Interner Bericht
DESY F56-73/1
September 1973

Interface-System für Prozeßrechner

Band 1 - Der UNIBUS der PDP-11-Familie

von

H.-J. Stuckenberg

Diese Reihe besteht aus 2 Bänden:

Band 1 - Der UNIBUS der PDP-11 Familie

Band 2 - Das CAMAC-BUS-System

H.-J. Stuckenberg

Hamburg, September 1973

1. Einleitung

Prozeßrechner spielen eine wichtige Rolle in der Kontrolle und Führung von Prozessen. Ihr Anwendungsbereich erstreckt sich vom on-line-Betrieb in naturwissenschaftlichen Laborexperimenten, wo der Rechner das zentrale Element ist, bis zum einfachen Aufsammeln von industriellen Meßdaten.

In der Architektur der Prozeßrechner steckt das gleiche Grundkonzept wie bei allen Rechnern, sie haben einen Prozessor, einen Speicher sowie In/Output-Kanäle, die den Verkehr zu den Peripheriegeräten abwickeln. Als charakteristische Unterschiede zu Mittel- und Großrechnern sind die kürzere Wortlänge sowie ein beschränkter Befehlsvorrat zu nennen.

Als typische Wortlänge hat sich eine Einheit von 16 Bit ergeben, selten werden 12 oder 24 Bit angeboten. Die Register, die Arithmetik, der Speicher und die Bussysteme werden auf die Wortlänge abgestimmt. Manchmal entstehen durch kurze Wortlängen Probleme im Befehlsformat, durch geschickte Adressiermethoden können aber trotzdem leistungsfähige Befehle erzeugt werden. Eine typische Prozeßkontrolle ist mit 16 Bit-Worten voll möglich.

Die Beschränkung des Befehlsvorrats zeigt sich im Fehlen der Gleitkommaarithmetik, erst bei einigen neueren Rechnern, die nach Aufbau und Preis sich bereits der Mittelklasse nähern, ist dieses vorhanden. Außerdem fehlen oft eine Dezimalarithmetik sowie Bytebehandlungsbefehle.

Prozeßrechner sind sehr flexibel in der Konfiguration, durch modularen Aufbau kann diese leicht den entsprechenden Problemen angepaßt werden.

Die Speichergröße richtet sich nach dem vorhandenen Adressenraum, meist ist dieser auf 32 K-Worte beschränkt, bei neueren Modellen bis 128 K erweitert. Die Speicher werden meist in Blöcken zu 4 K-Worten angeboten, teils noch Kernspeicher mit Zykluszeiten von 800-2000 nsec, teils schon Halbleiterspeicher mit Zykluszeiten von 450-950 nse.

Das Interface-System, das den Rechner mit seiner Peripherie verbindet und den Transport von Daten und Kontrollsignalen in beiden Richtungen ausführt, kann auf verschiedene Weise realisiert werden. Zwei neuere Konzepte, die in sich geschlossen wirken und die sich in einer großen Zahl von wissenschaftlichen und

industriellen Anwendungen durchzusetzen beginnen, sind das UNIBUS-Konzept der PDP-11 Familie der Firma Digital Equipment sowie das rechnerunabhängige, international entwickelte CAMAC-System. Beide Systeme sollen in ihrer Wirkungsweise beschrieben sowie einige wesentliche Anwendungen erläutert werden. Im zweiten Band, der das CAMAC-System beschreibt, wird auch die Verbindungs-Hardware beider Systeme gezeigt.

Als Literatur für den ersten Band dienen vor allem die "Processor Handbook" der Rechner PDP 11/20, 11/15, 11/r20 sowie das entsprechende für die PDP 11/45 und das "Peripherals and Interfacing Handbook" der PDP-11 Familie. Alle Bücher sind von der Firma DEC im Jahre 1972 herausgegeben.

2. Interface, Prinzip und Probleme

Wirtschaftlichkeit und Nutzen eines Computers hängen wesentlich von der Art und dem Umfang der Peripheriegeräte ab, die mit ihm in Dialog treten können. Leider ist es nicht möglich, diese Geräte mit einem einfachen Steckkontakt an den Computer zu schließen, man benötigt eine Zwischeneinheit aus Hard- und Software, das Interface. Wie kompliziert eine solche Einheit ist, hängt von den Input-/Output (I/O) -Eigenschaften des Rechners ab.

Der Zweck von Peripheriegeräten ist es, Arbeiten im Rahmen der Datenverarbeitung zu übernehmen, die der Prozessor (Central Processor Unit CPU) nicht ausführen kann, wie z.B. Daten aufsammeln, sie in großen Mengen zu speichern, Analog-Digital-Konversionen auszuführen usw. In den meisten Fällen sind die Peripheriegeräte allgemeine Entwicklungen, d.h. nicht für einen speziellen Rechner konzipiert; daher muß das Interface die Lücke zwischen diesen Geräten und dem Rechner schließen.

Je nach den I/O-Eigenschaften des Rechners kann das Interface nach folgenden Gesichtspunkten ausgelegt werden:

- der Entwickler kann sich zwischen einer mehr Hardware- oder mehr Software-orientierten Lösung entscheiden
- der Entwickler kann das Interface nach den unbedingt benötigten Eigenschaften entwerfen.

2.1. Programmierter Transfer oder direkter Speicherzugriff?

Die meisten Prozeßrechner können den Datenverkehr zu den Peripheriegeräten auf zweierlei Art ausführen, durch

- programmkontrollierten Transfer und durch
- Transfer mit direktem Speicherzugriff (DMA).

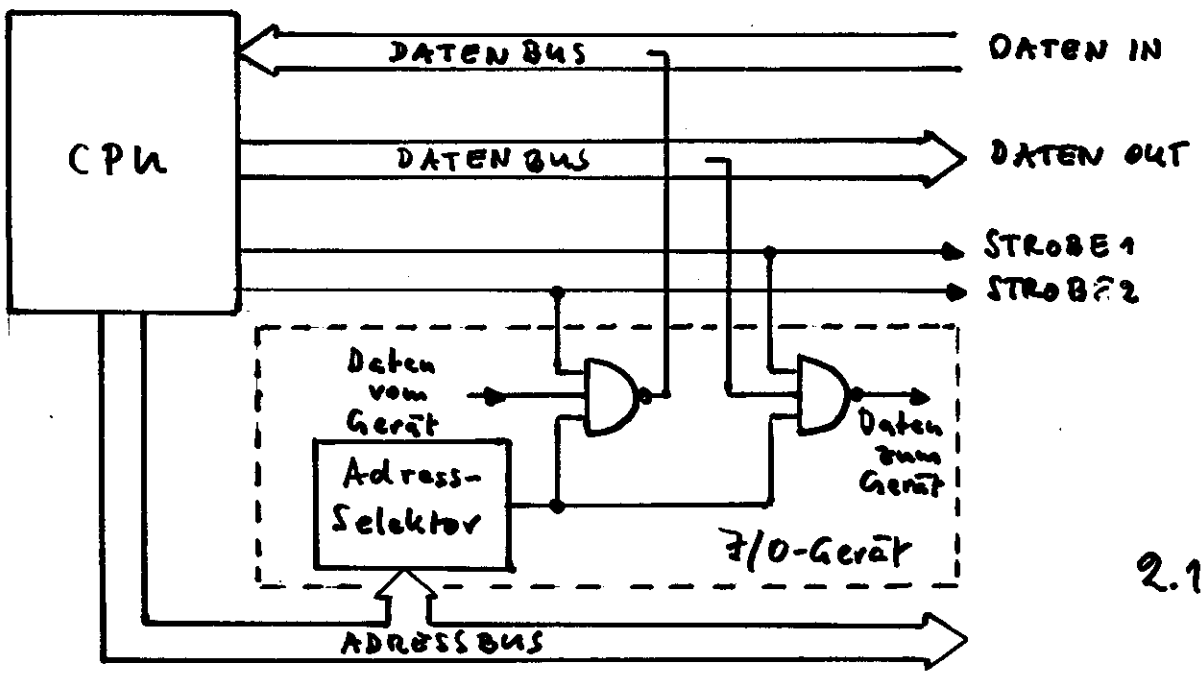
Beim programmkontrollierten Transfer muß der Rechner einen oder mehrere Befehle ausführen, um die Übertragung zwischen einem Prozessorregister und einem Peripheriegerätregister abzuwickeln.

Beim DMA-Transfer überträgt das externe Gerät die Information zum oder vom Speicher ohne Kontrolle durch den Prozessor, dieser wird benachrichtigt, wenn

die Übertragung beendet ist. Der DMA-Transfer wird meist zur Übertragung grösserer Datenblöcke und hoher Folgefrequenz zwischen Speicher und Peripheriegerät eingesetzt.

Programmierter Transfer ist die einfachste Art des Datenverkehrs. Alles, was dazu benötigt wird, ist eine Anzahl von Leitungen zwischen dem Prozessor und dem Interface für die Daten und Adressen sowie verschiedene Kontrolleitungen, die die Signale zum Ein- bzw. Ausstroben der Daten führen. Zur Ausführung des Transfers benötigt das Programm lediglich geeignete I/O-Befehle, die die Adresse sowie die Kontrollsignale spezifizieren.

Die zum programmierten Transfer benötigten Elemente zeigt Bild 2.1.



2.1

Die Geräteadresse wird in jedem Interface decodiert. Hat ein Gerät seine eigene Adresse erkannt, erzeugt es ein Enablesignal, das, zusammen mit den Kontrollsignalen STROBE, das Gaten der Daten vom oder zum Datenbus ermöglicht. Jedes Interface benutzt hochohmige Eingangsschaltungen und niederohmige Ausgangsschaltungen, letztere meist als WIRED OR verbunden. Fast alle Interface-Systeme verwenden TTL-Signalpegel. Die CPU überwacht den zeitlichen Ablauf (Timing) der Transfers, es muß darauf geachtet werden, daß alle Signale auf den Leitungen eingeschwungen sind, bevor sie gelesen werden.

Der DMA-Transfer ist komplizierter organisiert. Wenn eine solche Übertragung abgewickelt werden soll, muß das Interface die DMA-Kontrolle benutzen, die zunächst mit dem Prozessor die Prioritätsreihenfolge klärt und, nachdem der Transfer freigegeben ist, die Speicheradresse festlegt, an die oder von der der Transfer beginnen kann. Gehen die Daten in den Speicher, muß das Interface sie zur Übertragung bereitstellen, kommen die Daten vom Speicher, müssen sie vom Interface angenommen und in das Peripheriegerät weitergeleitet werden, dazu muß ein interner Registerverkehr stattfinden. Natürlich benutzt auch das DMA-Interface die Möglichkeiten des programmierten Transfers, da das Laden seiner Register mit Busadressen, mit Anzahl der zu übertragenden Datenworte sowie bestimmter Bits des Kontroll- und Statusregisters vom Programm her erfolgt. Das Interface muß auch in der Lage sein, durch Setzen eines bestimmten Bits im Kontrollregister dem Programm mitzuteilen, daß der Blocktransfer beendet ist.

2.2. Interrupts und Prioritäten

In vielen Prozeßrechnern ist in der CPU neben der Hardware, die den I/O-Datentransfer ausführt, auch diejenige, die die Interrupts aus einem oder mehreren Interfacesystemen annimmt und sie nach der Entscheidung der Prioritätslogik beantwortet und abwickelt. Zum Beispiel möchte das laufende Programm wissen, wann eine früher gestartete Operation im Interface beendet ist, damit eine neue Operation beginnen kann. Dazu setzt das Interface zu gegebener Zeit eine "Flagge", d.h. ein bestimmtes Bit, diese Aktion wird vom I/O-System registriert, worauf das laufende Programm unterbrochen wird und der Prozessor zu einem speziellen Teil des Programms, der Serviceroutine, dirigiert wird, in der enthalten ist, was mit dem Interrupt anzufangen ist. Dieser Vorgang wird Programm-Interrupt genannt, durch seine Einführung ist es unnötig, ständig alle Peripheriegeräte durch Programmroutinen zu überwachen.

Die Interrupteigenschaften von Prozeßrechnern reichen von sehr einfach bis aufwendig. Die einfachste Methode zur Unterruptbehandlung beginnt mit dem ODER'n aller Interfaceflaggen. Ist irgendeine Flagge gesetzt, wird das Programm unterbrochen, falls im Prozessor z.Zt. keine Operation mit höherer Priorität abläuft. Hat der Prozessor den Interrupt angenommen, muß das Programm alle Interruptflaggen einzeln prüfen, um festzustellen, wer den Interrupt verursacht hat. Diese Softwareprüfung wird polling, d.h. abzählen oder abstimmen, genannt.

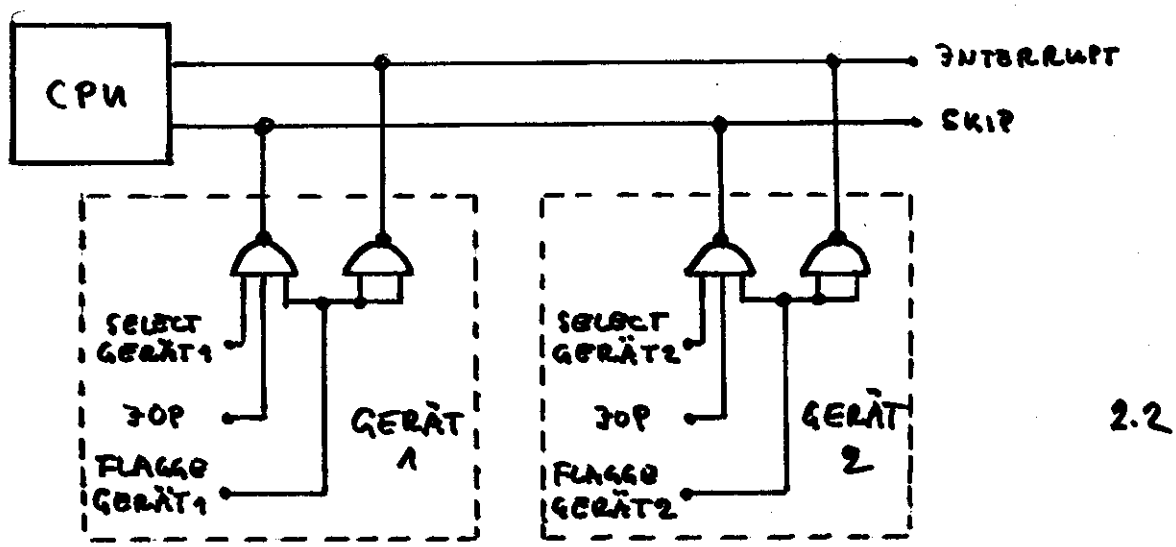
Diese umständliche Methode wird schneller, wenn jedes Gerät, das einen Interrupt auslösen kann, sich im I/O-System selbst identifiziert, indem man jeden dieser Geräte eine feste Nummer zuordnet.

Melden mehrere Geräte gleichzeitig eine Interruptanforderung, muß eine Prioritätsordnung entscheiden, in welcher Reihenfolge die Anforderungen behandelt werden. Einige Rechensysteme definieren die Priorität durch die Reihenfolge, in der die Interfacesysteme am I/O-Bus angeschlossen sind, d.h. je näher sie der CPU sind, desto höher die Priorität. Merkt die I/O-Hardware, daß ein oder mehrere Geräte einen Interrupt wünschen, sendet sie ein Signal, das Grant (Quittung) genannt wird, dan Bus entlang. Das erste Interface, das seine Flagge gesetzt hat, stoppt das Signal, d.h. verhindert die Weiterleitung zum nächsten Interface.

Eine differenziertere Prioritätsbehandlung benutzt mehrere Anforderungs- und Quittungsleitungen, denen jeweils eine spezifische Priorität, eine Prioritätsebene zugeordnet wird. Die Priorität ist dann nicht von der Lage am I/O-Bus abhängig, es sei denn, innerhalb einer Ebene sind mehrere Interfacesysteme angeschlossen.

Ist vom Betriebssystem her dem Prozessor eine bestimmte Priorität zugeordnet, erlaubt das I/O-System nur dann Interrupts, wenn die Anforderung von einem Gerät mit höherer Priorität kommt. Solche Anwendungen treten häufig in Real-time-Systemen auf, wo erst kritische Programme abgelaufen sein müssen, bevor relativ unkritische Programme starten dürfen.

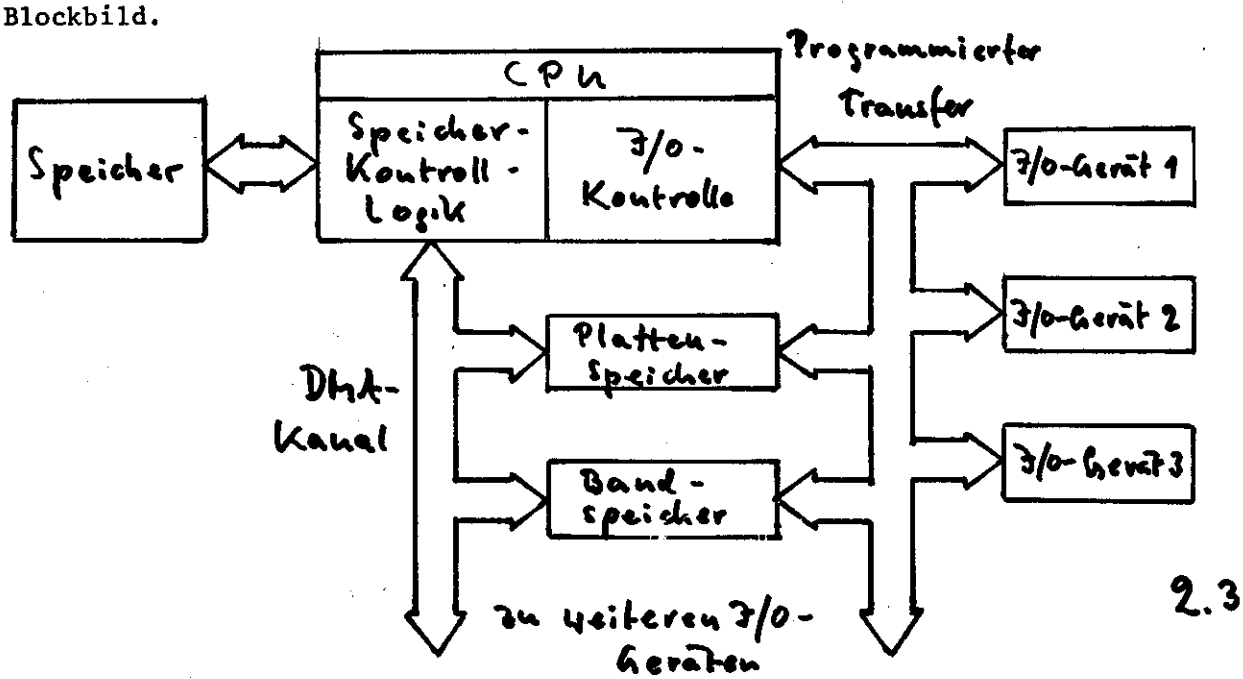
Bild 2.2 zeigt eine einfache Interrupt-Struktur.



2.2

Jedes Gerät treibt die Interruptleitung als Open Collector-Ausgang, d.h. das Signal auf dieser Leitung ist das ODER aller Flaggen. Die SKIP-Leitung, die gesetzt wird durch die Koinzidenz zwischen dem Geräte-SELECT und dem I/O-Puls veranlaßt das Programm, seinen nächsten Befehl zu streichen, um das Gerät zu identifizieren, das den Interrupt auslöst.

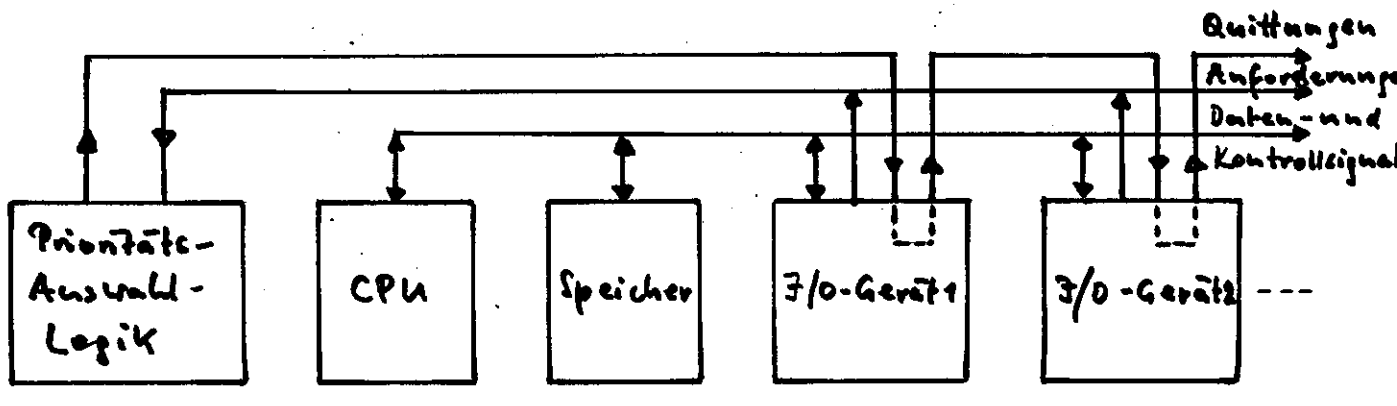
Rechnerhersteller bauen sehr unterschiedliche I/O-Systeme. In manchen Prozeßrechnern sind die Leitungen des programmierten Transfer, des DMA-Transfer sowie die Leitungen zum Speicher vollkommen getrennt. Bild 2.3 zeigt ein solches Blockbild.



2.3

In anderen Rechnern ist der programmierte und der DMA-Transfer kombiniert, bestimmte Signalleitungen, wie z.B. Daten und Adressen können für beide Übertragungsarten benutzt werden.

Wenn der Speicher und alle I/O-Geräte an einem gemeinsamen Bus arbeiten, wird das System wesentlich effizienter. Bild 2.4 zeigt diese Anordnung



2.4

Die Interfaceregister werden jetzt genau so wie Speicherregister behandelt, es gibt also keine speziellen I/O-Befehle.

Grundlage eines solchen Systems ist eine Master-Slave-Beziehung zwischen den am Bus angeschlossenen Geräten. Nur ein Gerät, entweder die CPU oder ein Interface, kann zur Zeit Master auf dem Bus werden, d.h. Datenübertragungen ausführen. Im Normalfall ist der Prozessor der Busmaster.

Ein Interface kann aus zwei Gründen Master werden, und zwar, um einen DMA-Transfer auszuführen oder einen Programminterrupt zu erzeugen. Es erhält die Buskontrolle, indem es auf einer geeigneten Leitung eine Anforderung schickt. Der Prozessor prüft periodisch alle Requestleitungen daraufhin durch. Sind eine oder mehrere Anforderungen eingetroffen, muß die Prioritätslogik entscheiden, ob eine Quittung und auf welcher Leitung geschickt wird. Nach deren Aussendung und Rückmeldung vom fordernden I/O-Gerät gibt der Prozessor die Kontrolle ab und das I/O-Gerät wird neuer Busmaster. Die gemeinsame Busleitung hat manche Vorteile, z.B. wird das Timing wesentlich vereinfacht, die Geschwindigkeit, mit der Übertragungen in die Geräteregister stattfinden, wird praktisch auf die Speicherzykluszeit gesteigert. Hinzu kommt, daß der Master einen Gerätefehler relativ leicht entdecken kann, falls der Slave kein Synchronisationssignal gesendet hat.

Dieses Konzept des gemeinsamen Bus wird UNIBUS genannt, es wird in der PDP11-Familie benutzt und in Abschnitt 5 ausführlich beschrieben.

3. Busleitungen

Der Störabstand der meisten digitalen Schaltkreise ist so groß, daß ohne Schwierigkeiten digitale Daten über 10-20 cm Draht direkt übertragen werden können. Der Transport fehlerfreier Daten über lange Entfernungen in gestörten Räumen erfordert jedoch spezielle Leitungstreiber und Empfänger sowie eine sorgfältige Wahl der Übertragungsleitung; ein Problem, das immer wieder auftritt, wenn Rechner und Peripheriegeräte weit auseinander stehen.

Mit ihrem Wellenwiderstand abgeschlossene einseitig geerdete Leitungen haben den Vorteil, daß die Signalreflektionen praktisch beseitigt werden und hohe Datenraten übertragen werden können. Ein Zweidrahtleitungssystem dagegen kann die Störungen wesentlich besser unterdrücken, einmal, weil sie schon primär im Gleichtakt sind, zum anderen werden sie durch einen Empfänger mit Differentialeingang noch stark reduziert.

Abhängig von der Länge der Leitung bestimmen auch die Kosten die Wahl des Leitungstyps. Bei kurzen Entfernungen kann man sich die relativ teuren abgeschirmten Koaxialkabel leisten, bei großen Längen wird man auf die billigeren Zweidrahtsysteme ausweichen, die als Paralleldraht- oder Twistedpair-Leitungen benutzt werden. Die Kosten können weiter durch ein "Party-Line"-System gesenkt werden, in denen mehrere Treiber und Empfänger die gleiche Leitung benutzen.

Bei der Entscheidung für eines dieser Systeme müssen die folgenden Kabeleigenschaften berücksichtigt werden:

- Verzögerungszeit pro Einheitslänge
- Kabeldämpfung
- Übersprechen
- Reflektionen, die aus Fehlanpassungen kommen, verursacht durch Stecker, Widerstände und Kabeltoleranzen.

Unterschiedliche Verzögerungszeiten in parallelen Kabeln verursachen Fehler beim Timing der Übertragungen. Bei langen Leitungen wird die Gesamtübertragungszeit beeinflußt, wenn man auf Rückmeldungen warten muß.

Die Dämpfung ist frequenzabhängig, sie erscheint als Ansteigen der Impedanz mit zunehmender Frequenz. Dabei wird die Anstiegszeit der Signale länger, die Signalamplitude sinkt.

Das Übersprechen entsteht durch die ungewollte Kopplung des Signals eines Draht auf einen benachbarten. Durch einen gekoppelten Puls entsteht ein Unterschwingen, das den Störabstand verringert. Gute Erde bzw. Abschirmung der Kabel ist die beste Gegenmaßnahme. Bei Zweidrahtleitungen ist dieses Problem praktisch vernachlässigbar.

Reflektionen verringern ebenfalls den Störabstand. Man muß im einzelnen Fall untersuchen, wie konstant man die Impedanz eines Kabels über die gesamte Länge halten kann.

3.1 Koaxialkabel

Koaxialkabel sind zur Übertragung hochfrequenter Signale geeignet. Ihre normalerweise gut definierte Impedanz ermöglicht einen guten Abschluß, d.h. die Reflektionen können gering gehalten werden. Die Abschirmung dämpft das Überspechen stark.

Die für die Übertragung benötigte Bandbreite ergibt sich aus

$$f = \frac{0.35}{T_R} \qquad [\text{MHz, wenn } T_R \text{ in } \mu\text{sec}] \ .$$

Für eine Anstiegszeit von 10 nsec benötigt man also ein Kabel mit 35 MHz-Bandbreite. Hieraus, sowie aus der zulässigen Dämpfung des Signals (gegeben durch den Störabstand) erhält man die mögliche Kabellänge.

Die Abhängigkeit der Dämpfung von der Frequenz für einige typische Koaxialkabels ist in der folgenden Tabelle 2.1 angegeben.

Tabelle 3.1

Dämpfung in dB pro 30 m

MHz	RG 59/U	RG 58/U	RG 188 A/U
1	0.2	0.4	0.6
3	0.4	0.7	1.2
10	0.9	1.5	2.3
30	1.8	2.7	4
100	3.6	5.4	8
300	7	10.5	15

Normale TTL-Schaltungen haben einen Störabstand von 0.4 V, die Bandbreite der Kabel sollte etwa 30 MHz betragen. Daraus folgt z.B. für das RG 58/U-Kabel eine Dämpfung von 2.7 dB/30 m. Der zulässige Verlust der TTL-Schaltung wäre

$$\text{Verlust (max)} = 20 \log \left(\frac{2.4V}{2V}\right) = 1.6 \text{ dB},$$

die maximale Kabellänge wird dann

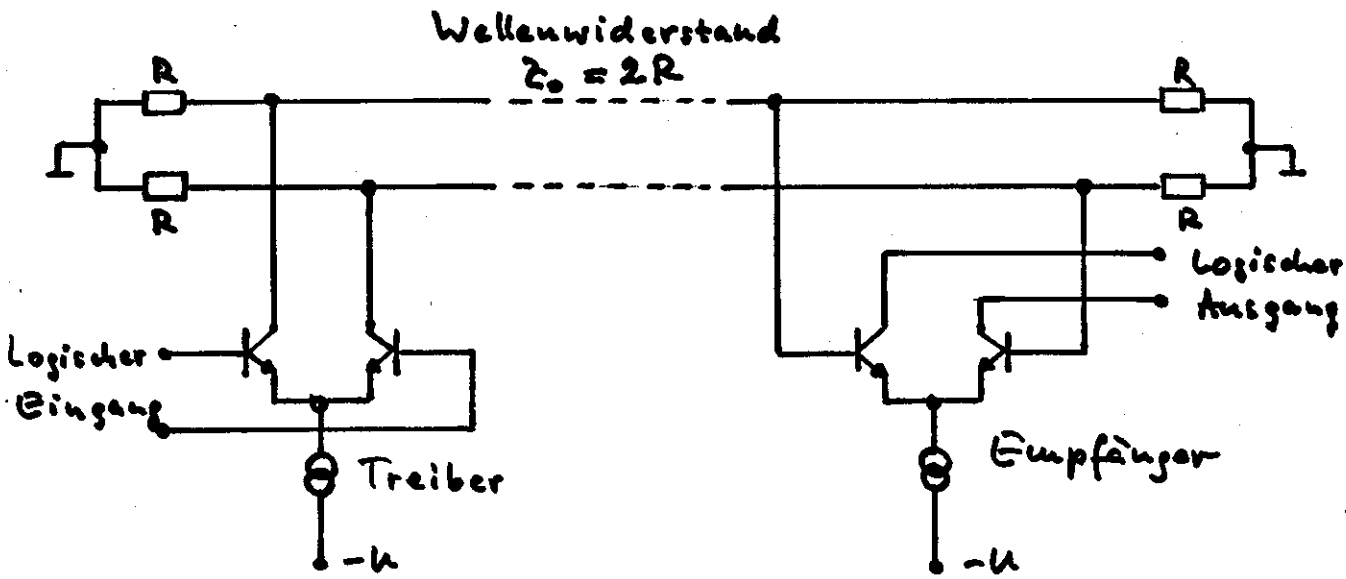
$$\text{Kabellänge (max)} = 30 \text{ m} \quad \frac{1.6dB}{2.7dB} \approx 18 \text{ m} \ .$$

Die Ausbreitungsgeschwindigkeit ist in Koaxialkabeln sehr hoch, sie beträgt typische 5 nsec/m.

3.2 Zweidraht-Twistedpair-Leitungen

Twistedpair-Kabel kann aus normalen isolierten Drähten hergestellt werden, das etwa einmal pro cm verdrillt wird. Je nach Drahtyp ergeben sich Wellenwiedertandswerte zwischen 100 und 115 Ω.

Twistedpair-Kabel wird differentiell angeschlossen, sowohl im Treiber als auch im Empfänger. Es bietet die günstigsten Störeigenschaften, da alle Störungen auf beiden Drähten gleichgerichtet sind. Da der Empfänger nur Differenzsignale annimmt, werden die Störungen unterdrückt. Eine typische Anschaltung zeigt Bild 3.1

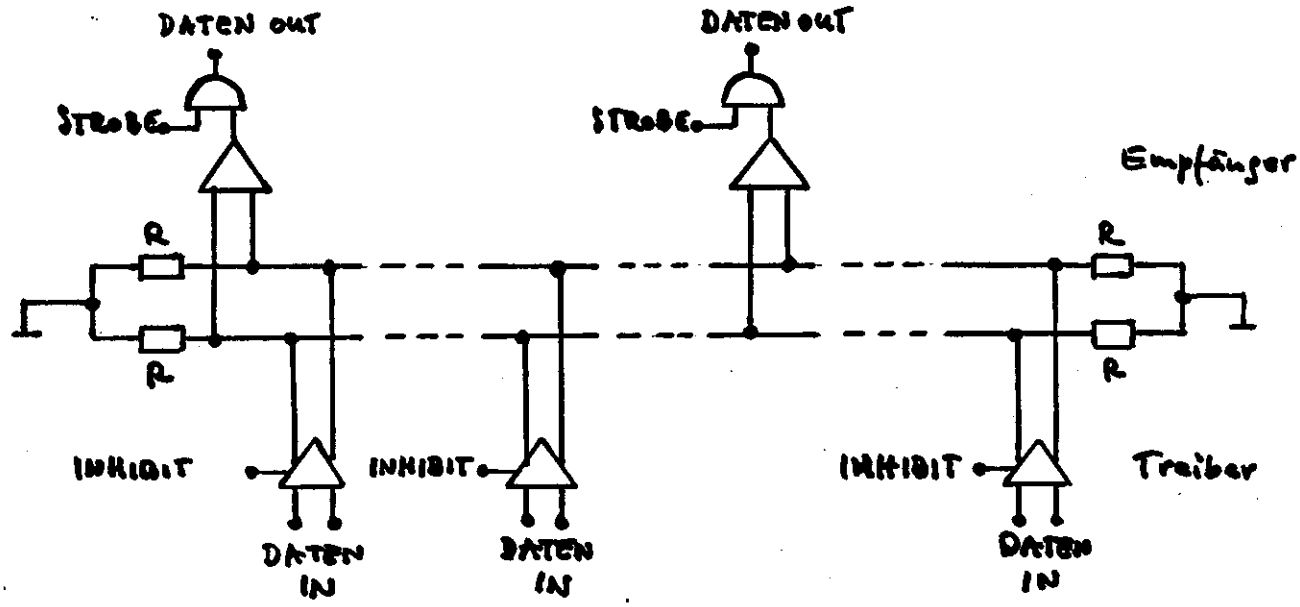


3.1 TWISTED PAIR - LINE

Eine Zweidraht-Twistedpair-Leitung ist an jedem Ende abgeschlossen. Die Widerstände erzeugen auch eine Vorspannung von einigen Hundert Millivolt. Der Treiber konvertiert die logische Eingangsspannung in eine Spannung, die den Stromschalter steuert; dadurch wird das Potentialgleichgewicht auf dem Kabel verschoben, der Empfänger detektiert die Differenz.

Treibt der Sender z.B. 600 mV (min) in das Kabel und hat der Empfänger eine Eingangsempfindleichkeit von 10 mV (max), darf der maximale Verlust auf der Leitung 36 dB betragen. Bei 30 MHz ist die typische Dämpfung im Twistedpair-Kabel 3 dB/30 m, d.h. die maximale Kabellänge darf in diesem Fall 360 m sein.

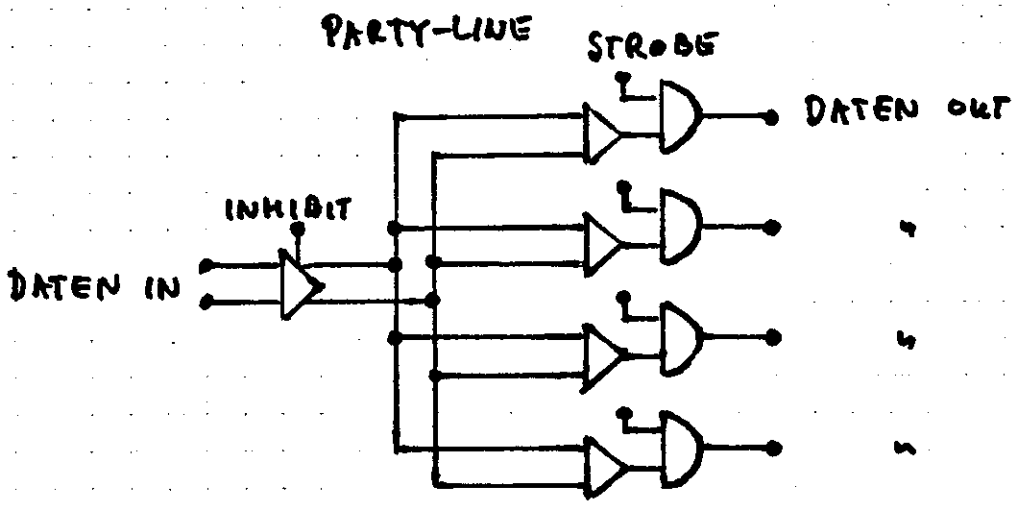
Bei vielen Treibertypen kann man verhindern, daß der Treiberstrom auf der Leitung erscheint, wenn kein Signal am Treibereingang liegt. Durch diesen INHIBIT-Mode kann man mehrere Treiber und Empfänger an einem Twistedpair-Kabel anschließen, man erhält eine Party-line. Bild 3.2 zeigt eine solche Ausschaltung.



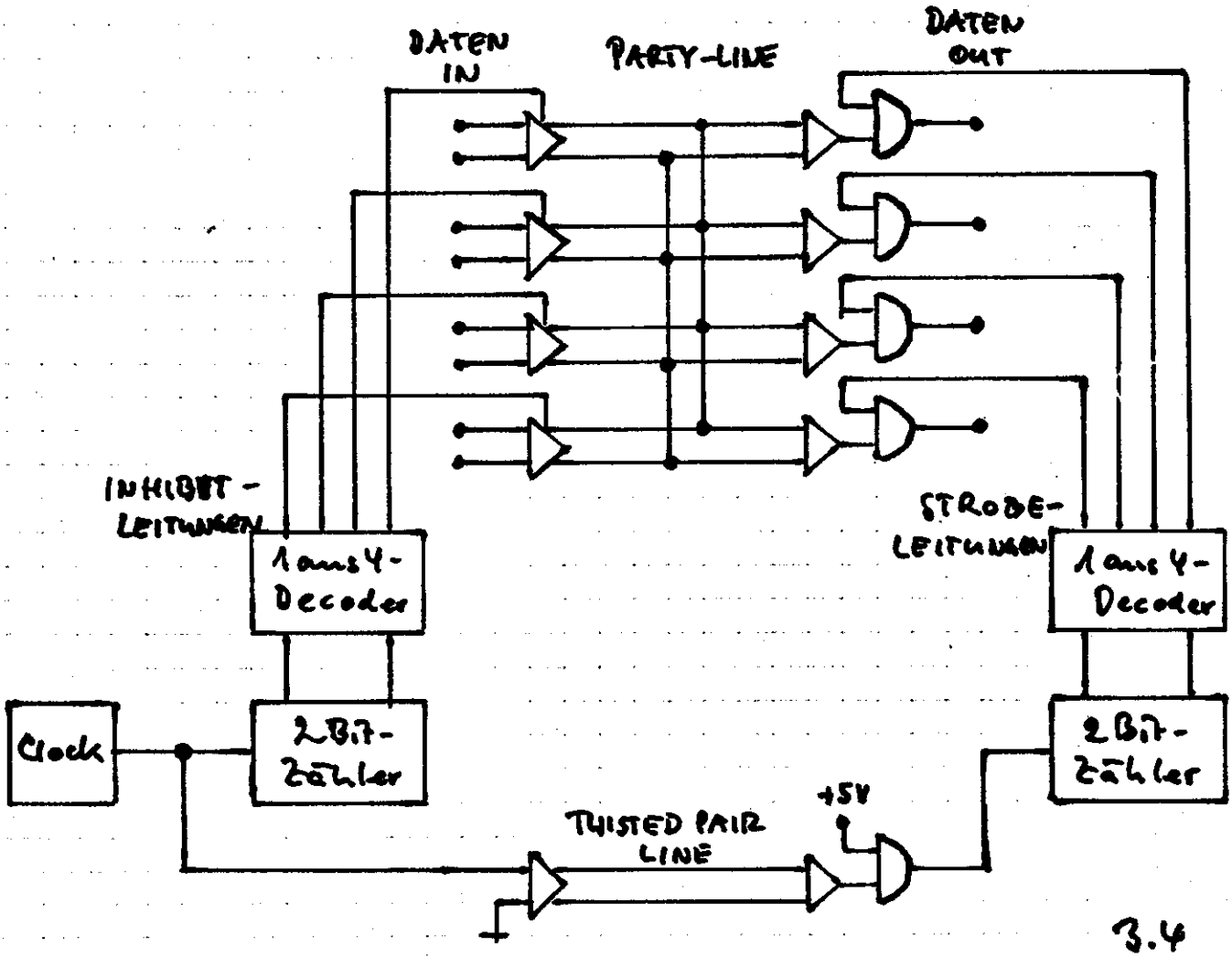
3.2 AUFBAU EINER PARTY-LINE (BUS)

Jeder selektierte Treiber kann entweder mit einem, mit mehreren oder mit allen Empfängern Daten austauschen, je nach dem, welche INHIBIT-Leitung im Treiber und welche STROBE-Freigabe im Empfänger gesteuert wird.

Die Party-line, allgemein auch Datenbus genannt, kann auf vielerlei Weise angeschlossen werden. Bild 3.3 zeigt, wie ein Treiber über eine Party-line vier Empfänger ansteuert, Bild 3.4 wie mehrere Treiber mehrere Empfänger steuerte, wobei eine Clock dafür sorgt, daß die verschiedenen Datenquellen nacheinander aufgerufen werden und ihre Daten über die Busleitungen an die mit gleicher Clock gesteuerten Empfänger gehen. Die Daten auf dem Bus können im Zeit-Multiplex-Verfahren übertragen werden.


PARTY-LINE
STROBE
DATEN OUT
INHIBIT
DATEN IN
4
4
4

3.3


DATEN IN
PARTY-LINE
DATEN OUT
INHIBIT-LEITUNGEN
STROBE-LEITUNGEN
1 aus 4-Decoder
1 aus 4-Decoder
Clock
2 Bit-Zähler
2 Bit-Zähler
+5V
TWISTED PAIR LINE

3.4

4. Kurzbeschreibung der PDP-11 Familie

Die PDP-11 Familie umfaßt mehrere Prozessoren, eine große Anzahl von Peripheriegeräten sowie eine umfangreiche Software. Die Rechner der Familie sind in der Architektur gleichartig und hard- bzw. softwarekompatibel.

PDP-11 Rechner haben allgemein eine Wortlänge von 16 Bit, die Prozessoren arbeiten mit variabler Wortlänge, sie können direkt 32 K-Worte je 16 Bits oder 64 K-Bytes je 8 Bit adressieren, über eine spezielle 18 Bit-Adressierung (im 11/45-Rechner) sind 128 K-Worte bzw. 256 K-Bytes anwählbar. Alle Verbindungen zwischen den Systemkomponenten gehen über einen Bus, den UNIBUS. Zur Standardausrüstung der 11/05, 11/15 und 11/20 Rechner gehören 8 allgemeine Register im Prozessor, die 11/45-Ausführung hat zwei Sätze davon, also 16 Register, die als Akkumulatoren, Indexregister oder Pointer (Adreßzeiger) benutzt werden können. Alle Rechner haben ein Interruptsystem mit mehreren Prioritätsebenen.

4.1. UNIBUS

Der UNIBUS verbindet den Prozessor, den Speicher sowie alle Peripheriegeräte; er überträgt Adressen, Daten und Kontrollsignale.

Die Art des Datenaustausches ist für alle Geräte, die am Bus angeschlossen sind, gleich; jedes Gerät, Speicherplätze, Prozessorregister und Register in den I/O-Geräten eingeschlossen, erhält eine Busadresse. Z.B. sind die Register der I/O-Geräte in den letzten 4 K-Worten der Adressenliste untergebracht. Diese Adressierung bedeutet, daß alle Register genau so flexibel behandelt werden können wie Speicherplätze. Die meisten UNIBUS-Leitungen sind bidirektional ausgeführt, d.h. daß alle Register in den I/O-Geräten sowohl für Eingangs- wie für Ausgangsoperationen benutzt werden können.

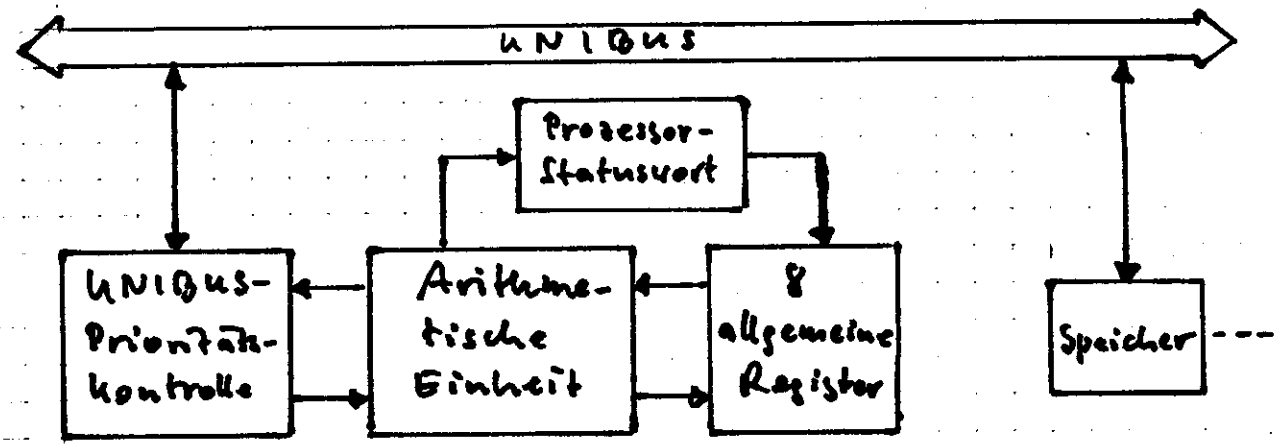
Der Busverkehr wird nach dem Master-Slave-Prinzip abgewickelt, d.h. ein Gerät, das die Buskontrolle besitzt, also der Busmaster, führt einen Datenverkehr mit einem anderen Gerät am Bus aus, dem Slave. Der Slave muß entweder Daten vom Master empfangen oder an ihn senden. Dieser Verkehr wird als Interlock- oder Handshake-Verfahren abgewickelt, d.h. jedes Kontrollsignal, das der Master aussendet, muß vom Slave beantwortet werden. Dadurch wird der Verkehr unabhängig von der physikalischen Buslänge und jedes Gerät kann mit seiner eigenen Geschwindigkeit arbeiten.

Um Master zu werden, muß das Gerät die Kontrolle anfordern. Da aber gleichzeitig mehrere Geräte diese Absicht haben können und die derzeitige Kontrolle von einem anderen Gerät ausgeübt wird, muß es eine Prioritätskontrolle geben.

4.2. Zentralprozessor CPU

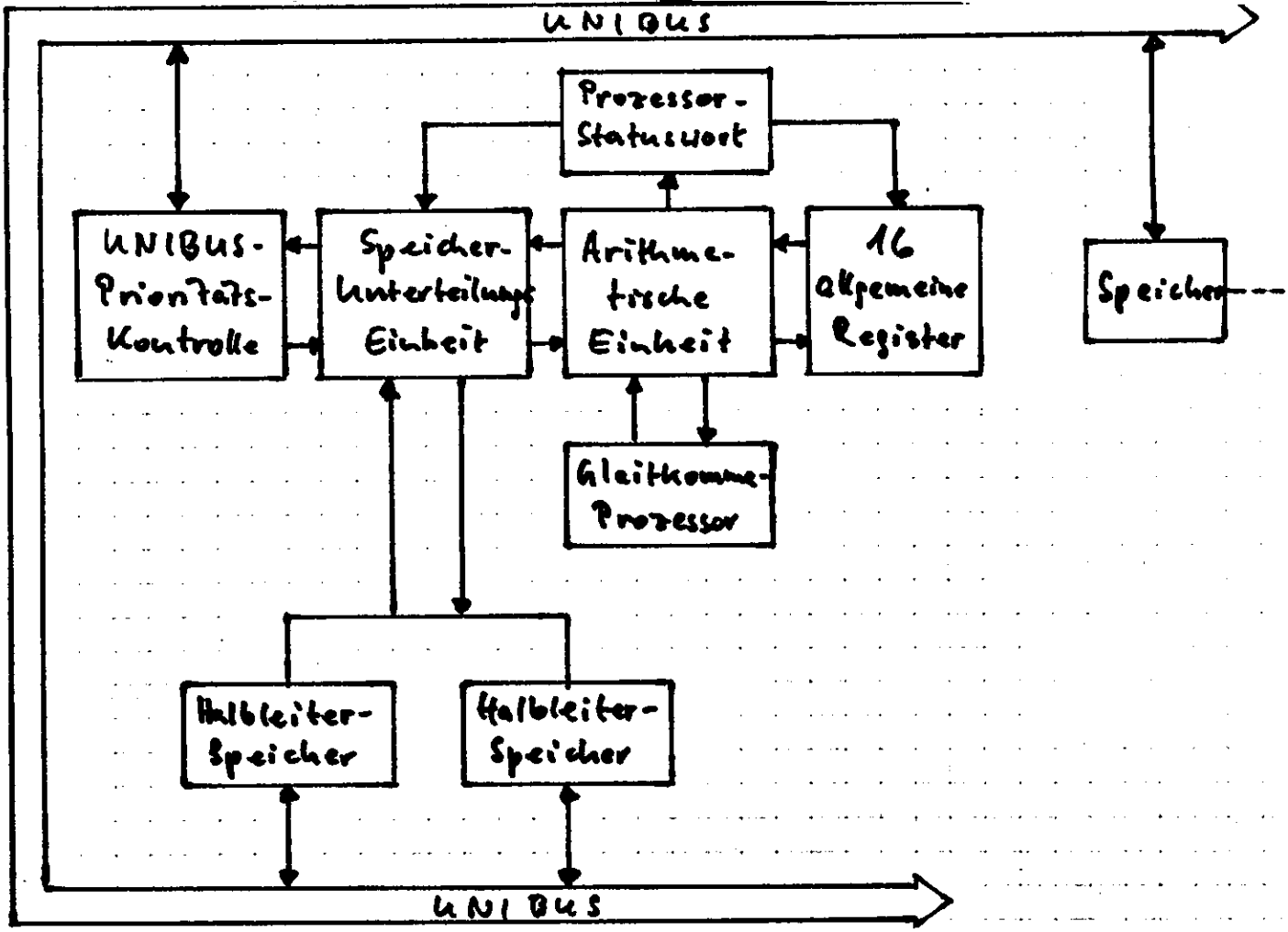
Der Zentralprozessor der PDP 11/05, 11/15 und 11/20 ist in 3 Funktionsblöcke unterteilt, die in Bild 4.1 dargestellt sind:

- die allgemeinen Register
- die arithmetische Einheit
- die UNIBUS- und Prioritätskontrolle



4.1 PROZESSOR DER PDP 11/05, 11/15, 11/20

Bei der PDP-11/45 enthält die Festkomma-Arithmetik eine hardwareverdrahtete Multiplikation und Division. In ihrer CPU kann zusätzlich ein Gleitkommaprozessor und eine Speicherunterteilungs-Einheit vorhanden sein, die den Speicherraum von 28 K auf 124 K Worte ausdehnen sowie eine wirksame Unterteilung für Multiprogramming ermöglicht. Die Einheit kontrolliert entweder 1 oder 2 sehr schnelle Halbleiterspeicher mit einer Zykluszeit von 300 (bipolar) bzw. 450 (MOS) nsec und 4 oder 8 K (bipolar) bzw. 16 oder 32 K (MOS) Speicherplätzen. Bild 4.2 zeigt den Aufbau dieser CPU.



4.2 Prozessor der PDP 11/45

Der Befehlsumfang umfaßt über 400 hardwareverdrahtete Befehle, deren Flexibilität durch die Benutzung der allgemeinen Register sehr hoch ist. Es gibt nicht die übliche Einteilung der Befehle in drei Klassen, nämlich speicherbezogene, arithmetische und I/O-Befehle, sondern alle Operationen werden mit einem Befehlssatz ausgeführt. Da die I/O-Geräteregister den Speicherplätzen gleichgestellt sind, kann ihr Inhalt von der CPU direkt getestet oder verändert werden, ohne ihn erst in den Speicher zu transportieren. Man kann mit den Registern direkt logische oder arithmetische Operationen ausführen, dadurch fallen spezielle I/O-Befehle weg. Alle Instruktionen sind ausführlich im "Processor-Handbook PDP-11" erläutert.

Die in Abschnitt 4.1. erwähnte Prioritätskontrolle ist ein automatisches Mehrebenensystem, in jeder Ebene können beliebig viele I/O-Geräte angschlossen werden. Jedes dieser Geräte hat einen Hardware-Pointer für das ihm zustehende Speicher-Doppelwort, eines der Worte weist auf die Service-Routine hin, das andere enthält die Information für den neuen Prozessorstatus. Diese Art der Identifikation macht das Durchsuchen aller möglichen Geräte, die den Interrupt ausgelöst haben

können, unnötig. Die Interruptpriorität des Gerätes und die der Service-Routine sind unabhängig voneinander. Dadurch kann, besonders in Real-time-Anwendungen, die Priorität der Service-Routine dynamisch dem Problem angepaßt werden.

Dieses Interruptsystem macht es möglich, daß der Prozessor kontinuierlich seine eigene Priorität mit der des den Interrupt auslösenden Gerätes vergleicht und dem Gerät den Interrupt gewährt, das den höchsten Prioritätslevel oberhalb des Prozessors besitzt. Ein Gerät, dessen Interrupt akzeptiert wurde, kann durch einen Interrupt mit höherem Level wieder unterbrochen werden. Die Interruptbehandlung wird ausführlich in Abschnitt 5.2.3. erläutert.

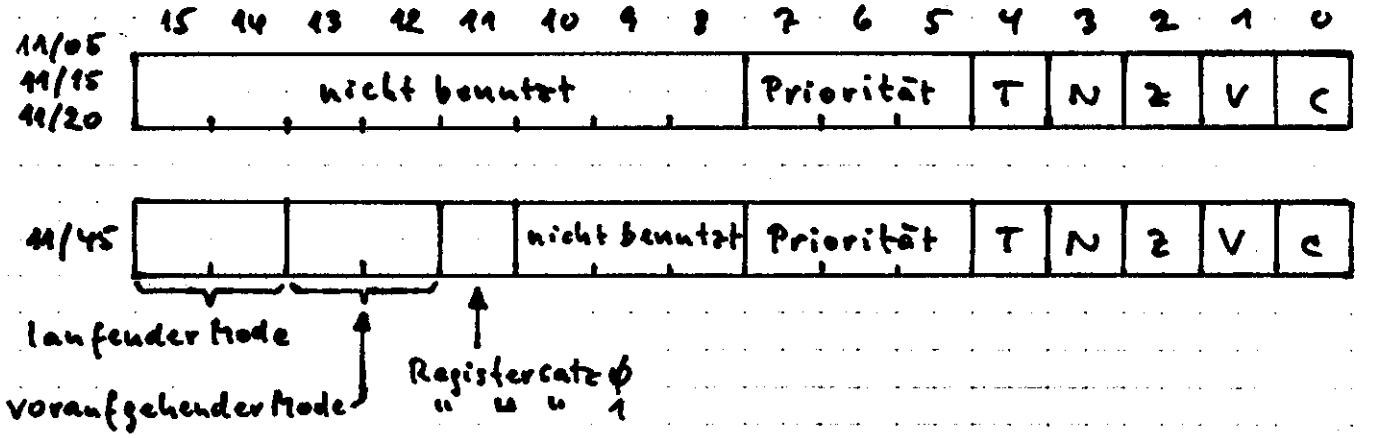
Alle PDP-11 Rechner sind für direkten Datenzugriff (DMA) ausgerüstet. Geräte, die DMA-Transfer ausführen wollen, erhalten die höchste Priorität, so daß direkte Datenübertragungen von und zum Speicher mit voller Speicherzyklusgeschwindigkeit ausgeführt werden können.

Innerhalb des Speichers können temporäre Datenbereiche existieren, in denen die für ein bestimmtes Programm sehr häufig benutzten Daten enthalten sind. Solche Bereiche heißen Stacks. Erhält der Prozessor einen Interrupt, wird das Prozessor-Statuswort und der Inhalt des Programmzählers (Adresse des nächsten Befehls) in einem Stack abgelegt. Dann wird ein neues Statuswort aus dem Speicherbereich geholt, der für Interruptinstruktionen zuständig ist (Vektorbereich). Nach Ablauf des Interrupts wird das ursprüngliche Statuswort und der Inhalt des Programmzählers geholt und das unterbrochene Programm automatisch fortgesetzt.

Die PDP-11/05, 11/15 und 11/20 haben einen Satz von 8 allgemeinen Registern, von denen die Register R0, R1 ... R5 als Akkumulatoren, Indexregister und Pointer dienen, R6 als Hardware-Stackpointer und R7 als Programmzähler.
Der PDP-11/45 Prozessor kann wegen der Multi-Programmierung in 3 Modes arbeiten, und zwar in Kernel-Mode, als Supervisor und als User. Ist der Prozessor im Kernel-Model, bedeutet dies, daß das Programm den Rechner vollständig kontrolliert; in den beiden anderen Modes wird der Prozessor daran gehindert, bestimmte Befehle auszuführen, außerdem kann er zu den I/O-Geräten nicht mehr direkt zugreifen. Es gibt daher zwei Sätze von allgemeinen Registern R0, R1 ... R5, drei R6-Register als Stackpointer für die drei Modes und R7 als Programmzähler.

Das Prozessor-Statusregister enthält 16 Bits, von denen die Bits 0 bis 4 Ergebnisse der vorangegangenen Operationen enthalten, Bits 5 bis 7 die Priorität des Prozessors, die Bits 8 bis 15 werden bei den Rechnern 11/05, 11/15 und 11/20

nicht benutzt, bei dem 11/45 Rechner sind Bits 8 bis 10 ungenutzt, Bit 11 sagt, welche Gruppen von Registern gemeint sind, Bits 12 bis 13 geben den voraufgehenden Mode an, Bits 14 bis 15 den laufenden. Bild 4.3 zeigt die beiden Statusworte.



4.3 PROZESSOR - STATUSWORT

Die Bedeutung der arithmetischen Bits, wenn sie gesetzt sind, ist:

- C, wenn die Operation einen Übertrag im MSB hat,
- V, wenn die Operation einen arithmetischen Overflow hat,
- Z, wenn das Ergebnis Null ist,
- N, wenn das Ergebnis negativ ist,
- T ist ein Prozessor-Trap-Bit, es kann vom Programm gesetzt oder gecleart werden. Es gibt eine Reihe von Fehlern und Programmbedingungen, die den Prozessor zu festverdrahteten Speicherplätzen führen, in denen spezielle Instruktionen stehen. Die Trapbehandlung ist im "Processor Handbook" ausführlich beschrieben.

Die Leistung der PDP-11 Familie wird entscheidend durch die Art der Adressierung beeinflußt. Es gibt mehrere Möglichkeiten, sie auszuführen:

- sequentielle Adressierung,
- Index-Adressierung,
- volle 16 Bit-Wort-Adressierung,
- 8 Bit-Byte-Adressierung,
- Stack-Adressierung.

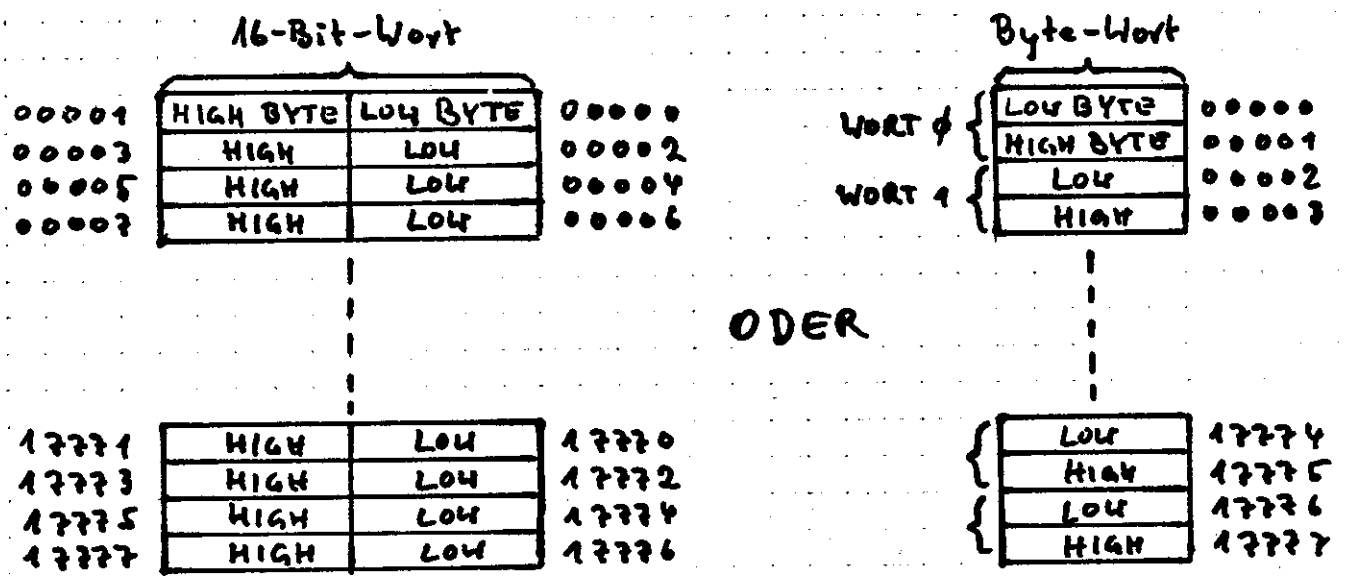
Durch die variable Länge des Befehlsformats wird nur eine minimale Bitzahl für jede Adressierung benötigt.

4.3. Speicher

Für die PDP-11 Familie gibt es mehrere Kernspeichertypen mit Zugriffszeiten zwischen 350 und 500 nsec, Zykluszeiten von 850 bis 1200 nsec. Sie werden als Blöcke von 4 K Worten angeboten, entweder bis 24 K in den 11/05, 11/15, 11/2o Rechnern oder bis 124 K in der 11/45. In dem letztgenannten Rechner sind außerdem die bereits beschriebenen Halbleiterspeicher enthalten. Ein I/O-Gerät kann sowohl den Kernspeicher als auch, im Zugriff der CPU, die Halbleiterspeicher lesen und beschreiben. Auch der Prozessor kann beide Speichertypen benutzen.

Normalerweise werden die Speicherzellen innerhalb eines Blocks fortlaufend nummeriert. Jeder Zugriff zu einer neuen aufsteigenden Adresse benötigt einen kompletten Speicherzyklus. Um diesen Ablauf zu verkürzen, kann man fortlaufende Adressen immer abwechselnd in zwei Speicherblöcken unterbringen, so daß der Schreibezyklus in einem Speicherblock mit dem Lesezyklus im anderen zusammenfällt. Diese Methode wird Interleaving (Durchschießen) genannt. Für den Benutzer sehen die zwei 4 K-Blöcke wie ein kontinuierlicher 8K-Block aus. Je nach Speichertyp gewinnt man 25-30 % effektiver Zykluszeit.

Die Speicher können entweder als 16 Bit-Worte oder als doppelt so viel 8 Bit-Bytes organisiert sein. Aufeinanderfolgende Worte erhalten so geradezahlige Adressen. Bild 4.4 zeigt diese beiden Darstellungsmöglichkeiten:

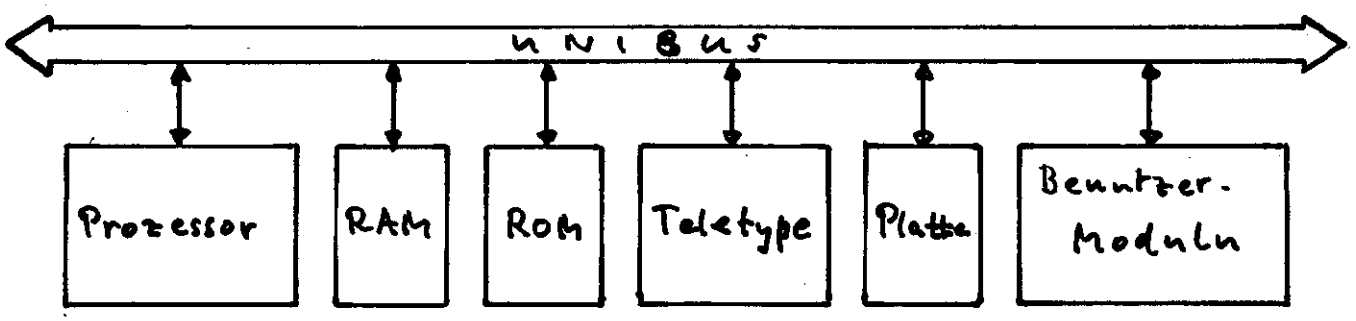


4.4 SPEICHERORGANISATION

5. UNIBUS-Interface

5.1 Einleitung, Prinzip des Einzel-Bus-Systems

Der UNIBUS ist ein Bus-System, das im PDP-11 System den Informationsaustausch zwischen dem Prozessor, dem Speicher sowie allen peripheren Geräten ausführt. Es ist ein Einzel-Bus-System, an das alle Rechnerkomponenten gleichberechtigt angeschlossen sind. Über den UNIBUS werden Adressen, Daten und Kontrollinformationen ausgetauscht. Bild 5.1 zeigt das Blockdiagramm des PDP-11 System und des UNIBUS.



5.1 DER UNIBUS

Die Operationen, die den Informationsaustausch begleiten, sind für alle Geräte, die am UNIBUS angeschlossen sind, gleich. Der Prozessor benutzt die gleichen Signale zum Verkehr mit dem Speicher wie zu allen anderen Peripheriegeräten. Analoges gilt für die übrigen Geräte.

Die meisten UNIBUS-Leitungen sind bidirektional, d.h. die Eingangsleitungen können auch als Ausgangsleitungen betrieben werden. Das bedeutet, daß das gleiche Register in irgendeinem Peripheriegerät sowohl für Eingangs- als auch für Ausgangsoperationen benutzt werden kann. Alle Befehle, die die Daten im Speicher betreffen, können ebenso auf Register in irgendwelchen Peripheriegeräten angewandt werden.

Der Austausch der Informationen zwischen zwei Geräten am Bus wird nach dem Master-Slave-Prinzip abgewickelt. Das bedeutet, daß während einer Bus-Operation ein Gerät, der jeweilige Bus-Master die Kontrolle über den Bus hat. In einem typischen Beispiel arbeitet der Prozessor als Bus-Master beim Übertragen von Daten an den Speicher, der in dieser Operation als Slave arbeitet.

Da der UNIBUS für alle I/O-Geräte und den Prozessor die einzige Verbindung ist, muß eine Prioritätsregelung die Bus-Kontrolle handhaben. Jedes Gerät am UNIBUS, das Bus-Master werden kann, muß also eine definierte Priorität besitzen. Sind zwei Geräte am Bus mit gleicher Priorität angeschlossen und versuchen beide gleichzeitig, Bus-Master zu werden, so hat das Gerät Vorrang, das elektrisch dem Bus am nächsten ist. Der Informationsaustausch zwischen Master und Slave erfolgt nach dem Handshake-Verfahren, d.h. jedes Kontrollsignal, das der Master aussendet, muß vom Slave mit einer Antwort (Response) versehen werden, um den Transfer ausführen zu können. Der Datenverkehr wird damit unabhängig von der physikalischen Länge des Bus. Die maximale Übertragungsrate auf dem Bus ist ein 16Bit-Wort alle 400 nsec oder 2.5 Millionen 16Bit-Worte pro Sekunde.

Da, wie schon erwähnt, Register in peripheren Geräten ähnlich wie Speicherzellen adressierbar sind, können alle PDP-11-Befehle, die die Zellen adressieren, auch I/O-Befehle werden, d.h. die Geräteregister können alle arithmetischen Möglichkeiten des Prozessors in Anspruch nehmen. Auch Kontrollfunktionen können an solche Register übertragen werden, so daß individuelle Bits im Register selektive Kontrolloperationen ausführen können. Das Gleiche gilt für Statusbedingungen.

Ein am Bus angeschlossenes externes Gerät, also nicht der Prozessor, benutzt, wenn es Bus-Master wird, den Bus im allgemeinen aus zwei Gründen:

- um einen Nicht-Prozessor-Transfer von Daten direkt zum oder vom Speicher zu machen,
- um das laufende Programm zu unterbrechen (Interrupt) und das Programm im Prozessor zu veranlassen, zu einer Adresse zu springen, wo die Interrupt-Routine abgelegt ist.

Die Anforderung (Request) und Zuteilung (Grant) von Bus-Master-Zyklen erfolgt parallel mit dem Datentransfer über einige hiervon unabhängige Busleitungen. Während ein Gerät den Bus benutzt, wird so bereits die nächste Anforderung auf Priorität geprüft und der nächste Benutzer festgelegt. Dadurch können sukzessive Datenübertragungen von verschiedenen Mastergeräten mit voller UNIBUS-geschwindigkeit abgewickelt werden.

Wenn ein Gerät versucht, Bus-Master zu werden, wird die Anforderung nach den Prioritätsebenen behandelt. Dabei ist folgendes zu beachten:

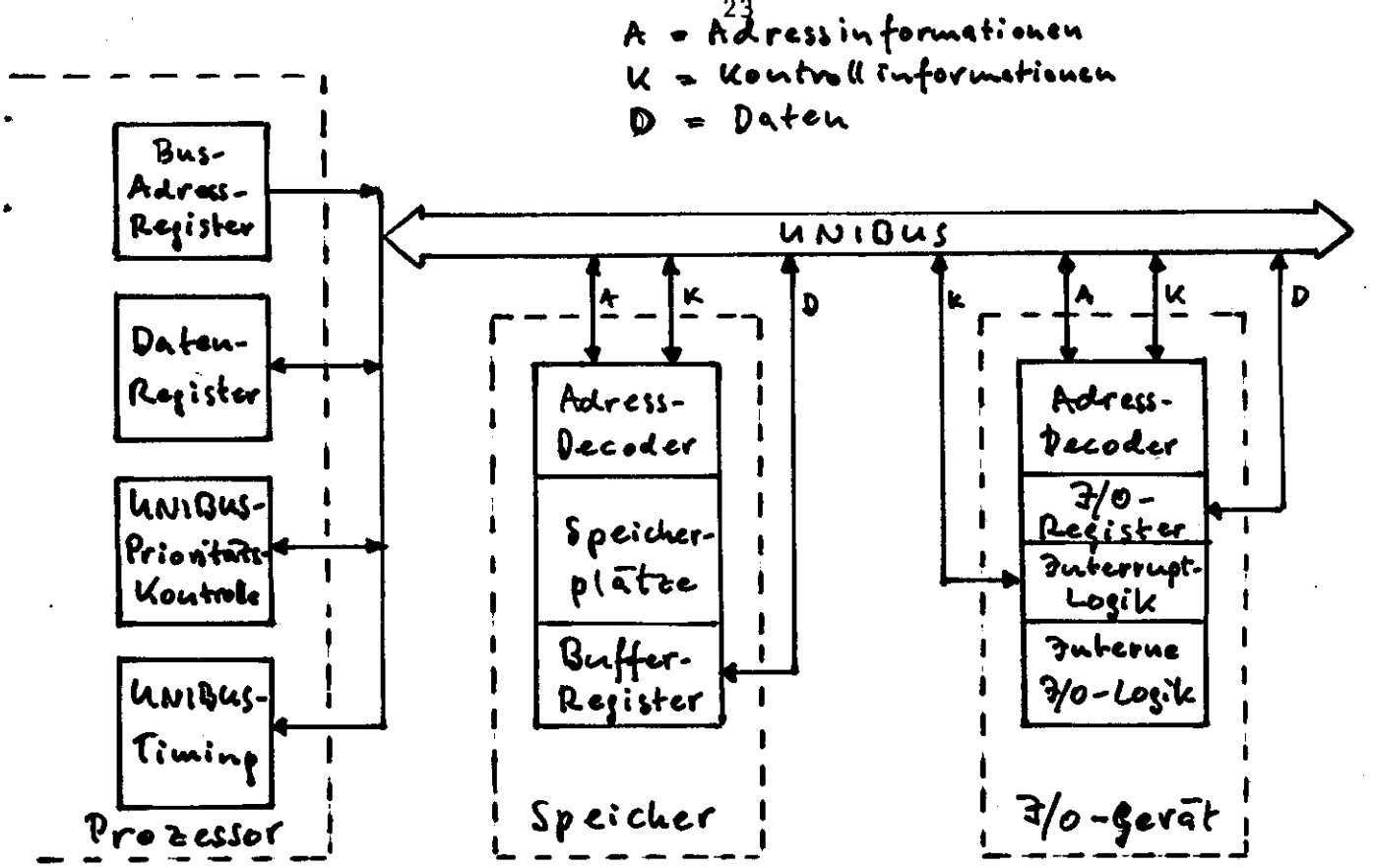
- Die Priorität des Prozessors wird durch das Programm auf eine von acht möglichen Ebenen festgelegt. Dies geschieht durch 3 Bits im Statusregister des Prozessors. Falls der Anforderer den gleichen oder niedrigeren Prioritätslevel als der Prozessor hat, hat letzterer Vorrang.
- Von externen Geräten können Bus-Anforderungen auf einer von fünf Anforderungsleitungen gemacht werden. Eine Nicht-Prozessor-Anforderung (NPRequest=NPR) hat die höchste Priorität. Anforderung 7 (BR7) hat die nächst höhere Priorität, Anforderung 4 (BR4) die niedrigste.
- Sind mehrere Geräte an die gleiche Bus-Anforderungsleitung angeschlossen, hat das Gerät die höhere Priorität, das elektrisch näher am Prozessor ist. An eine BR- oder NPR-Leitung können beliebig viele Geräte angeschlossen werden.

Direkter Datenverkehr zwischen zwei externen Geräten kann ohne Prozessor - Überwachung geschehen, man nennt sie daher NPR-Transfers. Geräte, die Bus-Master durch Anforderung auf den BR-Leitungen BR7, BR6, BR5 und BR4 werden, können alle Leistungen des Prozessors durch einen Programm-Interrupt benutzen. Der Prozessor unterbricht die laufende Aufgabe (task), die Interrupt-Serviceroutine wird gestartet. Nach Ablauf der Anforderung kehrt der Prozessor zur unterbrochenen task zurück. Interrupt-Anforderungen können nur ausgeführt werden, wenn die Bus-Kontrolle über die BR-Level erhalten wurde; ein NPR erzeugt keinen Programm-Interrupt.

5.2 UNIBUS-Operationen

5.2.1 UNIBUS-Signalleitungen

Der PDP-11-UNIBUS enthält Leitungen, an die alle Geräte, auch der Prozessor parallel angeschlossen sind. (vgl. Bild 5.2). Von den Leitungen sind 51 bidirektional und 5 unidirektional.



5.2 INTERFACE BLOCK DIAGRAMM PDP-11

5.2.1.1 Datentransferleitungen

Für den Datatransfer, d.h. für Operationen zwischen zwei Geräten, von denen das eine Bus-Master ist und den Transfer zu oder von einem Gerät kontrolliet, das als Slave arbeitet, werden 40 bidirektionale Leitungen benutzt. Der Prozessor ist immer dann Bus-Master, wenn kein anderes Gerät den Bus benutzt. Die Datentransfersignale sind in der Tabelle 5.1 zusammengestellt. Alle Signale auf dem UNIBUS werden in negativer Logik betrieben, d.h. die logische 1 wird durch L (Low), die logische 0 durch H (High) dargestellt.

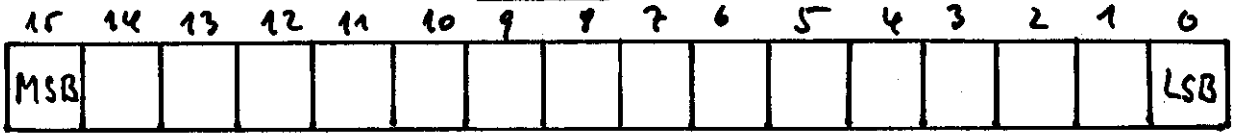
Tabelle 5.1 Datentransfersignale

Name	Mnemonic	Zahl der Leitungen
DATA	D < 15 : 00 >	16
ADDRESS	A < 17 : 00 >	18
CONTROL	C < 1 : 0 >	2
MASTER SYNC	MSYN	1
SLAVE SYNC	SSYN	1
PARITY BIT LOW	PA	1
PARITY BIT HIGH	PB	1
		40

Bei jedem Datentransfer werden Daten gesendet und empfangen, das Mastergerät erzeugt die Adresse des Slavegerätes sowie Kontroll- und Timing-Signale.

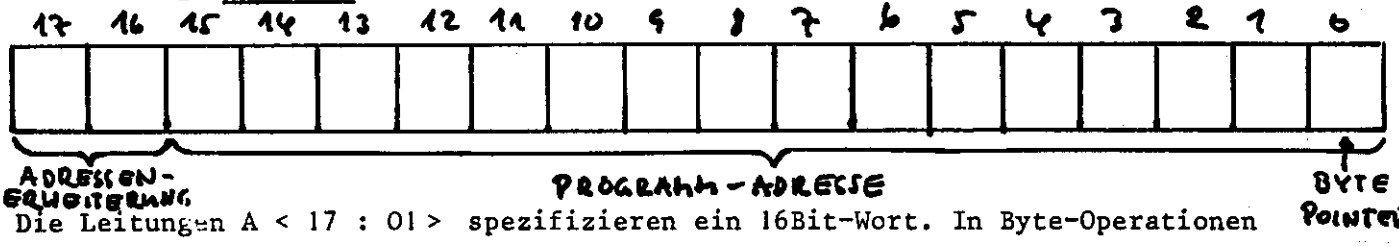
5.2.1.1.1 Datenleitungen (D < 15 : 00 >)

Die 16 Datenleitungen werden zum Transfer von Daten zwischen Master und Slave benutzt. Das Bitformat zeigt Bild 5.3



5.2.1.1.2 Adreßleitungen (A < 17 : 00 >)

Auf 18 Adreßleitungen kann der Master die Adresse des Slavegerätes wählen, d.h. eine individuelle Speicher- oder Geräteregisteradresse. Mit 18Bits können in der erweiterten PDP-11-Familie bis zu 262 144 Bytes adressiert werden. Das Bitformat zeigt Bild 5.4.



Die Leitungen A < 17 : 01 > spezifizieren ein 16Bit-Wort. In Byte-Operationen beschreibt A00 das jewertige Byte. Ist ein Wort durch die Adresse X festgelegt, wo X eine gerade Zahl sein muß, da Worte nur mit geraden Grenzen adressiert werden können, kann das Byte mit niedriger Ordnung durch X, das mit höherer Ordnung durch X+1 adressiert werden.

Vom Programm werden normalerweise nur 16Bits als Speicheradresse verwendet, nämlich A < 15 : 00>. Im Prozessor sind die Leitungen A17 und A16 für Adressen zwischen 160 000 und 177 777 vorgesehen, dazu gehört A15=A14=A13=1. In diesem Fall, der nur in den größeren Computern der PDP11-Familie vorkommt, wandelt der Prozessor die 16Bit-Adresse in eine volle 18Bit-Adresse um. Periphere Geräte heben normalerweise Adressen innerhalb des Adressen-Raumes 760 000 bis 777 777.

5.2.1.1.3 Kontrollsignale

Man benötigt drei Gruppen von Kontrollsignalen:

- Signale, die Datenübertragungsoperationen bestimmen,
- Signale, die Master und Slave synchronisieren und
- Signale, die Paritätsprüfungen ausführen.

Die Kontrolleitungen (C < 1 : 0 >) werden vom Master codiert, um den Slave zu

einer der folgenden vier Operationen zu veranlassen:

C1	CO	Operation
0	0	DATI - Data IN
0	1	DATIP - Data IN, Pause
1	0	DATO - Data OUT
1	1	DATOB -Data OUT, Byte

Der Bus-Master erzeugt ein Synchronisationssignal (MSYN), das dem Slave anzeigt, es sind Adressen- und Kontrollinformationen vorhanden. Der Slave antwortet mit einem SSYN-Signal.

Sind am UNIBUS Geräte angeschlossen, die einen Paritycheck ausführen, können sie dazu die beiden Paritätsbits PA und PB benutzen. PA gibt die Parität des niedrigwertigen Byte auf D < 07 : 00 > , PB die des höherwertigen auf D < 15 : 00 > an.

5.2.1.2 Übertragungsleitungen für Prioritätsverkehr

13 Leitungen des UNIBUS sind für die Anforderung- und Zuteilungssignale der verschiedenen Prioritätsebenen vorgesehen. 5 davon, nämlich BR < 7 : 4 > und NPR dienen der Anforderung, 5 weitere, nämlich BA < 7 : 4 > und NPG der Zuteilung. Sind mehrere Geräte gleicher Priorität an einer der Leitungen angeschlossen, läuft das Zuteilungssignal (Grant) durch die Kontrolle der elektrisch vor ihm liegenden Geräte hindurch bis zu dem Gerät, das die Anforderung geschickt hat. Dieses blockiert die Weiterleitung des Signals und wird Bus-Master.

3 weitere Leitungen, als SACK, BBSY und INTR bezeichnet, dienen der Kontrolle.
Alle 13 Leitungen werden noch einmal aufgeführt, es sind:

- Bus Requestleitungen (BR < 7 : 4 >). Diese 4 Leitungen werden von externen Geräten benutzt, um Bus-Master zu werden.
- Bus Grantleitungen (BG < 7 : 4 >). Diese 4 Leitungen sind die Antwort des Prozessor auf einen Bus Request. Sie werden nur am Ende eines ausgeführten Befehls erteilt und nur unter Beachtung der Priorität.
- Nicht-Prozessor-Request (NPR). Dieses Signal ist eine Busanforderung von einem externen Gerät an den Prozessor.
- Nicht-Prozessor-Grant (NPG). Dieses ist die Antwort des Prozessors auf einen NPR. Sie wird am Ende eines Bus-Zyklus gegeben.

- Selection Acknowledge (SACK). Diese Empfangsbestätigung sendet das Gerät aus, das eine Busanforderung angemeldet und eine Buszuteilung erhalten hat. Das Gerät erhält die Buskontrolle, wenn der Bus-Master-Vorgänger seine Busoperation beendet hat. Trifft nicht innerhalb von 10 µsec nach dem Bus Grantsignal die SACK-Quittung beim Prozessor ein, erzeugt dieser ein Zeit-Aussignal, der Grant wird automatisch gestrichen.
- Interrupt INTR. Dieses Signal erzeugt der Bus-Master, wenn er einen Programm-Interrupt vornehmen möchte.
- Bus Busy (BBSY). Um anzuzeigen, daß der Bus vom Master benutzt wird, gibt dieses das BBSY heraus.

5.2.1.3 Zusätzliche Kontroll-Signale

Es gibt 3 weitere Kontrolleitungen, die von allen Geräten benutzt werden können, dies sind:

- Initialization (INIT). Vom Prozessor wird dieses Signal erzeugt, wenn die START-Taste auf der Konsole gedrückt wird, wenn eine RESET-Befehl ausgeführt wird oder wenn die Stromversorgung ausgefallen war und wieder einsetzt.
- AC Line Low (AC LO). Dieses Signal zeigt an, daß die Wechselspannung beginnt, auszufallen. Es bleibt solange bestehen, bis die Spannung wieder steigend einen Schwellwert überschreitet.
- DC Line Low (DC LO). Dieses Signal zeigt an, daß die Gleichspannung unter die Spezifikationsgrenze fällt; es verschwindet, wenn die Spezifikationen wieder erreicht werden. Es wird durch den UNIBUS an alle Geräte geführt.

5.2.2 Datenübertragungsoperationen auf dem UNIBUS

Alle Aktivitäten auf den Bus sind asynchron und hängen vom Wechselspiel mit den Kontrollsignalen ab. Durch dan Handshake-Betrieb zwischen Master und Slave sind alle kritischen Zeitabhängigkeiten der Signale auf dem UNIBUS eliminiert, die Übertragungsgeschwindigkeit richtet sich nach den Anfragen und Antwortzeiten zwischen den beiden Partnern.

Die vier Transferarten auf dem Bus sind in Tabelle 5.3 beschrieben:

Tabelle 5.3

Datentransferarten auf dem Bus

Name	Mnemonic	C1	C0	Funktion	Octal Code
Data IN	DATI	0	0	Daten vom Slave zum Master	0
Data IN, pause	DATIP	0	1	In nichtzerstörungsfreien Leseeinrichtungen wird das Wiedereinschreiben verhindert. Pausen-FF wird gesetzt, er verhindert den Clear-Zyklus beim folgenden DATO(B). Es muß ein DATO oder DATOB folgen	1
Data OUT	DATO	1	0	Daten vom Master zum Slave	2
Dato OUT, Byte	DATOB	1	1	Daten vom Master in ein bestimmtes Byte im Slave. Für A00=1 auf D<15:08>, für A00=0 auf D<07:00>.	3

In DATI- und DATIP-Operationen holt der Master sich die Daten von einem durch die Adresse A < 17:00> spezifizierten Slave. Dabei unterscheidet der Master durch Festlegung von A00, ob er auf D < 15:08 > oder D < 07:00 > ein bestimmtes Byte oder auf D < 15:00 > ein ganzes 16Bit-Wort haben will. DATI und DATIP sind identisch in zerstörenfreien Leseeinrichtungen, wie z.B. Flip-Flop-Speichern. In nichtzerstörungsfreien Lesegeräten, wie z.B. Kernspeichern, wird durch DATIP der Slave informiert, daß dies der erste Teil eines IN-MODIFIZIEREN-OUT-Zyklus ist. Daher muß in solchen Operationen auch ein Data-OUT-Zyklus folgen, der Master muß bis zu dessen Ende die Bus-Kontrolle behalten.

DATO- und DATOB-Operationen übertragen Daten vom Master zum Slave. Bei DATO wird die Slaveadresse durch A<17:01> beschrieben, d.h. der Slave ignoriert A00, die Daten vom Master kommen auf D<15:00>. Bei DATOB wird die Adresse durch A<17:00> festgelegt und A00 entscheidet, welches Byte übertragen wird.

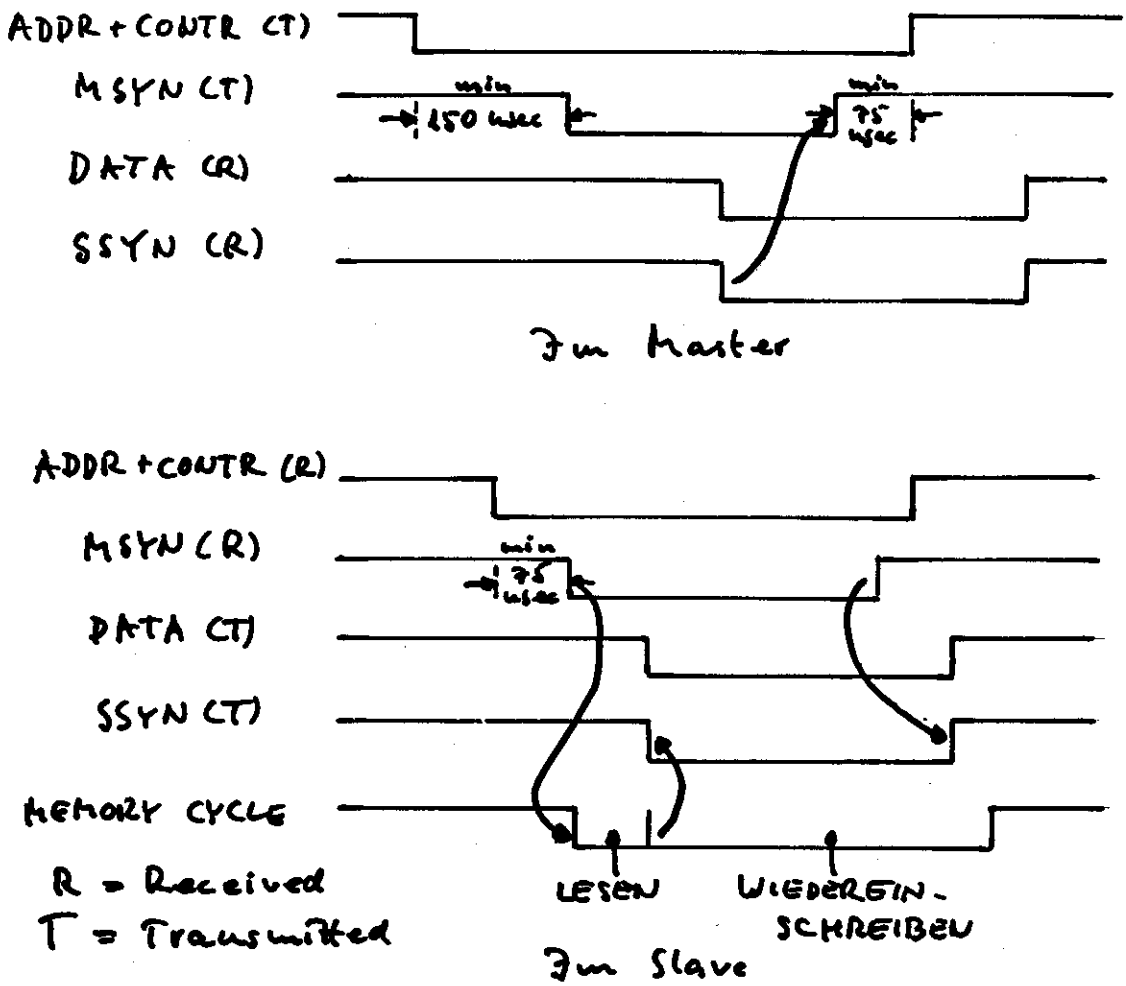
5.2.2.1 Timing beim Datentranfer

Trotz des Handshake-Betriebs gibt es gewisse Zeitvorschriften bei der Ausssendung und Decodierung von Adressen, Kontrollinformationen und Daten. Bei allen Übertragungen ist eine Zeittoleranz von 75 nsec für das Einschwingen der Signale und die Laufzeiten in Sendern und Empfängern vorgesehen. Aus diesem Grund verzögert der Master sein MSYN-Signal um 75 nsec, damit es nicht vor den Daten oder Adressen

im Slave erscheint. Zusätzliche 75 nsec Verzögerung sind für das Decodieren im Slave-Decoder vorgesehen. Nach dem Ende des MSYN-Signals nimmt der Master auch nach frühestens 75 nsec seine Adress- und Kontrollinformationen wieder zurück, um genügend Zeit zum sicheren Ausschwingen ohne erneutes Adressieren zu haben.

5.2.2.2 DATI- und DATIP-Ablauf

Alle Datenübertragungsfunktionen sind auf den Master bezogen, daher sind DATI(P)-Operationen in der Richtung vom Slave zum Master. Bild 5.5 zeigt die Signale als Funktion der Zeit.



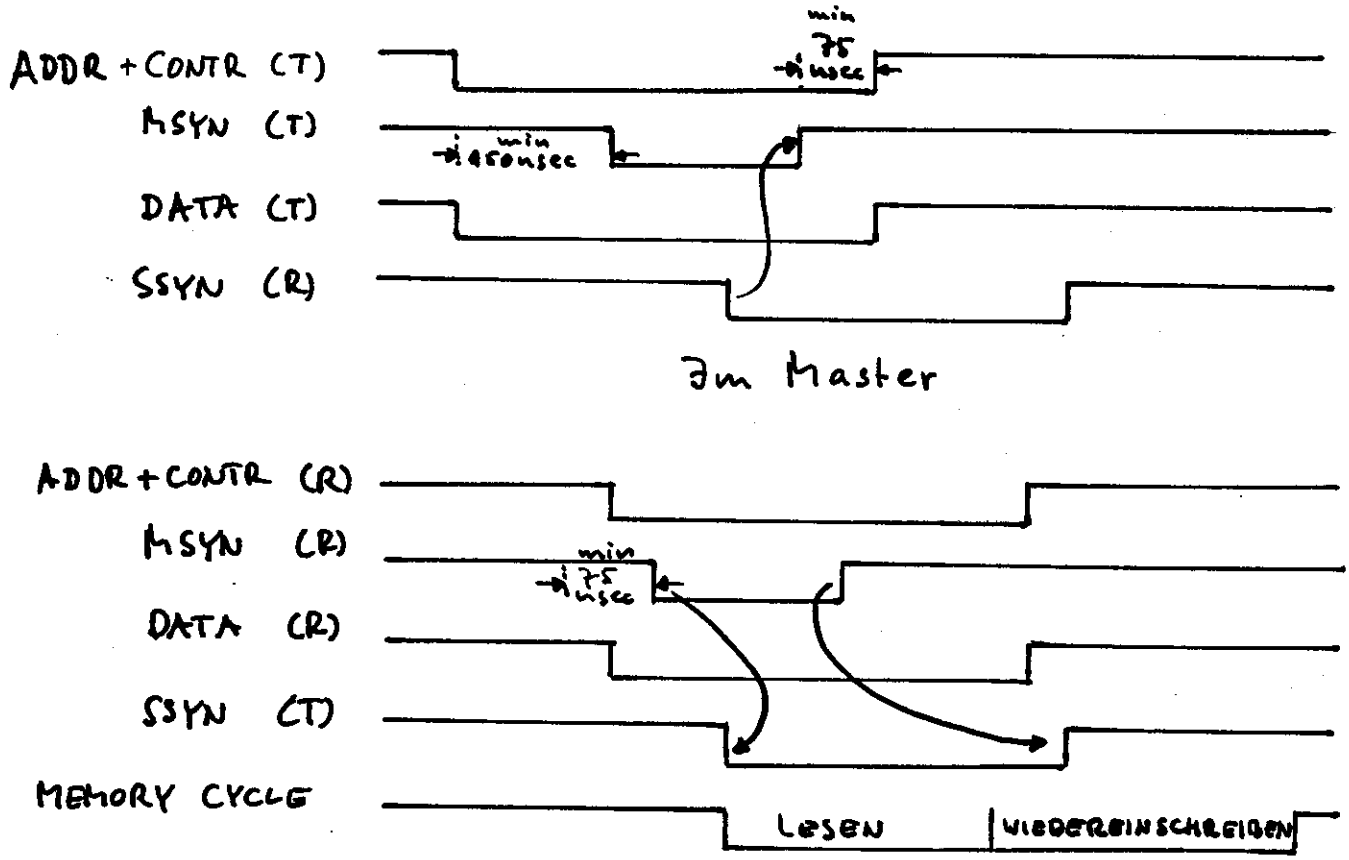
5.5 DATI - UND DATIP TIMING DIAGRAMM

Die Operation läuft nach folgendem Schema ab:

- Master setzt C<1:0>=00 für DATI bzw. C<1:0>=01 für DATIP sowie auf A<17:00> die Slave Adresse
- Master wartet mindestens 150 nsec (75 nsec Zeittoleranz, 75 nsec zum Decodieren der Adresse. Damit kommt A<17:00> und C<1:0> mindestens 75 nsec vor MSYN am Ausgang der Bus-Empfänger im Slave.
- Master wartet, bis der Bus leer ist, falls SSYN noch von einem vorigen Zyklus ansteht. Das wird automatisch erfüllt, da BBSY erst erscheinen kann, wenn SSYN verschwunden ist (s. Schaltung im Interrupt Control Modul M7820)
- Master erzeugt MSYN
- Erkennt der gewünschte Slave seine Adresse und MSYN, bereitet er die Daten zur Übertragung an den Master vor. Für Kernspeicher bedeutet dies den Lesezyklus, für Flip-Flop-Register sind die Daten unmittelbar bereit.
- Sind die Daten vorhanden, setzt der Slave sie auf D<15:00> und erzeugt SSYN. Hat der Slave eine nicht zerstörungsfreie Auslese, macht er den Wiedereinschreib-Zyklus, falls es eine DATI-Übertragung ist; ist es eine DATIP-Übertragung, setzt der Slave die Pausen-Flagge (über einen Flip-Flop) und wartet auf die folgende DATO- oder DATOB-Übertragung, in der neue, geänderte Daten eingeschrieben werden sollen. Der SSYN muß innerhalb von 25 µsec erzeugt werden.
- Master empfängt SSYN und wartet mindestens 75 nsec
- Master strobt die Daten herein, die auf D<15:00> stehen
- Nach der Datenannahme nimmt der Master MSYN zurück
- Nach 75 nsec ändert der Master die Signale auf den Adress- und Kontrollleitungen oder läßt sie verschwinden. Folgt eine weitere DATI-Operation, beginnt der Master die beschriebene Sequenz erneut. Folgt eine DATO- oder DATOB-Operation, beginnt die dort beschriebene Sequenz. Ist es der letzte Bus-Zyklus der Operation des Masters gewesen und sind die Signale der A- und C-Leitungen wieder zurückgenommen, cleart der Master BBSY und gibt die Bus-Kontrolle zurück
- Slave erkennt das Verschwinden von MSYN und nimmt die Daten von den D-Leitungen, SSYN wird gecleart.

5.2.2.3 DATO- und DATOB-Ablauf

Eine DATO(B)-Operation geht in der Richtung vom Master zum Slave. Bild 5.6 zeigt die Signale als Funktion der Zeit.



5.6 DATO- oder DATOB-TIMING DIAGRAMM

Der DATO- bzw DATOB-Verkehr läuft nach folgendem Schema ab:

- War der vorige Zyklus ein DATI- oder DATIP-Verkehr, wartet der Master, bis SSYN verschwindet. Dies geschieht wiederum automatisch (s. Schaltung M7820)
- Master setzt C<1:0>=10 für DATO oder C<1:0>=11 für DATOB, A<17:00> zur Festlegung der Slaveadresse und Daten, die an den Slave übertragen werden sollen auf D<15:00>
- Master wartet mindestens 150 nsec (75 für den Zeitausgleich, 75 nsec für Decodieren. Dadurch erscheinen A<17:00>, D<15:00> und C<1:0> garantiert mindestens 75 nsec vor MSYN am Ausgang der Slave-Busempfänger.
- Master wartet, daß der Bus frei wird, d.h. daß SSYN vom letzten Zyklus verschwindet.
- Master setzt MSYN
- Sieht der adressierte Slave seine Adresse und MSYN und ist er bereit, die Daten auf den D-Leitungen zu akzeptieren (falls er also nicht durch eine interne Operation noch beschäftigt ist), strobt er die Daten herein und

setzt SSYN. Die Daten auf den D-Leitungen bleiben garantiert mindestens 75 nsec nach Beginn von SSYN stehen. Der Prozessor bricht die Operation ab und zieht MSYN zurück, falls nach 25 µsec noch kein SSYN gekommen ist.

- Master empfängt SSYN und nimmt MSYN zurück
- Nach mindestens 75 nsec schaltet der Master die A-, C- und D-Leitungen ab oder ändert sie. Folgt ein weiterer DATO- oder DATOB-Zyklus, wiederholt der Master die beschriebare Sequenz. Ist es der letzte Bus-Zyklus des Masters und sind die A-, C- und D-Leitungen gecleart, setzt der Master BBSY zurück und gibt die Bus-Kontrolle ab. Folgt eine IN-Operation, startet der nächste Zyklus wie zu Beginn der DATI- oder DATIP-Folge.
- Der Slave sieht das Verschwinden von MSYN und schaltet SSYN ab.

5.2.3 Abwicklung des Prioritätsverkehrs

Die Übertragung der Bus-Kontrolle von einem Gerät auf ein anderes wird durch die Prioritätsauswahllogik bestimmt, die im Prozessor eingebaut ist. Requests für Bus-Kontrolle können zu jeder Zeit gestartet werden (asynchroner Betrieb), entweder auf den Bus-Request-(BR-) Leitungen oder auf der Nicht-Prozessor-Request-(NPR-) Leitung. Während jedes Bus-Zyklus prüft die Logik, ob ein NPR vorhanden ist, da dieser die Prozessorpriorität übersteigt. Ist dies der Fall, erzeugt die Logik ein NPG-Signal und empfängt das Selection acknowledge-(SACK-) Signal zurück. Diese Prozedure geschieht parallel zum laufenden Datentransfer. Ist das Gerät festgelegt, das nun Bus-Master werden soll, wartet es solange, bis der Bus frei wird, d.h. bis der vorige Bus-Master das Signal BBSY zurücknimmt. Darauf erzeugt der neue Bus-Master sein BBSY.

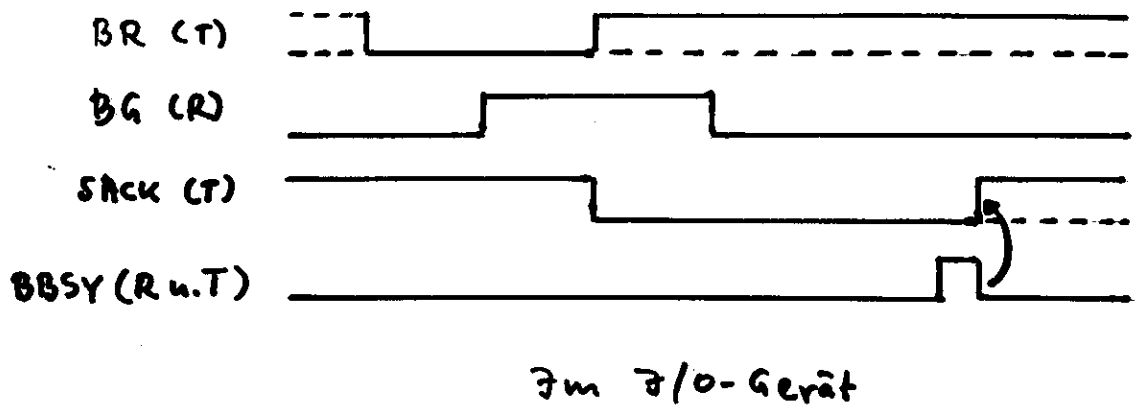
Eine ähnliche Prozedure geschieht am Ende jedes Befehls, wenn die Priortätslogik prüft, ob auf den BR-Leitungen Anforderungen sind und evtl. vorhandene mit der Prozessorpriorität vergleicht (diese liegt in 3 Bits im Prozessorstatusregister). Ist die Anforderungspriorität höher als die des Prozessors, wird auf der zugehörigen Leitung zwischen 2 Befehlen ein Grantsignal erzeugt. Dies ist jedoch nicht der Fall, wenn der Prozessor gerade einen Fehlerbefehl bearbeitet, z.B. wenn die Stromversorgung abschaltet. Dann wird zunächst die Fehlerfolge abgearbeitet, nach Ende des ersten, sich anschließenden Befehls wird nun das Grantsignal abgegeben. Dies Signal läuft seriell durch alle Geräte hindurch, die an der gleichen BG-Leitung (gleiche Priorität) angeschlossen sind. Ist das Gerät, das den Request ausgelöst hat, erreicht, blockiert dieses das Weiterlaufen des Grantsignals und erreicht damit Bus-Kontrolle. Das dem Prozessor nächstliegende Gerät der gleichen BG-Leitung hat also die höchste Subpriorität.

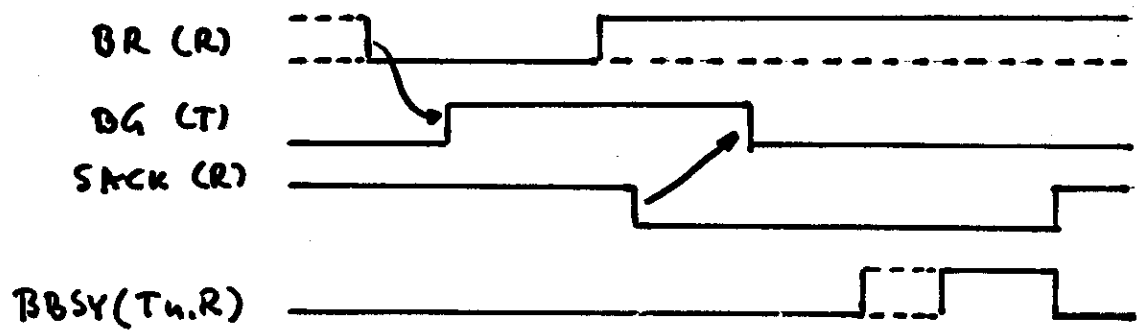
5.2.3.1 Prioritätstransfer-(PTR-) Abwicklung

Die PTR-Operation ist eine Signalfolge, die eines der Geräte, die eine Anforderung ausgesandt haben, zum nächsten Bus-Master macht. Die Operation, die in Bild 5.7 gezeigt ist, wird nach folgendem Schema abgewickelt:

- Das Gerät, das die Bus-Kontrolle benötigt, sendet einen Request auf der zugehörigen BR- oder NPR-Leitung.
- Der Prozessor empfängt ein oder mehrere BR-Signale, die in der Prioritätslogik den Vergleich der Prioritätsebenen auslösen. Ist die einlaufende Priorität höherwertiger als die des Prozessors, und ist die SACK-Leitung frei, erzeugt der Prozessor das zugehörige BG- oder NPG-Signal und zwar das NPG-Signal während der laufenden Bus-Operation, das BG-Signal erst am Ende des laufenden Befehls.
- Jedes an die aktivierte BG-Leitung angeschlossene Gerät läßt das BG-Signal durch, es sei denn, es selbst hat den Request ausgelöst.
- Das auf der BG-Leitung angesprochene Gerät erzeugt das SACK-Signal, blockiert das BG-Signal von den nachfolgenden Geräten und nimmt das BR-Signal zurück.
- Der Prozessor empfängt das SACK-Signal und cleart die BG-Leitung. Ist das SACK-Signal nicht innerhalb von 10 µsec eingetroffen, wird die Bus-Zuteilung automatisch gecleart.
- Der noch laufende Bus-Master vollendet seinen Datentransfer und cleart zusammen mit den A- und C-Leitungen auch sein BBSY.
- Der neue Bus-Master setzt sein BBSY, wenn die Leitungen BBSY, BG und SSYN am Ende des vorangehenden Zyklus frei werden. Falls der neue Bus-Master einen Interrupt macht, kann die INTR-Leitung auch während des Zyklus gesetzt werden.
- Das SACK-Signal wird zur gleichen Zeit zurückgestellt, wenn der neue Master den Interrupt macht. Falls der Bus-Master erst Daten überträgt, wird das SACK-Signal erst vor dem Beginn des letzten Bus-Zyklus gelöscht.
- Hat der neue Bus-Master seine Datenübertragung abgeschlossen, nimmt er BBSY zurück und ein neuer Bus-Master kann die Kontrolle erhalten. Ist kein weiteres Gerät vorhanden, das einen Request schickt, setzt der Prozessor sein BBSY und fährt in seinem Programm fort.

Da ein NPR während eines Befehls mit einem Grantsignal beantwortet wird, darf das den NPR auslösende Gerät keinen Prozessor-Interrupt machen, da sonst die im Prozessor enthaltende Information verloren geht.



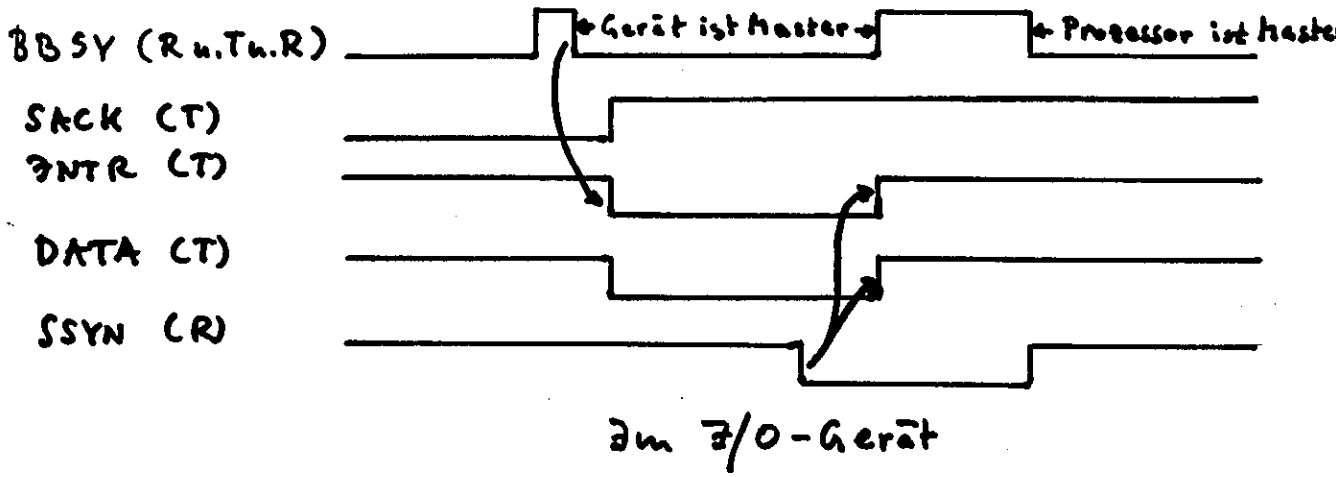


R = Received
T = Transmitted

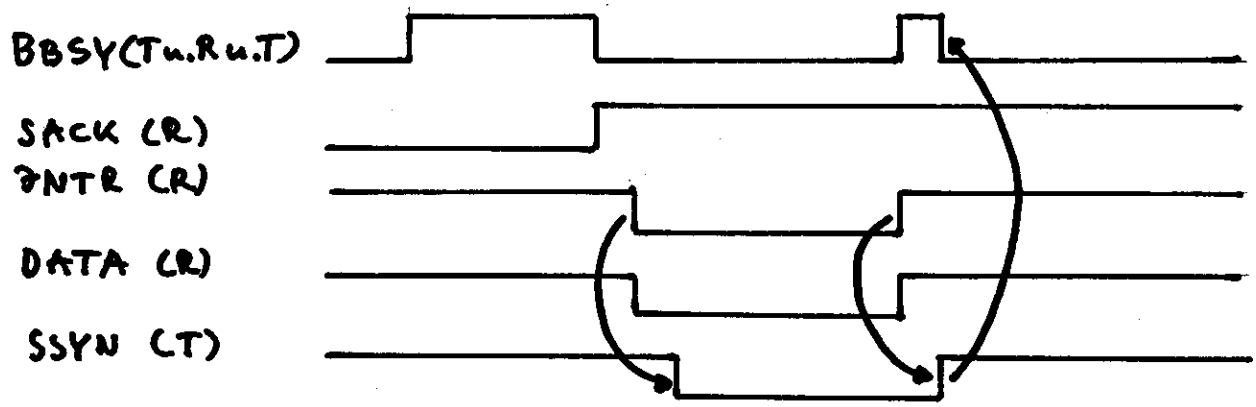
5.7 TYPISCHER PTR-ABLAUF
PROZESSOR IST BUS-MASTER

5.2.3.2 Interrupt Abwicklung

Ein externes Gerät, das über die BR-/BG-Leitungen Buskontrolle erhalten hat, kann entweder sofort zu Beginn oder nach einer oder mehreren Datenoperationen einen Interrupt auslösen. Dies geschieht nach folgendem Schema (s. auch Bild 5.8):



im E/O-Gerät



im Prozessor

R = Received
T = Transmitted

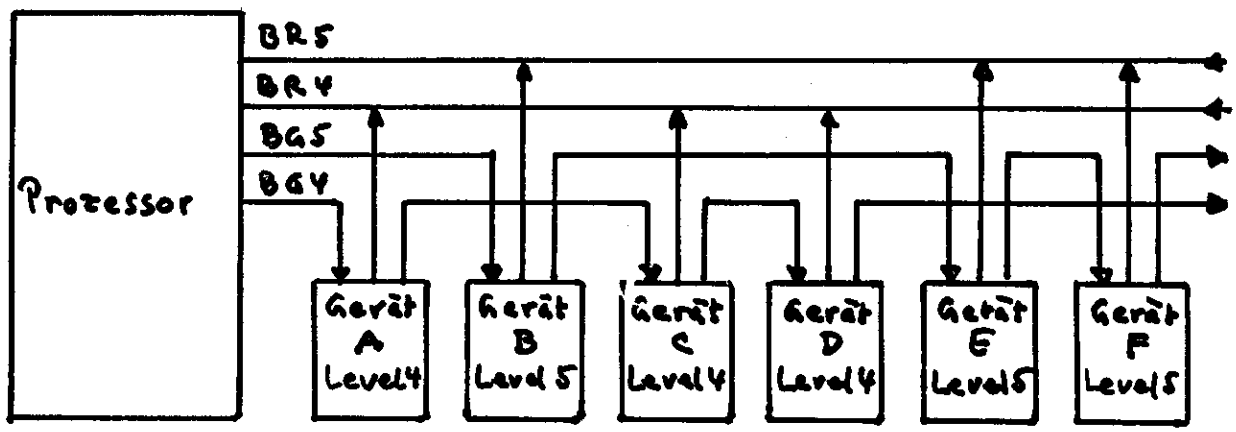
5.8 INTERRUPT TIMING

- Wird der Interrupt sofort ausgelöst, setzt der neue Bus-Master das INTR-Signal und eine Vektor-Adresse auf die D-Leitungen, zur gleichen Zeit wird von ihm BBSY erzeugt und SACK gecleart. Werden vorher noch Daten übertragen, bleibt das SACK-Signal so lange bestehen, bis INTR erscheint.
- Der Prozessor erhält das INTR-Signal, wartet 75 nsec, um sicher zu sein, daß alle Bits der Interrupt-Vektoradresse vorhanden sind und setzt SSYN, während die Daten eingelesen werden.

- Der Bus-Master, der den Interrupt ausgelöst hat, empfängt SSYN und cleart INTR, BBSY und die D-Leitungen. Damit geht die Interrupt-Verwaltung auf den Prozessor über.
- Ist INTR gecleart, nimmt der Prozessor auch SSYN zurück und startet die Interrupt-Folge durch Wegspeichern des Inhalts des Prozessor-Status-Registers (PS) und des Programm-Zählers (PC) und Neuladen der beiden mit dem Inhalt der Zelle, die durch die Vektoradresse spezifiziert ist.

5.2.3.3 Verketten von Prioritäten

Im PDP-11 System werden, wie bereits erwähnt, die Prioritäten der Geräte, die auf gleichem Level liegen, verkettet. Dies kann an einem Beispiel in Bild 5.9 noch verdeutlicht werden.



5.9 PRIORITÄTS-KETTE

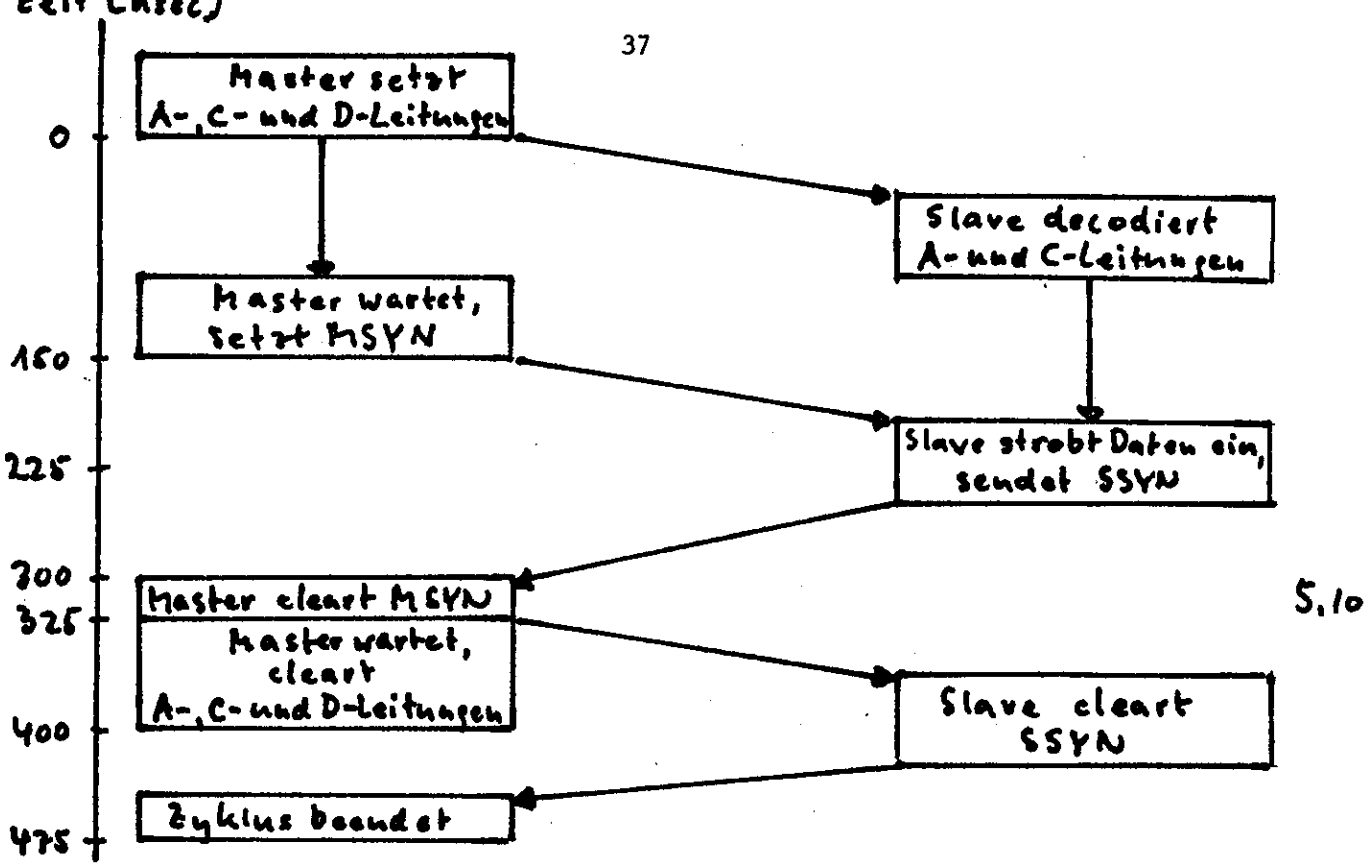
Drei der Geräte haben die Priorität 4, nämlich A, C und D, oder die Priorität 5, nämlich B, E, F. Wäre z.B. die Priorität des Prozessor 5, würde keine dieser Bus-Anforderungen beantwortet. Nehmen wir z.B. an, der Prozessor habe die Priorität 2 und die Geräte C, E und F fordern während einer Befehlsausführung den Bus, dann macht am Ende des Befehls der Prozessor eine PTR-Operation. Da zwei der anfordernden Geräte die Prioritäts 5 haben, wird der BR5-Request angenommen und mit BG5 beantwortet. Dieses Signal läuft zuerst zu Gerät B, dort wird das Interrupt-Kontroll-Schieberegister geclockt und festgestellt, daß Gerät B keine Anforderung gesendet und dann das Signal zu Gerät E weitergeleitet. Dieses hält das BG-Signal, löscht sein BR-Signal und bekommt Kontrolle über den Bus. Die Anforderungen BR5 aus Gerät F und BR4 aus Gerät C bleiben auf dem Bus stehen, bis die Anforderung

des Gerätes E abgearbeitet sind. Sie werden dann behandelt. Hat Gerät E eine Interrupt-Operation ausgelöst, erhält Gerät F die Bus-Kontrolle nach Ausführung des ersten Befehls der Interrupt-Routine, es sei denn, die Interrupt-Operation erhöht die Priorität des Prozessors. Dies kann dadurch geschehen, daß die Interrupt-Operation zu Beginn ein neues Prozessor-Status-Registerwort PS erzeugt, in dem in drei dafür vorgesehenen Bits die neue Priorität enthalten ist.

5.2.4 Timing in UNIBUS-Operationen

Als die DATI(P)- oder DATO(B)-Operationen beschrieben wurden, wurde bereits erwähnt, daß gewisse Verzögerungen nicht zu vermeiden sind. Sie entstehen durch unterschiedliche Durchlaufzeiten in den Schaltkreisen sowie verschiedenen Busweglängen. Um diese zu berücksichtigen, wird eine Standardverzögerung von 75 nsec eingeführt. Da viele Geräte am Ein- bzw. Ausgang Übertragungsgates haben, die durch Strobe-Signale geöffnet werden, muß auch die dadurch bedingte Ansprechverzögerung beachtet werden. Dies geschieht durch eine weitere Standardverzögerung von 75 nsec. Braucht eines der Slave-Geräte mehr Zeit zum Decodieren, muß es selbst den MSYN intern weiterverzögern.

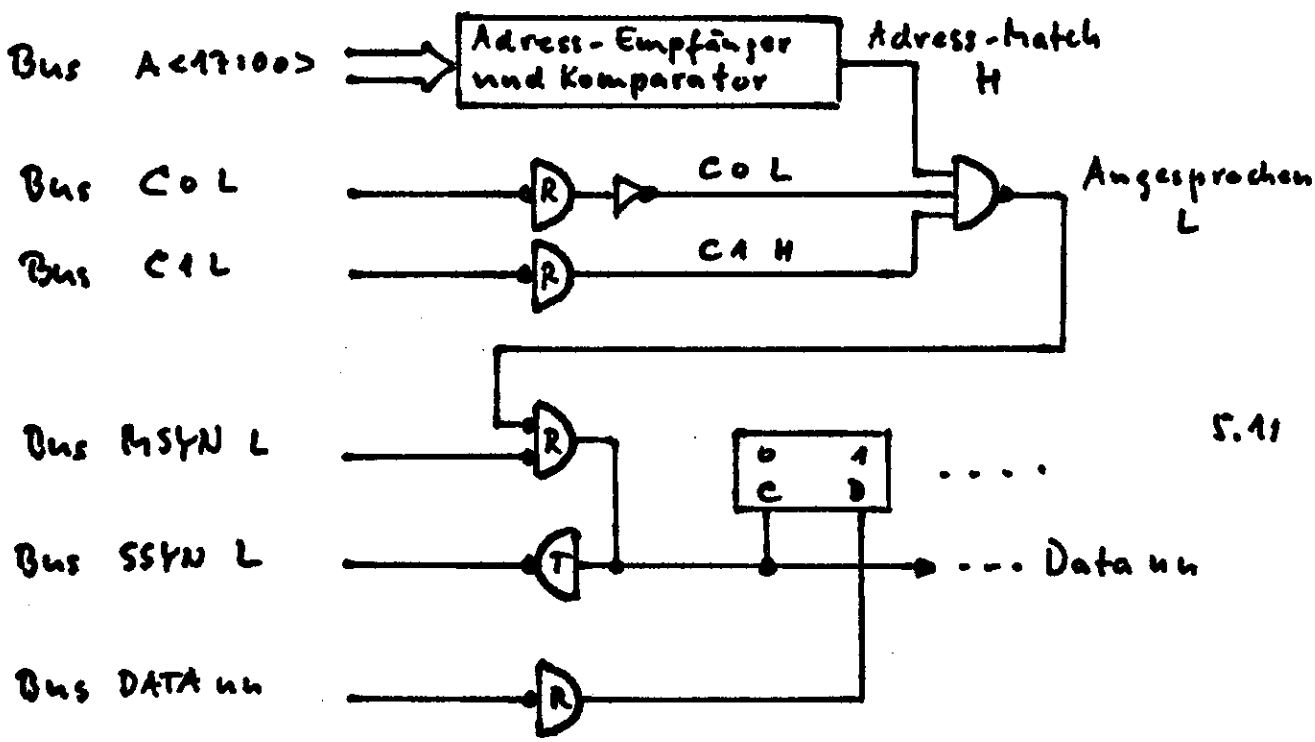
In einem typischen Beispiel sollen die auftretenden Zeiten erläutert werden. Wir nehmen eine DATO-Operation an einen Slave an, der ein Flip-Flop-Register hat. Der Bus-Master setzt die A-, D- und C-Leitungen, wartet 150 nsec und sendet MSYN. Nach weiteren 75 nsec erkennt der Slave MSYN, strobt die Daten in sein Register und sendet SSYN. Sieht der Master SSYN, cleart er sein MSYN (Dauer etwa 25 nsec) und wartet 75 nsec, bevor er die A-, C- und D-Leitungen wieder zurückstellt. Auf die Wegnahme von MSYN löscht der Slave sein SSYN, so daß, wie aus Bild 5.10 zu erkennen ist, nach 400 nsec ein neuer Zyklus beginnen kann.



5.10

TYPISCHES DATO-TIMING

In Bild 5.11 ist eine typische Slave-Logik für den DATO-Verkehr gezeigt.



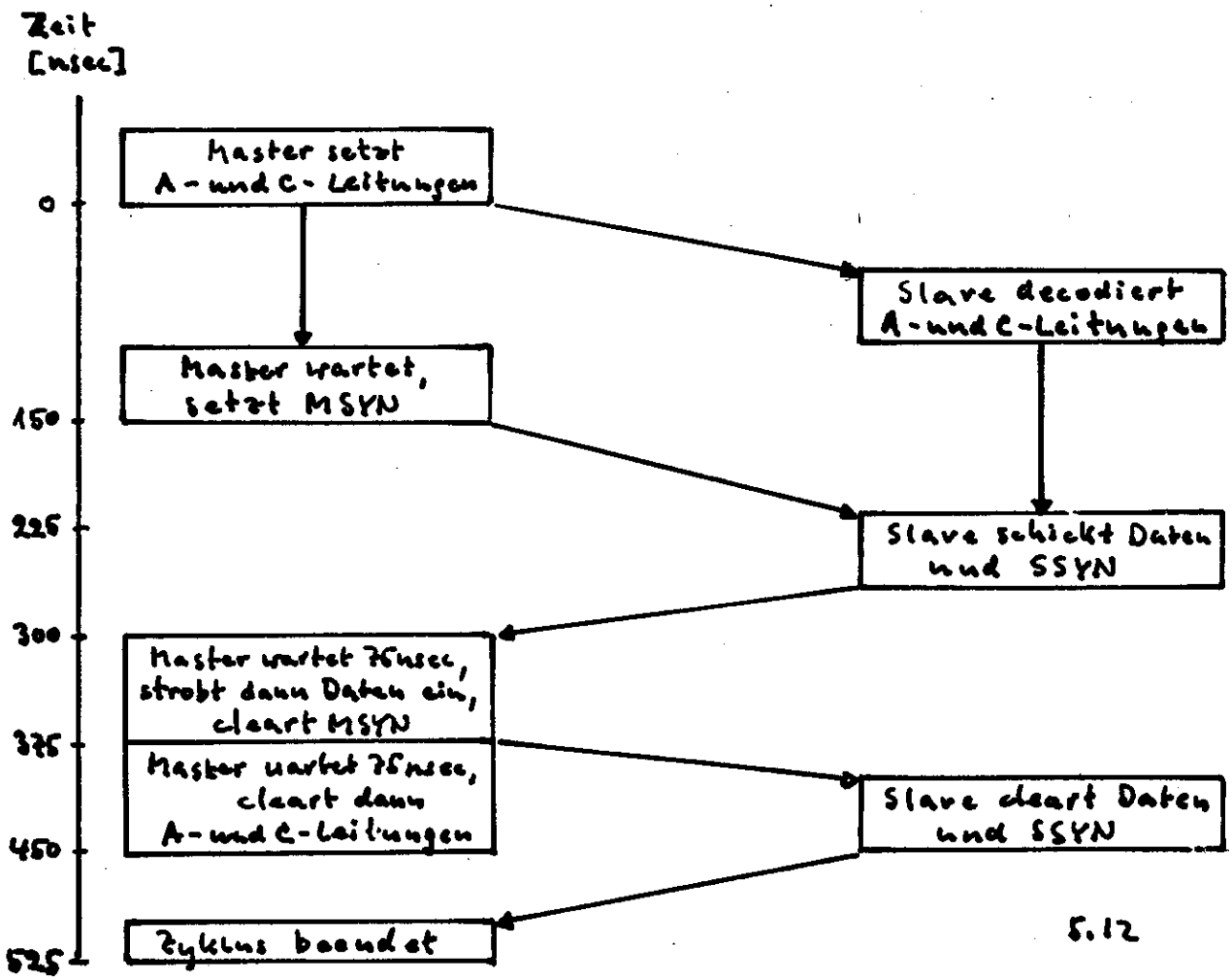
5.11

SLAVE LOGIK FÜR DATO

Bild 5.12 zeigt das typische Timing-Diagramm für den DATI-Verkehr. Dies entspricht dem Diagramm beim DATO, mit Ausnahme der folgenden vier Punkte:

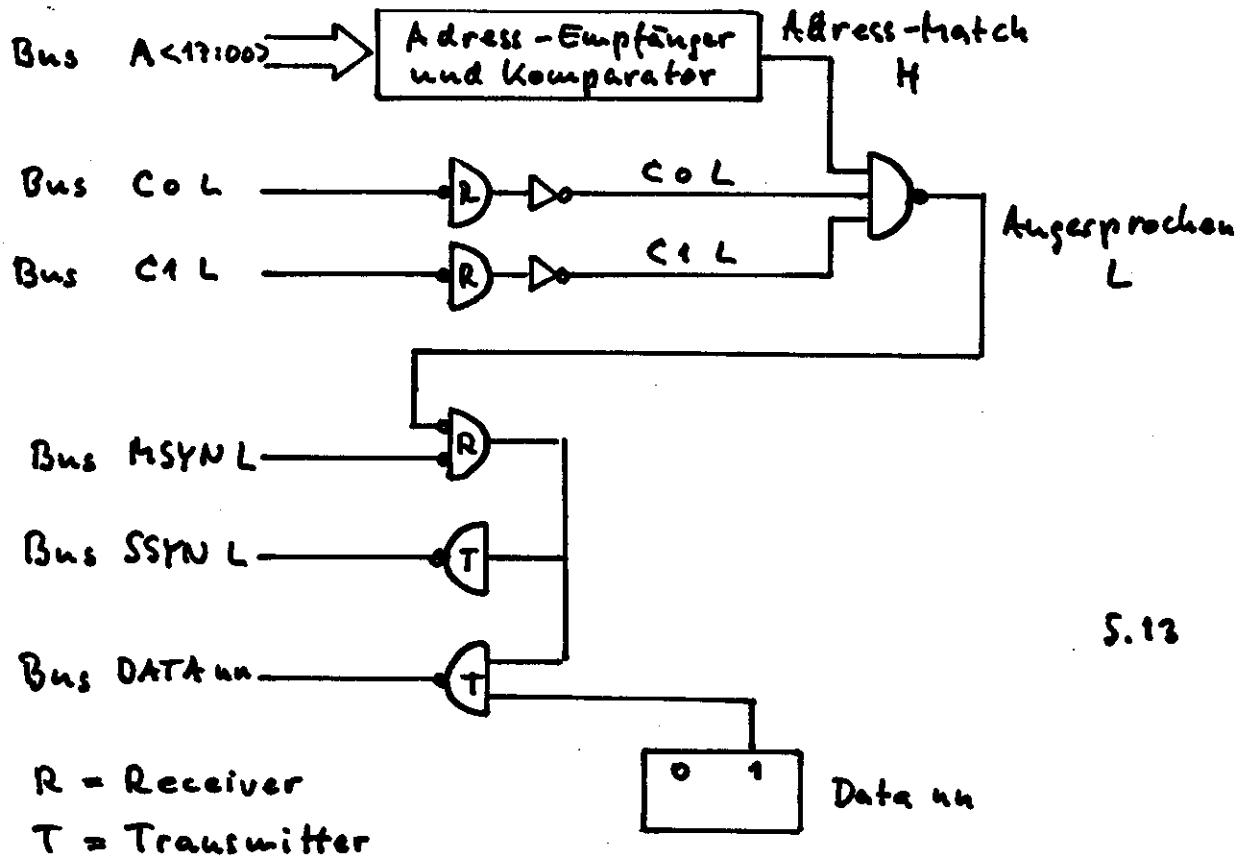
- Der Master setzt nur A- und C-Informationen
- Der Slave schickt die Daten gleichzeitig mit SSYN auf den Bus.
- Der Master wartet 75 nsec, bevor er die Daten einstrobt und MSYN löscht.
- Der Slave löscht die Daten mit der Rücknahme von SSYN.

Die DATI-Operation ist normalerweise nach 450 nsec beendet.



TYPISCHES DATI-TIMING

Eine typische DATI-Logik im Slave zeigt Bild 5.13.



5.13

SLAVE LOGIK FÜR DATI

Ein Gerät, das Master werden möchte, um Daten an ein anderes Gerät zu übertragen, hat normalerweise ein Bus-Adressregister, das zunächst vom Programm mit einer Anfangsadresse geladen wird. Diese Adresse muß dann für jedes übertragene Datenwort inkrementiert werden. Lädt das Programm jedoch eine falsche Anfangsadresse oder entstehen beim Inkrementieren Adressen, die im Slave-Speicher garnicht mehr vorhanden sind, wird kein SSYN zurückkommen. Um zu verhindern, daß sich das System dann selbst aufhängt, wird empfohlen, daß ein 10-25 µsec Univibrator

jedesmal getriggert wird, wenn der Master MSYN setzt. Ist diese Zeit vorüber, bevor SSYN eintrifft, sollte der Master den Verkehr abbrechen durch Löschen vom MSYN, BBSY und weiteren, von ihm gesetzten Signalen. Der Master muß dann eine Fehler-Flagge in seinem Status-Register setzen.

5.2.5 Adreßlisten

Eine PDP11-Adressenliste findet sich in den Datenbüchern. Die Adressenzahlen sind im allgemeinen im Oktalcode angegeben.

Über den UNIBUS können 2^{18} = 256k Adressenplätze belegt werden, jeder Platz enthält 8Bits. In den kleinen PDP11-Maschinen sind nur 16Bits, d.h. 2^{16} =64k Plätze unter Programm-Kontrolle. Da die Wortlänge und Busbreite 2 Bytes sind, adressieren die meisten Busoperationen 2 Plätze gleichzeitig; die angegebene Adresse ist die des geradzahligen Platzes. Byteoperationen können jedes einzelne Byte adressieren. Bei einer DATI-Operation mit der Adresse 400 wird die Information in die Plätze 400 und 401 eingeschrieben, bei einer DATOB-Operation nur in Platz 400. Grundsätzlich kann ein 16Bit-Wort nicht in einen ungeradzahligen Platz eingeschrieben oder gelesen werden.

Bei 18 Bit-breiten Adressen werden im Prozessor die Bits A(17:16) auf 1 gesetzt, falls A(15:13) auch auf 1 liegt. Das bedeutet, daß die letzten 8k Bytes die höchsten Plätze sind, die der Bus akzeptiert. In diesen 8k Plätzen sind alle Geräteadressen und interne Prozessoradressen untergebracht.

5.2.5.1 Interruptvektorplätze

In den ersten 400 Plätzen sind u.a. die Interruptvektoren untergebracht, und zwar von Adresse 58 bis 400. Speziell die Vektoradressen 170, 174, 270 und 274 sind für vom Benutzer entwickelte Interface-Systeme vorgesehen. Jeder Vektor benötigt 4 Plätze, d.h. 2 Worte, die Vektoradressen müssen geradzahlige Grenzen haben, d.h. der Vektor muß auf 4 oder 0 enden. Man verwendet Vektoradressen, die Bit 0 oder 1 nicht spezifizieren. Da die unteren Bits immer 0 sind, legt Bit 2 entweder 0 oder 4 fest.

5.2.5.2 Speicherplätze

Speicherplätze, und zwar Lesen/Schreiben oder ROM, beginnen bei 0 und enden bei 157 777. Der 8k-Block mit der höchsten Nummer wird für Geräteregister und interne Prozessorregister als Adresse benutzt.

5.2.5.3 Geräteregisterplätze

Jedes Gerät hat ein oder mehrere Register, die grundsätzlich geradzahlige Adressen erhalten, d.h. A00 ist 0, obgleich in Byte-Operationen jede Hälfte adressierbar ist. Die oberen 8k-Byteplätze sind für Geräteregister vorgesehen, davon die oberen 4k-Bytes (770 000 bis 777 777) für Prozessoradressen und Standardperipherie von DEC. Die 2k-Byteadressen von 764 000 bis 767 777 sind für Geräte des Benutzers vorgesehen, sie werden von DEC nicht benutzt.

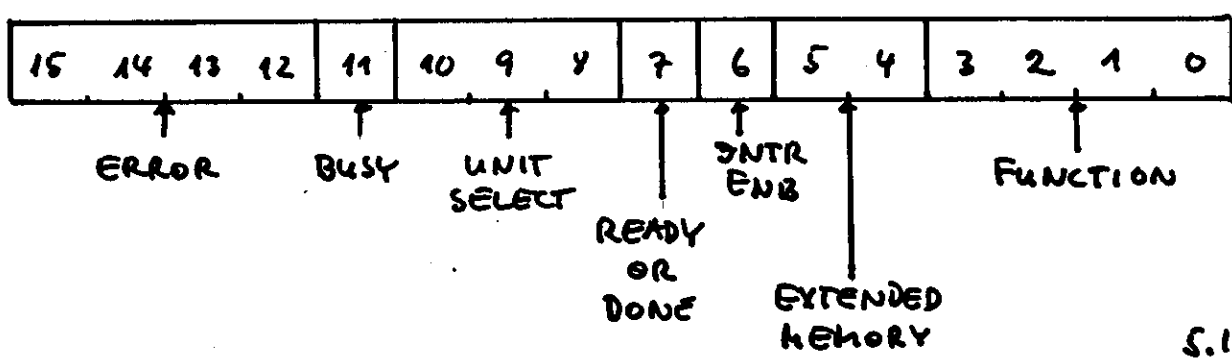
5.2.5.4 Prozessorplätze

Nur 2 Prozessorregister sind explizit adressierbar, das Konsolen-Schalterregister (Adresse 777 570 und 777 571) kann für programmkontrollierten Datenverkehr benutzt werden (nur zum Lesen) und das Prozessor-Statusregister PS (Adresse 777 776 und 777 777). Die 16 Prozessor-Speicherregister liegen auf den Adressenplätzen 777 700 bis 777 717, jede Adresse bedeutet ein volles 16Bit-Wort. Diese Adressen sind jedoch nicht vom Bus adressierbar.

5.2.6 Geräteregister

Der Datenverkehr zwischen einem Gerät und dem UNIBUS erfolgt über ein oder mehrere Geräteregister. Diese können entweder durch Flip-Flop-Speicherregister oder dynamische Signale dargestellt werden, die während eines Transfers auf den UNIBUS gegatet werden. Die Registerbits können für verschiedene Operationen benutzt werden, z.B. für Lesen/Schreiben beim DATI- oder DATO-Verkehr, nur zum Schreiben bei DATO-Operationen oder nur zum Lesen bei DATI-Verkehr.

Das Registerformat sollte sich an den in Bild 5.14 dargestellten Standard halten, die Funktionen der einzelnen Bits werden wie folgt erläutert:



5.14

FUNCTION: Gerätefunktion, wie z.B. Lesen, Schreiben, Drucken, Suchen. Geräte mit nur einer Funktion sollen Bit 0 benutzen

EXTENDED MEM: Wird benutzt, um A<17:16> festzulegen, falls Gerät Datenübertragungen in Adressenplätze ausführt, die nicht in den ersten 64k liegen

INTR ENB: Interrupt Enable, falls nicht gesetzt, wird Interrupt-Anforderung verhindert.

READY or DONE: Dieses Bit wird gesetzt, wenn die im Gerät vorgenommenen Abläufe beendet sind und das Gerät bereit zum Datentransfer ist.

UNIT SELECT: Wird zum Selektieren eines von mehreren Geräten benutzt, die von einem Kontroller bedient werden.

BUSY: Zeigt an, daß das Gerät interne Abläufe ausführt.

ERROR: Zeigt die Quelle oder Ursache eines Fehlers an. Bit15 wird für Fehler aus einer einzelnen Quelle oder als OR mehrerer Fehlerquellen benutzt.

Als Geräteregister finden 4 typischen Register Verwendung:

CSR - Gerätefunktion, Status, Interruptkontrolle

DBR - Datenbufferregister für Informationsaustausch

MAR - Speicher Adressregister für Blocktransfer, wird nach jedem Worttransfer inkrementiert

WCR - Word Countregister, wird vom Programm gesetzt mit der Anzahl der zu übertragenden Wort im Blocktransfer

Diese Register sollen wie folgt adressiert werden:

Adresse (Oktal)	
N	CSR
N+2	DBR
N+4	MAR
N+6	WCR

Werden mehrere Register für die gleiche Funktion benutzt, sollen sie mit aufsteigenden Adressen versehen werden und in der ebengenannten Reihenfolge sortiert werden, wie z.B.

CSR1
CSR2
DBR1
DBR2
DBR3
MAR
WCR

5.3 Interface-Schaltungen

5.3.1 UNIBUS-Kabel

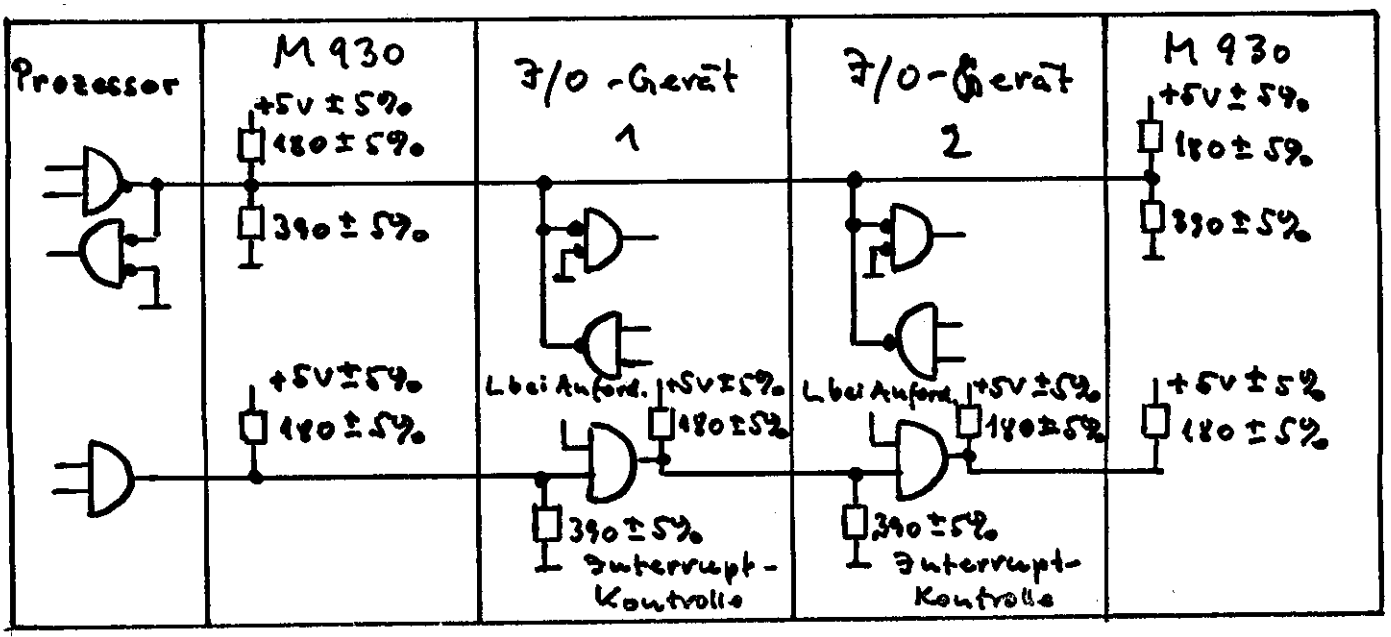
Der UNIBUS ist ein vieladriges Drahtsystem mit bestimmten Wellenwiderstand, das durch Sender und Empfänger betrieben wird. Es werden verschiedene Kabeltypen eingesetzt. Eine kurze Kabelbrücke (Typ M920, 56 Signalleitungen und 14 Erdleitungen) verbindet benachbarte Moduln direkt. Crates oder Mountingboxes (Typ BA11) werden mit flexiblen Kabeln (Typ BC11A) verbunden, deren Wellenwiderstand 120Ω±15% und deren DC-Widerstand ca 0.4Ω pro m beträgt. Das Kabel enthält 56 Signalleitungen und 64 Erdleitungen. Für längere Kabelverbindungen wird twisted pair, evtl. auch Koaxkabel empfohlen.

An jedem Ende ist der UNIBUS durch eine Widerstandskombination abgeschlossen, mit Ausnahme der Grantsignale, die mit einem einzigen Widerstand beschaltet sind. 2 Abschlußmoduln Typ M930 sind in jedem System vorhanden.

5.3.2 UNIBUS Signalpegel

Der Ruhezustand für alle UNIBUS Signalleitungen, mit Ausnahme der Grantleitungen BG<7:4> und der NPG-Leitung ist die logische 0 auf +3.4V. Der gesetzte Zustand, d.h. die logische 1, liegt zwischen 0 und +0.8V. Die Signalpegel der Grantsignale sind genau umgekehrt. Die Empfängersignalschwelle liegt bei +2V. In der üblichen Terminologie werden die Buchstaben H (High) und L (Low) für +3.4V bzw. 0 V gebraucht, die Buchstaben in den Schaltungen geben den gesetzten Zustand an. Eine UNIBUS-Datenleitung wird daher BUS D00 L genannt, eine Grantleitung Bus BG4 H. Alle Signale, die nicht den UNIBUS betreffen, sind Standard-TTL-Signale.

Bild 5.15 zeigt den Bus mit Sendern und Empfängern in bi- und unidirektionalen Leitungen.

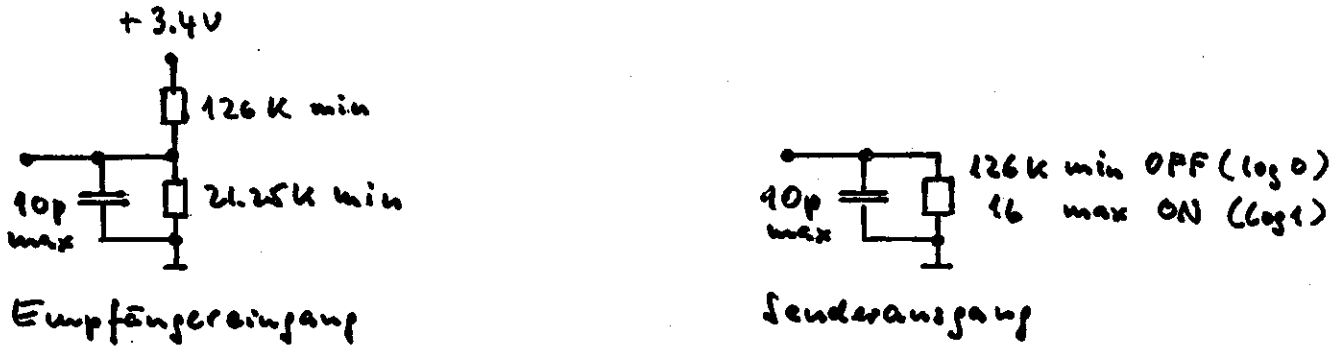


5.3.3 UNIBUS-Länge und Belastung

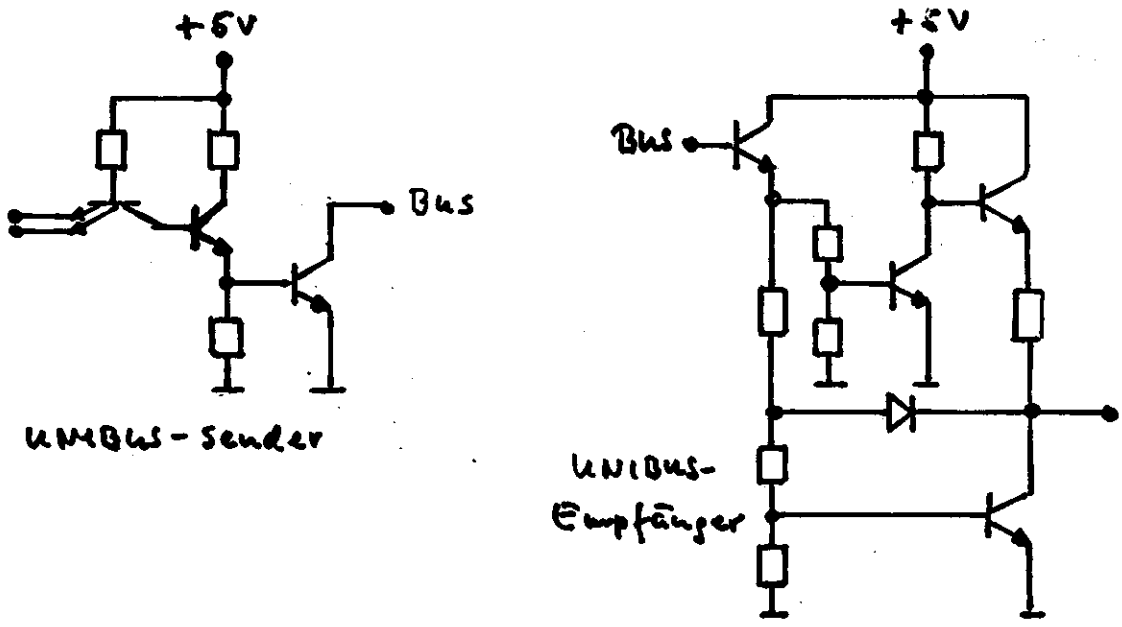
Die maximale UNIBUS-Länge hängt vom Kabeltyp, der Belastung sowie der Verteilung von Sender- und Empfängeranschlüssen auf dem Bus ab. Da der Transfer auf dem Bus asynchron ist, spielt die Signalverzögerung durch die Buslänge keine Rolle. Mit dem flexiblen Kabel ist die maximal zulässige Länge 15m weniger die Längen der Zuleitungen von den Sendern und Empfängern. Die UNIBUS-Leitungen können mit je 20 TTL-Eingängen belastet werden, sonst muß ein UNIBUS Repeater (Typ DB11-A) dazwischen geschaltet werden.

5.3.4 Busempfänger- und -sender-Schaltungen

Bild 5.16 zeigt die äquivalenten Netzwerke der Standardempfänger- und -sender. Bild 5.17 die typischen Schaltungen der integrierten Kreise, der Empfänger ist DEC380A, der Sender DEC8881 (selektierte Sprague-Typen).



5.16 ÄQUIVALENTSCHALTUNG

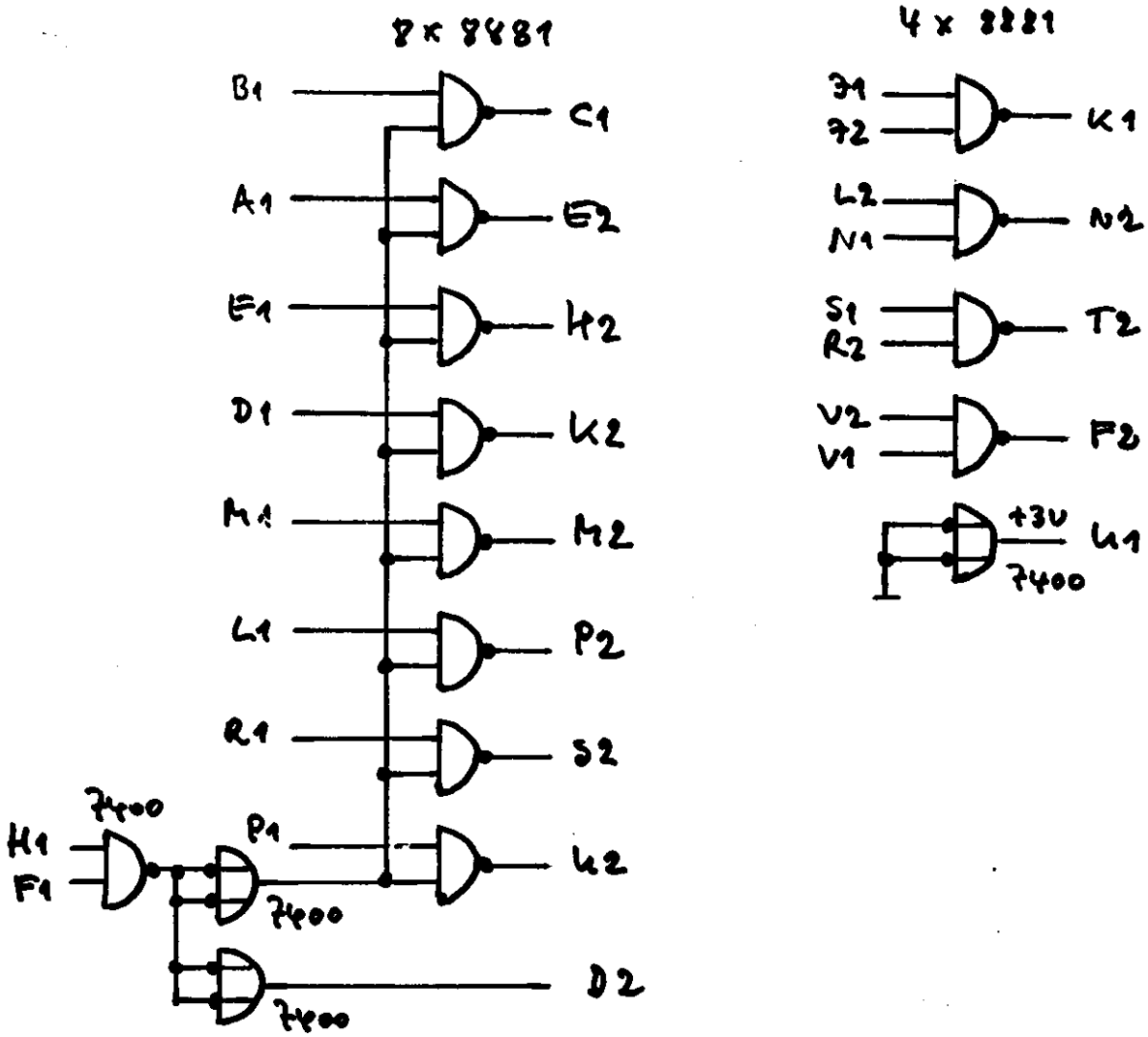


5.17 REALE SCHALTUNG

Verschiedene Sender-, Empfängermoduln, Kombinationen beider sowie Treibermoduln sind entwickelt worden.

5.3.4.1 M783 UNIBUS-Transmittermodule

Diese Einheit enthält 12 Treiberstufen, 8 davon haben eine gemeinsame Strobeleitung, 4 haben 2-Eingang-positive-AND-Gates. Die Schaltung zeigt Bild 5.18

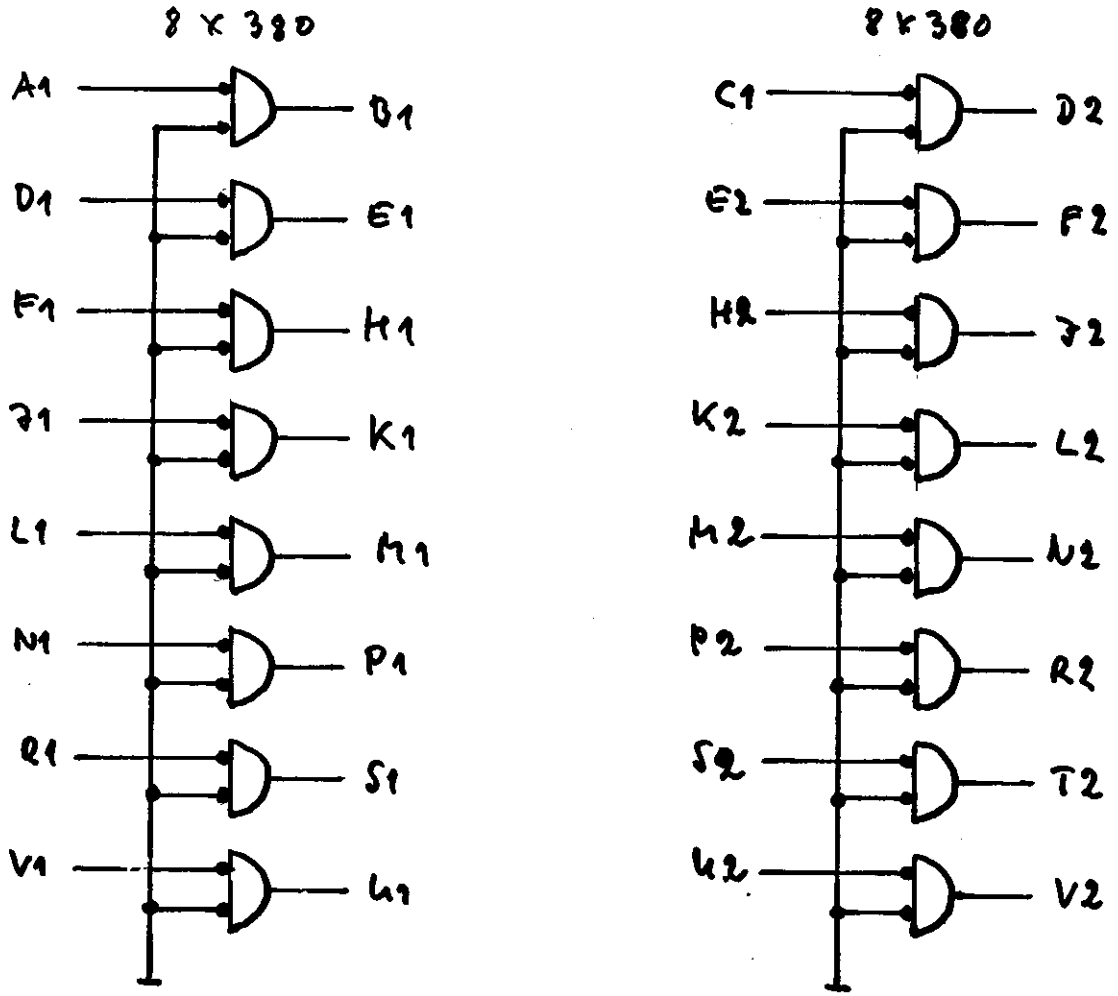


Karte benötigt +5V, 200 mA
+5V auf A2
Erde auf C2, T1

5.18 SCHALTUNG DES M783 UNIBUS TRANSMITTERS

5.3.4.2 M784 UNIBUS-Receivermodule

Dieser besteht aus 16 DEC 380A-Invertern, die das UNIBUS-Eingangsignal empfangen und gepuffert an das Gerät abgeben. Bild 5.19 zeigt die Schaltung

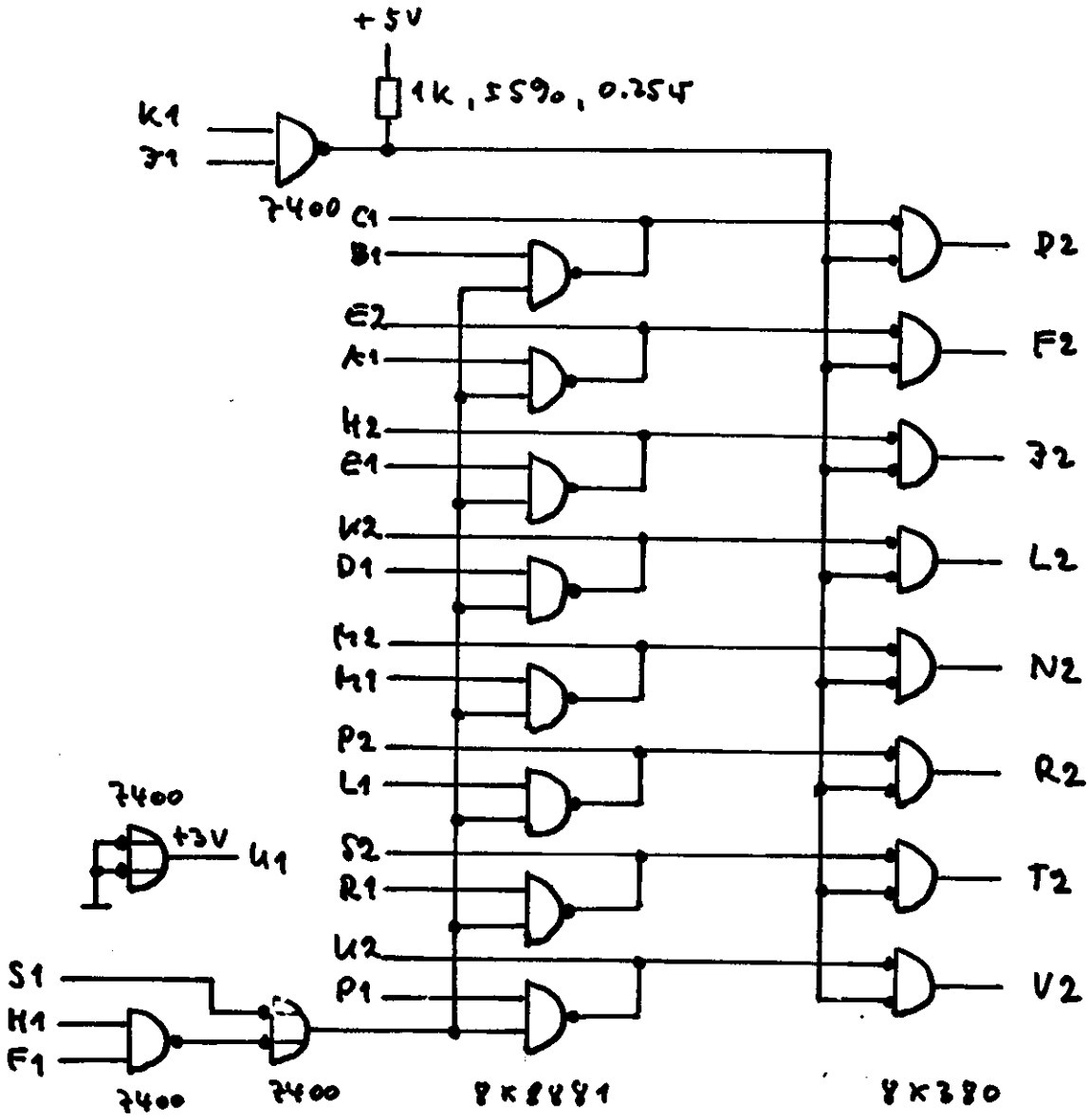


Karte benötigt +5 V, 200 mA
+5 V auf A2
Erde auf C2, T1

Bild 5.19 SCHALTUNG DES M784 UNIBUS-RECEIVERS

5.3.4.3 M785 UNIBUS-Transceivermodule

Diese Einheit enthält je 8 DEC 8881 Sender und DEC 380 Empfänger, die für bidirektionalen UNIBUS-Betrieb gedacht sind. Die Sender sowie die Empfänger haben je eine Strobeleitung. In Bild 5.20 ist die Schaltung dargestellt.

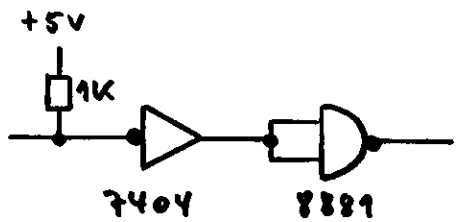


Karte benötigt +5V, 225mA
+5V auf A2
Erde auf C2, T1

5.20 SCHALTUNG DES M785 UNIBUS-TRANSCEIVERS

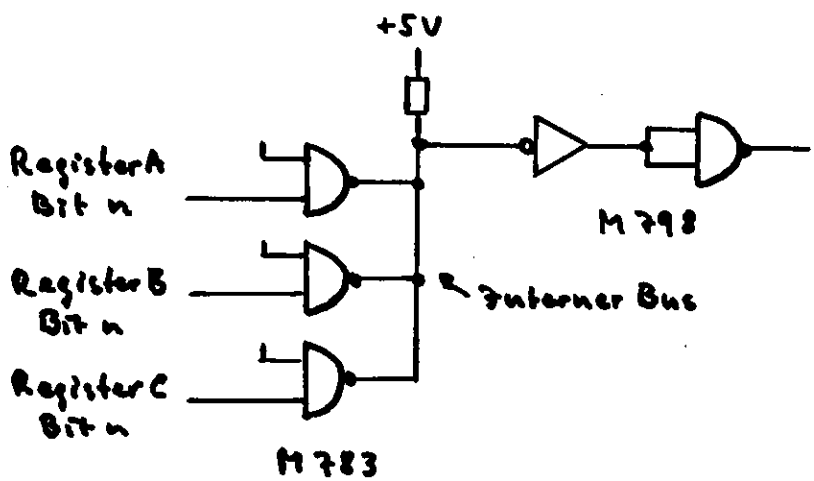
5.3.4.4 M798 UNIBUS Drivermodule

Diese Einheit besteht aus 16 nichtinvertierenden UNIBUS-Treibern, sie wird eingesetzt, wenn mehrere Geräte an der gleichen UNIBUS-Leitung angeschlossen sind. Bild 5.21 zeigt eine solche Treiberschaltung, d.h. eine von 16 auf der Karte.



5.21

Durch diese Schaltung kann der UNIBUS von Standard-opencollektor-TTL-Gates gesteuert werden. Jeder Treibereingang wird über 1kΩ auf +5V gezogen. Bild 5.22 zeigt eine Anschaltungsmöglichkeit für eine Registeranordnung mit internem Bus zum Anschluß an den Treibereingang.



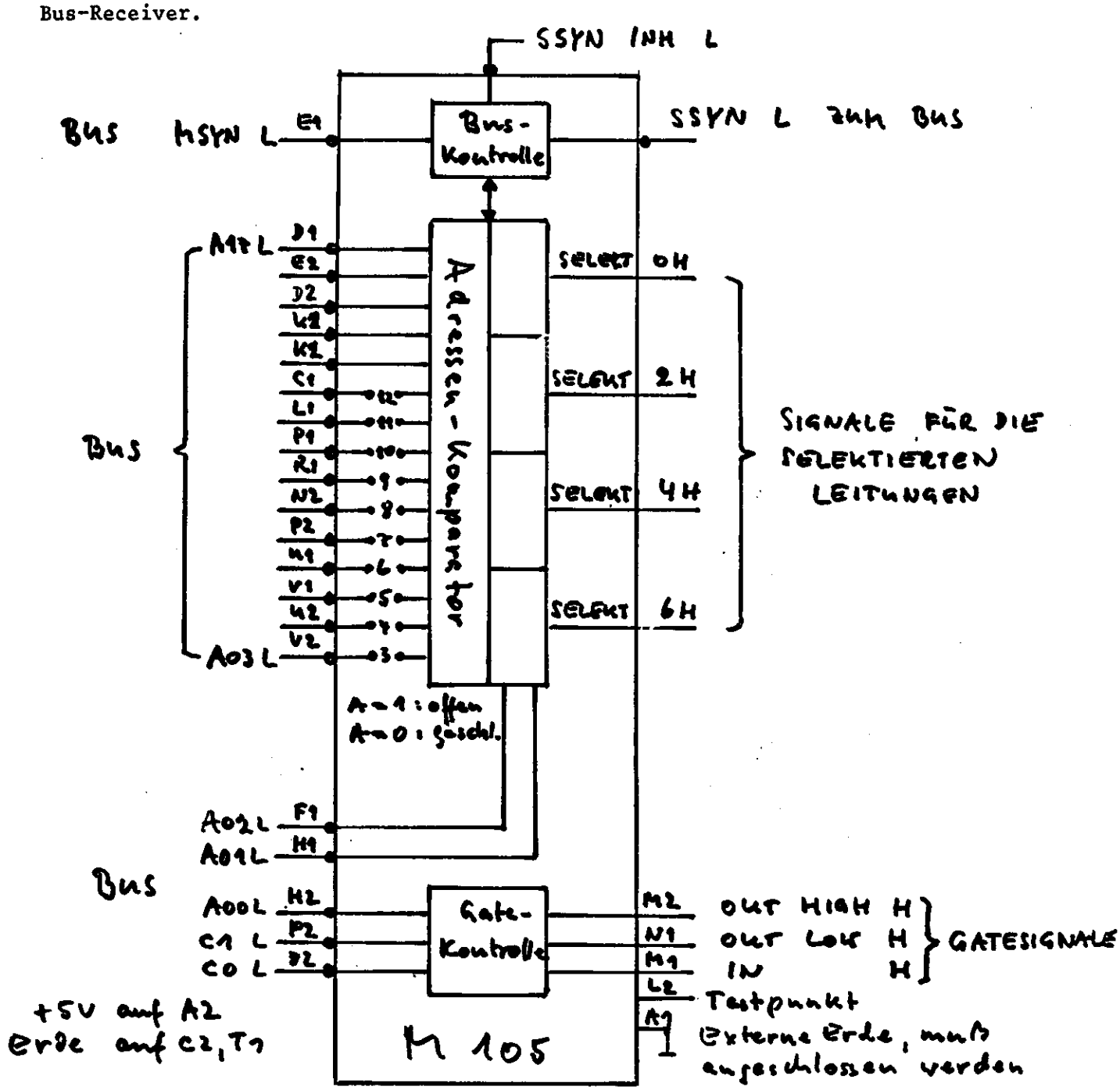
5.22

Die Eingänge können mit je 4 TTL-Belastungen beschaltet werden, die Ausgänge treiben je 50 mA bei 0.8 V Max. Der Opencollektor-Leckstrom ist 25 µA.

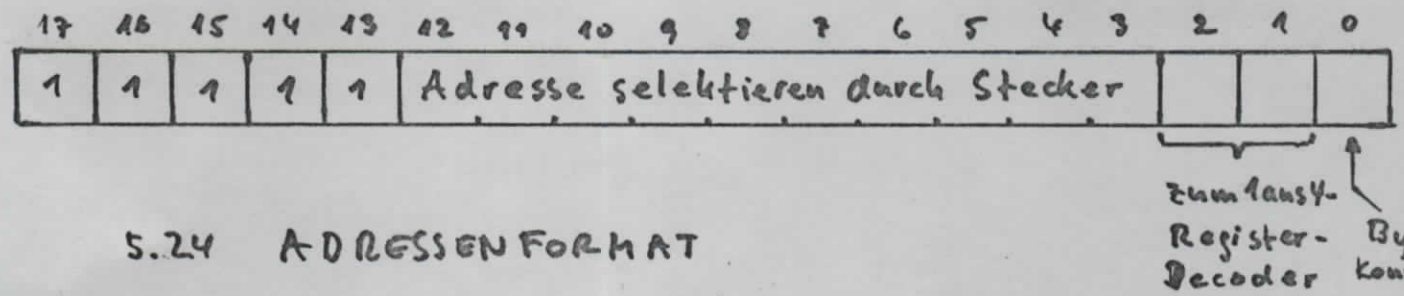
5.3.4.5 M105 Adress Selectormodule

Dieser Einschub vergleicht die vom Bus kommende Adresse mit der eigenen und selektiert 1 von 4 Geräteregistern. Das Blockdiagramm zeigt Bild 5.23 , die Bezeichnungen IN oder OUT sind auf den Master bezogen, wenn dieses Modul in einem Peripheriegerät steckt, geht der OUT-Verkehr vom Master, z.B. dem Prozessor zum Gerät, der IN-Verkehr zum Master.

Die M105-Eingangssignale enthalten 18 Adressleitungen A<17:00>, 2 Buskontrollleitungen C<1:0> und die Mastersynchronisation MSYN. Der Adreßselektor dekodiert die 18Bit-Adresse, deren Format Bild 5.24 zeigt. Alle Eingänge sind Standard-Bus-Receiver.



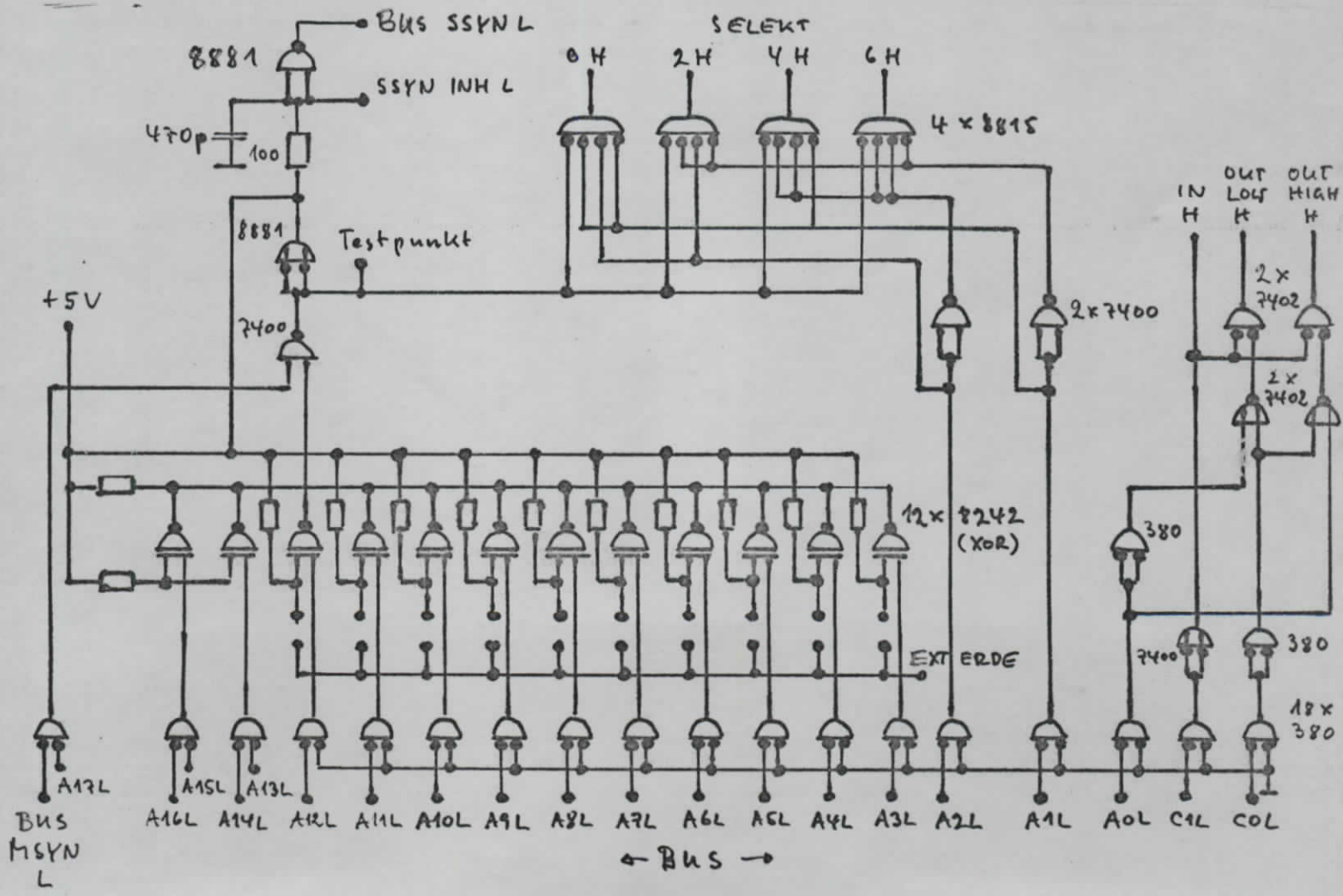
5.23 BLOCKBILD DES ADRESSENSELEKTORS



5.24 ADRESSENFORMAT

Die Leitung A00 wird für die Bytekontrolle benutzt. Leitungen A01 und A02 werden decodiert und selektieren eines der vier adressierbaren Register. Die Geräteadresse wird durch die Verbindungen festgelegt. Ist die Leitung verbunden, stellt sie eine 0 dar, ist sie offen, eine 1. Die Adressenleitungen A<17:13> müssen alle auf 1 liegen, damit die Adresse im zugeteilten Adressenraum ist (vgl. Abschnitt 5.2.5).

Bild 5.25 zeigt das Schaltbild des M105-Moduls.



Alle Widerstände 1K, 0.25 W, 5%, wenn nicht anders angegeben
Benötigt +5V, 340 mA; +5V auf A2, Erde auf C2, T1

5.25 SCHALTBILD DES M105-MODULS

Ist SSYN INH geerdet, verhindert es das Erzeugen des SSYN, der normalerweise im Adressen-Selektor M105 entsteht. In diesem Fall muß ein anderes Gerätden SSYN produzieren. Ist SSYN INH nicht geerdet, wird SSYN 100 nsec, nachdem das Selektieren des Registers erfolgt ist, an den Master geschickt. Die Zeit kann bis 400 nsec verlängert werden, wenn zwischen SSYN INH und Erde ein Kondensator geschaltet wird.

Die Ausgangssignale des M105 selektieren 1 von 4 Registern und liefern drei Gerätesignale vom oder an den Master.
In Tabelle 5.4 sind die Eingangskombinationen zum Selektieren der Register angegeben, in Tabelle 5.5 die Gatekontrollsignale.

Tabelle 5.4
M105-Selektierte Leitungen

Eingangsleitungen A<02:01>	Selektiert (+3V)
0 0	0
0 1	2
1 0	4
1 1	6

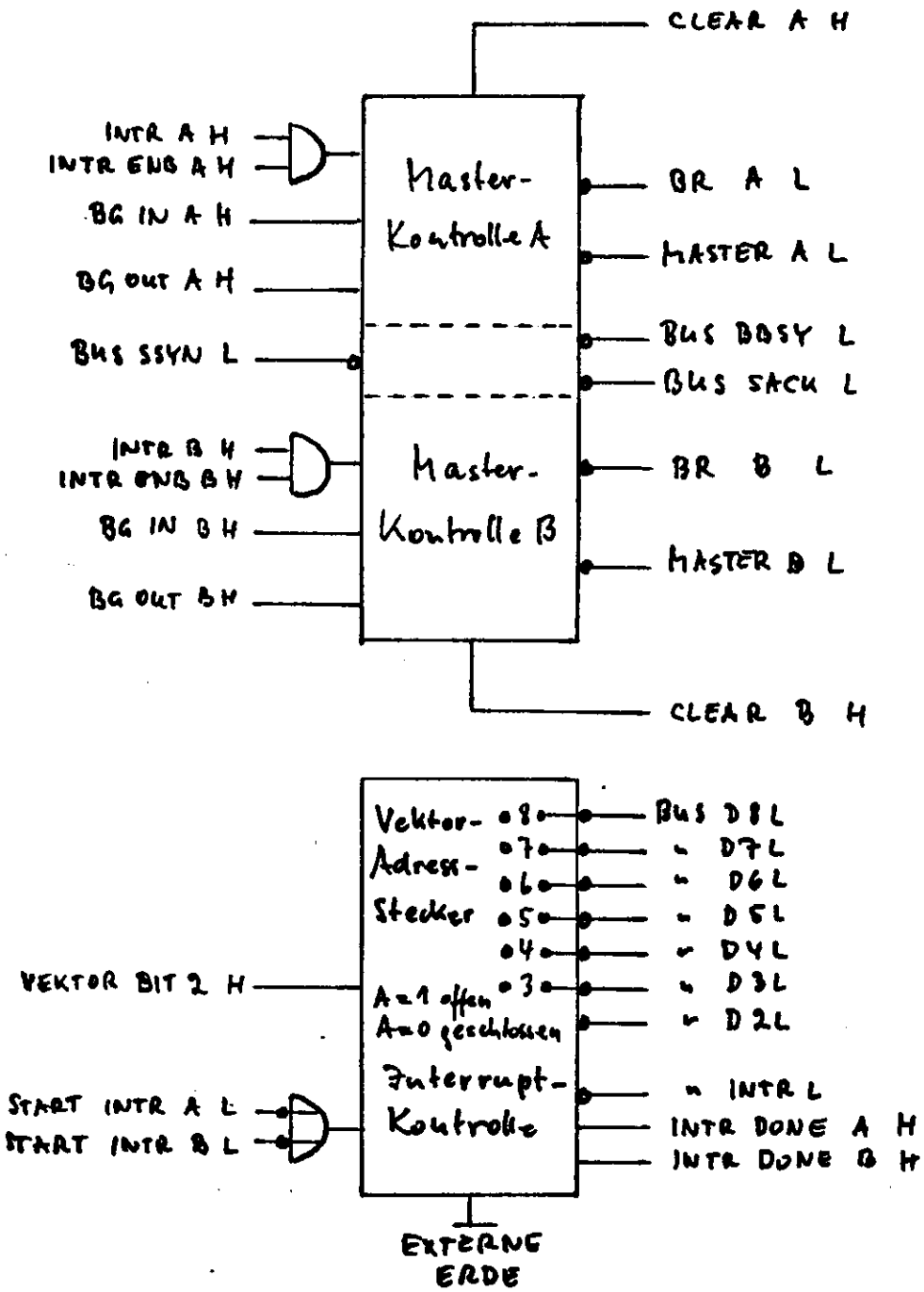
Leitungen A<17:13> sind alle 1, d.h. OV auf dem Unibus.
Leitungen A<12:03> werden durch Verbindungen bestimmt.

Tabelle 5.5
M105-Gatekontrollsignale

Richtungskontrolle C <1:0>	Bytekontrolle A 00	Gatesignal "Wahr" (+ 3V)	BUS-Operation
0 0	0	IN	DATI
0 0	1	IN	DATI
0 1	0	IN	DATIP
0 1	1	IN	DATIP
1 0	0	OUT LOW, OUT HIGH	DATO
1 0	1	OUT LOW, OUT HIGH	DATO
1 1	0	OUT LOW	DATOB
1 1	1	OUT HIGH	DATOB

5.3.4.6 M7820 Interrupt-Control-Module

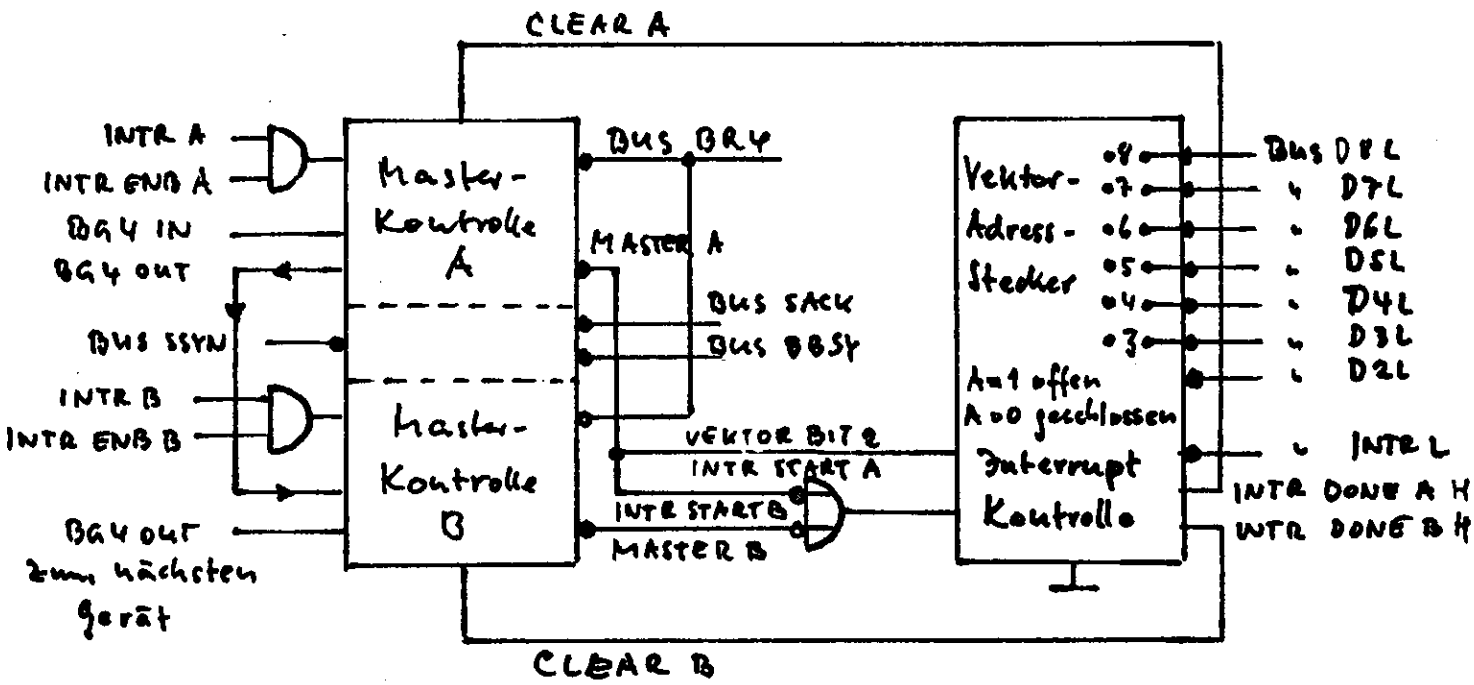
Diese Einheit enthält die Schaltungen zur Erzeugung des Busreguests und zum Erlangen der Kontrolle über den Bus. Zusätzlich sind Logikbausteine enthalten, die die Interruptvektoren bereitstellen. Der Modul enthält zwei unabhängige Request- und Grantlogikeinheiten (Kanal A und B). Die Interruptvektoren werden für beide Kanäle erzeugt. Bild 5.26 zeigt ein Blockbild der Einheit.



5.26 M 7820 INTERRUPT KONTROLLE

Wird INTR und INTR ENB gleichzeitig gesetzt, wird auf der zugehörigen BR-Leitung (in Kanal A oder B) ein Busrequest ausgelöst, der durch die Prioritätslogik des Systems je nach der BR-Leitungsnummer erkannt und durch ein Busgrantsignal beantwortet wird. Darauf erzeugt der M7820-Modul das SACK-Signal. Sind alle Voraussetzungen gegeben, um Busmaster zu werden, setzt der Modul das BBSY-Signal sowie ein internes Mastersignal.

Ist die Einheit Master geworden, kann ein Interrupt erzeugt werden. Dazu wird die Einheit, die in diesem Beispiel die Priorität 4 hat, wie folgt zusammengeschaltet (vgl. Bild 5.27):



Beispiel: Bus-Request erscheint auf BR4
Interrupt A auf Vektoradresse 100
Interrupt B auf Vektoradresse 104

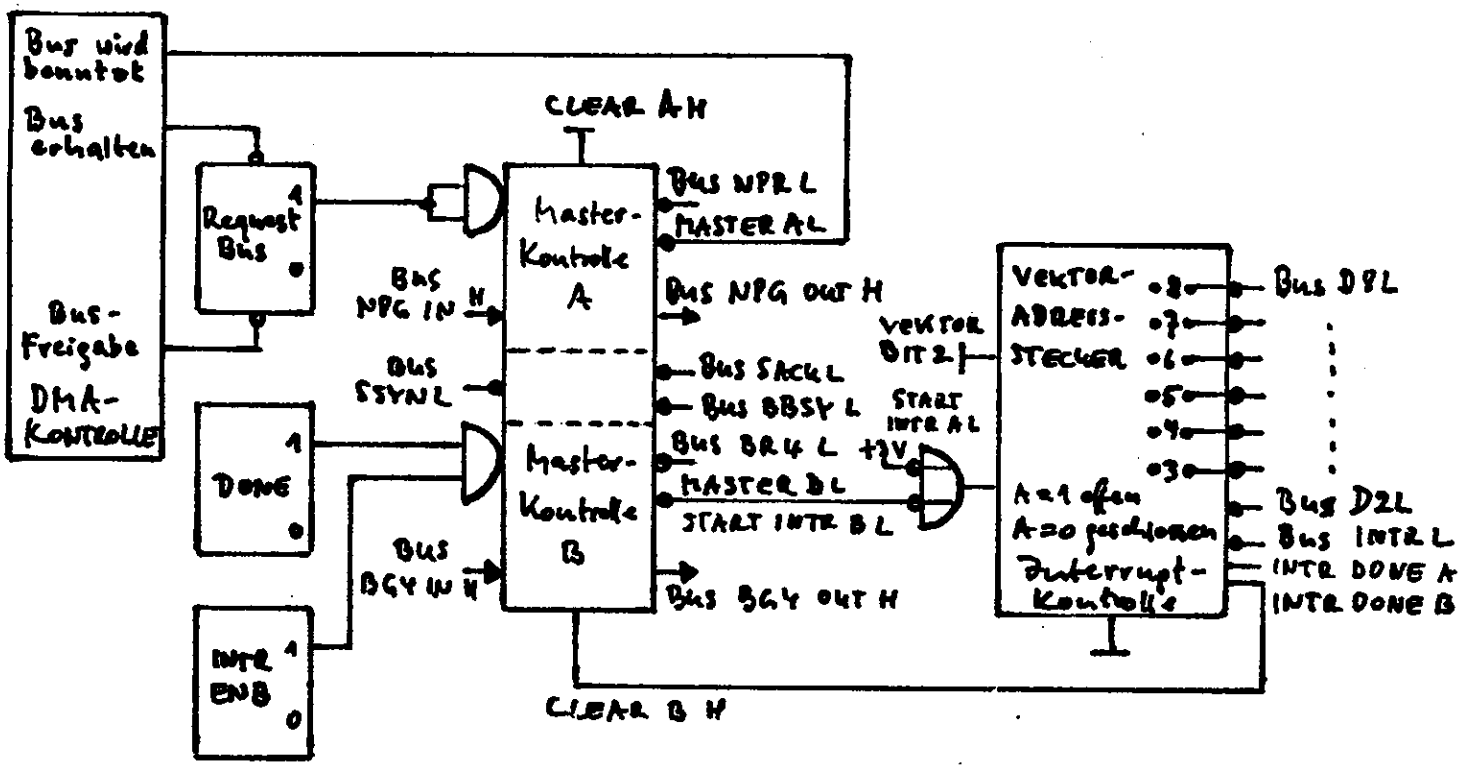
5.27 M7820 - SCHALTUNG MIT INTERRUPT AUF 2 KANÄLEN

In diesem Bild wird gezeigt, wie die zwei Kanäle benutzt werden, um erst den Request zu erzeugen und dann den Interrupt zu starten. Der Request von Kanal A hat eine etwas höhere Priorität als der von Kanal B, da das Grantsignal zuerst durch A läuft.

Die Vektoradresse wird durch die gesteckten Verbindungen bestimmt. Da der Vektor ein Block aus zwei Worten (vier Bytes) ist, braucht man die Bits 0 und 1 nicht festzulegen, sondern nur die restlichen sieben Bits. Die LSB-Leitung wird durch

das VEKTOR-BIT 2-Signal beschrieben. Ist das Signal gesetzt, wird auch die DO2-Leitung gesetzt (hier auf L). Ein Interrupt in Kanal A erzeugt einen Vektor der Platznummer 100, einer in Kanal B zur Nummer 104.

Das nächste Bild 5.28 zeigt die interne Verbindung des M7820-Moduls, das sich in einem Gerät befindet, das direkten Datentransfer zum Speicher (DMA-Mode) betreibt und dann einen Interrupt erzeugt, wenn der Transfer beendet ist. Der Kanal A ist mit der NPR- und NPG-Leitung verbunden, ist also für Modul-zu-Modul-Transfer (MMT) vorgesehen; Kanal B soll die Buskontrolle anfordern, wenn ein Interrupt erzeugt werden soll.



5.28 M7820 FÜR DMA-BETRIEB

Jeder Master-Kontroll-Kanal hat zwei Flip-Flops, die vier Zustände annehmen können, von denen allerdings nur zwei benutzt werden und zwar

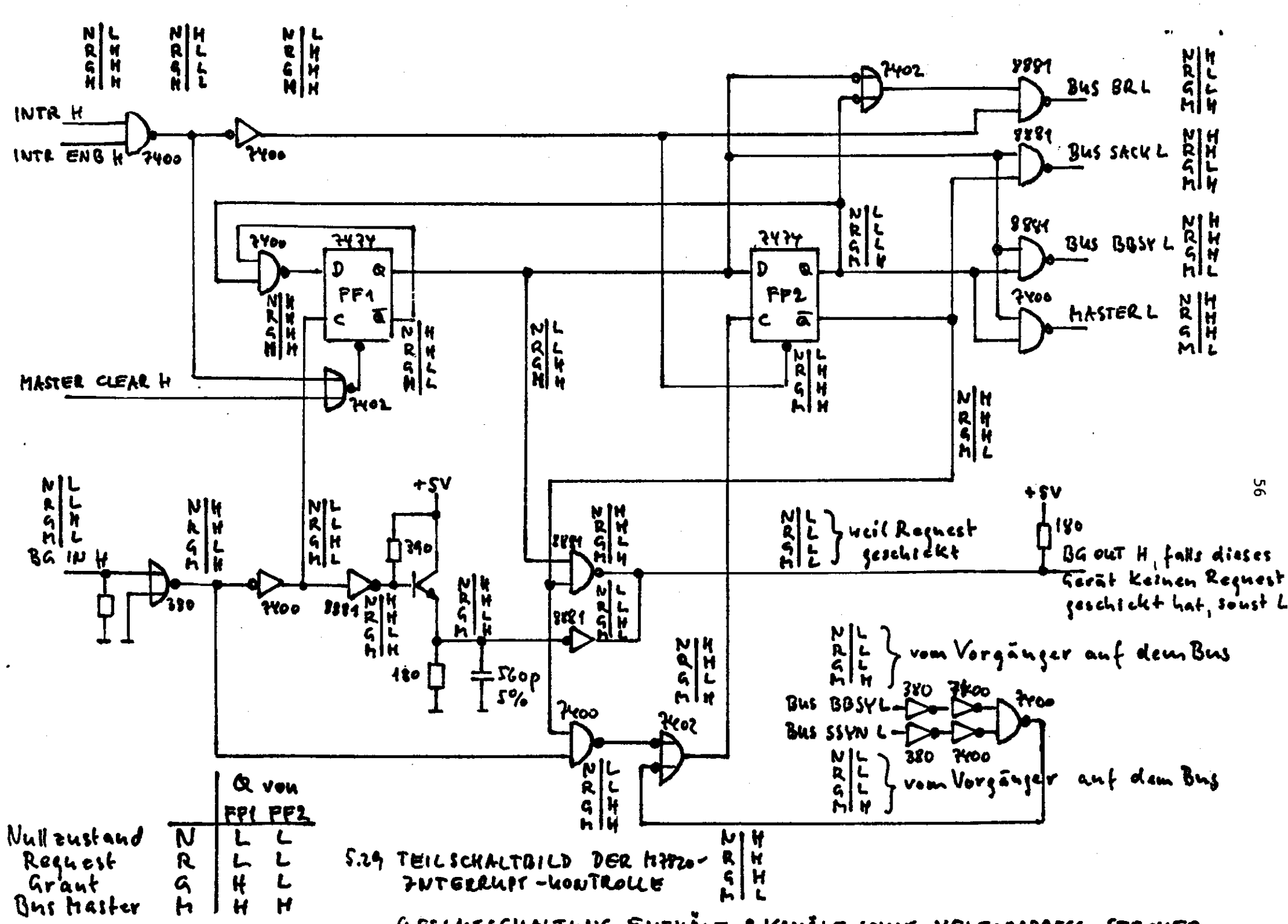
- Nullzustand bzw. Aussenden eines Requests
- Grantsignal-Annahme mit SACK-Antwort
- Busmasterzustand mit BBSY-Signal

Die Gesamtschaltung des Moduls zeigt Bild 5.29.

Soll ein Request ausgesendet werden, werden sowohl der INTR als auch der INTR ENB gesetzt (die Buchstaben H, L in der Schaltung beziehen sich immer auf gesetzte Signale, die geclearten Signale haben umgekehrte Polarität). Dadurch werden die im Nullzustand geclearten Flip-Flops freigegeben, am Clockeingang des FF1 liegt L, an FF2 H, dadurch liegen beide Q-Ausgänge an L, die Requestleitung geht auf L. Von der Prioritätslogik wird das Grantsignal geschickt, das einerseits den FF1-Clockeingang auf H setzt und das an D liegende H auf den Q-Ausgang schaltet (Flip-Flop SN7474 ist ein flankengetriggerter ⌐↓ D-Typ., andererseits über die RC-Verzögerung im Emitter des Transistors dieses Umschalten abwartet. Hätte diese Einheit keinen Request ausgelöst, wäre der Flip-Flop nicht geschaltet, da L an Clear liegen würde, und somit wäre das BG-Signal ungehindert zum nächsten Gerät weitergelaufen. Nach dem Schalten jedoch hat das NAND-Gate am Ausgang ein L-Signal und damit bleibt die BG-Leitung auf L.
Das Schalten des Q von Flip-Flop 1 auf H läßt die SACK-Leitung auf L gehen, d.h. das Grantsignal wird quittiert. Während dieser Phase ist der Clockeingang von Flip-Flop 2 auf L, ebenfalls vom Grantsignal gesteuert. Der BBSY vom Vorgänger auf dem Bus geht dann von L auf H, wenn dieser seine Operation beendet, damit wird der Clockeingang von Flip-Flop 2 auf H geschaltet, so daß dessen D (auf H liegend) an das Q übertragen wird. Jetzt, nach Erreichen des dritten Zustands, ist das Gerät Master geworden, das BBSY-Signal wird gesetzt.

Der Request kann nun zurückgenommen werden und zwar entweder durch Wegnahme von INTR bzw. INTR ENB oder durch Setzen des Master Clear-Signals. Die zweite Methode verhindert weitere Busrequests, auch wenn INTR und INTR ENB weiterhin gesetzt bleiben. Um einen neuen Busrequest zu erzeugen, müssen die beiden Signale erst zurückgenommen und dann erneut gesetzt werden. Dadurch werden mehrfache Interrupts des Bus-Masters verhindert. Die erste Methode wird angewendet, wenn der Bus nach einem NPR wieder freigegeben wird.

Eine neuere Version des M7820, der M7821, läßt die Möglichkeit des mehrfachen Interrupts zu. Diese Einheit, die voll kompatibel zu der M7820 ist, hat zwei nicht identische Masterkontrollteile. Wenn, über ein NPR-Signal, das Gerät Master wer-



	Q von FF1	FF2
Nullzustand	N	L L
Request	R	L L
Grant	G	H L
Bus Master	M	H H

5.29 TEILSCHALTBILD DER M7820-7NTERRUPT-KONTROLLE

GESAMTSCHALTUNG ENTHÄLT 2 KANÄLE SOWIE VEKTORADRESS-STECKER MIT BUSTREIBERN

den will, muß die obere Hälfte dafür benutzt werden, während die untere Hälfte für den BR-Verkehr verwendet werden muß. Die NPR-Hälfte hat die Möglichkeit, das SACK-Signal zu belassen, falls mehr als ein Datenzyklus pro Request ausgeführt werden soll. Dies erreicht man durch Hochlegen des Pin I2 (Erdpunkt der Vektoradreßverbindungen in M7820) bis zum Beginn des letzten Bus-Zyklus. Sobald I2 auf L geht, verschwindet SACK, das Signal an I2 kann ein Puls oder ein Level sein. I2 ist nur aktiv, wenn das Mastersignal gesetzt ist, daher kann I2 auch dauernd geerdet sein, falls nur ein Zyklus pro Request ablaufen soll (Kompatibilität zun M7820).

Die BR-Hälfte schaltet das SACK-Signal grundsätzlich zurück, falls BBSY gesetzt ist.

Die beiden Hälften können aber unabhängig voneinander für NPR- oder BR-Operationen verwendet werden, dann aber beide nur für einen Datenzyklus pro Request, im letzteren Fall muß I2 geerdet werden.

Die Interruptschaltung ist auch leicht geändert. Die Verbindungen werden für Vektoradresse = 0 geöffnet, für 1 geschlossen.

5.3.4.7 M796 UNIBUS Master-Control-Module

Diese Einheit kontrolliert die Datenoperationen auf dem UNIBUS, wenn das Gerät als Master arbeitet. Neben der Kontrolle der 4 Operationsarten DATI, DATIP, DATO und DATOB erzeugt die Einheit Strobe- und Gatesignale zum Senden bzw. Empfangen von Daten und Adressen. Außerdem werden die Zeitverzögerungen generiert, die einige Kontrollsignale warten müssen, bis sich Daten oder Adressen auf dem UNIBUS befinden und eingeschwungen haben. Dieses Problem wurde in dem Abschnitt 5.2.2.1 besprochen. Außerdem enthält der Modul Schaltungen für Time-out-Signale zum Schutz gegen Datentransfer in nichtexistierende Geräte und einen Flip-Flop und Univibrator für spezielle Kontrollfunktionen des Benutzers.

Den Busrequest macht das vorher beschriebene Interrupt Control-Module auf den BR- bzw. NPR-Leitungen, erhält das Grantsignal vom Prozessor, setzt das SACK-Signal, wartet, bis der Vorgänger den Bus freigibt, setzt das BUS BBSY und erhält so die Bus-Kontrolle. Damit wird das Gerät frei für den Datentransfer. Damit dieser ausgeführt werden kann, muß das Gerät BUS C<1:0> setzen, um die Transferrichtung anzugeben, die Adresse des Slave setzen, das MMSYN senden und den Empfang des SSYN abwarten. Die Daten werden dann entweder auf D<15:00> gegatet oder

von den D-Leitungen in das Gerät hineingestrobt. Alle diese Kontrollfunktionen führt die M796-Einheit aus.

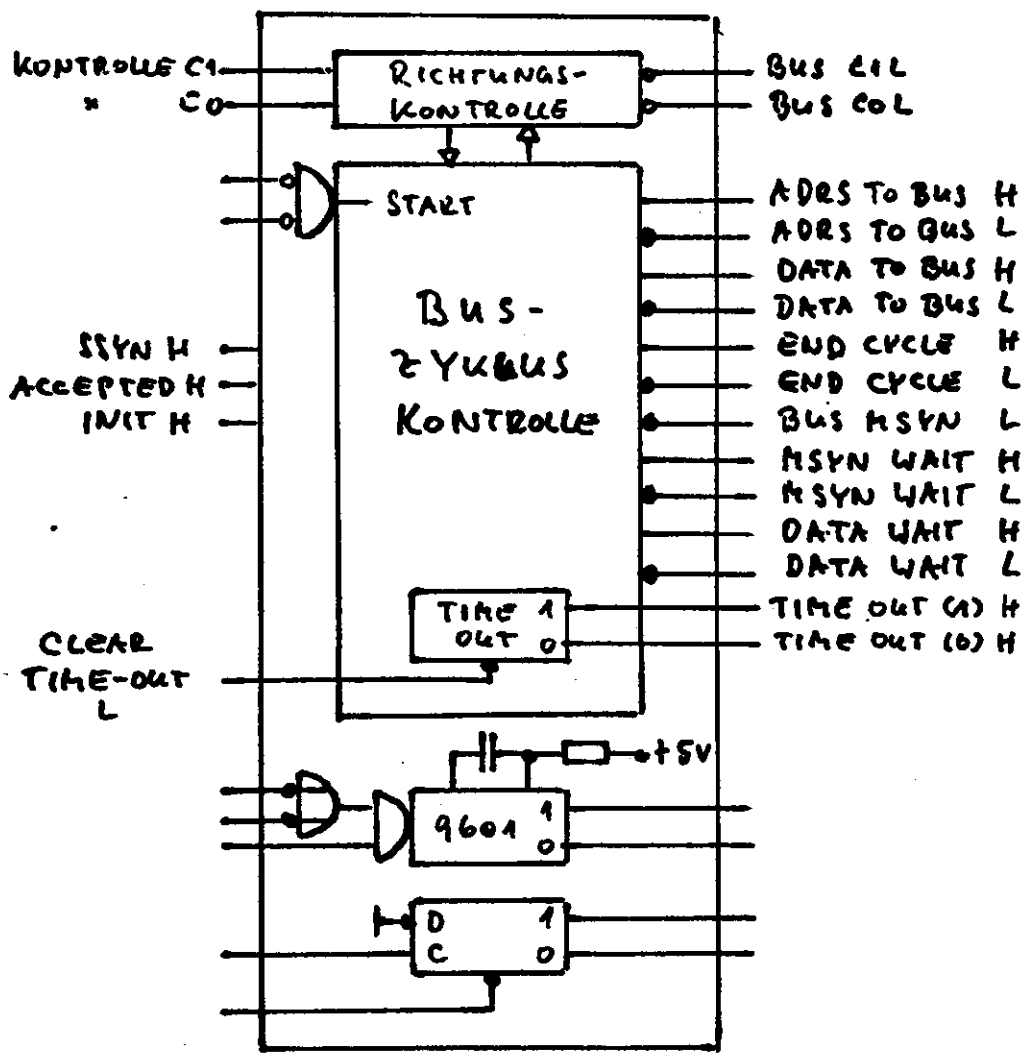
Bild 5.30 stellt das Blockdiagramm des Master-Control-Module dar. Die BUS C1 und BUS CO-Ausgänge können direkt den UNIBUS steuern, sie werden von Kontrolleingängen C1 und CO gesetzt. Tabelle 5.6 zeigt die vier Zustände der C-Leitungen und die Art des Transfers, die daraus folgt.

Tabelle 5.6

Kontrolleingänge des M796

C1	Co	Bus-Zyklus
0	0	DATI
0	1	DATIP
1	0	DATO
1	1	DATOB

O bedeutet L, also 0 V, 1 bedeutet H, also + 3 V.



5.30 BLOCKDIAGRAMM DES M796

Der Datentransfer wird ausgelöst durch die UND-Bedingung der zwei L-Level, die meist verbunden werden und vom MASTER-Signal aus der Interrupt-Kontrolle gesteuert werden, zusammen mit BBSY. Das darauf entstehende START-Signal wird intern in der Master-Kontrolle benutzt. Dadurch werden BUS C1 und BUS CO gesetzt, ebenfalls ADRS TO BUS, so daß die Slave-Adresse auf BUS A<17:00> gegatet wird.

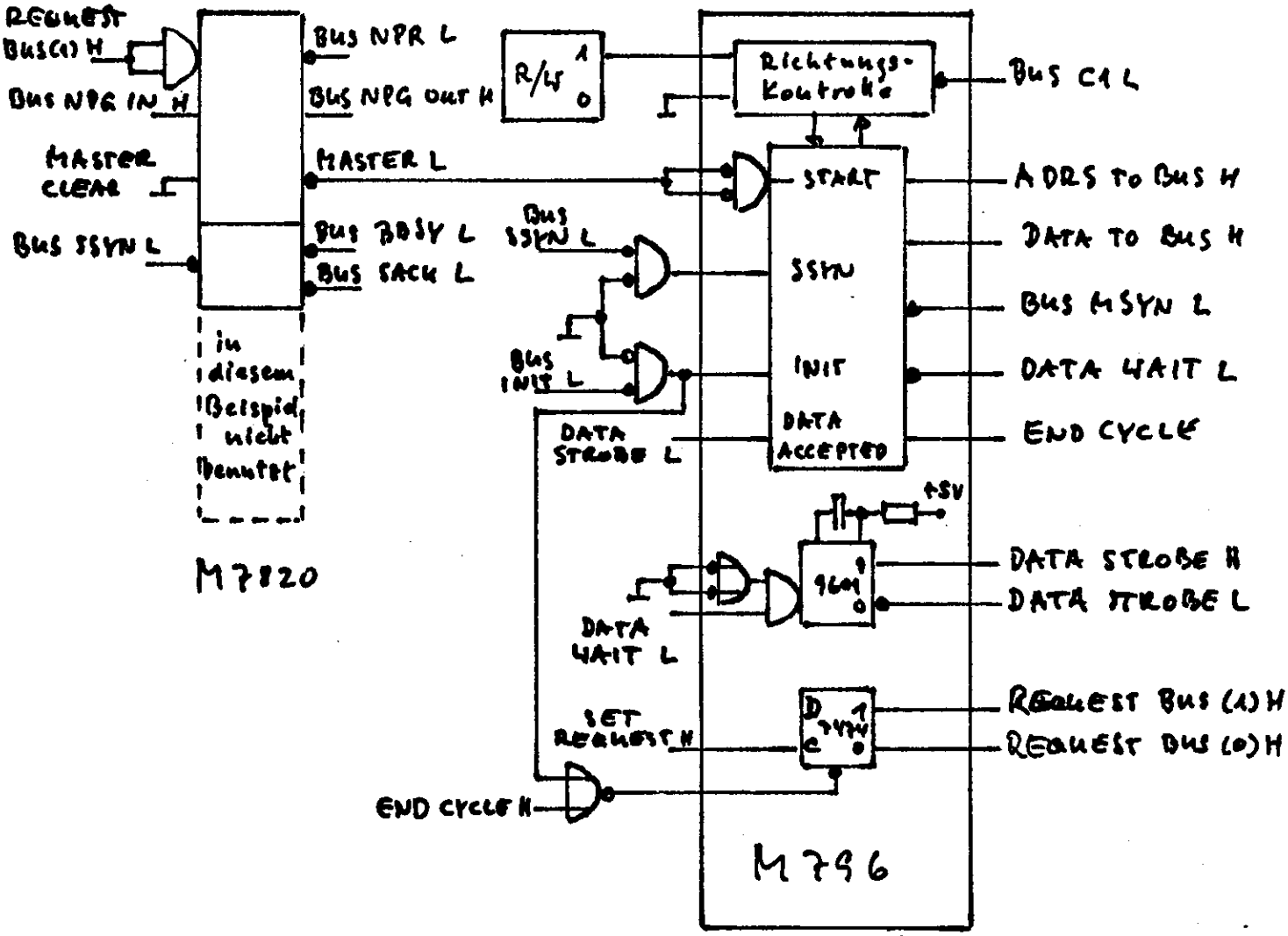
Handelt es sich bei dem Transfer um einen OUT-Zyklus (C1=1), wird das DATO TO BUS-Signal gesetzt, das die Daten für den Slave auf die D<15:00>-Leitungen des BUS gatet. BUS MSYN wird 200 nsec nach dem START gesendet, die Master-Kontrolle wartet dann auf SSYN vom Slave. Kommt dieses, verschwindet MSYN sofort, während BUS C1, BUS CO, ADRS TO BUS und DATA TO BUS erst nach 100 nsec zurückgenommen werden. Dadurch erscheint der END CYCLE-Puls, der die Freigabe der Bus-Kontrolle anzeigt.

Ist der Transfer ein IN-Zyklus (C1=0), erzeugt das SSYN-Signal mit seiner Vorderflanke einen 200 nsec-Puls, der DATA WAIT genannt wird. Während dieser Zeit formieren sich die ankommenden Daten auf dem Bus. Die Rückflanke des DATA WAIT kann zum Hereinstroben der Daten in den Master benutzt werden. Sonst muß ein gesonderter Strobepuls aus der Rückflanke von DATA WAIT und dem eingebauten Univibrator 9601 erzeugt werden. Sind die Daten im Master, geht eine positive Flanke vom Strobe auf DATA ACCEPTED, worauf zunächst BUS MSYN und 100 nsec später ADRS TO BUS, BUS C1 und BUS CO verschwinden.

Der TIME-OUT Flip-Flop wird gesetzt, falls SSYN nicht innerhalb von 20 µsec nach Setzen von BUS MSYN empfangen wird. In diesem Fall wird der Bus-Zyklus nicht vollendet. Der Flip-Flop muß durch Setzen von CLEAR TIME-OUT zurückgestellt werden.

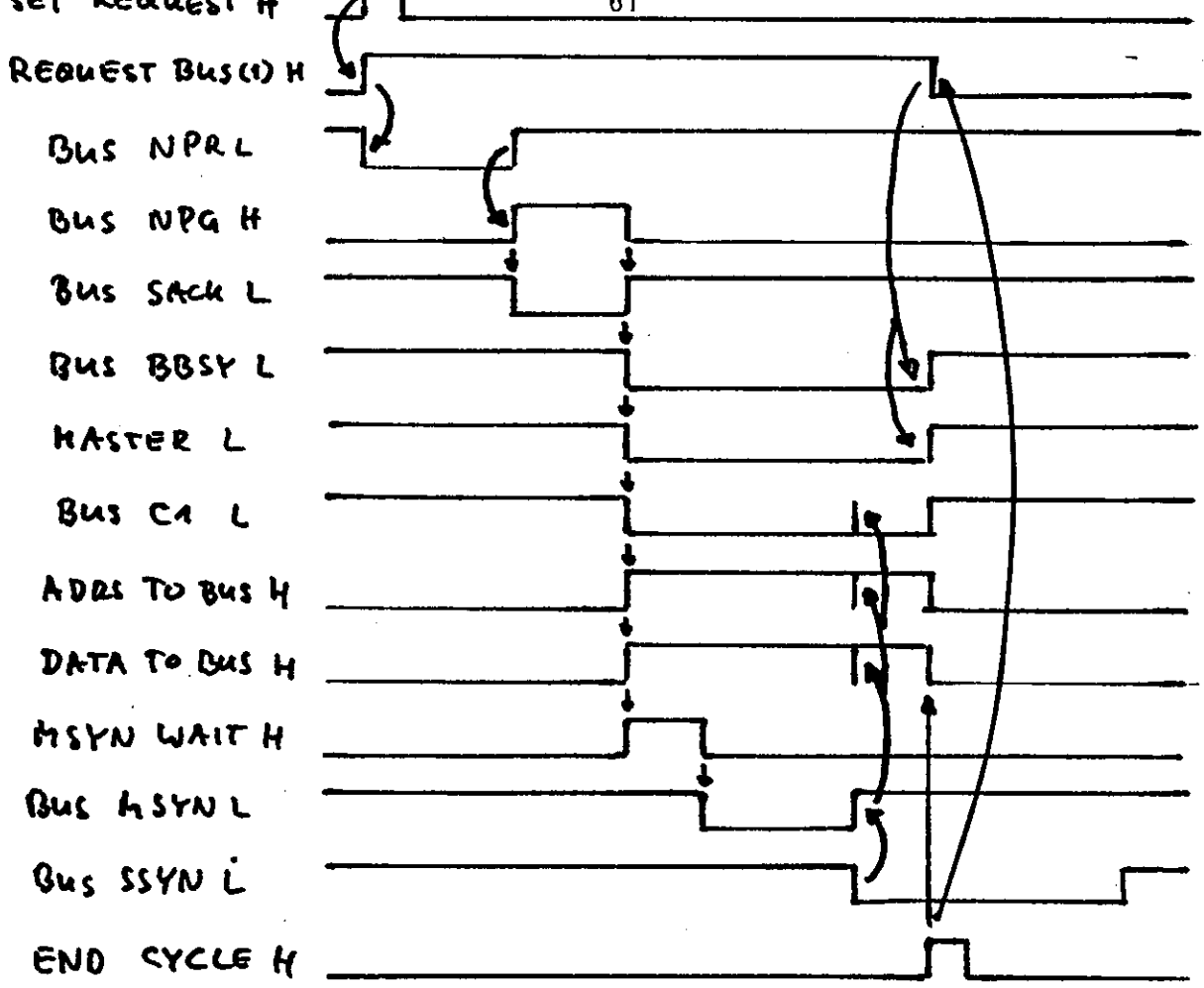
Ein zusätzlicher flankengetriggerter Flip-Flop kann für Kontrollfunktionen des Benutzers verwendet werden. Der ebenfalls gesondert eingebaute Univibrator hat eine Pulsdauer von 150 nsec, die durch externe Kondensatoren verlängert werden kann.

Bild 5.31 zeigt eine typische Kombination der Master-Kontroll-Einheit und der Interrupt-Kontroll-Einheit.

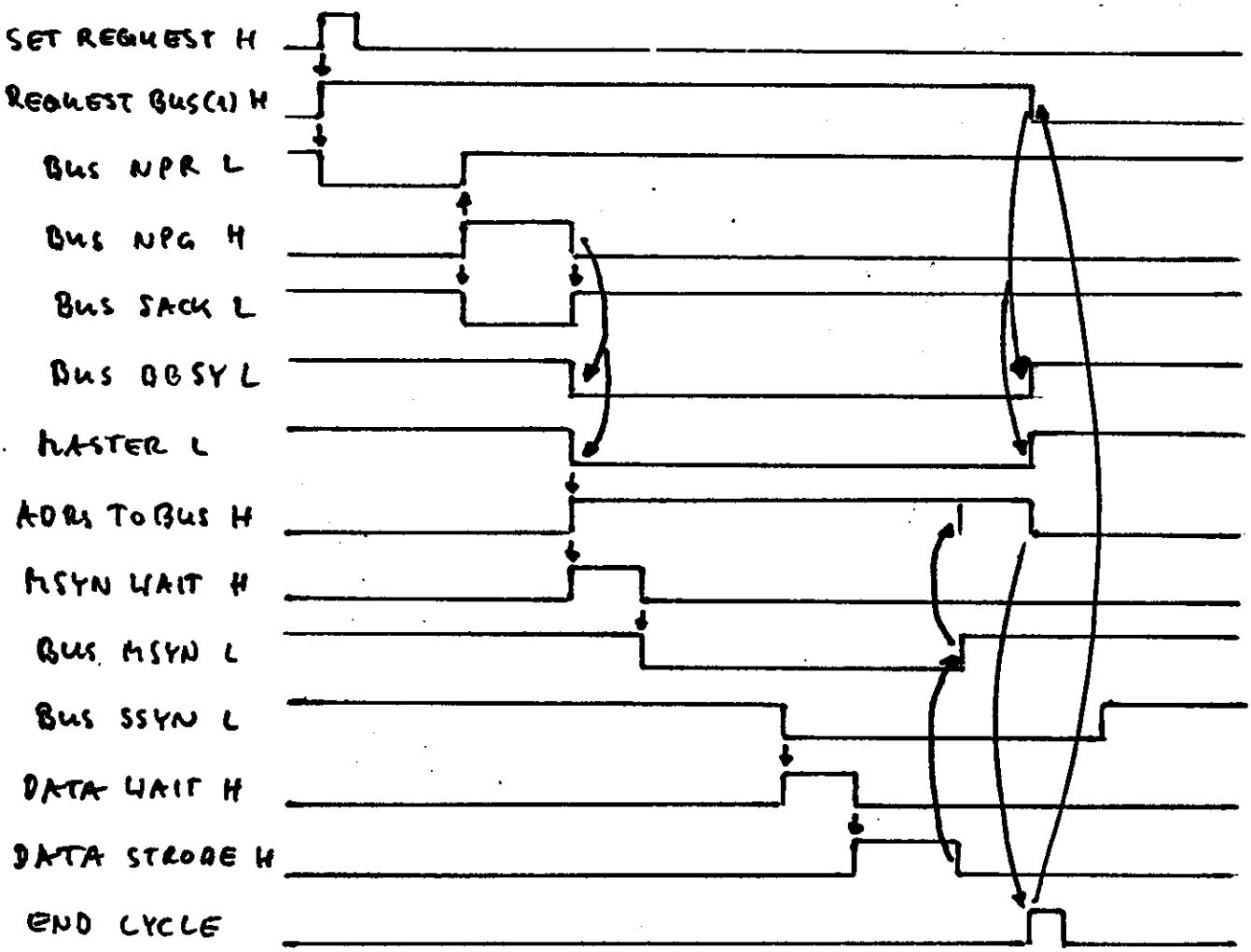


5.31 TYPISCHE SCHALTUNG ZWISCHEN INTERRUPT- UND MASTER-KONTROLLEINHEIT

Der Read/Write (R/W)-Flip-Flop ist Teil der Geräteinterfacelogik und bestimmt die Richtung des Datentransfer (gesetzt bei DATO, clear bei DATI). Der Datentransfer wird eingeleitet durch einen Puls auf SET REQUEST, wodurch REQUEST BUS gesetzt wird. Dieses erzeugt einen NPR, der, wenn er erteilt wird, die Bus-Kontrolle an den neuen Master gibt, was durch das Erscheinen des MASTER-Signals angezeigt wird. Dieses wiederum erzeugt das innere START-Signal mit den schon beschriebenen Folgesignalen. Zeitabläufe für DATO und DATI sind in den folgenden Bildern 5.32 und 5.33 gezeigt.



5.32 ZEITDIAGRAMM FÜR DATO MIT M796-MODUL



5.33 ZEITDIAGRAMM FÜR DATI MIT M796-MODUL

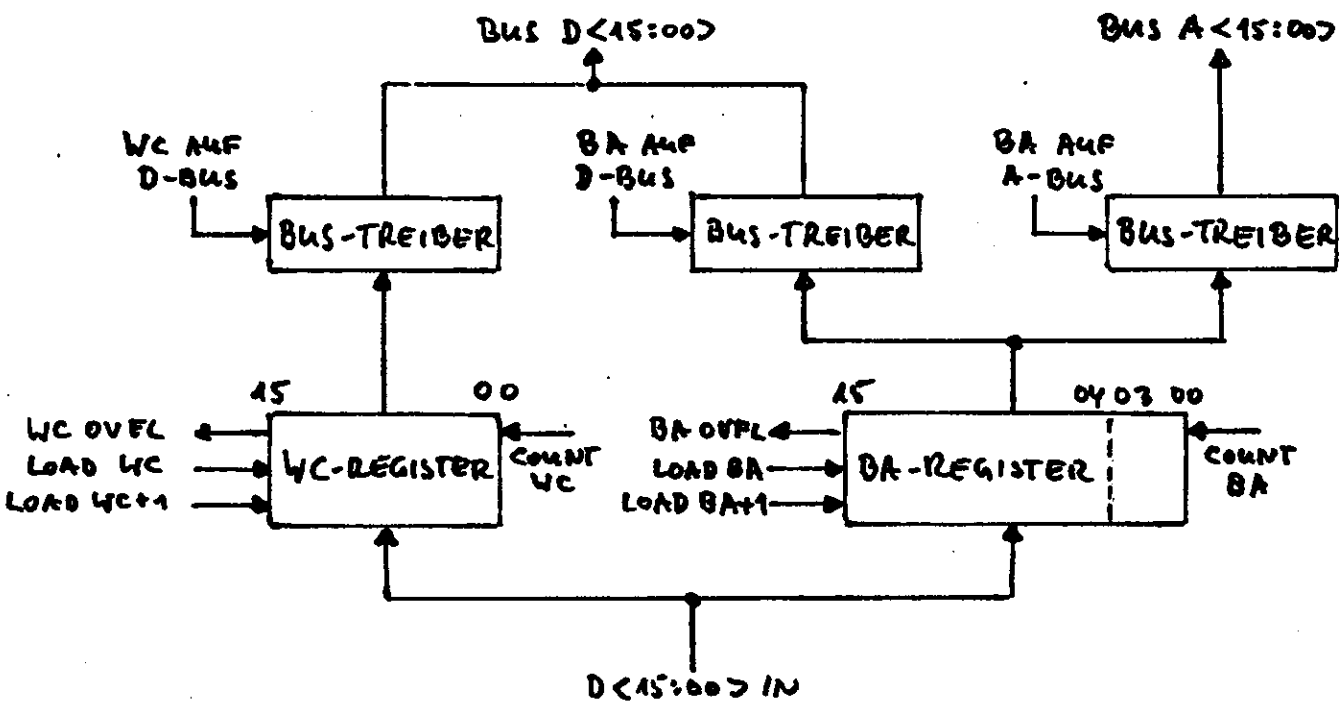
In der DATI-Operation wird das DATA WAIT-Signal erzeugt, wenn BUS SSYN erscheint. Die Rückflanke von DATA WAIT triggert den Univibrator, der daraufhin das DATA STROBE-Signal liefert, mit dem die auf dem Bus stehenden Daten in das Gerät gestrobt werden. Die positiv gehende Rückflanke des Univibrators geht an DATA ACCEPTED, worauf das BUS MSYN gecleart wird.

5.3.4.8 M795 Word Count and Bus Address Module

Diese Einheit wird benutzt, um solche Geräte an den Bus zu schalten, die im Blocktransfer Direct-Memory-Access (DMA) ausführen wollen. Sie enthält zwei 16-Bit-Zähler, einen, der die Zahl der Datentransferzyklen registriert, den anderen, um die jeweilige Busadresse anzuzeigen, in die die Daten transferiert werden.

Da beim Blocktransfer meist Worte übertragen werden, heißt der erste Zähler Word-Count (WC). Er wird zu Beginn der Übertragung mit dem 2er Komplement der Zahl der zu übertragenden Worte geladen. Nach jedem beendeten Zyklus wird der Inhalt inkrementiert. Ist der neue Inhalt des Zählers 0, angezeigt durch einen Overflow, wird der Transfer beendet. Da über den UNIBUS auch Bytes transportiert werden können, muß das Register auch als Byte -Countregister arbeiten.

Das Bus-Adress-Register (BA) wird zu Beginn mit der Adresse der Speicherzelle geladen, in die das erste Wort transportiert wird. Nach jedem erfolgten Zyklus wird der Inhalt inkrementiert, d.h. die Adresse um 1 erhöht.
Bild 5.34 zeigt das Blockdiagramm.



5.34 BLOCKDIAGRAMM DES WORDCOUNT- UND BUSADRESSREGISTERS

Beide Register enthalten je 16 Flip-Flops, die mit den Daten aus den D-Leitungen geladen werden können, wenn das entsprechende LOAD-Signal angelegt wird. Es gibt vier getrennte LOAD-Signale:

- WC highbyte
- WC lowbyte
- BA highbyte
- BA lowbyte

Die Ausgänge aller Flip-Flops in beiden Registern können über DEC 8881-Treiber auf den Bus gegated werden, wenn das zugehörige Gatesignal

- WC TO D BUS
- BA TO D BUS
- BA TO A BUS

gesetzt wird. Die Treiberausgänge des WC- bzw. BA-Registers für die D-Leitungen sind in WIRED OR verbunden. Die Flip-Flops sind nicht flankengesteuert, die Daten müssen während der Dauer des Ladepulses präsent sein.

Sowohl das WC- als auch das BA-Register können zur Byteübertragung um 1 oder 2 im Inhalt erhöht werden. Dazu haben sie einen Kontrolleingang, an dem entweder + 3V oder Erde gelegt werden muß, um dies zu erreichen. Die Inkrementierung wird während der Rückflanke eines positiven Zählpulses am Eingang erreicht. Der Pulsoverflow wird angezeigt; durch ein CLEAR können beide Register auf Null gestellt werden.

Beim BA-Register ist der Zählübertrag zwischen Bit 03 und 04 unterbrochen und auf Pins im Modul geführt. Normalerweise werden diese Pins überbrückt, um eine Zählung bis 16 Bit zuzulassen. Ist die Verbindung unterbrochen, können nur 4 Bit gezählt werden. Dies dient der wiederholten Adressierung von 16 aufeinanderfolgenden Bytes und ist manchmal nützlich beim Modul-zu-Modul-Transfer. Tabelle 5.7 stellt die Eingangssignale des M795-Moduls zusammen, Tabelle 5.8 die Ausgangssignale.

Tabelle 5.7

Eingangssignale

Signal	gesetzter Pegel	Zahl der Signale	Operationen
D <15:00> IN	+ 3V = 1	16	Dateneingang zu den Registern
LOAD WC LOAD WC + 1	0V	2	Lädt die Daten in den Eingang des selektierten Bytes im Register
LOAD BA LOAD BA + 1	0V	2	L-Pulse von mindestens 250 nsec Dauer
WC TO D BUS BA TO D BUS BA TO A BUS	0V	3	Gatet das selektierte Register auf den Bus
CLEAR WC + BA	+ 3V	1	Setzt alle Bit zurück H-Pulse von mindestens 1 µsec Dauer
BA INC CONTROL WC INC CONTROL	+ 3V = 1 0V = 2	2	Kontrolliert die Inkrementbildung
COUNT WC COUNT BA	+ 3V	2	Rückflanke des positiven Pulses inkrementiert. Mindestens 100 nsec Dauer
BA CARRY IN	0V	1	Übertrag geht auch in die oberen 12 Bit des BA-Register

Tabelle 5.8
Ausgangssignale

Signal	gesetzter Pegel	Zahl der Signale	Operationen
BA CARRY OUT	OV	1	Ausgang der unteren 4 Bit
BA OVFL WC OVFL	OV	2	Register-Overflow-Anzeige
BUS D <15:00>	OV = 1	16	Treibt Datenleitungen
BUS D <15:00>	OV = 2	16	Treibt Adressen

5.4. Interface-Beispiele

Die hier gebrachten Beispiele benutzen den UNIBUS, so wie er beschrieben wurde, sowie die vorgestellten Geräte. Nach einem Grundinterface, das die einfachste Lesen/Schreiben-Operation enthält, folgt ein programmgesteuertes Interface, das aus dem Grundbeispiel durch Erweiterung hervorgeht, dann ein interruptgesteuertes Interface und schließlich eines für DMA-Verkehr. Die Schaltungen bauen aufeinander auf.

5.4.1. Grundinterface

Dieses einfache Lesen/Schreiben-Interface wird benutzt, wenn Daten während der Busoperation zu oder von einem Register übertragen werden. Es enthält nur ein Speicherregister und die Gateschaltungen.

5.4.1.1. Interface-Operation

Die Daten werden unter Programmkontrolle und unter Benutzung der Registeradresse übertragen. Der Prozessor arbeitet als Master, das Register als Slave. Benutzt man ein 16 Bit-Register, kann es als Einwortregister oder als Zweibyteregister adressiert werden.

Bild 5.35 zeigt die Teile dieses Interfaces. Zum Decodieren der UNIBUS-Adresse benötigt man den M105-Adress Selector. Das Register enthält auf der Eingangsseite nichtgegatete Empfänger, das Strobesignal zum Laden wird aus der M105-Einheit entnommen. Die Ausgänge des Registers sind zum UNIBUS gegatet und werden über Treber angeschlossen. Die Gates sind nötig, damit der Inhalt des Registers nicht am UNIBUS liegt, während dort andere Operationen ablaufen.

5.4.1.2. Datenübertragungen

Das Interface kann alle 4 Transferarten DATI, DATIP, DATO oder DATOB ausführen.

Wenn der Prozessor einen DATO-Transfer ausführen möchte, adressiert er das Register und schickt die Daten bis zum Registereingang. Die M105-Einheit (s.Abschnitt 5.3.4.5.) erzeugt die Signale OUT HIGH H und OUT LOW H. Ist MSYN vom Prozessor gesetzt und die Adresse decodiert, gaten die beiden OUT H Signale das SELECT 0-Signal, das die Daten mit der positiven Flanke in das Register lädt.

Eine DATOB-Operation läuft ähnlich ab, jedoch wird nur ein Byte des Registers geclockt, denn für A00=0 setzt das M105-Modul nur OUT LOW H, für A00=1 nur OUT HIGH H.

Bei einem DATI-Verkehr adressiert der Prozessor das Interface und setzt MSYN. Darauf erzeugt das M105-Modul wieder das SELECT 0-Signal, das jedoch vom IN-Signal gegatet wird. Dadurch entsteht ein GATE DATA TO BUS-Signal, das die Registerdaten auf den UNIBUS schickt. Das SSYN, das anzeigt, daß die Daten im Interface-Register bereitstehen, wird vom M105-Modul erzeugt.

5.4.1.3. Beispiel einer Interfaceschaltung

Bild 5.36 zeigt eine mögliche Schaltung mit DEC-Moduln. Es werden benutzt:

- M105 Address Selector Module (einmal)
- M785 Bus Transceiver Module (zweimal)
- M206 General Purpose Flip-Flop-Module (dreimal, da jedes Module nur 6 D-Flip-Flops enthält)
- M617 Power NAND Module (einmal)

Die M206 Flip-Flops bilden das Register, das von den Gates des M785-Moduls angesteuert wird. Die Ausgänge werden auf den Bus durch die Treiber des M785-Moduls gegatet. Wie man sieht, ist dieses Modul genau auf die typischen Ein-Ausgangsverhältnisse eines 8 Bit-Lesen/Schreiben-Registers abgestimmt.

5.4.1.4. Programmierung des Interface

Der gesamte Datenverkehr dieses Interface-Systems ist unter Kontrolle des Prozessors, d.h. auch alle speicherbezogenen Befehle können das Interface direkt adressieren.

Es sei REG die mnemonische Bezeichnung für die Registeradresse, dann bedeutet der Befehl MOV REG, R4, daß die im Register gespeicherten Daten gelesen und in das Register R4 des Prozessors transportiert werden sollen, also eine DATI-Operation. Der Befehl MOV R4, REG beschreibt den umgekehrten DATO-Datenfluß. Jeder Befehl, der zu einer Busadresse zugreift, kann Daten mit dem Register austauschen. So kann z.B. der Registerinhalt inkrementiert werden durch den INC, REG-Befehl oder durch ADD VALUE, REG um einen beliebigen Wert erhöht werden.

5.4.2. Programmkontrolliertes Interface

In diesem Beispiel wird beschrieben, wie ein Analog-Digital-Konverter (ADC) als externes Gerät über das Interface unter Programmkontrolle angeschlossen wird. Der Eingang des ADC soll der Ausgang eines Multiplexers (MUX) sein, der 64 analoge Datenquellen auf den ADC schalten kann, wo deren Information digitalisiert wird. Das Interface muß in diesem Beispiel folgende 7 Kontrollsignale an den ADC geben:

- 1 Start-Konversionssignal, positiv gehend, verursacht, daß der ADC die Konversion beginnt.
- 6 Signalleitungen für die MPX-Adresse, so daß das MUX-Register (ADMUX) 1 aus 64 Eingängen wählen kann.

Das Interface erhält 11 Signale:

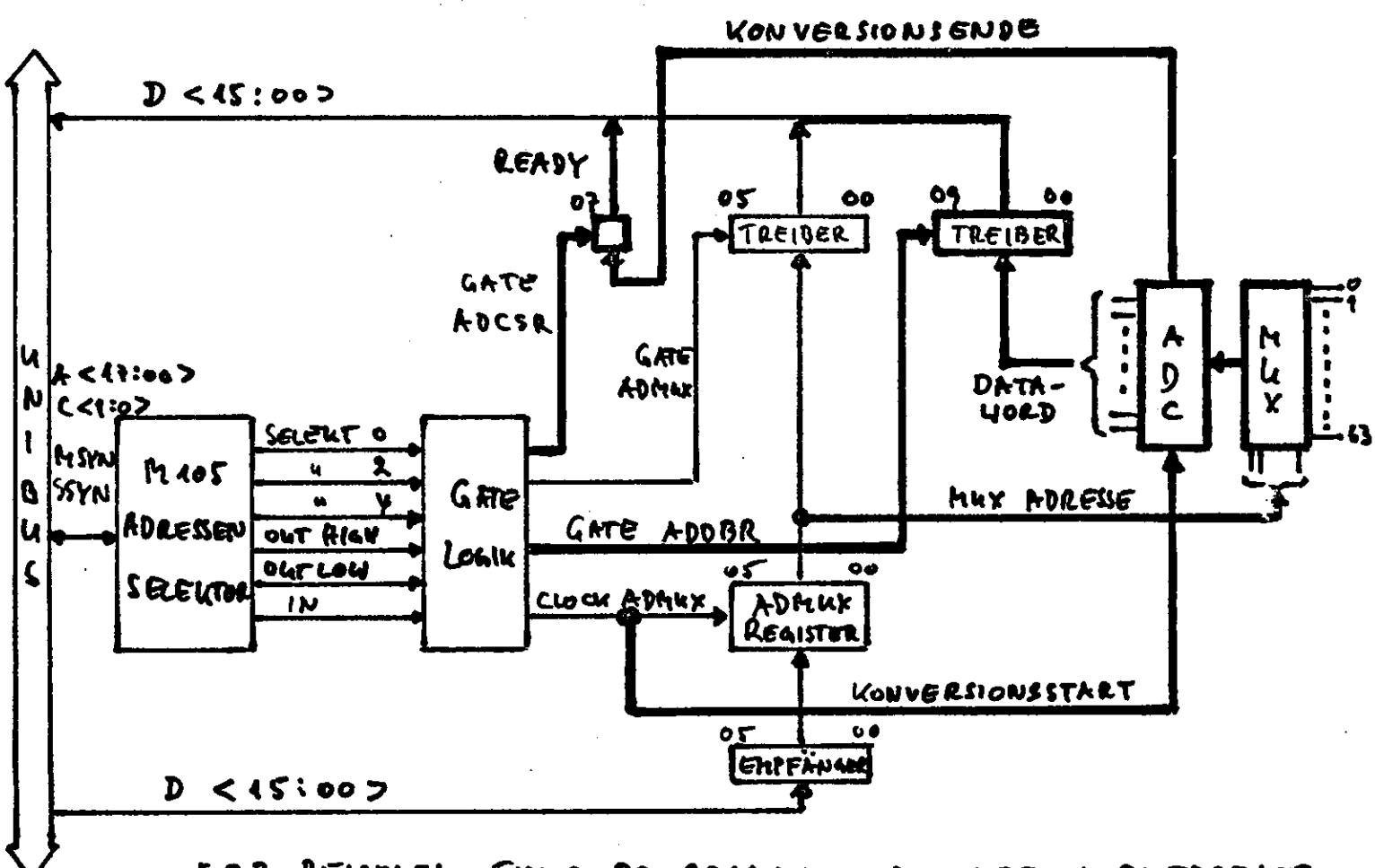
- 1 Konversionsendesignal, das vom Start der Konversion bis zum Ende auf 0 V liegt, dann auf + 3V schaltet.
- 10 Datenleitungen vom Konverterregister. Der für dies Beispiel gewählte ADC soll die analoge Information in 10 Bits konvertieren, d.h. auf 1024 Kanäle verteilen. Auf den Datenleitungen bedeutet eine 0 auch 0 V, eine 1 entsprechend + 3 V. Diese

Pegel sind übliche TTL-Pegel und auch interner DEC-Standard; auf dem UNIBUS sind sie jedoch invertiert (s. Abschnitt 5.2.1.1.).

5.4.2.1. Interface-Operation

In diesem Interface wählt das Programm einen der möglichen analogen Eingänge aus, der ADC digitalisiert das zugehörige Signal. Das Interface schickt das erzeugte Datenwort sowie das Konversionsendesignal auf die UNIBUS-Datenleitungen, so daß beide in den Master übertragen können, zuerst das Konversionsendesignal, das als READY-Signal geschickt wird, dann die Daten.

Bild 5.37 zeigt eine mögliche Schaltung. Die fettgezeichneten Linien sind die Leitungen oder Schaltkreise, die zu den Grundinterface hinzugefügt werden müssen.



5.37 BEISPIEL EINES PROGRAMMKONTROLLIERTEN INTERFACE

Das Interface arbeitet mit 3 Busadressen, und zwar:

- für das MUX-Register (ADMUX), das vom Typ her wie das im vorigen Beispiel ist.
- für das Datenregister im ADC-Ausgang (ADDBR), das vom Interface nur gelesen werden kann.
- für das 1 Bit-Kontroll- und Statusregister (ADCSR).

Der Adressen-Selektor erzeugt eines der drei Signale, durch die zu den Registern zugegriffen werden kann. Diese Signale werden durch IN H bzw. OUT LOW H so gegatet, daß die 4 Signale

- GATE ADCSR
- GATE ADMUX
- GATE ADDBR
- CLOCK ADMUX

erzeugt werden. Nur das ADMUX-Register erhält Daten von UNIBUS über die Empfänger. Die Ausgänge aller drei Register werden durch getrennte Bustreiber auf den UNIBUS gegatet.

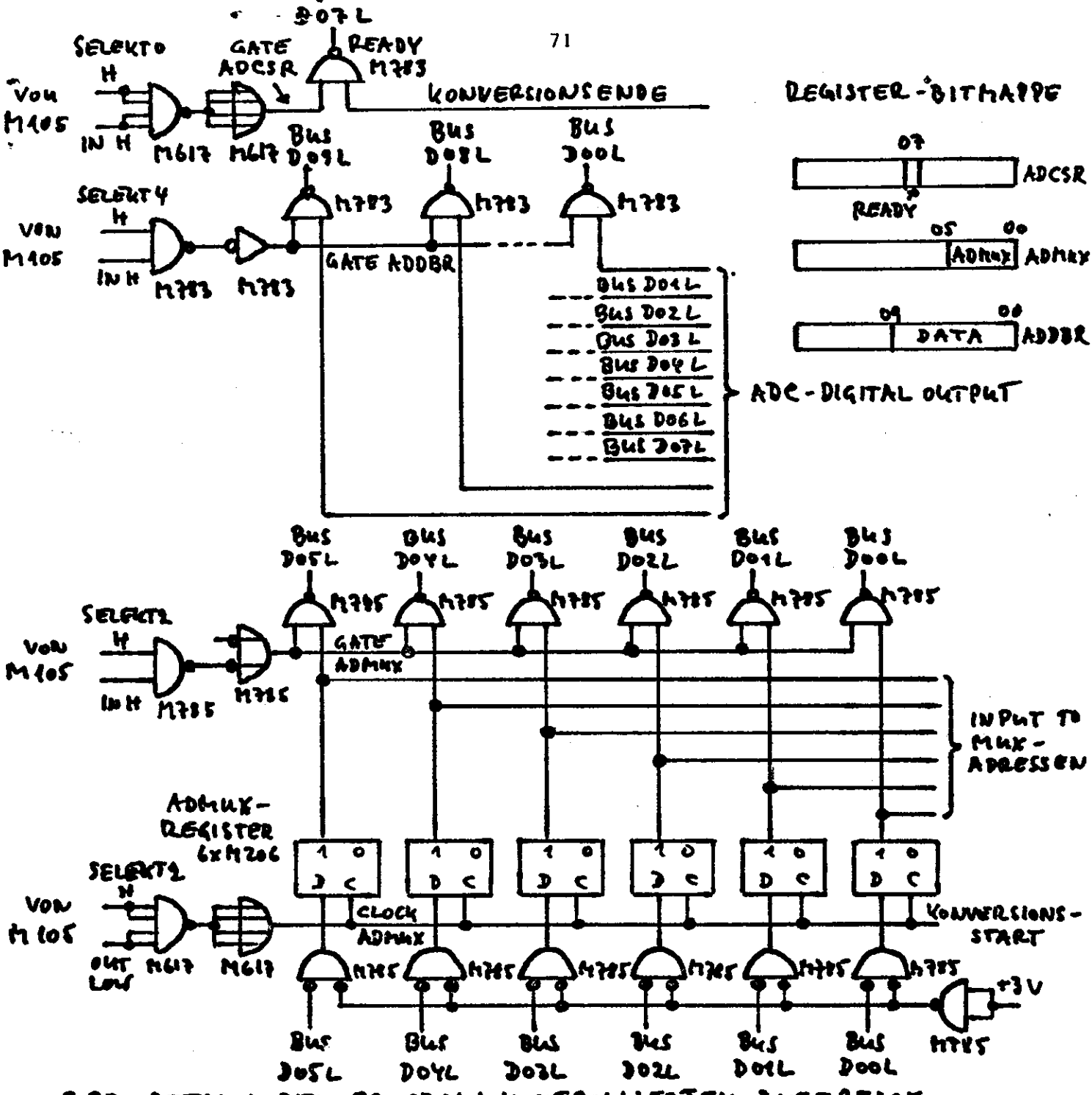
5.4.2.2. Datenübertragungen

Grundsätzlich ist die Behandlung der Datentransferarten die gleiche wie beim Grundinterface.

Jedes der drei Register kann während einer DATI-Operation gelesen werden, das ADMUX-Register mit einer DATO-Operation geladen werden. Wird eines der drei Register während eines DATO-Verkehrs adressiert, erzeugt das M105-Modul das SSYN-Signal, ohne das die Busoperation nicht abgeschlossen werden kann.

5.4.2.3. Beispiel einer Interfaceschaltung

Bild 5.38 zeigt eine Aufbaumöglichkeit. Das M105 und der ADC sind in der Schaltung nicht explizit enthalten, wohl aber die von ihnen erzeugten bzw. benötigten Signale. Die UNIBUS-Verbindung kann durch ein M785-Transceiver Module für das ADMUX-Register und ein M783-Transmittermodule für das Daten- und Kontrollregister realisiert werden. Das READY-Signal, das aus dem CONVERSION COMPLETE-Signal hervorgeht, muß mit einem der vier individuellen Bustreiber des M783 gegatet werden.



5.38 DETAILS DES PROGRAMMKONTROLLIERTEN INTERFACE

5.4.2.4. Programmierung des Interface

Das START CONVERSION-Signal, das die Datenannahme und den Transferzyklus in Gang setzt, wird aus dem CLOCK ADMUX-Signal gewonnen, mit dem das Multiplexer - register geladen wird. Der ADC beginnt die Konversion, das READY-Signal wird dabei gecleart und erst am Ende der Konversion wieder gesetzt. Anschließend beginnt der Transfer des Digitalwortes aus dem ADC an den Prozessor. Eine mögliche Befehlsfolge wird nun beschrieben. Das Programm wählt einen MUX-Eingang, wartet, bis der ADC das Signal konvertiert hat und überträgt dieses an das Register 4 im Prozessor.

```
        MOV     INPUT, ADMUX        ;   SELECT ANALOG INPUT
READY:  TSTB    ADCSR               ;   CHECK FOR CONVERSION COMPLETE
        BPL     READY               ;   NO, TEST AGAIN
        MOV     ADDBR, R4           ;   YES, OBTAIN DATA
```

Darin bedeutet INPUT die Nummer des gewünschten Analogeingangs; BPL Branch on Plus, d.h. prüfe den Status des Bit N (in unserem Beispiel N = 07) und verzweige, wenn N gecleart ist (N wird aber über NAND-Gate abgefragt, darum verzweigen, wenn N gesetzt ist!)

Sollen mehrere Eingänge des Multiplexers nacheinander digitalisiert und ausgelesen werden, kann eine Subroutine (Multiplexer-Scannen) durch den Befehl JSR R 7, MUXSCN (Jump to Subroutine im Prozessorregister 7) eingeleitet werden. Diese setzt den Inhalt des Registers R4 als Anfangsandresse des ersten von 64 Worten im Bufferregister ein, cleart das Multiplexer-Adressregister (ADMUX) und schaltet nacheinander alle MUX-Eingänge auf den ADC, wo sie konvertiert und in den Buffer eingelesen werden. Nach Ablauf dieser 64 Zyklen geht durch ein RTS R7 (Return from Subroutine) die Kontrolle wieder an das laufende Programm zurück. Eine solche Befehlsfolge könnte sein:

```
MUXSCN: MOV BUFADR, R4          ; LOAD DATA POINTER (START ADDRESS)
        CLR ADMUX               ; SELECT INPUT LINE ZERO
LOOP:   TSTB ADCSR              ; CHECK FOR CONVERSION COMPLETE
        BPL LOOP                ; NO, TEST AGAIN
        MOV ADDBR, R4           ; YES, PLACE DATA IN BUFFER
        CMP ADMUX,#77₈          ; LAST LINE?
        BEQ DONE                ; YES, GO TO DONE
        INC ADMUX               ; NO, GO TO NEXT INPUT
        BR LOOP                 ; GO TO LOOP
DONE:   RTS R 7                 ; EXIT FROM SUBROUTINE
```

Darin bedeuten?

- BUFADR die Speicherzelle, die die Adresse des ersten Wortes im 64-Wort-Buffer enthält,
- CMP (Compare), vergleiche gegenwärtige MUX-Adresse mit Endadresse,
- BEQ (Branch on equal), prüfe den Status des Komparatorausgangs und verzweige, wenn die verglichenen Adressen gleich sind. Dieser Befehlt folgt also auf CMP.

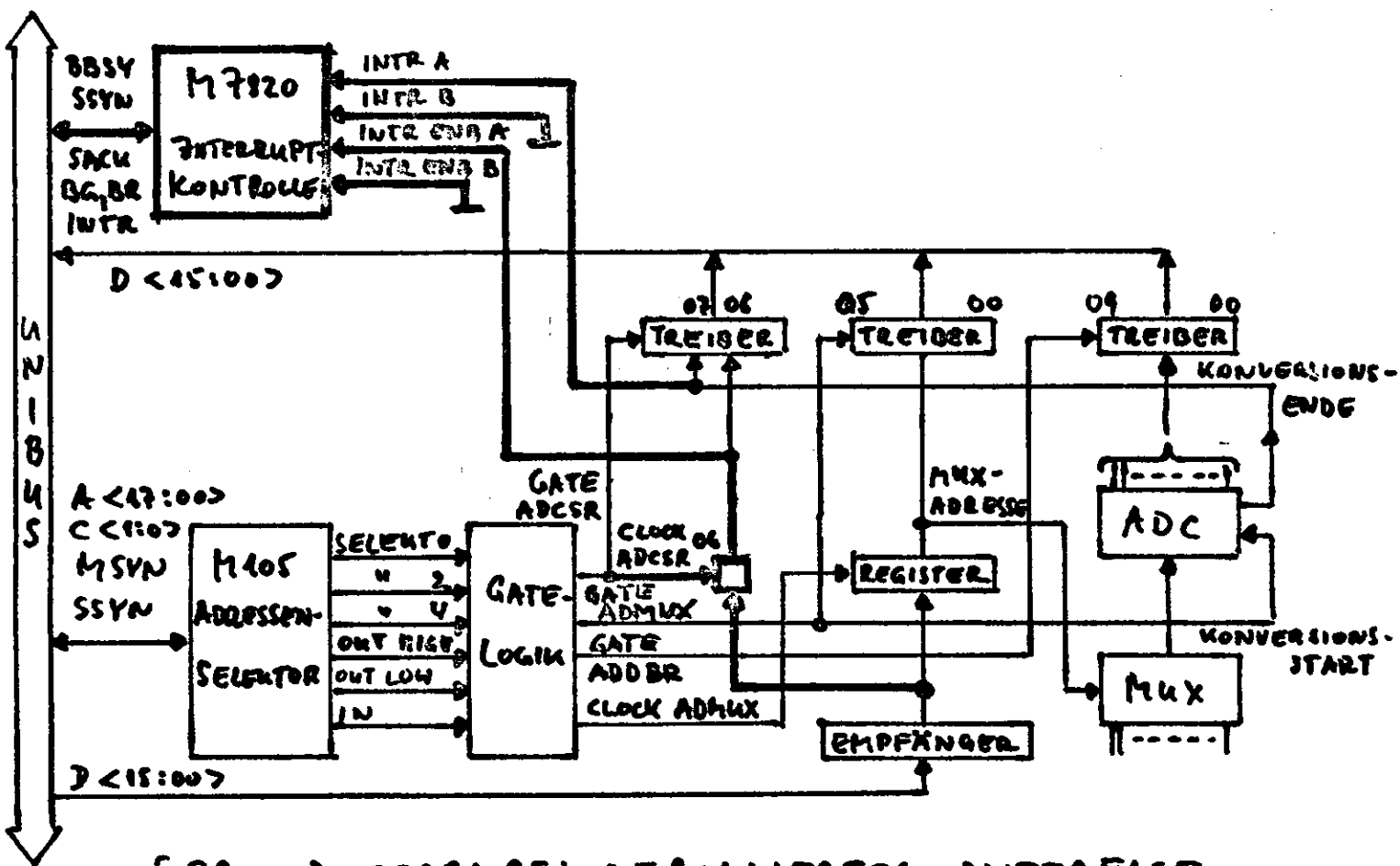
- INC (Increment), erhöhe den Registerinhalt um 1.
- BR (Unconditional branch), einfache Programmverzweigung zum Beginn der Schleife.

5.4.3. Interruptkontrolliertes Interface

Im vorigen Beispiel mußte der Prozessor solange sein laufendes Programm unterbrechen, bis die vom ADC konvertierten Daten von ihm übernommen sind. Das kann, je nach ADC-Typ, einige hundert Mikrosekunden dauern, was für die Programmbearbeitung sehr unwirtschaftlich ist. Fügt man zu dem bisherigen Interface noch eine Interrupt-Kontroll-Einheit M7820 sowie einen Interrupt-Enable-Flip-Flop hinzu, kann der Prozessor bis zum Setzen des READY-Signals seine laufende task weiterbearbeiten. Damit fällt auch das Testen des READY-Signals im vorigen Beispiel weg.

5.4.3.1. Interface-Operation

Bild 5.39 zeigt ein Blockdiagramm eines interruptkontrollierten Interface. Dieses kann entweder wie im vorigen Beispiel oder mit Interruptmeldung arbeiten, wie es in den Abschnitten 5.2.3.2. und 5.3.4.6. beschrieben ist.



5.39 INTERRUPTKONTROLLIERTES INTERFACE

5.4.3.2. Programmierung des Interface

Das folgende Programm stellt eine typische Interruptroutine dar, bei der die Daten der Konversion nach Ende des letzten Konversionszyklus angenommen werden. Es seien

- ADCSR, ADMUX und ADDBR wie bisher die Interface-Register,
- ADCSER die Serviceroutine des ADC-Interface, die durch die Adresse ADCVEC aufgerufen werden kann. Diese Adresse ist durch die Verbindungen in der M7820 Interrupt-Kontrolle festgelegt.
- BUFSTRT die Startadresse des Buffers und
- BUFADR der Speicherplatz, der von der Serviceroutine benutzt wird.

Dann lautet die Programmfolge:

```
ADCVEC:     ADCSER                      ;  SET UP ADC VECTOR AREA
             24 Ø                       ;  STATUS INCLUDES PRIORITY LEVEL 5
                                        ;  MAIN PROGRAM FOLLOWS
BEGIN:      MOV BUFSTRT, BUFADR         ;  INITIALIZE BUFFER POINTER
            CLR ADMUX                   ;  START MUX AT CHANNEL Ø
            MOV #1ØØ , ADCSR            ;  ENABLE INTERRUPT
ADCSR:      MOV ADDBR, BUFADR           ;  COLLECT DATA
            CMP BUFADR, BUFSTRT+174₈    ;  LAST ONE ?
            BEQ DONE                    ;  YES, GO TO DONE
            ADD #2, BUFADR              ;  NO, INCREMENT POINTER
            INC ADMUX                   ;  INC MUX AND START CONVERSION
            RTI                         ;  RETURN TO MAIN LINE
DONE:       CLR ADCSR                   ;  CLEAR INTERRUPT ENABLE
```

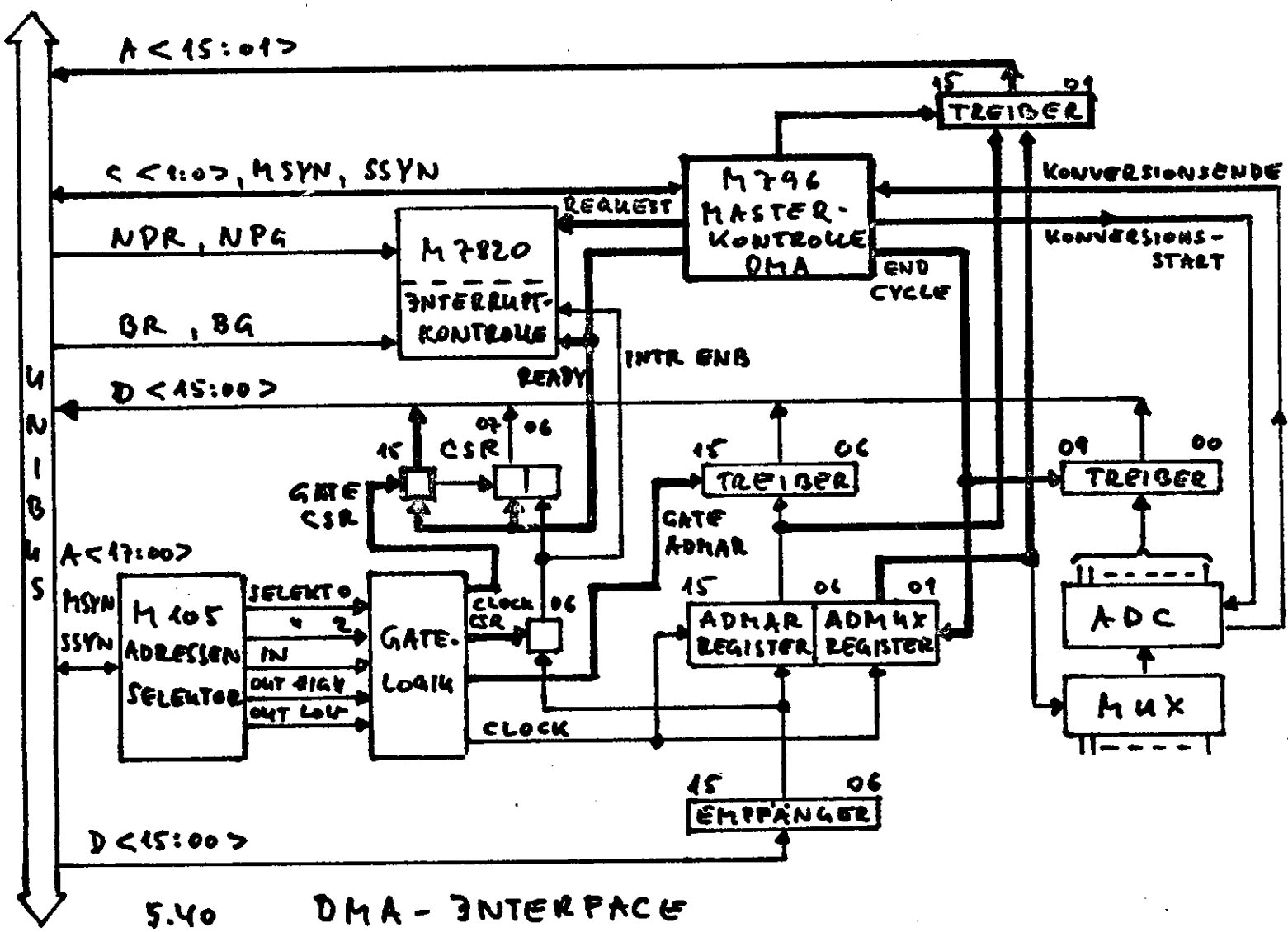
Nach den Anfangsbefehlen des Hauptprogramms zwingt der Interrupt den Prozessor, die ADCSER-Routine auszuführen.

Die CLR ADMUX-Anweisung sollte dem MOV #1ØØ, ADCSR-Befehl vorausgehen, um einen sofortigen Interrupt zu verhindern, der eintreffen würde, wenn das Interrupt-Enable-Bit gesetzt ist und das READY-Signal erscheint.

5.4.4. Interface mit direktem Speicherzugriff (DMA)

Das DMA-Interface überträgt in eigener Regie Daten vom externen Gerät direkt in den Speicher. Ohne den Prozessor zu benutzen, wird dabei eine große Zahl von Worten übertragen, am Ende der Prozessor benachrichtigt, so daß das entsprechende Programm anlaufen kann.

Bild 5.40 zeigt das Blockdiagramm des DMA-Interface, wieder auf den ADC bezogen, d.h. als Erweiterung des bisherigen Interface. Hinzukommen Treiberstufen, eine UNIBUS-Master(DMA)-Kontrolle sowie die Neuaufteilung der Register. Neben dem ADCSR-Register, das Flagge und Fehlerbits enthält, gibt es nun ein kombiniertes ADBAR/ADMUX-Register, in dem die Busadresse und die MUX-Adresse stehen.



5.40 DMA-INTERFACE

5.4.4.1. Interface-Operation

Das Programm lädt das Busadressenregister (ADBAR) mit der Speicheradresse, in die das erste Datenwort kommen soll. Dann startet das Interface die erste ADC-Konversion. Ist diese beendet, schickt das Interface einen NPR, nach dessen Quittung (NPG) und Erreichen der Buskontrolle wird das erste ADC-Datenwort zum Speicher übertragen. Anschließend wird das ADMUX-Register inkrementiert, der nächste Eingangskanal selektiert, ebenfalls im ADBAR-Register die Adresse des nächsten Speicherplatzes durch Inkrement erzeugt. Nun startet der neue ADC-Zyklus. Wenn alle Eingangskanäle abgearbeitet und die Datenworte im Speicher untergebracht sind, setzt das Interface den READY-Flip-Flop, der den Interrupt verursacht.

5.4.4.2. Interface-Aufbau

Das Interface wird durch die Masterkontrolle erweitert, dadruch ergibt sich ein geänderter Funktionsablauf. Das MUX-Register, das auf 15 Bit erweitert wird, arbeitet auch als Busadressregister. Neun Bits, und zwar <15:07>, sind unter Programm-Kontrolle, sie stellen die Basisadresse für diejenigen Speicherplätze dar, in die die 64 Datenworte des ADC transportiert werden sollen. Die restlichen 6 Bits, also< 06:01>, arbeiten als Zähler für die 64 MUX-Adressen ebenso wie für die 64 Speicherplätze, die an die Basisadresse angehängt werden. Auf diese 6 Bits hat der UNIBUS keinen Zugriff, d.h. sie können vom Programm weder gelesen noch geändert werden. Jedesmal, wenn das Adressregister ADBAR geladen wird, werden die 6 ADMUX-Bits gecleart, d.h. jeder Transfer startet mit Endadresse 0 und endet mit 63.

Das Laden des ADBAR-Registers geschieht in 2 Bytes (für 9 Bits) mit der UND-Bedingung SELECT 2 • OUT HIGH sowie SELECT 2 • OUT LOW, dabei wird neben dem ADMUX-Register auch der READY-Flip-Flop gecleart, wodurch das CONVERSION START H gesetzt wird, d.h. ein neuer ADC-Zyklus beginnt.

Ist das CONVERSION COMPLETE H-Signal erschienen, setzt es den REQUEST BUS Flip-Flop, wodurch das NPR-Signal entsteht. Ist dieses beantwortet (NPG), setzt die Interrupt-Kontrolle BBSY und das MASTER A L-Signal. Dieses MASTER-Signal initiiert in der M796 Master-Kontrolle das START-Signal und es beginnt ein DATO-Zyklus, weil C1 auf H, CO auf L (geerdet) ist (vgl. Abschnitt 5.3.4.7 und Bilder 5.30 und 5.31). Das ADRS TO BUS H wird erzeugt, das die 9 Bits des ADBAR sowie die 6 des ADMUX-Registers auf die Bus-Adress-Leitungen A<15:01> gatet. DATA TO BUS

setzt die Digitalworte aus dem ADC auf die Datenleitungen D<09:00>. Nach wenigstens 150 nsec wird BUS MSYN L gesetzt.

Antwortet der Slave, in diesem Fall der Speicher, mit BUS SSYN L, verschwinden ADRS TO BUS und DATA TO BUS sowie auch BUS MSYN L. Der Puls END CYCLE H wird zum Clearen des REQUEST-Flip-Flops benutzt, damit verschwindet BUS BBSY.

Das END CYCLE L-Signal triggert den im M796 vorhandenen freien Univibrator für die Dauer von 600 nsec, um ein COUNT IN für den ADMUX-Zähler zu erzeugen. Ist der READY-Flip-Flop noch nicht durch den Overflow des Zählers gesetzt, wird durch die Rückflanke des Univibrators von H nach L ein CONVERSION START H erzeugt. Ist jedoch der Overflow gekommen und der READY-Flip-Flop gesetzt, ist die ADC-Operation beendet, der Interrupt Bus Request wird gestartet.

Der TIME-OUT-Flip-Flop im M796 wird gesetzt, falls der Slave nicht innerhalb von 20 µsec auf das BUS MSYN L geantwortet hat. In diesem Fall wird der Buszyklus unterbrochen, READY gesetzt und weitere Konversionen verhindert. Das Signal TIME-OUT ERROR wird durch eine 1 in Bit 15 des ADCSR angezeigt, es wird durch Laden des ADCSR mit einer 0 wieder gecleart.

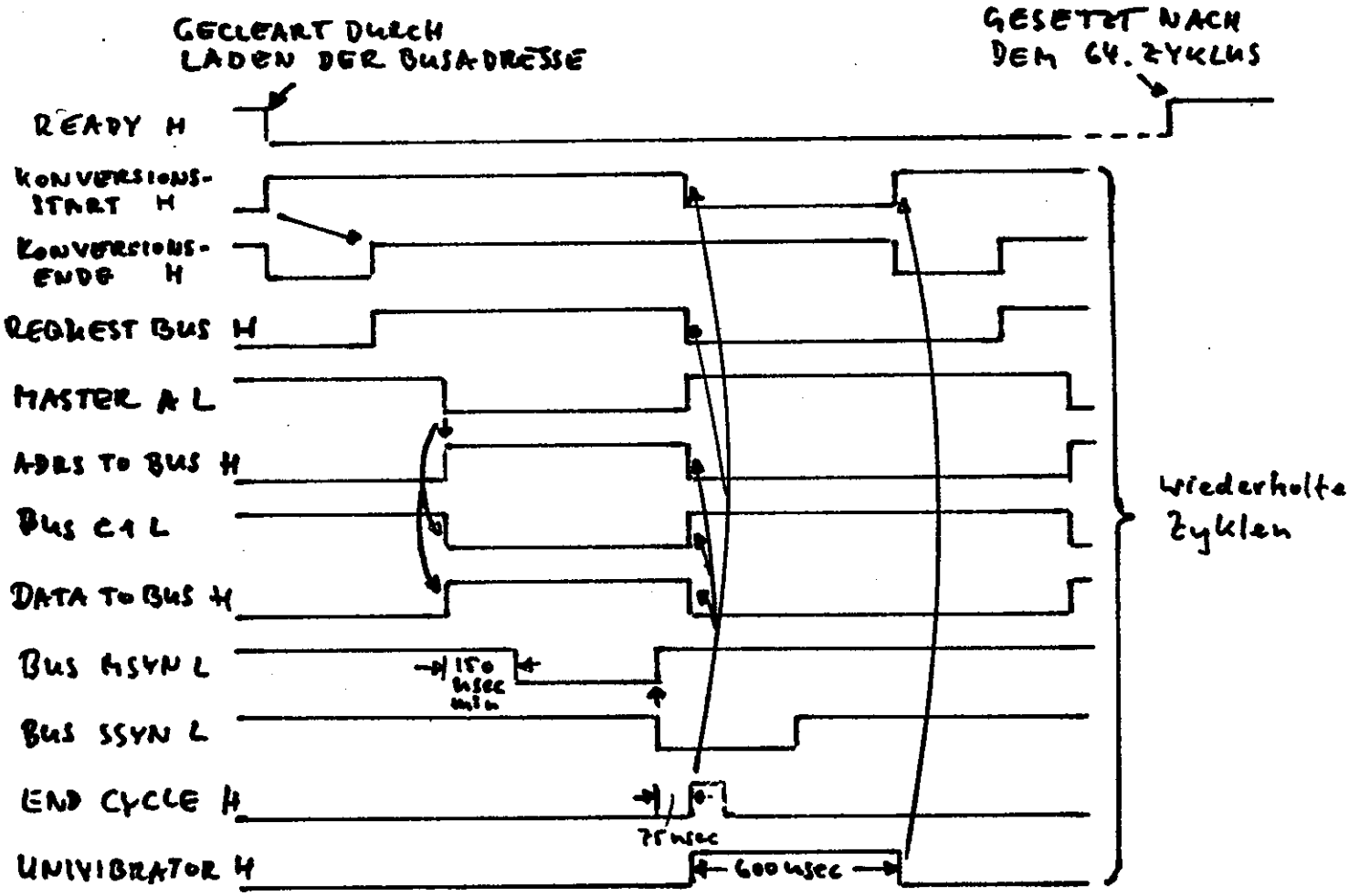
5.4.4.3. Programmierung des Interface

Die Interrupt-Routine ist der im vorigen Beispiel beschriebenen äquivalent. Um die Operation zu starten, beginnt das Programm mit:

```
MOV#BUFADR, ADBAR        ;    LOAD ADDRESS AND START
MOV#1ØØ, ADCSR           ;    ENABLE INTERRUPT
```

5.4.4.4. Zeitdiagramm der Interface-Operation

Bild 5.41 zeigt den zeitlichen Ablauf der DMA-Operation, der das Setzen und Verschwinden der Signale deutlich macht.



5.41 ZEITDIAGRAMM DES DMA-INTERFACE

5.4.5. Output-Interface mit Interruptkontrolle

Die bisherigen Beispiele beschrieben Interfacesysteme, die Dateneingänge in die UNIBUS-Leitungen haben. In diesem Beispiel holt das Interface Daten aus dem Speicher über den UNIBUS und steuert damit einen Digital-Analog-Konverter (DAC).

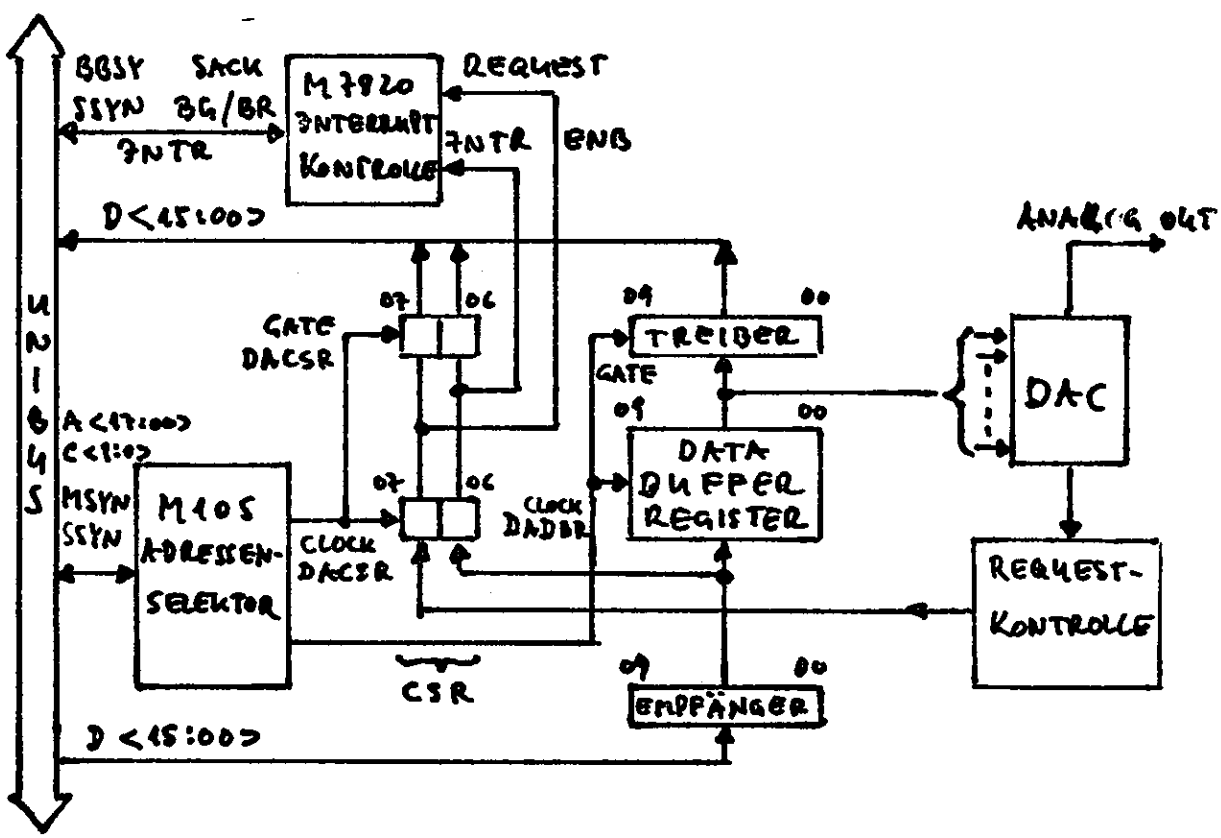
5.4.5.1. Gerätebeschreibung

Der DAC nimmt Daten über den UNIBUS und konvertiert jedes ankommende Binärwort in eine Analogspannung. Die digitalen Datenworte sollen 10 Bit lang sein, die

binäre 0 wird durch OV, die 1 durch +3V repräsentiert. Nach Erreichen des Analogwertes, der dem Datenwort entspricht, liefert der DAC eine Anforderung (Request), mit dem er ein neues Datenwort verlangt. Dies ist ein + 3V-Pulssignal, zwischen zwei Requests ist der Pegel 0 V.

5.4.5.2. Interface-Operation

Das Interface enthält ein Bufferregister, um Daten in den DAC zu geben sowie eine Interrupt-Kontrolle, um über das Requestsignal die Interruptroutine aufzurufen. Bild 5.42 zeigt das Blockdiagramm.



5.42 OUTPUT-INTERFACE MIT INTERRUPTKONTROLLE

Neben dem M105-Adressen-Selektor sind vorhanden das Datenbufferregister DADBR und, wie immer, das Kontroll- und Statusregister DACSR. Das Anforderungsbit 7 des DACSR kann über den Bus gelesen, aber nicht geladen werden, alle anderen Bit sind direkt unter Buskontrolle.

Adressiert der UNIBUS während eines DATO-Transfers das Bufferregister, clockt das Interface diese Daten herein, die Konversion beginnt. Zur gleichen Zeit wird das Bit 7 des DACSR gecleart. Ist die Konversion ausgeführt, und der DAC wünscht einen neuen Digitalwert, wird der Request-Flip-Flop (Bit 7) gesetzt und die Interrupt-Kontrolleinheit schickt einen Request, falls Bit 6 des DACSR durch den Interrupt-Enable-Flip-Flop gesetzt ist. Das Gerät wird dann Busmaster, das Interface schickt einen Interrupt, um die Programm-Kontrolle auf die Service-Routine zu übertragen. Diese lädt neue Daten ins Bufferregister und gibt die Kontrolle an das unterbrochene Programm zurück.

5.4.5.3. Programmierung des Interface

Das Programm kann z.B. den DAC so laden, daß eine zeitlich veränderliche Analogspannung entsteht, die zyklisch wiederholt wird. Ein typisches Beispiel solcher Operation ist die Erzeugung der Ablenkspannungen eines Rechner-Displays.

Das Programm lädt das Bufferregister DADBR mit dem Anfangswert und ändert diesen schrittweise durch kleine Additionen bis zum Maximalwert, durch anschließende Subtraktionen wieder zum Ursprungswert. Bei jedem Schritt wird der Inhalt des DADBR gelesen, durch die arithmetische Operation geändert und neu eingeschrieben. Diese Operationen laufen unter voller Prozessorkontrolle.

Die Anzahl der periodischen Wiederholungen wird in einen Zyklenzähler eingeschrieben (DACNT), der nach jedem Zyklus decrementiert wird.

Der Prozessor startet mit dem Clearen des Bufferregisters und des Aufwärts-/Abwärtsschalters, der entscheidet, ob eine Addition oder Subtraktion ausgeführt wird. Dann wird der Zyklenzähler DACNT geladen:

```
CLR DADBR                    ;   CLEAR DATA BUFFER REGISTER
CLR DASW                     ;   RESET UP/DOWN SWITCH
MOV #N, DACNT                ;   SET CYCLE COUNTER TO N
MOV #100, DACSR              ;   SET INTERRUPT ENABLE
```

Die Interrupt-Serviceroutine setzt zuerst den Pointer auf die Serviceroutine und gibt die Prozessorpriorität an, z.B. 5:

```
DAVEC:   DASERV          ;   POINTER TO SERVICE ROUTINE
         24 Ø            ;   PROCESSOR PRIORITY = 5
```

Das DASERV-Programm richtet sich nun nach der speziellen Aufgabe, z.B. der Erzeugung der Ablenkspannung. Nach Ablauf dieses Programmteils kehrt die Programmkontrolle über einen RTI-Befehl zum Hauptprogramm zurück.

5.4.6. Output-Interface mit DMA-Kontrolle

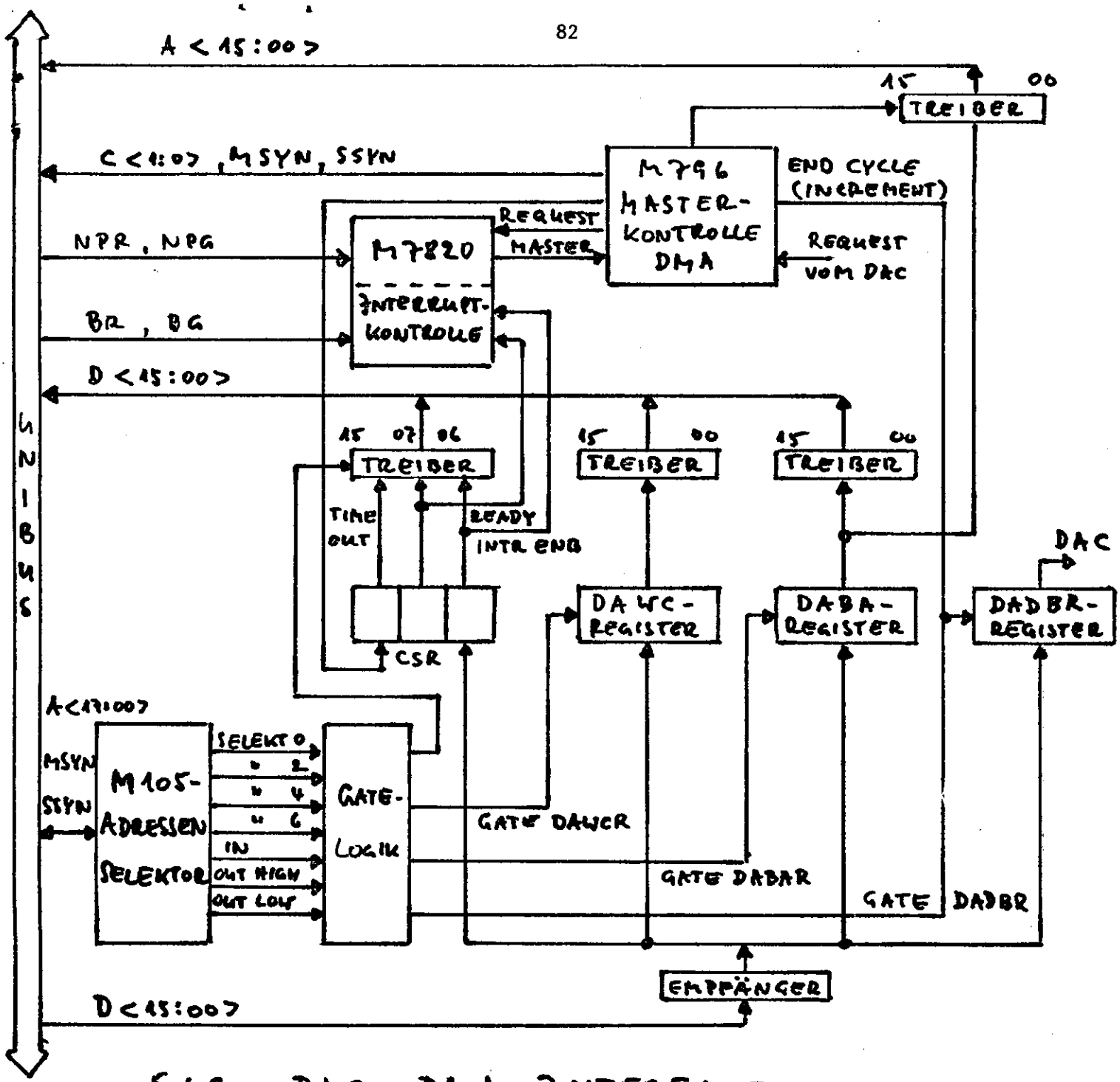
Im vorigen Beispiel wird viel Prozessorzeit verbraucht; durch Zufügen einer Masterkontrolle für direkten Speicherzugriff (DMA) kann diese Zeit wesentlich reduziert werden.

In diesem, letzten Beispiel erweitern wir das Interface des vorigen Abschnitts durch die Masterkontrolle, einen Wortzähler (word count register DAWC) sowie ein Busadressregister (DABA). Dadurch kann eine festgelegte Anzahl von Worten aus einem bestimmten Speicheradressenbereich unabhängig von der Prozessorkontrolle direkt in das Interface übertragen werden.

5.4.6.1. Interface-Operation

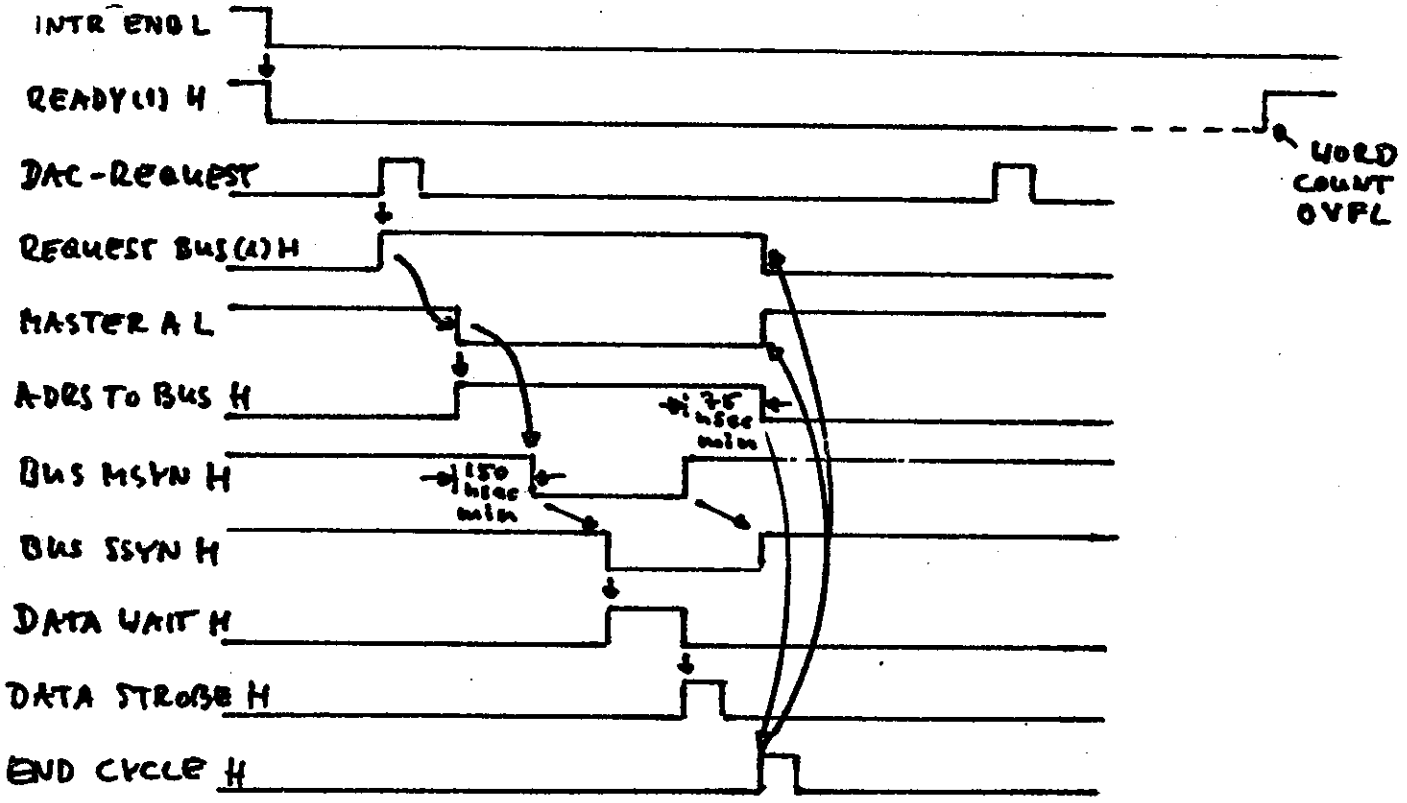
Bild 5.43 zeigt das Blockdiagramm eines DAC-Interface mit DMA-Kontrolle. Das Interface enthält vier Register:

- das DAC-Kontroll- und Statusregister DACSR,
- das DAC-Wortzählregister DAWC, das das 2er-Komplement der zu übertragenden Worte enthält,
- das DAC-Busadressenregister DABA, das anzeigt, wo der Datenblock im Speicher steht, sowie
- das DAC-Datenbufferregister DADB, das die Daten während eines Buszyklus hält und das vom Programm geladen werden kann.



5.43 DAC - DMA - INTERFACE

Der Operationsablauf beginnt mit dem Laden des DAWC- und DABA-Registers. Im nächsten Schritt wird Bit 6 im DACSR gesetzt. Datenworte werden dann sequentiell aus dem Speicher geholt und ins DADBR geladen, dies geschieht mit einer Folgefrequenz, die entweder der DAC oder eine externe Clock steuert. Nach jedem Transfer werden die Inhalte der DAWC und DABA incrementiert. Dies geschieht so lange, bis das DAWC einen Overflow anzeigt (alle Bits gehen auf 0). Dadurch wird das READY-Bit gesetzt, das den Interrupt auslöst, vorausgesetzt, Bit 6 (INTR ENB) ist gesetzt. Der Interrupt zeigt dem Prozessor das Ende des Blocktransfer an. Bild 5.44 liefert das Zeitdiagramm des besprochenen Ablaufs.



5.44 ZEITDIAGRAMM DES INTERFACE AUS 5.43