleinen Kondensators ab. Diese Ladung muss, auf Grund der Selbstentladung, periodisch erneuert werden, dieser Vorgang wird mit REFRESH (Auffrischen) bezeichnet. Das Auffrischen wird bereits erreicht, wenn der Speicher gelesen wird. Die CPU U880 unterstuetzt diesen Vorgang durch Aussenden einer REFRESH-Information, um die zeitlichen Bedingungen zum Auffrischen unter allen Umstaenden zu gewaehrleisten. Werden die Zellen der dynamischen RAM's nicht spaetestens nach 2 Millisekunden aufgefrischt, geht ihre gespeicherte Information verloren.

Trotz des nicht unerheblichen Mehraufwandes werden dynamische RAM's verwendet, da sie bei gleichen Abmessunngen der Bausteine eine groessere Speicherkapazitaet und kleinere Leistungsaufnahme gegenueber statischen RAM's aufweisen.

2.2.3. Ein/Ausgabe-Einheiten

Unter externen Geraeten sollen im folgenden alle Geraete verstanden werden, mit denen Informationen in den Mikrorechner eingegeben oder vom Mikrorechner ausgegeben werden. Damit ist es moeglich, sowohl Daten als auch Programme in den Mikrorechner zu bringen und die Ergebnisse fuer den Nutzer sichtbar zu machen. Solche Geraete sind Lochbandleser und -stanzer, Magnetbandtechnik, Tastaturen, Bildschirm usw.

Die Verbindung dieser Geraete mit dem Mikroprozessor erfolgt ueber sogenannte E/A-Funktionseinheit en, in denen spezielle integrierte Schaltungen enthalten sind. Diese Funktionseinheiten steuern selbstaendig die Arbeit der Geraete und treten mit der CPU nur zur Informationsuebermittlung in Kontakt. Damit wird die CPU entlastet und die Programmabarbeitung wesentlich effektiver.

Weiterhin koennen auch Funktionseinheiten angeschlossen werden, die beliebig zur Verfuegung gestellte Meldesignale aus zu ueberwachenden Prozessen aufnehmen und sie fuer den Mikroprozessor aufbereiten. Gleichermassen ist die Abgabe von Steuersignalen zur Beeinflussung bestimmter zu steuernder Prozesse moeglich. Als integrierte Schaltkreise werden dazu im MRB Z1013 parallele E/A-Schaltkreise (Parallel Input OutputPIO) vom Typ U855 verwendet.

2.2.4. Verbindung der Funktionseinheiten

Der Mikroprozessor (CPU) sendet Signale ab und wertet bestimmte empfangene Signale aus. Diese Signale werden i. a. in allen angeschlossenen Funktionseinheiten benoetigt.

Die Leitungen zur Uebermittlung von Daten, Adressen und Systemsignalen, wie z. B. LESEN, SCHREIBEN, werden entsprechend ihrer Funktion zu Leitungsbuendeln zusammengefasst.

Da diese Leitungen die Daten und Informationen zwischen den einzelnen Funktionseinheiten transportieren, wurde der Begriff „Bus“ fuer ein solches Leitungsbuendel gepraegt.

Demzufolge bezeichnet man die Datenleitungen als DATENBUS, die Adressleitungen werden als ADRESSBUS und die Systemleitungen als STEUERBUS bezeichnet. Gelegentlich steht der Begiff „SYSTEMBUS“ auch fuer alle Leitungen innerhalb des Mikrorechnersystems.

Durch Verwendung eines einheitlichen Systembusses ist es moeglich, beliebige Funktionseinheiten einem bestehenden System hinzuzufuegen, d. h. das System staendig zu erweitern. Voraussetzung ist die Uebereinstimmung der elektrischen Anschluesse der jeweiligen Einheiten.

2.3. Programmabarbeitung

2.3.1. Ablauf in der CPU

Wie bereits gesagt, beneetigt die CPU zur Lossung ihrer Aufgaben Anweisungen, die den gesamten Ablauf des Mikrorechners steuern. Diese Anweisungen oder Befehle findet die CPU in den angescohlossenen Speichereinheiten in einer ganz bestimmten, fuer sie verstaendlichen Form, die als Maschinenkode bzw. MC bezeichnet wird. Alle Anweisungen an die CPU muessen also in Form dieses MC vorliegen bzw. sind in diese Form zu bringen. Im Anhang befindet sich eine Uebersicht, in der die Befehle des U880 sowie deren Darstellung im Maschinenkode enthalten sind.

Um eine bestimmte Anweisungsfolge abzuarbeiten, ist es notwendig, der CPU mitzuteilen, in welchem Speicherbereich diese Befehle zu finden sind. Die CPU liest in diesem Bereich den Speicher und versucht die gelesenen Informationen als Befehl auszufuehren. Dazu werden die gelesenen Informationen ins Befehlsregister transportiert und steuern von hier den Ablauf in der CPU. In Abhaengigkeit vom konkreten Befehl werden entweder zusaetzliche Informationen aus dem Speicher gelesen, werden Daten zum oder vom Speicher transportiert oder bestimmte logische Verknuepfungen in der ALU vorgenommen. Alle diese Aktivitaeten der CPU sind mit dem Aussenden bestimmter Steuersignale verbunden, die die jeweilige Art der Operation anzeigen.

Einen Ueberblick ueber die Steuersignale bei einigen ausgeaehlten Operationen gibt Anlage 10. Zwischenzeitlich waehrend der Befehlsverarbeitung sendet die CPU mit Hilfe des REFRESH-Registers eine Information zum Auffrischen des Speicherinhaltes eventuell angeschlossener dynamischer Speicher aus. Waehrend des Refresh-Zyklus wird der Befehl ausgefuehrt.

War der eben abgearbeitete Befehl ein Verarbeitungs- oder Transportbefehl, dann wird die Verarbeitung mit dem im Speicher folgenden Befehl fortgesetzt. Einige Befehle veraendern aber diese Abarbeitungsreihenfolge, sie teilen der CPU mit, in welchem Speicherbereich der naechste Befehl zu finden ist.

Nach erfolgter Programmabarbeitung kann die CPU anhalten oder ein Steuerprogramm bearbeiten, mit dem z. B. die naechste Aufgabe ausgewaehlt werden kann.

2.3.2. Holen der Befehle

Fuer die Organisation der Befehlsabarbeitung besitzt die CPU ein besonderes Register, den Befehlszaehler (PC). Der Befehlszaehler besitzt 16 Bitstellen entsprechend der Anzahl der Adressleitungen. Beim Betaetigen der RESET-Taste wird dieses Register auf Null gesetzt. Damit liest die CPU den ersten abzuarbeitenden Befehl auf dem Platz Null.

Aus diesem Grund muss ein Programm ab dieser Stelle beginnen. Um ein solches Programm ab Null nach dem Einschalten zur Verfuegung zu stellen, ist ein nicht fluechtiger Speicher, z. B. ein EPROM notwendig. Befindet sich in diesem Speicherbereich kein Programmspeicher, so findet die CPU zufaellige Bitkombinationen, die als Befehl aufgefasst und abgearbeitet werden.

Es kann also immer nur ein Programm nach dem Einschalten gestartet werden. Das wird im Normalfall ein Steuerprogramm sein, mit dem andere Programme aktiviert werden koennen. Soll ein anderes Steuerprogramm verwendet werden, ist der entsprechende Programmspeicher auszuwechseln. Da Programme auch in den Schreib-Lese-Speicher (RAM) geladen werden koennen, kann der Speicherbereich ab Null als RAM ausgelegt werden. Dann muss aber durch die Hardwareschaltung das Erreichen eines Steuerprogramms sichergestellt werden, welches in einem beliebigen Speicherbereich stehen kann. Es koennen nun beliebige Betriebsprogramme in den Bereich ab Null geladen und verwendet werden, ohne jedesmal den Speicher auswechseln zu muessen. Damit ist ein solches System jeder Aufgabenstellung anpassbar.

Der Bereich ab Null ist noch aus einem anderen Grunde besonders fuer Betriebsprogramme geeignet. Er enthaelt einige ausgewaehlte Adressen, die sowohl von Programmen (sogenannte RESTART-Befehle) als auch im Resultat von externen Ereignissen (sogenannten Programmunterbrechungen) benoetigt werden.

Das Lesen der Befehle oder auch anderer Informationen geschieht durch Aussenden einer Adresse, begleitet von bestimmten Steuersignalen.

Durch eine Speicherverwaltung werden aus bestimmten Stellen dieser Adresse die Auswahl der entsprechenden Speichereinheit sowie eines Speicherbereiches vorgenommen. Der niederwertige Teil der Adresse wird verwendet, um in dem betreffenden Speicherbereich den konkreten Platz zu adressieren.

War die dort vorgefundene Information ein Befehl fuer die CPU so wird automatisch der Befehlszaehler entsprechend der Befehlslaenge erhoeht (inkrementiert) und damit die neue Befehls-adresse bereitgestellt. Wurde der Befehl als ein Verzweigungsbefehl erkannt, wird im Befehlszaehler die neue Adresse bereitgestellt und dann erneut durch Aussenden dieser Adresse ein bestimmter Speicherplatz ausgewaehlt.

2.3.3. Die Darstellung von Informationen im Speicher

Bisher wurde immer nur allgemein von „Informationen“ gesprochen, die in einem „Speicher“ zu finden sind. Diese Informa- tionen waren sowohl Daten als auch Befehle, die in unterschiedlichen Speichertypen aufbewahrt wurden (ROM bzw. EPROM oder RAM).

Hinsichtlich ihrer Darstellung im Speicher unterscheiden sich diese Informationen auch nicht; es waere auch meeglich, Daten als Befehle zu betrachten und umgekehrt. Bei einer Abarbeitung durch die CPU kommen dabei selten sinnvolle Ergebnisse zustande.

Es wurde bereits der Speicher mit einer endlichen Anzahl Faecher eines Schranken verglichen, in denen Informationen abgelegt werden koennen. Durch eine Adresse wird die Nummer eines konkreten Faecher bereitgestellt.

Wenn Informationen sowohl gelesen als auch abgelegt werden koennen, entspricht das dem Prinzip des Schreib-Lese-Speichers. Als Information kann das Vorhandensein eines Zeichens, einer Markierung oder dergleichen gedeutet werden. Ist diese Markierung dauerhaft (eingraviert), so handelt es sich um einen Nur-Lese-Speicher. In einem Kanten koennen auch mehrere Informationen enthalten sein. Analog dazu sind die Speicher im Mikrorechner aufzufassen.

Eine Funktionseinheit „Speicher“ besteht aus einer bestimmten Anzahl adressierbarer Plaetze, wobei jeder Platz zwei verschiedene Zustaende annehmen kann. Diese beiden Zustaende werden durch die Dualziffern „0“ und „1“ repraesentiert. Die Auswahl dieser Plaetze erfolgt ueber sogenannte Adressleitungen, die Anzahl der Leitungen richtet sich nach der Kapazitaet des Speichers. Der einfachste Aufwand ergibt sich bei der Festlegung der Speicherkapazitaet, d. h. der Anzahl der adressierbaren Speicherplaetze, als ein Vielfaches einer Potenz zur Basis 2.

Eine Adressleitung kann zwei Zustaende annehmen, entweder hohen Spannungspegel, sogenannten „H-Pegel“ (logisch „1“), als auch niedrigen Spannungspegel, sogenannten „L-Pegel“ (logisch „0“). Damit waeren zwei verschiedene Speicherplaetze adressierbar. Zwei Adressleitungen koennen zusammen bereits 4 Zustaende annehmen, damit sind 4 verschiedene Speicherplaetze (mit den Adressen 00, 01, 10 und 11) adressierbar. Demzufolge werden bei 10 Adressleitungen 2 hoch 10=1024=1K Speicherplaetze adressiert.

Damit ergibt sich:

```
     Kapazitaet = 2 hoch n (n = Anzahl der Adressleitungen)
```

Die CPU U880 hat 16 Leitungen fuer die Bildung von Adressen zur Verfuegung, d. h. sie kann maximal 2 hoch 16 = 65536 ^= 64K Speicherplaetze adressieren.

Eine weitere Eigenschaft einer Speichereinheit ist die Aufrufbreite. Hierunter wird verstanden, wieviel Speicherplaetze gleichzeitig mit einer Adresse angesprochen werden koennen, um die Information parallel zu verarbeiten. Diese Aufrufbreite ist verschieden: bei ROM's und EPROM's betraegt sie 8 Stellen, bei statischen RAM's 1, 4 oder 8 Stellen und bei dynamischen RAM's im allgemeinen eine Stelle. Um die moegliche Verarbeitungsbreite des Mikroprozessors U880 mit 8 Datenleitungen zu nutzen, muessen in einer Speichereinheit mehrere Speicherschaltkreise kombiniert werden, um damit die gewuenschte Aufrufbreite zu realisieren. Das geschieht, indem die Adressanschluesse von acht Speicherschaltkreisen mit den jeweiligen Adressleitungen der CPU verbunden werden. Jeder der Speicherschaltkreise wird an einer Datenleitung angeschlossen, die Auswahl erfolgt fuer alle acht Schaltkreise mit einem gemeinsamen Auswahlsignal.

2.4. Grundbegriffe der Software

2.4.1. Darstellung von Zahlen

Das Wesentliche bei der Programmabarbeitang besteht in der Veraenderung der eingegebenen Zahlen, um die gewuenschten Ergebnisse zu erhalten. Das gewohnte Dezimalsystem ist fuer die

Zahlendarstellung im Mikrorechner nicht geeignet; die zwei moeglichen Zustaende fuehren auf ein anderes Zahlensystem, das sogenannte Dualsystem. Dieses kennt nur die Ziffern 0 und 1, welche mit dem „L“- und „H“-Pegel der Informationsspeicherung identisch sind.

Da aber die Bildung von Zahlen sowohl im Dezimalsystem als auch im Dualsystem nach gleichen Gesetzmaessigkeiten verlaeuft, ist eine Umrechnung unproblematisch und kann durch entsprechende Programme vom Mikroprozessor vorgenommen werden.

Diese Zahlenbildung kann mit folgender Gleichung beschrieben werden:

```
                n        n-1           1        0
      Z = d * x  + d * x   +...+ d * x  + d * x
```

wobei bedeuten:

- Z = Zahlenwert
- d = Ziffern innerhalb des Wertebereichs im Zahlensystem
- x = Basis des Zahlensystems
- n = ganzzahliger Exponent

Im Dezimalsystem kann d die Ziffern 0 … 9 annehmen, x ist dann gleich 10. Im Dualsystem ist d entweder 0 oder 1, die Basis ist gleich 2.

Ein Beispiel soll das verdeutlichen.

```
                2        1        0
    123 = 1 x 10 + 2 x 10 + 3 x 10          (dezimal)
              6       5       4       3       2
        = 1 x 2 + 1 x 2 + 1 x 2 + 1 x 2 + 0 x 2 +
              1       0
          1 x 2 + 1 x 2
        = 11111011B                         (dual)
```

Das „B“ hinter der Dualzahl soll zur Unterscheidung zur Dezimalzahl, die ohne Kennzeichnung geschrieben wird, dienen. „B“ bedeutet „binaer“, abgeleitet von den zwei Zustaenden. Nun waere eine solche Umrechnung per Hand kompliziert. Es gibt jedoch ein einfaches Umrechnungsverfahren, das am deutlichsten durch ein Beispiel wird.

```
123 : 2 = 61                Rest: 1----------------------
   -------|                                             |
   |                                                    |
   V                                                    |
 61 : 2 = 30                      1-------------------- |
   -------|                                           | |
   |                                                 | |
   V                                                 | |
 30 : 2 = 15                      1------------------ | |
   -------|                                        | | |
   |                                              | | |
   V                                              | | |
```

```
 15 : 2 =  7                    1--------------- | | |
   -------|                                     | | | |
    |                                          | | | |
    V                                          | | | |
  7 : 2 =  3                    1------------- | | | |
   -------|                                    | | | | |
    |                                         | | | | |
    V                                         | | | | |
  3 : 2 =  1                    1------------ | | | | | |
   -------|                                   | | | | | |
    |                                        | | | | | |
    V                                        | | | | | |
  1 : 2 =  0                    1--------- | | | | | | |
                                           | | | | | | |
                                          V V V V V V V V
                                Binaer:   1 1 1 1 0 1 1  B
                                          ==============
```

Die Speichereinheiten in U880 Systemen, wie dem Z1013, besitzen in der Regel eine Aufrufbreite von 8 Bit. Das heisst, auf einem Speicherplatz sind gleichzeitig 8 Bit, die zu einem Byte zusammengefasst werden, gespeichert. Ein Byte kann demzufolge 2 hoch 8 = 256 verschiedene Werte annehmen.

Fuer ein Byte ergeben sich die folgenden Wertigkeiten fuer die einzelnen Bits:

Byte:

```
                                          Wertigkeit oder
  +---------------------------------------+  Exponent zur
  |   7 | 6  |  5 |  4 |  3 |  2 |  1  |  0 |  Basis 2
  +---------------------------------------+
  | 128 | 64 | 32 | 16 |  8 |  4 |  2  |  1 |  Zahlenwert
  +---------------------------------------+
  |   hoeherwertiges   |   niederwertiges   |  Halbbyte
  +---------------------------------------+  (BCD-Ziffer)
```

Die in der Bytedarstellung eingetragenen Ziffern geben die Numerierung der einzelnen Bit's an. Das Bit 0 besitzt die niedrigste Wertigkeit, das Bit 7 die hoechste.

Eine vorzeichenlose ganze Zahl mit der Bitfolge 01001010B kann auch in der Form:

```
Z = 0*128 + 1*64 + 0*32 + 0*16 + 1*8 + 0*4 + 1*2 + 0*1 = 74
```

geschrieben werden.

Sollen auch negative Zahlen dargestellt werden, besitzt das Bit 7 die Funktion des Vorzeichens.

Eine Dualzahl 10110110B kann als

```
Z = -1*128 + 0*64 +1*32 + 1*16 + 0*8 + 1*4 + 1*2 + 0*1 = - 74
```

aufgefasst werden, diese Art bezeichnet man als Zweierkomplement. Damit ergibt sich ein Zahlenbereich fuer ganze vorzeichenlose Zahlen von 0 bis 255 und fuer vorzeichenbehaftete Zahlen von -128 ueber 0 bis +127.

Sollen greessere Zahlen dargestellt werden, muessen 2 und mehr Byte dafuer genutzt werden. Die Zusammenfassung von 2 Byte wird als Wort bezeichnet, analog dazu 4 Byte als Doppelwort.

Die einzelnen Byte des Maschinenkodes werden als Dualzahlen, d. h. als Ziffemfolgen von „0“ oder „1“ dargestellt. Insbesondere bei grossen Programmen ergibt sich damit ein sehr grosser Schreibaufwand, um diese Dualzahlen zu notieren. Deshalb hat sich ein anderes Zahlensysten, das sogenannte Hexadezimalsystem fuer die Darstellung von Zahlen und Programmen bei Mikrorechnern durchgesetzt. (Die Bezeichnung Hexadezimalsystem ist umgangssprachlich, exakt heisst es Sedezimalsystem.) Im Hexadezimalsystem werden 4 benachbarte Dualziffern zusammengefasst und durch eine Hexadezimalziffer dargestellt. Mit vier Dualziffern koennen 16 verschiedene Zustaende dargestellt werden. Die Zahlen „0“ bis „9“ sind gleich den Dezimalzahlen, groesser als „9“ werden die ersten Buchstaben des Alphabets verwendet. Die folgende Tabelle enthaelt eine Gegenueberstellung von Dual-, Dezimal- und Hexadezimalziffern.

```
 DUAL          DEZ   HEX      |     DUAL          DEZ   HEX
-------------------------------+---------------------------
                               |
 0 0 0 0        0     0        |     1 0 0 0        8    8
 0 0 0 1        1     1        |     1 0 0 1        9    9
 0 0 1 0        2     2        |     1 0 1 0       10    A
 0 0 1 1        3     3        |     1 0 1 1       11    B
 0 1 0 0        4     4        |     1 1 0 0       12    C
 0 1 0 1        5     5        |     1 1 0 1       13    D
 0 1 1 0        6     6        |     1 1 1 0       14    E
 0 1 1 1        7     7        |     1 1 1 1       15    F
```

Da in einem Byte (mit 8 Bit) zwei sogenannte Halbbyte zu je 4 Bit enthalten sind, kann ein Byte mit 2 Hexadezimalziffern dargestellt werden. Die binaere Darstellung der Dezimaizahlen von 0 bis 9 nennt man auch BCD-Zahlen, Auch mit dieser Zahlendarstellung kann gerechnet werden. Dabei muss aber eine Dezimalkorrektur vorgenommen werden. Warum und wie, wird bei der Erlaeuterung des DAA-Befehls im Befehlssatz genauer erklaert. Zur besseren Unterscheidung zu den Dezimalzahlen verden die Hexadezimalzahlen in Protokollen oder Drucklisten durch ein nachgestelltes Zeichen „H“ gekennzeichnet.

Nehmen wir z. B. ein Byte in Binaerdarstellung:

```
    0111   1011B   =  7BH   = 123
     1.     2.     Halbbyte
```

Dabei sind die Wertigkeiten der einzelnen Bits in einem Halbbyte:

```
    3 2 1 0 Wertigkeit
    -------
    8 4 2 1 Zahlenwert
```

Die Umwandlung einer Hexadezimalzahl in die entsprechende Dezimalzahl geschieht am einfachsten auf folgende Weise:

```
              1      0
    7BH  = 7x16 + Bx16
              1       0
         = 7x16 + 11x16
         = 123
```

Die Umrechnung Dezimal- in Hexadezimalzahl erfolgt nach einem analogen Schema wie die Umrechnung Dezimal- in Dualzahl, z. B.

```
                                      Dez.   Hex.
 45 346  :  16  =  2 834      Rest     2      2 -----------
  2 834  :  16  =    177              2      2 --------- |
    177  :  16  =     11              1      1 ------- | |
     11  :  16  =     11             11      B ----- | | |
                                                 | | | |
                                                 V V V V
                              Hex.-Zahl:         B 1 2 2  H
                                                 =======
```

Um den Vorteil dieser Schreibweise deutlich werden zu lassen, hier zum Vergleich diese Zahl in Binaerdarstellung:

```
    1011 0001 0010 OO1OB.
```

2.4.2. Logische Operationen

Mit den Dualzahlen lassen sich verschiedene logische Operationen durchfuehren. Bei den logischen Verknuepfungen werden die Dualzahlen als vorzeichenlose, ganze Zahlen aufgefasst.

Die wichtigsten dieser Operationen sind:

- **Komplementbildung (NEGATION)**:
  Eine Dualzahl wird in ihr Komplement ueberfuehrt, indem alle Bitstellen einzeln auf den entgegengesetzten Wert gebracht werden.

  ```
  Zahl:         0 1 0 0 1  0 1 0
  --------------------------------
  Ergebnis:     1 0 1 1 0  1 0 1
  ```

  Diese Operation wird nur mit einer Dualzahl durchgefuehrt.
  In Stromlaufplaenen finden Sie dafuer das folgende Sinnbild:

  - **UND-Verknuepfung (KONJUNKION, AND)**
    Eine Konjunktion wird mit zwei Dualzahlen durchgefuehrt. Dabei bleibt nur in der Bitposition eine „1“ stehen, in welcher in der ersten und in der zweiten Dualzahl eine „1“ stehen.

    ```
    1. Zahl:     0 1 0 0 1 0 1 0
    ```

```
2. Zahl:     0 0 0 1 1 1 1 1
------------------------------
Ergebnis:    0 0 0 0 1 0 1 0
```

Sinnbild:
Zur besseren Darstellung der logischen Operationen ist es ueblich, sich eine beliebige Bitposition auszuwaehlen und in einer Wertetabelle alle moeglichen Kombinationen und deren Ergebnisse zu erfassen. Die Wertetabelle der Konjunktion besitzt danach folgendes Aussehen (gleiche Bitposition vorausgesetzt):

```
1. Zahl       | 2. Zahl        | Ergebnis
-----------------------------------------
   0          |    0           | 0
   0          |    1           | 0
   1          |    0           | 0
   1          |    l           | 1
```

Besonders bei komplizierten Verknuepfungen stellt die Wertetabelle ein sehr einfaches Hilfsmittel dar.

- **NICHT-UND-Verknuepfung (NAND)**
  Diese Verknuepfung stellt eine Konjunktion mit anschliessender Negation dar.

  ```
  1. Zahl    |  2. Zahl    |  Ergebnis
  -----------------------------------
     0       |     0       |  1
     0       |     1       |  1
     1       |     0       |  1
     1       |     1       |  0
  ```

  Sinnbild:

- **ODER-Verknuepfung (DISJUNKION, OR)**
  Zwei disjunktiv verknuepfte Dualzahlen liefern im Ergebnis eine „1“, wenn in der ersten oder zweiten Dualzahl in der jeweiligen Bitposition eine „1“ steht.

  ```
  1. Zahl    |  2. Zahl    |  Ergebnis
  -----------------------------------
     0       |     0       |  0
     0       |     1       |  1
     1       |     0       |  1
     1       |     1       |  1
  ```

  Sinnbild:

- **NICHT-ODER-Verknuepfung: (NOR)**
  Diese Verknuepfung stellt eine Disjunktion mit anschliessender Negation dar.

  ```
  1. Zahl    |  2. Zahl    |  Ergebnis
  -----------------------------------
  ```

```
    0       |     0       |  1
    0       |     1       |  0
    1       |     0       |  0
    1       |     1       |  0
```

Sinnbild:

- **Exklusiv- ODER bzw. (ANTIVALENZ, EXOR)**
  In der jeweiligen Bitposition der Ergebnisse wird eine „1“ eingetragen, wenn sich in dieser Bitposition die beiden Dualzahlen unterscheiden.

```
1. Zahl    |  2. Zahl    |  Ergebnis
-------------------------------------
   0       |     0       |  0
   0       |     1       |  1
   1       |     0       |  1
   1       |     1       |  0
```

  Sind beide Dualzahlen gleich, wird das Ergebnis auf Null gesetzt. Das wird besonders verwendet, um einen bestimmten Zwischenspeicher, z. B. das A-Register der CPU, zu loeschen, indem der Inhalt des A-Registers mit sich selbst durch einen XOR-Befehl verknuepft wird (X0R A).

2.4.3. Arithmetische Verknuepfungen

Zu den arithmetischen Operationen gehoeren Addition und Subtraktion. Die Multiplikation zweier Dualzahlen kann durch fortlaufende Addition einer Dualzahl bei gleichzeitiger Verringerung der anderen Dualzahl, bis diese Null ist, vorgenommen werden. Auch eine teilweise Addition, kombiniert mit Verschiebung von Ergebnis und Operand ist ueb]ich. Die Division kann analog dazu als eine fortlaufende Subtraktion einer Dualzahl von einer anderen durchgefuehrt werden. Dabei wird der Dividend solange vom Divisor subtrahiert und der Quotient jeweils um 1 erhoeht, bis der Divisor kleiner als der Dividend geworden ist. Der Quotient als Ergebnis enthaelt damit die Anzahl der benoetigten Subtraktionsschritte, im Divisor ist der Rest enthalten.

Nachfolgend die arithmetischen Operationen im einzelnen:

Es empfiehlt sich, die Beispiele mit anderen Zahlen selbst noch einmal nachzuvollziehen.

- **ADDITION:**
  Die Addition zweier Dualzahlen liefert folgendes in der Wertetabelle sichtbare Ergebnis. Dabei wird der Uebertrag in der letzten Spalte in der naechsthoeheren Bitposition ausgewertet.

```
 0   +   0       =   0
 0   +   1       =   1
 1   +   0       =   1
 1   +   1       =   0  Uebertrag 1
```

  Sollen z. B. die Zahlen 26 und 43 miteinander addiert werden, ergibt das folgende Rechnung:

```
26:               0 0 0 1 1 0 1 0
43:               0 0 1 0 1 0 1 1
Uebertraege:          1 1 1   1
---------------------------------

Ergebnis:         0 1 0 0 0 1 0 1  = 69
```

Werden zwei Zahlen addiert, deren Ergebnis den Zahlenbereich ueberschreitet, kommt es zum Üeberlauf, d. h. das ermittelte Ergebnis ist falsch. An der Addition der Zahlen 69 und 73 soll das im Rechenschema gezeigt werden.

```
69:               0 1 0 0 0 1 0 1   Zahlenbereich:
73:               0 1 0 0 1 0 0 1   -128 <= x <= 127
Uebertraege:      1           1
-----------------------------------

Ergebnis:         1 0 0 0 1 1 1 0   = -114
```

Im Ergebnis entsteht die Zahl -114, obwohl die Addition dieser Zahlen zu dem Ergebnis 132 fuehren muesste. Dieser Ueberlauf ist dadurch charakterisiert, dass ein Uebertrag in die Vorzeichenstelle ein-, aber kein Uebertrag aus der Vorzeichenstelle herauslaeuft.

- **SUBTRAKTION**
  Die Subtraktion zweier Dualzahlen verlaeuft aehnlich der der Dezimalzahlen, d. h. wenn die Subtraktion einen negativen Wert in der Bitposition ergibt, muss von der hoeherwertigen Stelle etwas „geborgt“ werden, es entsteht ein Uebertrag.
  Daraus resultiert folgende Wertetabelle:

```
1. Zahl 2. Zahl Ergebnis

   0   -   0       =   0
   0   -   1       =   1   (0-1 => 10-1 => 1+Uebertrag)
   1   -   0       =   1           |
   1   -   1       =   0           '->geborgte 1'
```

Subtrahiert man die Dualzahl 26 von der 43, so ergibt sich folgendes Rechenschema:

```
43:               0 0 1 0 1 0 1 1
26:               0 0 0 1 1 0 1 0
Uebertraege:          1
-----------------------------------
Ergebnis:         0 0 0 1 0 0 0 1 =  17
```

Subtrahiert man die Zahlen in anderer Weise, d. h. die Dualzahl 43 von der 26, so kann man auch einen Vorzeichenwechsel beobachten.

```
26:               0 0 0 1 1 0 1 0
43:               0 0 1 0 1 0 1 1
Uebertraege: 1<=  1 1   1 1 1 1
-----------------------------------
```

```
  Ergebnis:          1 1 1 0 1 1 1 1 = -17
```

Da hier aber ein Uebertrag sowohl in die Vorzeichenstelle hinein als auch ein Uebertrag aus der Vorzeichenstelle heraus erfolgt, handelt es sich um keinen Ueberlauf und das Ergebnis ist korrekt. Dieser herauslaufende Uebertrag wird bei Zahlen im Wort- oder Doppelwortformat weiterverwendet.

- **ZWEIERKOMPLEMENT:**
  Eine Ergebnisdarstellung wie im vorangegangenen Subtraktionsbeispiel wird Zweierkomplement genannt. Jede Zahl kann in ihr Zweierkomplement ueberfuehrt werden, wenn diese Zahl zuerst in ihr Komplement umgewandelt (negiert) wird und anschliessend zur niederwertigsten Bitposition eine „1“ addiert wird.

```
 -17:              1 1 1 0 1 1 1 1
  Negation:         0 0 0 1 0 0 0 0
  Addition:                       1
  ----------------------------------
  Ergebnis:         0 0 0 1 0 0 0 1 = 17
```

Das Zweierkomplement wird verwendet, um eine Subtraktion auf eine Addition zurueckzufuehren.
Die Addition einer Zahl und ihres Zweierkomplements liefert als Ergebnis immer eine Null.

Abschliessend noch ein Beispiel zur Multiplikation, die hier als eine fortlaufende Addition betrachtet werden soll.

Es werden die Dualzahlen 9 und 5 miteinander multipliziert:

```
     Ausgangawerte:  5: 0 0 0 0 0 1 0 1
                     9: 0 0 0 0 1 0 0 1
                                        Multiplika-
                                        tor 5

                     9: 0 0 0 0 1 0 0 1     5
                    +9: 0 0 0 0 1 0 0 1     4
                    --------------------
                     =  0 0 0 1 0 0 1 0     3
                    +9: 0 0 0 0 1 0 0 1
                    --------------------
                     =  0 0 0 1 1 0 1 1     2
                    +9: 0 0 0 0 1 0 0 1
                    --------------------
                     =  0 0 1 0 0 1 0 0     1
                    +9: 0 0 0 0 1 0 0 1
                    --------------------
     Ergebnis:       =  0 0 1 0 1 1 0 1 = 45
```

Die Multiplikation mit teilweiser Addition und Verschiebung kann analog zur Multiplikation von Dezimalzahlen dargestellt werden:

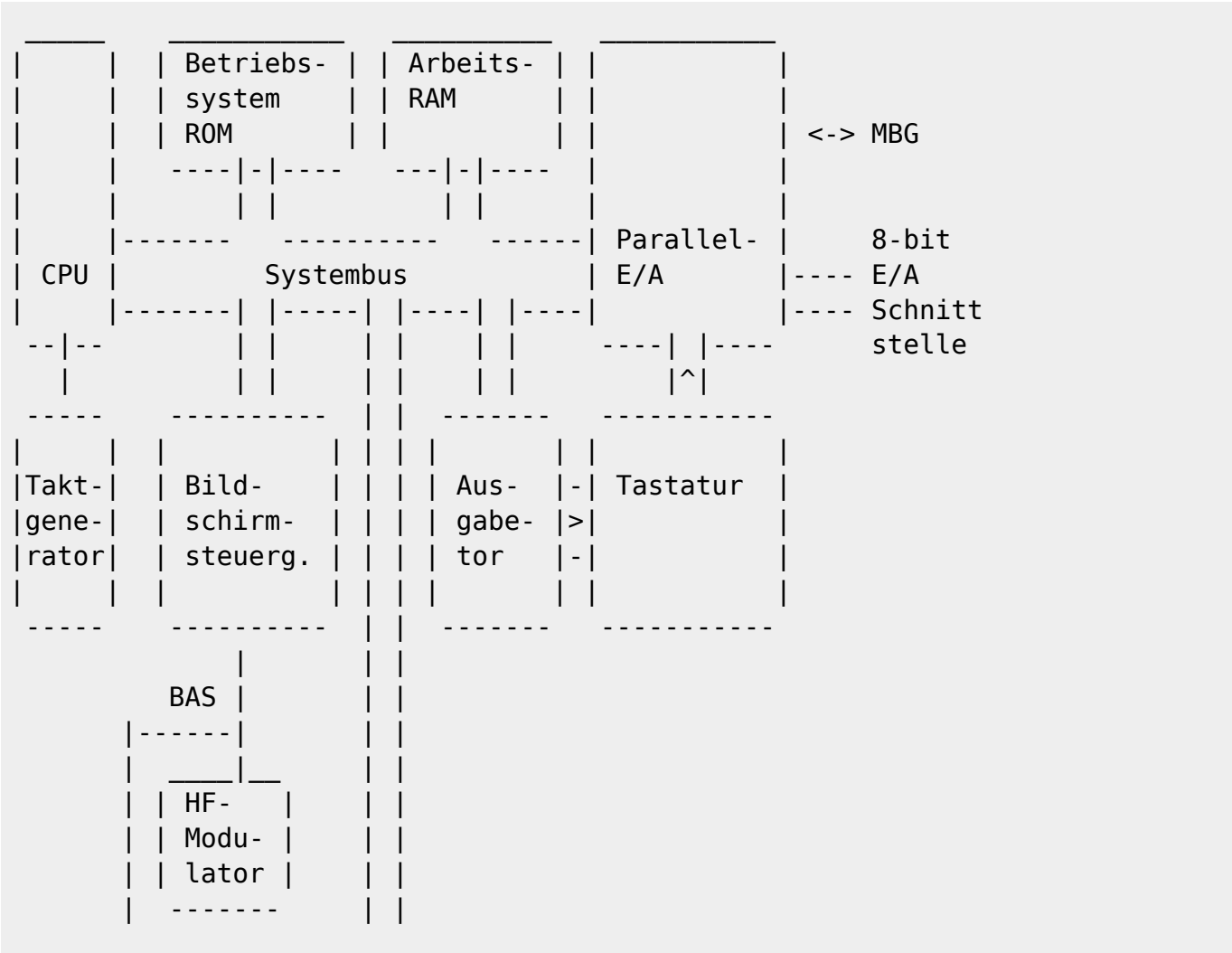
```
 Ausgangswerte:     0 0 0 0 1 0 0 1  *  0 1 0 1
 ---------------------------------------------
                    0 0 0 0 1 0 0 1  |  1
                  0 0 0 0 0 0 0 0    |  0 = 5
                0 0 0 0 1 0 0 1      |  1
              0 0 0 0 0 0 0 0        |  0
 ---------------------------------------------
 Ergebnis:    0 0 0 0 0 1 0 1 1 0 1       = 45
```

Bei der teilweisen Addition kann ebenfalls ein Uebertrag auftreten, d. h. der Zahlenbereich ueberschritten werden. Wenn die Kontrolle nicht in jedem Zwischensohritt vorgenommen wird, ist mit fehlerhaften Ergebnissen zu rechnen.

3. Hardware des Z1013

Am konkreten Beispiel des MRB Z1013 soll in diesem Kapitel die Arbeitsweise eines Mikrorechners erlaeutert werden. Grundlage dafuer bilden Stromlaufplaene des Z 1013, die Sie in der, Anlage 16 finden.

3.1. Blockschaltbild



```
       |     |        | |
     TV-Geraet     Systemsteckverbinder
```

3.2. Steuerung des Mikroprozessors

3.2.1. Beschreibung der Steuersignale

Um den ordnungsgemaessen Betrieb der CPU zu gewaehrleisten, sind bestimmte Steuersignale notwendig. Andere Signale werden von der CPU gebildet und kennzeichnen bestimmte Zustaende waehrend der Abarbeitung von Befehlen. Im folgenden werden alle Steuersignale der CPU und des Systembusses beschrieben.

Signale, die mit einem Schraegstrich beginnen, sind sogenannte LOW-aktive Signale, die normalerweise H-Pegel fuehren und bei ihrer Aktivierung L-Pegel zeigen.

Nach dem Signal steht in Klammern ein „A“ fuer von der CPU ausgesandte, ein „E“ fuer von der CPU empfangene und ein „B“ fuer Signale, die sowohl von der CPU empfangen als auch ausgesandt werden koennen.

(A-Ausgabe, E-Eingabe, B-bidirektional, d. h. sowohl Ein- als auch Ausgabe)

- **A0 bis A15 (A)**
  Sie bilden den 16-Bit-Adressbus. Sie werden in der CPU gebildet und bei der Arbeit mit den Speichern als Speicheradresse sowie die Leitungen A0 bis A7 bei der E/A-Arbeit als E/A-Adresse verwendet.
- **A0 bis A6 (A)**
  Sie dienen zum Auffrischen dynamischer Speicher.
- **D0 bis D7 (B)**
  Diese Leitungen stellen den 8 Bit-Datenbus dar. Die Signale koennen sowohl von der CPU gebildet werden (bei Ausgabe oder Speicherschreiben) oder sie werden von den ausgewaehlten Funktionseinheiten erzeugt (bei Eingaben oder Speicherlesen).
- **/MREQ (A)**
  Dieses Signal wird benoetigt, um eine auf dem Adressbus ausgesandte Adresse zur Speicheradresse zu erklaeren und einen Speicherzugriff durchzufuehren.
- **/IORQ (A)**
  Das Signal kennzeichnet die auf dem Adressbus anliegende Adresse als Adresse einer E/A-Funktionegruppe. Dabei werden nur die Adreseleitungen A0 bis A7 in die Auswahl einbezogen.
- **/M1 (A)**
  Es charakterisiert den Maschinenzyklus 1. Dieses Signal wird von der CPU ausgesendet und kennzeichnet in Verbindung mit dem Signal /MREQ, dass vom Speicher ein Befehl geholt wird. In Verbindung mit dem Signal /IORQ wird gekennzeichnet, dass von einem interrupterzeugenden Baustein (s. 4.4.) der sogenannte Interruptvektor gelesen wird (Vektorlesen).
- **/RD (A)**
  Es wird in den angeschlossenen Funktionseinheiten ausgewertet und legt die Richtung des Datentransportes als „Lesen“, d. h. zur Eingabe in die CPU fest.
- **/WR (A)**

Es kennzeichnet die Richtung des Datentransportes fuer die angeschlossenen Funktionseinheiten als „Schreiben“, d. h. die CPU sendet Daten aus.

- **/RFEH (A)**
  Zeigt den angeschlossenen Speichereinheiten in Verbindung mit /MREQ, dass auf dem Adressbus eine Refresh-Information verfuegbar ist. Diese Refresh-Information besteht aus einer 7-Bit-Adresse (A0 bis A6), die festlegt, welche Speicherzellen in den dynamischen Speichern; aufgefrischt werden sollen. Das Adressbit 7 kann durch den Programmierer gesetzt oder rueckgesetzt werden und ist Bestandteil der Refresh-Information. Auf dem hoeherwertigen Teil des Adressbusses wird der Inhalt des I-Registers ausgesandt.
- **/HALT (A)**
  Wird von der CPU ausgesandt, wenn der soeben gelesene Befehl den Operationskode 76H hatte. Die Abarbeitung wird unterbrochen, der Befehlszaehler zeigt auf den naechsten Befehl. Die Refresh-Steuerung wird aufrechterhalten, eine Fortsetzung der CPU-Arbeit ist nur nach Reset oder Interrupt moeglich.
- **/WAIT (E)**
  Wird von der CPU zu bestimmten Zeiten abgetastet. Ist dieses Signal Low, wird die Arbeit der CPU angehalten, die Informationen auf dem Systembus bleiben erhalten. Anwendung findet dieses Signal vor allem bei der Anpassung der Verarbeitungsgeschwindigkeit von langsamen Funktionseinheiten, indem die Arbeitsgeschwindigkeit der CPU durch solche WAIT-Zyklen der entsprechenden Funktionseinheit angepasst wird. Waehrend des WAIT-Zustandes findet kein Refresh-Zyklus statt.
- **/INT (E)**
  Wird von der CPU am Ende eines Befehls abgetastet und signalisiert, dass eine angeschlossene Funktionseinheit das gerade abzuarbeitende Programm unterbrechen moechte, damit von der CPU die Ursache dieser Unterbrechung analysiert und bearbeitet werden kann. Die Ursachen dieser Unterbrechung koennen ein notwendiger Datentransport zwischen CPU und Interface-Baustein sein oder eine Ereignismeldung aus einem zu ueberwachenden Prozess. Die Funktionseinheiten sind untereinander ueber eine sogenannte Prioritaetskette miteinander verbunden, um die jeweils wichtigste Unterbrechung vorrangig zu behandeln. Die CPU kann ihrerseits die Annahme einer Unterbrechung sperren, um z. B. bestimmte Programmabschnitte stoerungsfrei abzuarbeiten. Nach Freigabe des Unterbrechungseinganges wird die dort eventuell gespeicherte Unterbrechung ausgewertet.
- **/NMI (E)**
  Dieser Eingang stellt aequivalent zum INT-Signal eine Unterbrechungsmoeglichkeit der laufenden CPU-Arbeit dar, die allerdings nicht gesperrt werden kann. Die Abarbeitung des Unterbrechungsbehandlungsprogramms beginnt ab der Adresse 66H, nachdem zuvor die Fortsetzungsadresse des gerade laufenden Programmes gerettet wurde.
- **/RESET (E)**
  Unterbricht jede weitere Arbeit der CPU, stellt einen Anfangszustand ein und gibt mit dem Uebergang nach H-Pegel die CPU wieder frei. Da das Reset-Signal meist manuell erzeugt wird, wird durch die Schaltung eine Verkuerzung dieses Signals vorgenommen, um angeschlossenen dynamischen Speichern die Refresh-Informationen zu garantieren.
- **C (E)**
  Stellt den der CPU zugefuehrten Systemtakt dar. Dieser Takt ist gleichzeitig in allen Funktionseinheiten verfuegbar und sichert die Synchronitaet aller Baugruppen.
- **/BUSRQ (E)**
  Diese Leitung wird am Ende eines Befehls durch die CPU abgetastet. Dieses Signal kennzeichnet, dass eine angeschlossene Funktionseinheit den Systembus benoetigt, um ihrerseits die Vorgaenge im Mikrorechner zu steuern. Die CPU unterbricht das laufende

Programm und setzt ihre Ausgaenge in den hochohmigen Zustand. Gleichzeitig wird ein Quittungssignal von der CPU aktiviert,welches den hochohmigen Zustand anzeigt. Dieser bleibt solange bestehen, wie das Signal BUSRQ aktiv ist, d. h. L-Pegel fuehrt. Danach wird das Quittungssignal von der CPU abgeschaltet, alle Ausgaenge nehmen wieder ihr erforderlich hohes Potential ein und die Abarbeitung wird fortgesetzt. Waehrend des hochohmigen Zustandes kann die CPU keine Refresh-Informationen aussenden.

- **/BUSAK (A)**
  Ist das Quittungssignal der CPU, welches den hochohmigen Zustand kennzeichnet und damit der den Systembus anfordernden Funktionseinheit den Zugriff erlaubt.

Weiterhin umfasst der Systembus folgende Signale, die nichtvon der CPU ausgesandt oder empfangen werden:

- **/MEMDI**
  Stellt ein Systemsignal dar, mit dem angeschlossene Funktionseinheiten den Zugriff auf Speichereinheiten auf der Leiterplatte der Grundausbaustufe verhindern koennen.
  Dieses Signal wird erzeugt, wenn Speichererweiterungen die festgelegten Speicheradressen des Grundgeraetes ebenfalls verwenden. Es wird verhindert, dass nicht mehr als eine Speichereinheit den Datenbus benutzen kann.
- **/IODI**
  Stellt analog zum MEMDI-Signal eine Moeglichkeit dar, bestimmte Adressbereiche auszublenden und Konflikte auf dem Datenbus bei der E/A-Arbeit zu verhindern.
- **/IEI** und **/IEO**
  Werden zur Bildung der Prioritaetskette der interrupterzeugenden Funktionseinheiten benoetigt. Jeweils der Ausgang (IEO) der hoeheren Prioritaet wird dem Eingang (IEI) dernaechstfolgenden Prioritaetsstufe zugefuehrt (vergleiche auch Abschnitt 4.4 Interruptbehandlung). Ein Interrupt kann von einer Funktionseinheit nur ausgeloest werden, wenn das zugefuehrte Signal IEI H-Pegel fuehrt. Gleichzeitig wird das abgegebene Signal IEO auf L-Pegel gehalten. Damit wird sichergestellt, dass immer nur die in der Prioritaetskette am weitesten am Anfang eingereihte Funktionseinheit eine Unterbrechung ausloesen kann.
- **/BAI** und **/BAO**
  Stellen analog zu den Signalen IEI und IEO die Signale einer Prioritaetskette dar, die alle Funktionseinheiten verbindet, die eine Anforderung auf den Systembus (BUSRQ) stellen keennen. Fuer die Benutzer des MRB Z1013 werden diese Signale kaum Bedeutung haben.
- **RDY**
  Stellt ein aehnliches Signal wie WAIT dar, um langsame Funktionseinheiten an die CPU anzupassen. Es kennzeichnet die Kommunikationsbereitschaft einer Funktionseinheit und kann mit der WAIT-Leitung verbunden werden. Im Gegensatz zu den meisten anderen Steuersignalen ist es nicht Low-aktiv.

3.2.2. Takterzeugung

Der Taktgenerator wird durch drei Gatter von A6, dem Kondensator C7.1 und den Widerstaenden R38 und R39 gebildet. Stabilisiert wird die Taktfrequenz durch den Schwingquarz Q1. Dieser schwingt mit einer Frequenz von 8 MHz. Der Takt wird dem Binaerteiler A3 zugefuehrt, an dessen Ausgaengen die Taktfrequenzen von 4 MHz, 2 MHz und 1 MHz anliegen. Der Z 1013.01 arbeitet standardmaessig mit 1 MHz Systemtakt, der Z 1013.12 mit 2 MHz.

Hinweis: Das Umruesten des Z 1013.01 auf 2 MHz fuehrt zum Erloeschen der Garantie. Die Taktfrequenz 4 MHz ist nicht zugelassen!

Je nach Lage von El erhaelt die CPU den Takt mit der Frequenz entsprechend folgender Zuordnung:

Lage	Systemtakt
E1.1	1 MHz
E1.2	2 MHz

Mittels des Widerstandes R52 erfolgt noch die erforderliche Pegelanpassung zur Speisung der CPU (A7) und des E/A-Schaltkreises A45.

Dieser Takt realisiert die Synchronitaet aller Zeitablaeufe.

3.2.3 RESET-Logik

Um einen definierten Anfangszustand der CPU zu erreichen, ist die RESET-Steuerung erforderlich. RESET kann von 3 Stellen ausgeloest werden:

1. Taste TAl auf der Leiterplatte (RESET-Taste)
2. Externe Tastatur ueber den Steckverbinderanschluss X2:A02
3. A20 des Systemsteckverbinders X1

Eine spezielle Schaltung sorgt dafuer, dass der Datenbustreiber A1 inaktiv wird, d. h. er wird vom Prozessor getrennt. Unmittelbar an der CPU werden die Datenleitungen ueber die Widerstaende R44 ... R51 auf Masse, d. h. L-Pegel gelegt.

Da die CPU nach aktiven RESET den Befehlszaehler auf die 0000H einstellt, werden nun auf dieser Adresse die Daten 00H gelesen. Das bedeutet fuer den Prozessor die Ausfuehrung eines sogenannten Leerbefehls (NOP, s. 4.3.15). Bei dessen Ausfuehrung wird der Befehlszaehler um eins erhoeht. Auf diese Art und Weise zaehlen die Adressen hoch, bis die Adresse des Betriebssystems erreicht wird und das Signal /CS aktiviert wird, das den Datenbus mit Hilfe der Logik wieder frei gibt. Als naechstes wird jetzt der erste Befehl des Betriebssystemprogrammes gelesen und dieses wird abgearbeiet.

Damit die Laenge des Reset-Impulses ven der Laenge der Betaetigung unabhaengig wird, wurde ein Monoflop verwendet. Damit wird eine zeitgerechte Aufffrischung der dynamischen Speicher gewaehrleistet. Einige periphere Schaltkreise besitzen keinen Reset-Anschluss. Sie werten das alleinige Auftreten des Signale /M1 als Resetimpuls. Damit auch diese Schaltkreise in einen definierten Anfangszuetand versetzt werden koennen, wurden die Signale /RESET und /M1 zum Signal /PM1 verknuepft, welches die Ruecksetzfunktion ausfuehrt.

3.3 Speichereinheiten

3.3.1. Anschluss

Der Anschluss der Speicherschaltkreise ist abhaengig vom Typ. Im MRB Z1013 werden drei Arten verwendet

In einem PROM U 2616 bzw. ROM U 2316 (A14) ist das Monitorprogramm enthalten. Dieser Schaltkreis besitzt eine Kapazitaet von 2048 (=2K) Speicherplaetzen, wobei bei jedem Zugriff acht Bit parallel gelesen werden. Um diese 2 KByte zu adressieren, sind 11 Adressleitungen (A0 ... A9) notwendig.

Die verwendeten statischen Schreib-Lese-Speicher besitzen eine Kapazitaet von 1024 (=1K) Plaetzen, wobei jeweils 4 Bit gleichzeitig angesprochen werden. Erst zwei dieser Schaltkreise besitzen deshalb eine Kapazitaet von 1 KByte, wobei 10 Adressleitungen (A0 bis A9) ausreichen.

Mit Hilfe dieser 11 bzw. 10 Adressbits wird jeweils nur ein Byte ausgewaehlt. Die verbleibenden Adressleitungen werden nun dazu verwendet, um einen oder mehrere Speichereschaltkreise auszuwaehlen, damit nur eine Information, und zwar die richtige, bearbeitet werden kann. Die Auswahl des betreffenden Speicherschaltkreisee erfolgt mit dem Adressdekoder A23, der aus einem Bereich von 8 KByte fuer jeden einzelnen 1 KByte-Bereich eine Auswahlleitung bereitstellt. Mit dem Gatter A 24/25 wird dieser Bereich auf den oberen Adressraum eingestellt. Dazu werden mit A25 die betreffenden Adressleitungen mit dem Speicherauswahlsignal MREQ verknuepft und damit der Adressdekoder frei gegeben, d. h. konkret

```
MREQ  ADR
      15 14 13  12 11 10     9 8 7 6 5 4 3 2 1 0
   0  1  1  1               Diese Leitungen werden an alle
                             Speicherschaltkreise gelegt
                0  0  0  --->/DK10 = E000H = RAM
                0  0  1  --->/DK11 = E400H
                  ...
                0  1  1  --->/DK13 = EC00H = Bildwieder-
                                             holspeicher
                1  0  0  --->/DK14 = F000H = Mit Dioden D9
                1  0  1  --->/DK15           ODER verknuepft
                                             fuer 2K-Monitor
```

Die so gebildeten Leitungen zur Bausteinauswahl (chip-select, CS) werden an den CS-Eingang der Speicherschaltkreise gefuehrt und geben diese frei.

Die Bildung der Auswahlsignale kann ueber das Signal MEMDI am Steckverbinder X1 von ausserhalb verhindert werden. Das wird dann sinnvoll sein, wenn der MRB Z1013 als Bestandteil eines Mikrorechnersysteme betrieben wird und in diesen Adressbe- reichen bereits Speichereinheiten angeschlossen sind.

Bei der Verwendung der dynamischen 16 KByte Speicher U 256 bzw. K 565 RU3 oder K 565 RU6 (A33 bis A40) ergibt sich folgende Adressauswertung:

```
ADR: 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
      0  0 Diese Adressbits werden intern zur
           Auswahl des Speicherplatzes ausgewertet
     Legt den Bereich ab Adresse 0000H fest.
```

Da diese Speicher pro Platz nur 1 Bit speichern, muessen hier 8 Schaltkreise parallel an den Datenbus angeschlossen werden. Der Anschluss ist ausserdem komplizierter, weil diese Schaltkreise nur 6 Adresseingaenge haben. Die Uebernahme der 14-stelligen Adresse erfolgt deshalb zeitlich gestaffelt.

Zuerst werden bei Auswahl dieses Speicherbereiches die sieben niederwertigen Adressbits in den Speicherschaltkeis uebernommen. Dazu wird das Signal RAS (ROW ADRESS STROBE, Reihenadressuebernahmeimpuls) am Schaltkreis aktiviert. Anschliessend werden die sieben hoeherwertigen Adressbits auf die Schaltkreisanschluesse geschalten und mit dem Signal CAS (COLUMN ADRESS STROBE, Spaltenadressuebernahmeimpuls) diese in die Schaltkreise eingetragen.

Die Erzeugung des Signals /RAS ist durch das Speicheranforderungssignal /MREQ gegeben. Das negierte Signal MREQ gibt ein FlipFlop (A17) frei. Mit der naechsten steigenden Flanke des Systemtaktes wird das FlipFlop umgeschalten steuert den Adressleitungsumschalter A28/41 (Multiplexer) und gibt ein zweites Flip-Flop frei, das mit der absteigenden Flanke des Systemtaktes das Signal CAS erzeugt und, sofern ein Zugriff in diesen Adressbereich (A14=A15=0) erfolgt, das Signal /CAS an den Schaltkreisanschluessen aktiviert. War der Zugriff zum Speicher nicht in dem Bereich der dynamischen RAM's oder wurde das Signal /REFRESH als Zeichen der Speicherauffrischungszeit aktiv, so wird zwar das Signal /RAS gebildet, aber kein Signal /CAS. Damit werden in den dynamischen RAM's nur Auffrischungsaktivitaeten ausgeloest.

In Verbindung mit den Signalen /RD oder /WR, die ebenfalls an die Speicherbausteine gefuehrt werden, werden entweder Daten oder Befehle gelesen und auf den Datenbus geschaltet oder die auf dem Datenbus vorhandenen Daten im Speicher eingetragen.

Im Anhang ist ein Schema der Speicherverteilung innerhalb des gesamten Adressraumes zu finden (Anlage 2).

3.3.2. Zusammenarbeit mit der CPU

Wenn die CPU auf den Speicher zugreifen moechte, sei es, um Befehle oder Daten zu holen oder um etwas abzuspeichern, ist das durch folgende Signale gekennzeichnet:

- /MREQ (Speicheranforderung) wird Low, d. h. aktiv und zeigt damit den Zugriff auf den Speicher an.
- A0 bis A15 (Adressen) geben den konkret adressierten Speicherplatz an.
- /WRt (Schreiben) wird aktiv, wenn die CPU etwas in den Speicher schreiben moechte.
- /RD (Lesen) wird beim Lesen von Daten oder Befehlen aktiv.
- /M1 (Befehlsholezyklus) kennzeichnet in Verbindung mit /MREQ und /RD das Holen des Operationskodes eines Befehls (s. Kapitel 4)
- D0 bis D7 enthalten entweder die abzuspeichernde oder gelesene Information.

Die detaillierten Zeitablaeufe koennen der Anlage 10 entnom- men werden, wo die Taktdiagramme fuer den Speicher-Schreib- und Speichen-Lese-Zyklus angegeben werden.

3.4. Ein- und Ausgabebaugruppen

3.4.1. Parallel E/A-Baustein U 855 PIO

3.4.1.1. Beschreibung der Steuersignale

Aus der Bezeichnung des Bausteins geht eigentlich seine Verwendung bereits hervor. Er dient bevorzugt zur parallelen Ein- bzw. Ausgabe, d. h. zum Beispiel, dass alle acht Bit des Datenbusses gleichzeitig ausgegeben werden koennen.

Man kann natuerlich auch Daten seriell, d. h. bitweise nacheinander aus- oder eingeben. Dazu ist aber ein gesondertes Programm notwendig.

Im MRB Z1013 kommt ein Baustein U 855 zum Einsatz. Ein Teil davon wird von den E/A-Baugruppen des Z1013 selbst genutzt (s. 3.4.2., 3.4.3.). Ueber den anderen Teil koennen Sie frei verfuegen. Dazu muessen Sie allerdings die Anschluss- und Funktionsweise einer PIO kennen. Das soll Inhalt dieses Abschnittes sein.

Die Anschlussbelegung des U 855 finden Sie in der Anlage 9. Es ist zu erkennen, dass die PIO rechnerseitig an den Datenbus angeschlossen wird und prozesseitig zwei Kanaele A und B, auch Tore oder Ports genannt, besitzt. Ausserdem verfuegt er ueber eine Reihe von Steuersignalen, deren Bedeutung hier kurz erlaeutert werden soll:

Es gelten die gleichen Vereinbarungen wie im Abschnitt 3.2.1.

- **B/A SEL (E)**
  Liegt dieser Eingang auf „L“, so wird das Tor A, liegt er auf „H“, dann das Tor B freigegeben. Ueblicherweise wird hieran die Adressleitung der CPU A1 gefuehrt.
- **C/D SEL (E)**
  Der U 855 ist ein programmierbarer E/A-Baustein, d. h. es muss vor der eigentlichen Nutzung fuer den Datentransfer zwischen Rechner und Prozess mitgeteilt werden, was er machen soll. Dazu gibt es eine Reihe von Steuerwoertern, die die Befehle der PIO darstellen. Diese Programmierung der PIO wird im allgemeinen als Initialisierung bezeichnet. Lesen Sie dazu den Abschnitt 3.4.1.2. Erhaelt dieser Eingang „L“-Begel, so sind die auf dem Datenbus befindlichen Informationen Daten, bei „H“-Pegel Steuerwoerter. Ueblicherweise liegt C/D SEL an der Adressleitung A0.
- **/CS (E)**
  Hiermit wird die PIO fuer den Datentransfer freigegeben. Die Bildung dieses Bausteinauswahlsignals erfolgt analog zur CS-Dekodierung fuer die Speichereinheiten, nur dass fuer die E/A-Dekodierung die Adressen AO bis A7 (Niederwertiger Teil des Adressbusses) ausgewertet werden. Es koennen also maximal 256 (2 hoch 8) E/A-Tore angeschlossen werden.
- **/IORQ (E)**
  Dient in Verbindung mit den anderen Signalen zur Kennzeichnung der E/A-Anforderung. Dieser Eingang wird direkt an den entsprechenden Ausgang der CPU gelegt.
- **/M1 (E)**
  Mit aktiven Ml bei nicht aktiven RD und IORQ wird die PIO in einen definierten Anfangszustand zurueckgesetzt. Geschieht dies nicht, arbeitet die PIO unkontrolliert. Anschliessend muss die Initialisierung erfolgen. Ausserdem synchronisiert dieses Signal in Verbindung mit IORQ die Interruptbehandlung durch die CPU. Damit beide Funk- tionen gewaehrleistet werden koennen, muss dieses M1 aktiv bei aktivem RESET der CPU oder bei Aussendung des CPU-M1 sein. Diese ODER-Verknuepfung wird durch die Bildung des /PM1 realisiert, welches an das PIO-M1 angeschlossen wird.
- **RD (E)**
  Wird die Datenbusinformation in den PIO geschrieben, muss RD inaktiv sein. Ist RD auf „L“, legt die PIO die vom Prozess gelesenen Daten, entsprechend den mit B/A SEL ausgewaehlten Tor,

auf den Datenbus.

- **C (E)**
  Systemtakt analog zur CPU
- **/ASTB (E)**, **/BSTB (E)**
  Diese Steuerleitungen werden zur Quittung des erfolgten Datenaustausches verwendet. Zur Ausgabe wird diese Leitung (Low-aktiv) vom angeschlossenen Geraet aktiviert und damit die Daten uebernommen. Bei der Eingabe wird mit dieser Leitung angezeigt, dass die anstehenden Daten in die PIO uebernommen werden koennen. Der Uebergang des Signals /STB aus dem aktiven Zustand in den H-Pegel kann zur Bildung des Interruptsignals verwendet werden.
- **ARDY (A)**, **BRDY (A)**
  Diese Steuerleitungen teilen dem angeschlossenen Geraet mit, dass bei der Ausgabe Daten bereitstehen, waehrend bei der Eingabe dieses Signale dem Geraet die Bereitschaft zur Datenuebernahme signalisiert.
- **/INT (A)**
  Dieses Signal liegt parallel zu allen anderen interruptausloesenden E/A-Baugruppen am INT-Eingang der CPU und meldet der CPU, dass eine Unterbrechung des aktuell laufenden Programms erwuenscht wird. Ursache dafuer kann eine Meldung vom Prozess sein, da hier eine Warnung ausgegeben wurde, die unbedingt eine Behandlung erfordert.
- **/IEI (E)**, **/IEO (A)**
  Hiermit werden die Prioritaeten bei der Behandlung von Unterbrechungsanforderungen durch Bildung einer Prioritaetskette (daisy chain).

```
        _________________         _________________
       | E/A-Baugruppe |       | E/A-Baugruppe |
       |       1       |       |       2       |
5P -----| IEI       IEO |-------| IEI       IEO |---->
        ----------------          ----------------
```

Die in einer solchen Kaskade am weitesten links stehende Baugruppe hat den groessten Vorrang. Wird an dieser E/A-Einheit eine Unterbrechung angemeldet, dann wird diese Kette unterbrochen (der Schalter oeffnet), so dass fuer die nachfolgenden Einheiten ein Interrupt gesperrt ist.
Intern besitzt das Tor A gegenueber Tor B hoehere Prioritaet.

3.4.1.2. Programmierung

Am Beispiel der im MRB Z1013 verwendeten E/A-Tore soll die Bildung der Auswahladresse erlaeutert werden. Fuer die Ergaenzung der Chip-select Signale wird ein Dekoder A27 eingesetzt, der mit dem E/A-Anforderungssignal die ersten acht Ausgaenge freigibt. Die Festlegung der jeweiligen aktiven IOSEL-Leitung erfolgt dann mit den Adressen A2, A3 und A4. Mit dem im vorigen Abschnitt zu den 0/13 SEL- ;4nd B/ASEL- Signalen gesagten ergibt sich folgende Adreseverteilung:

```
ADR:    7  6  5  4  3  2  1  0
                              C/D SEL
                              0, wenn Information Daten
                              1, wenn Information Steuerworte
                           B/A SEL
```

```
                        0  , wenn Tor A
      beliebig, z.B.    1  , wenn Tor B
      0  0  0  0  0  0  ==>IOSEL0, PIO
               0  1  0  ==>IOSEL2, Tastaturspaltentreiber
```

Damit ergeben sich die Adressen:

```
   Tor A (Anwenderport) - Daten:      00H
                        - Steuerwort: 01H
   Tor B (Systemport)   - Daten:      02H
                        - Steuerwort: 03H
```

Im Z1013 sind diese Adressen nicht eindeutig, da die Adressbits A7, A6, A5 auch 111 sein koennten. Da diese nicht ausgewertet werden, spielt das aber keine Rolle.

Die Arbeitsweise der PIO wird durch die Steuerworte festgelegt, die im folgenden erlaeutert werden sollen.

1. **Betriebsartenauswahl**

```
   Bit:        7   6   5   4   3   2   1   0
   Belegung:                   1   1   1   1  Kennzeichen
                       beliebig
               0   0   Betriebsart    0 :    Byteausgabe
               0   1                  1 :    Byteeingabe
               1   0                  2 :    Byteein/ausgabe
               1   1                  3 :    Bitein/ausgabe
```

   Betriebsart 0:
   Die durch die CPU bereitgestellten Daten werden waehrend des durch sie veranlassten Ausgabezyklus in das angesprochene Ausgaberegister geschrieben. Mit RDY zeigt die PIO dem Prozess an, dass Daten zur Uebernahme in PIO bereitstehen. Dieses RDY wertet das periphere Geraet aus, uebernimmt daraufhin die Daten und teilt mit STB der PIO die Datenuebernahme mit. Anschliessend loest die PIO ein Interrupt aus, um der CPU das Ende der Datenausgabe zu melden.

   Betriebsart 1:
   Die PIO teilt mit H-Pegel an RDY dem externen Geraet die Bereitschaft zur Datenuebernahne mit. Mit STB=L schreibt das Geraet die Daten in das entsprechende Eingaberegister. RDY=L sperrt eine weitere Eingabe bis die Daten durch eine Interruptbehandlung von der CPU uebernommen werden.

   Betriebsart 2:
   Diese bidirektionale Betriebsart ist nur mit dem Kanal A moeglich. Mit Kanal B ist dann nur noch Betriebsart 3 moeglich, da fuer die Abwicklung des Datentransfers alle vier Quittungssignare ARDY, ASTB, BRDY und BSTB benoetigt werden. ARDY und ASTB steuern die Ausgabe, die beiden anderen die Eingabe.

   Betriebsart 3:
   In dieser Betriebsart kann innerhalb eines Tores jedem Bit eine beliebige Datenflussrichtung

zugeordnet werden. Auf diese Weise koennen Stellsignale und Statusmeldungen fuer Prozess-Steuerungen aus- bzw. eingegeben werden.

2. **Ein-/Ausgabe Maskenwort**
   Soll die Bitstelle eine Eingabeleitung sein, muss an dieser Stelle eine 1 stehen, bei Ausgabe eine 0. Da dieses Steuerwort kein eigenes Kennzeichen besitzt, muss es unmittelbar auf das Betriebsauswahlsteuerwort folgen. Ist in diesem Betriebsart 3 festgelegt worden, liest die PIO das naechste Byte immer als E/A-Maskenwort.
3. **Interruptvektor**

```
Bit: 7   6   5   4   3   2   1   0
Belegung:                        0 Kennzeichen
     niederwertiger Teil des Interruptvektors
```

   (s. 4.4.)

4. **Interruptsteuerwert**

```
Bit:      7   6   5   4   3   2   1   0
Belegung:                 0   1   1   1  Kennzeichen
                      0   , naechstes Steuerwort ist kein
                            Maskenwort
                      1   , naechstes Steuerwort wird als
                            Maskenwort erkannt
                  0   , Interrupt bei H -> L Flanke
                  1   , Interrupt bei L -> H Flanke
              0   , die im folgenden Steuerwort festgelegten
                    Interrupt ausloesenden Bit sind ODER-
                    verknuepft, d. h. eine dieser Leitungen
                    kann bereits Interrupt ausloesen
              1   , UND verknuepft, d. h. alle festgelegten
                    Stellen muessen gleichzeitig die mit Bit
                    5 festgelegte Interruptbedingung erfuel-
                    len
          0  , Interrupt freigeben
          1  , Interrupt gesperrt
```

5. **Interruptmaskenwort**
   Die Bitzelle der Eingabeleitungen, die Interrupt ausloesen sollen, werden durch eine 0 gekennzeichnet, die kein Interrupt ausloesen sollen durch eine 1.
6. **Interrupt Ein/Aus**

```
Bit:    7 6   5   4   3   21  0
Belegung: beliebig    0   01  1   Kennzeichen
        0  ,Interrupt gesperrt
        1  ,Interrupt freigegeben
```

3.4.2. Tastaturanschluss

Elektrisch stellt die Tastatur nichts anderes als eine Matrix von Schaltern in folgender Anordnung dar:

Die Zeilen dieser Anordnung sind mit den Widerstaenden R11 bis R14 auf „H“-Pegel gelegt. Diese Leitungen sind mit dem Tor B, Bit 0 bis 3, des PIO verbunden, welche fuer Eingabe programmiert sind. Wird keine Taste gedrueckt, liest die PIO auf allen vier Leitungen eine 1.

Die acht Spaltenleitungen der Tastatur sind an ein separates Ausgabetor, das durch die Bausteine A47 (Speicher fuer Spaltennummer) und A46 (1 aus 8 Spaltenleitungen) gebildet wird, angeschlossen. Die Adresse dieses Tores ist 08H. Die Spatennummer steht im niederwertigen Halbbyte des Datenbusses binaer verschluesselt. Bei einer Ausgabe werden diese vier Bit entschluesselt und legt so eine Spalte auf „L“-Potential. Wird in dieser aktivierten Spalte nun eine Taste betaetigt, wird der L-Pegel auf die entsprechende Zeilenleitung durchgereicht. Der Rechner liest jetzt eine 0 in der entsprechenden Bit stelle.

Aus der ausgegebenen Spaltennunmer und der eingelesenen Zeilennummer ermittelt das Tastaturbedienprogramm des Betriebssystems den rechnerinternen Kode der gerade betaetigten Taste. Der Z1013 benutzt den sogenannten ASCII-Kode (s. Anlage 7).

3.4.3. Magnetbandanschluss

Von der auf der Leiterplatte installierten PIO wird eine Bitleitung (PB 7) zur Ausgabe eines seriellen Datenstromes ge- nutzt. Die erforderliche Parallel/Serienwandlung wird softwarenaessig realisiert. Das ausgegebene Signal wird ueber einen Spannungsteiler R27/28 zur Pegelanpassung abgeschwaecht; mit einem Kondensator C1.9 werden die Flanken verrundet, damit ein etwa sinusfoerniges Signal in Magnetbandgeraet aufgezeichnet werden kann.

Das Ausgangssignal eines Magnetbandgeraetes wird gleichspannutigsfrei einem Operationsverstärker A48 zugefuehrt. Das auf TTL-Pegel verstaerkte Signal wird an einen Anschluss der PIO (PB 6) geleitet, Durch entsprechende Software wird dieser Anschluss staendig abgefragt und aus dem ankommenden seriellen Datenstrom durch Serien/Parallelwandlung die urspruengliche Information wieder zurueckgewonnen.

3.4.4. Bildschirmsteuerung

Die Bildschirmsteuerung wandelt die vom Rechner auszugebende Information in ein CCIR-kopatibles Fernsehsignal, indem sie zusaetzlich die notwendigen Synchron- und Dunkeltastimpulse erzeugt. Um diesen Vorgang prinzipiell zu verstehen, sind einige Bemerkungen ueber den Aufbau des Fernsehsignals notwendig.

Beim Schreiben eines Fernsehbildes laeuft ein Elektronenstrahl, auf den die Bildinformation aufmoduliert wurde, ueber einen flueresziererenden Schirm. Fuer eine Zeile benoetigt er eine Zeit von 64 µs. Das entspricht einer Zeilenfrequenz von 15,625 kHz. Ein Zeilensynchronimpuls veranlasst den Strahlruecklauf, wobei der Strahl dunkelgesteuert wird. Um ein Flimmern der Anzeige zu vermeiden, muss das ganze Bild mit einer Frequenz von mindestens 25 Hz wechseln.

Da beim Fernsehen in dieser Zeit zwei Halbbilder geschrieben werden, im Z1013 aber ein Bild zweimal, ergibt sich hier eine Bildwechselfrequenz von 50 Hz.

Ein sogenannter Bildsynchronimpuls loest dann jeweils einen Strahlruecklauf zum oberen Bildrand aus. Die Bildschirmsteuerung des MRB Z1013 arbeitet nach folgendem Prinzip:

Die gesamte Erzeugung des fernsehgerechten Signals, des sogenannten BAS-Signals, wird durch die Zaehlkaskade ohne Mitarbeit der CPU gesteuert. Die Kaskade A3, A4, A5 und A12 wird mit dem 8 MHz-Takt des Taktgenerators gespeist. Eine Teilung durch 2 hoch 9 liefert z. B. die Zeilenfrequenz.

Aus dem Bildaufbau wissen wir bereits, dass eine Zeile aus 32 (=2 hoch 5) Zeichen besteht. Um diese abzuzaehlen, werden die 5 niederwertigen Adressen des Bildwiederholspeichers (BWS) A30/31 genutzt. Die hoeherwertigen Adresseingaenge zaehlen die Zeichenzeilen eines Bildes. Da die Zaehlkaskade immer zyklisch durchzaehlt, wird auch der BWS zyklisch ausgelesen.

Das aus dem BWS gelesene Byte, das den ASCII-Kode entsprechend Anlage 7 des darzustellenden Zeichens enthaelt, steht als hoeherwertiger Adressteil am Zeichengemerator A44. Mit den drei Ausgaengen des Linien pro Zeichenzaehlers, die an die niederwertigen Adresseingaenge von A44 gehen, werden nacheinander die Bildpunktzeilen an den nachfolgenden Parallel/Serien-Wandler A21/22 uebergeben. Hier wird das uebernommene Bitmuster mit dem 8 MHz-Takt seriell herausgeschoben. Dieser seriell Datenstrom bildet die Bildinformation des Bild-, Austast- und Synchronsignals (BAS-Signal).

Mit den Gattern der Schaltkreise A9, A10, A13 und A20 werden aus dem Zaehlfolgen entsprechend der Fernsehnorm die Synchronimpulse dekodiert.

Ausserdem wird durch diese Schaltung gesichert, dass fuer der Strahlruecklauf das Signal dunkelgesteuert wird, da dieser sonst auf dem Bildschirm sichtbar waere. Diese Impulse werden mit der Bildinformation gemischt und ergeben so das BAS-Signal.

In einem HF-Modulator wird das BAS-Signal auf eine HF-Traegerfrequenz, die auf den Fernsehkanal 3 abgestimmt ist, aufmoduliert. Der Ausgang dieses Modulators kann nun direkt mit dem Antenneneingang des Fernsehgeraetes verbunden werden.

Wie gelangen aber nun in diese selbstaendig arbeitende Einheit die darzustellenden Daten? Ueber die Adroesmultiplexer (A29, A42, AlS) kann die CPU einen Platz im BWS adressieren. Dazu wird mit einem Speicherbereichauswahlsignal der Multiplexer umgeschaltet. Ueber den Datentreiber A43 kann die CPU den BWS beschreiben oder lesen.

Damit ist auch deutlich gemacht, dass der BWS wie ein normaler Speicher behandelt werden kann. Die Anfangsadresse ergibt sich analog zu dem ROM-Auswahlsignal zu EC00H. Welche Position die einzelnen Speicherplaetze auf dem Bildschirm einnehmen, ist in der Anlage 8 schematisch dargestellt.

3.5. Stromversorgung

Fuer den Betrieb des MRB Z1013 sind drei verschiedene Versorgungsspannungen noetig.

Zur Versorgung aller Logikschaltkreise wird eine Spannung von + 5 V, die im folgenden mit 5P bezeichnet wird und etwa mit 1 A belastbar ist, verwendet. Die beiden anderen Span- nungen von + 12 (12P) und - 5 V (5N) werden fuer die Speichereinheiten sowie einige Spezialfaelle benoetigt. Sie werden nicht so stark belastet.

Um diese Spannungen zu erzeugen, besitzt der MRB Z1013 ein eigenes Netzteil. Eine zugefuehrte Wechselspannung von ca. 12 V wird mittels Dioden in Einweggleichrichtung gleichgerichtet. An den Ladekondensatoren C2.1, C3.1 und C5.1 sind jeweile Rohspannungen verfuegbar. Eine Ausnahme bildet die Erzeugung der Rohspannung fuer die 12P. Hier wird mit einer Spannungsverdopplerschaltung gearbeitet.

Die Erzeugung der 5P wird mit einem integrierten Festspanmungsregler A2 vorgenommen, der auf einem Chip alle benoetig- ten Bauteile enthaelt und kaum eine Aussenbeschaltung benoetigt. Lediglich ein Kondensator am Ausgang ist erforderlich. Da eine starke Belastung dieses Bauelementes erfolgt, wird eine angemessene Kuehlung benoetigt.

Die Spannung 5N wird mittels einer Z-Diode D4 stabilisiert. Diese einfache Widerstands/Z-Dioden-Kombination ist bei dem geringen Leistungsbedarf ausreichend.

Um die Spannung 12P zu erzeugen, wird eine verdoppelte und anschliessend mit einer Widerstands/Z-Dioden-Kombination stabilisierte Spannung der Basis eines Transistors V2 zugefuehrt. Dadurch ist am Emitter dieses Transistors eine stabilisierte Spannung verfuegbar, die staerker belastet werden kann.

3.6. Bussystem

Die wichtigsten Signale des Mikrorechners Z1013 sind an den Rand der Leiterplatte gefuehrt und dort fuer den Anschluss von Steckverbindern vorbereitet. Dabei haben diese Anschluesse folgende Bedeutung:

XI:	Systembus (Steckverbinder: StL 304-58 TGL 29331/03) *Enthaelt alle Signale des Systembusses und ist elektrisch kompatibel zum K1520-Systembus. (Anlage 6)*
X2:	Pruefkamm und Tastaturanschlusspunkte *(hier wird entsprechend den Hinweisen von Pkt.1.2.4.1. und 1.4. der Bedienungsanleitung das Tastaturbandkabel oder die Buchsenleiste BuL 202-26 TGL 29331/04 angeloetet)*
X3:	Wechselspannungszufuehrung (Flachsteckverbinder)
X4:	PIO Kanal A (Steckverbinder: BuL 402-15 TGL 29331/04) *Hier werden die Anschluesse des Kanals A der PIO herausgefuehrt. Ausser den Steuerleitungen ARDY und /ASTB des Kanals A wurden auch die des Kanals B (BRDY und /BSTB) auf den Steckverbinder gelegt, um die Betriebsart bidirektionale E/A realisieren zu koennen.*
X5:	Anschluss Magnetbandgeraet (Diodenbuchse)
X6:	HP-Ausgang des Modulators (Koaxialbuchse)

Die genaue Zuordnung der einzelnen Signale zu den jeweiligen Anschluessen ist der Anlage 6 zu entnehmen.