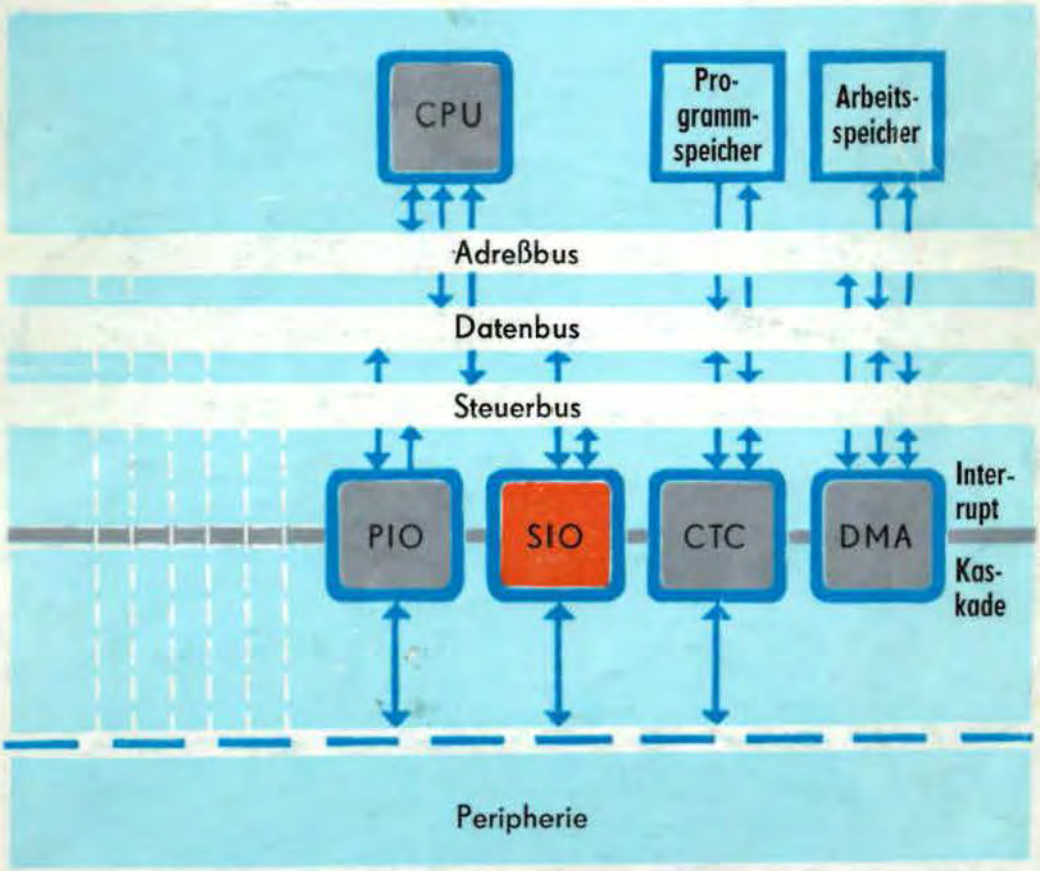
RFT

Mikroprozessorsystem

der II. Leistungsklasse

– Technische Beschreibung –



Schaltkreis für serielle Ein- und Ausgabe

SIO U 856 D

Die technischen Angaben dieses Handbuches tragen reinen Informationscharakter. Verbindliche technische Liefer- und Reklamationsgrundlage sind ausschließlich die Typstandards. Die vorliegende Dokumentation gibt keine Auskunft über Liefermöglichkeiten und beinhaltet keine Verbindlichkeiten zur Produktion.

Inhaltsübersicht

1. Einführung

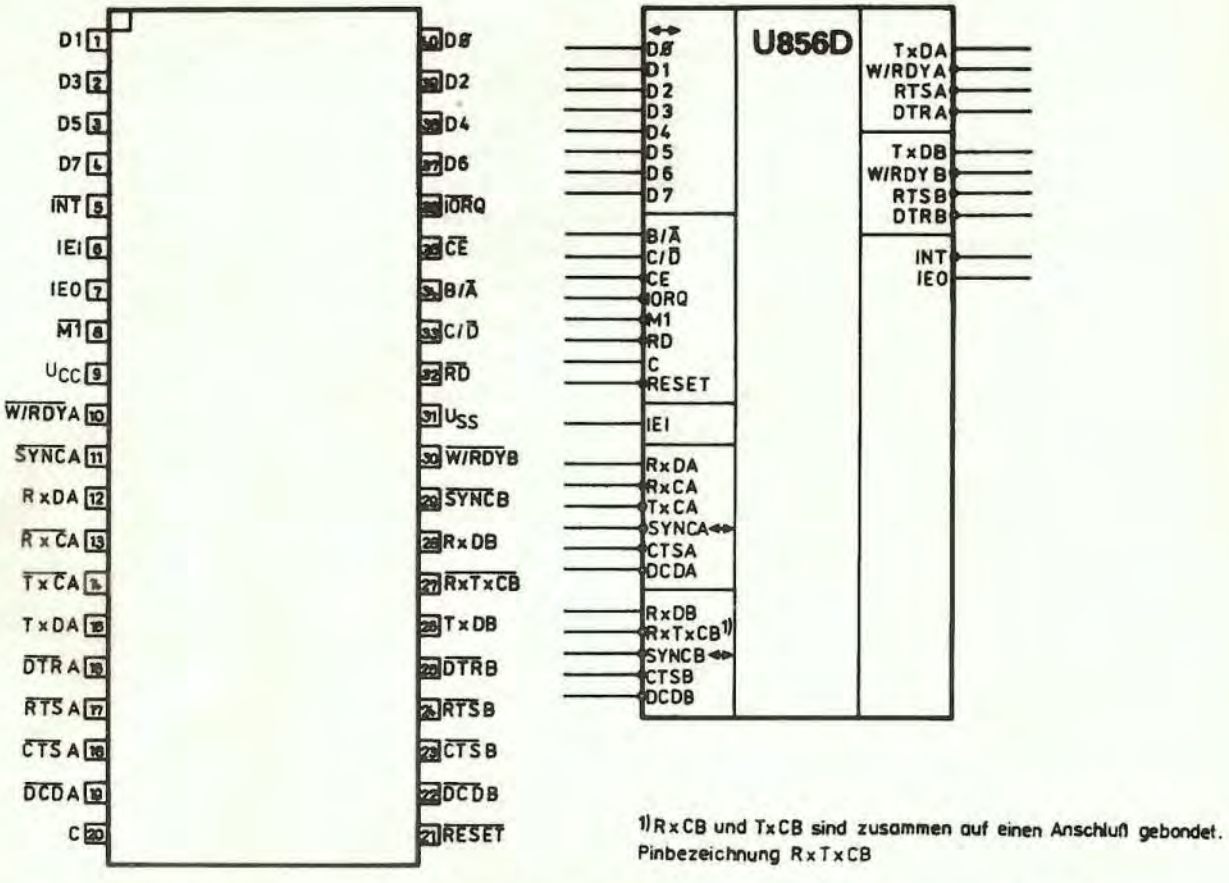
Der serielle Ein-Ausgabeschaltkreis U 856 D ist ein in n-Kanal-Silicon-Gate-Technologie gefertigter programmierbarer, zweikanaliger Baustein, der Daten in das für serielle Datenübertragung erforderliche Format umsetzt.
Der SIO ist in der Lage, asynchrone und synchrone byteorientierte Formate wie IBM-Bisync und synchrone bit-orientierte Formate wie HDLC und IBM SDLC zu verarbeiten. Der U 856 D kann durch die Systemsoftware in seiner Funktion in großer Varationsbreite an die Bedingungen eines speziellen seriellen Datenprotokolls angepaßt werden (z. B. Kassetten - oder Floppy-Disk-Interface). Der SIO kann CRC-Kodes in jeder synchronen Betriebsart erzeugen und prüfen. Desweiteren ist er als Modemsteuergerät in beiden Kanälen geeignet. In Anwendungsfällen, in denen diese Kontrollfunktionen nicht benötigt werden, kann die Modemsteuerung für allgemeine Ein-/Ausgabezwecke verwendet werden. Der U 856 D ist ein Schaltkreis innerhalb des Systems der 2. Leistungsklasse. Er ist vorwiegend für den Einsatz in Datenverarbeitungsanlagen und Anlagen der Steuerungs- und Regelungstechnik vorgesehen.

Eigenschaften

- 4 unabhängige serielle Ports: zwei Sender- sowie Empfängerports
- Datenübertragungsrate: 0 bis 550 kBit/s
- Empfänger-Datenregister vierfach gepuffert, Sender doppelt gepuffert
- Asynchrone Betriebsart:
  . 5, 6, 7 oder 8 Bits/Zeichen
  . 1, 1½ oder 2 Stoppbits
  . gerade, ungerade oder keine Parität
  . Taktvarianten: x1, x16, x32, x64
  . Break-Erzeugung und -Erkennung
  . Paritäts-, Überlauf-und Rahmenfehlererkennung
- Synchrone Betriebsart:
  . Interne oder Externe Zeichensynchronisation
  . Ein oder zwei Synchronisationszeichen in getrennten Registern
  . Automatisches Einfügen von Synchronisationszeichen
  . CRC-Erzeugung und Kontrolle
- HDLC- und IBM-SDLC-Betriebsart:
  . Null-Einfügung und -Ausblendung
  . Automatisches Flag-Einfügen
  . Adreßfeld-Erkennung
  . I-Feld Residuum-Behandlung
  . gültige empfangene Daten vor Überschreiben geschützt
  . CRC-Erzeugung und Kontrolle
- 8 Modem-Steuer-Ein- und Ausgänge
- CRC-16- oder CRC-CCITT-Blockkontrolle
- Die Interrupt-Logik (Daisy-Chain-Priorisierung) bietet eine automatische Interrupt-Vektorbildung ohne externe Logik.
- Der Modemstatus kann überwacht werden.

2. Aufbau der SIO U 856 D

Der SIO U 856 befindet sich in einem standardisierten Dual-In-Line-Gehäuse mit 40 Anschlüssen.
Bild 1 zeigt die Anschlußbelegung und das logische Schaltbild des U 856 D.



1) RxCB und TxCB sind zusammen auf einen Anschluß gebondet. Pinbezeichnung RxTxCB

Bild 1: Anschlußbelegung und logisches Schaltbild des U 856 D (Bondvariante U 8560)

Die Beschränkung auf ein 40poliges Gehäuse erlaubt es nicht, Empfangstakt, Sendetakt, Terminalbereitschaft und Synchronisationssignale für beide Kanäle herauszuführen. Daher muß Kanal B auf ein Signal verzichten oder zwei Signale werden zusammengebondet. Da die Forderungen der Nutzer unterschiedlich sind, werden zwei Bondmöglichkeiten angeboten. SIO/0 (U 8560 D) hat alle vier Signale, wobei jedoch $\overline{\text{TxCB}}$ und $\overline{\text{RxCB}}$ zusammengebondet sind (Bild 1). SIO/1 (U 8561 D) verzichtet auf $\overline{\text{DTRB}}$ und behält $\overline{\text{TxCB}}$, $\overline{\text{RxCB}}$ und $\overline{\text{SYNCB}}$ (Bild 2).

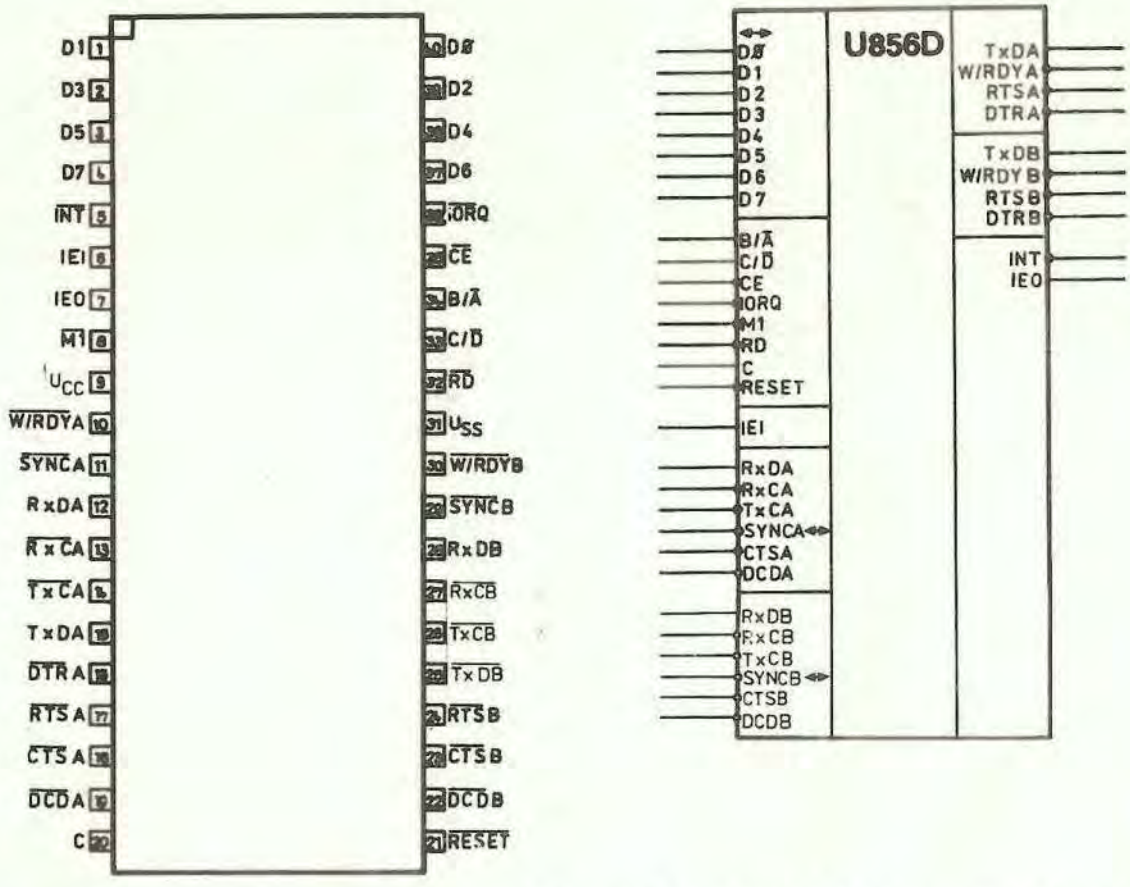


Bild 2: Anschlußbelegung und logisches Schaltbild des U 856 D (Bondvariante U 8561 D)

PIN-Beschreibung

Bezeichnung	Funktion	Kommentar
D0-D7	Datenbus	Bidirektionale Tri-State-Datenleitungen zum Anschluß an den Systemdatenbus
B/$\overline{A}$	Kanal "B/A-Select"	Eingang zur Kanalauswahl (High bedeutet "Kanal B selektiert")
C/$\overline{D}$	"Control/Data-Select"	Eingang Steuerwort/Datenübertragung (High bedeutet Steuerwort übertragen)
$\overline{CE}$	"Chip enable"	Eingang Low aktiv: Chip-Freigabe
$\overline{M1}$	"Machine Cycle 1"	Eingang Low aktiv: Maschinenzyklus M1 oder CPU-Steuerbussignal
$\overline{IORQ}$	"I/O-Request"	Eingang Low aktiv: Ein/Ausgabeanforderung von der CPU
$\overline{RD}$	"Read"	Eingang Low aktiv: Lesezyklus der CPU (Steuerbussignal)
C	"Clock"	Eingang: Systemtakt
$\overline{INT}$	"Interruptrequest"	Eingang, Low aktiv: Interruptanforderung von der Peripherie

IEI	"Interrupt enable In"	Eingang, High aktiv: Verbindung von IEI mit IEO des nächst höher priorisierten E/A-Schaltkreises ermöglicht Interruptprioritäts-Kaskadierung. High-Pegel an IEI bedeutet, daß momentan kein Interrupt höherer Priorität abgearbeitet wird.
IEO	"Interrupt enable Out"	Ausgang, High aktiv: IEO führt nur High-Pegel, wenn vom betreffenden und allen höher priorisierten Schaltkreisen der Interrupt-Prioritätskette kein Interrupt in Abarbeitung befindlich oder angemeldet ist. (Ausnahme: noch nicht bestätigte Interruptanmeldung eines höher priorisierten E/A-Schaltkreises während ED-Dekodierung)
$\overline{\text{RESET}}$	"Reset"	Eingang, Low aktiv: Sperrt sowohl Sender als Empfänger. TxDA und TxDB werden in den aktiven Zustand gebracht, Modem Control in den High-Zustand. Nach dem Rücksetzen müssen die Steuerregisterinformationen neu eingeschrieben werden, bevor irgendeine Datenübertragung stattfindet. Sämtliche Interrupts werden gesperrt.
$\overline{\text{W/RDYA}}$ $\overline{\text{W/RDYB}}$	"Wait/Ready"	Ausgang programmierbar Open-Drain oder Gegentaktausgang 1 Anschluß pro Kanal, der als "Ready-Leitung" für den Anschluß von DMA-Controllern oder als "Wait-Leitung" zur Synchronisation der CPU mit der Datenübertragungsrate programmiert werden kann.
$\overline{\text{CTSA}}$ $\overline{\text{CTSB}}$	"Clear to Send"	1 Anschluß pro Kanal, Schmitt-Trigger-Eingänge, Low aktiv: In der Betriebsart "Auto-Enable" sperrt dieses Signal den Sender seines Kanals. Wird der Anschluß nicht zur Senderfreigabe verwendet, steht er als allgemeiner Eingang zur freien Verfügung. Die beiden Leitungen haben Schmitt-Trigger-Eingänge, so daß auch Eingangssignale geringer Flankensteilheit einwandfrei verarbeitet werden.

$\overline{DCDA}$, $\overline{DCDB}$	"Data Carrier Detect"	1 Anschluß pro Kanal, Schmitt-Trigger-Eingänge, Low aktiv: Im "Auto-Enable" - Mode Eingang zur Empfängerfreigabe sonst frei verfügbarer Eingang
RxDA, RxDB	"Receive Data"	1 Anschluß pro Kanal, Eingänge high aktiv: serielle Dateneingänge
TxDA, TxDB	"Transmit Data"	1 Anschluß pro Kanal, Ausgänge high aktiv: Serielle Datenausgänge
$\overline{RxCA}$, $\overline{RxCB}$	"Receiver Clock"	1 Anschluß pro Kanal, Schmitt-Trigger-Eingänge, Low aktiv: Empfängertakteingang, als Taktgeschwindigkeit in der asynchronen Betriebsart können x1, x16, x32 oder x64 programmiert werden.
$\overline{TxCA}$, $\overline{TxCB}$	"Transmitter Clock"	1 Anschluß pro Kanal, Schmitt-Triggereingänge Sendertakteingang; als Taktgeschwindigkeit sind hier ebenfalls x1, x16, x32 oder x64 möglich.
$\overline{RTSA}$, $\overline{RTSB}$	"Request to Send"	1 Anschluß pro Kanal, Ausgänge low aktiv: Sobald das RTS-BIT eines Kanals gesetzt ist, geht die zugehörige RTS-Leitung in den Low-Zu stand über. Wird das RTS-BIT in der asynchronen Betriebsart rückgesetzt, geht die zugehörige RTS-Leitung in den High-Zustand, sobald das Senderregister leer ist. In der synchronen Betriebsart fungiert der Anschluß einfach als Ausgang, an dem dauernd der Wert des RTS-Bits liegt.
$\overline{DTRA}$, $\overline{DTRB}$	"Data Terminal Ready"	1 Kontakt pro Kanal, Ausgänge Low aktiv: Ausgang, an dem der Wert des DTR-Bits liegt.
$\overline{SYNCA}$, $\overline{SYNCB}$	"External Charakter Synchronisation"	2 Ein/Ausgabe-Anschlüsse, Low aktiv: Funktion des Anschlusses ist abhängig vom programmierten Übertragungsmodus: 1. Externe Synchronisation: Eingang; Aufbauen des Zeichens beginnt mit der steigenden Flanke von RxC, die auf die fallende Flanke des SYNC-Signals folgt. Eingang darf frühestens 2 Taktzyklen nach der steigenden Flanke von RxC aktiviert werden. Empfehlung: SYNC-Aktivierung nur mit der fallenden Flanke von RxC erfüllt diese Forderung.

2. Interne Synchronisation:
Ausgang; aktiv in dem Teil eines Taktzyklus, in dem ein SYNC-Zeichen (8- oder 16-Bit-Folge) erkannt wurde. SYNC-Bedingung wird nicht zwischengespeichert, d. h. jede SYNC-Bitfolge (unabhängig von Wortgrenzen) aktiviert SYNC-Ausgang.

3. Asynchroner Modus
Eingang, für allgemeine Eingabezwecke in Read-Register 0.

Struktur des Bausteins

Die interne Struktur des Bausteins umfaßt CPU-Interface, interne Steuerung, Interrupt-Logik und zwei Vollduplexkanäle (Bild 3). Jeder Kanal hat Lese- und Schreibregister und eine diskrete Steuer- und Statuslogik, die das Interface für Modems oder andere externe Geräte enthalten.

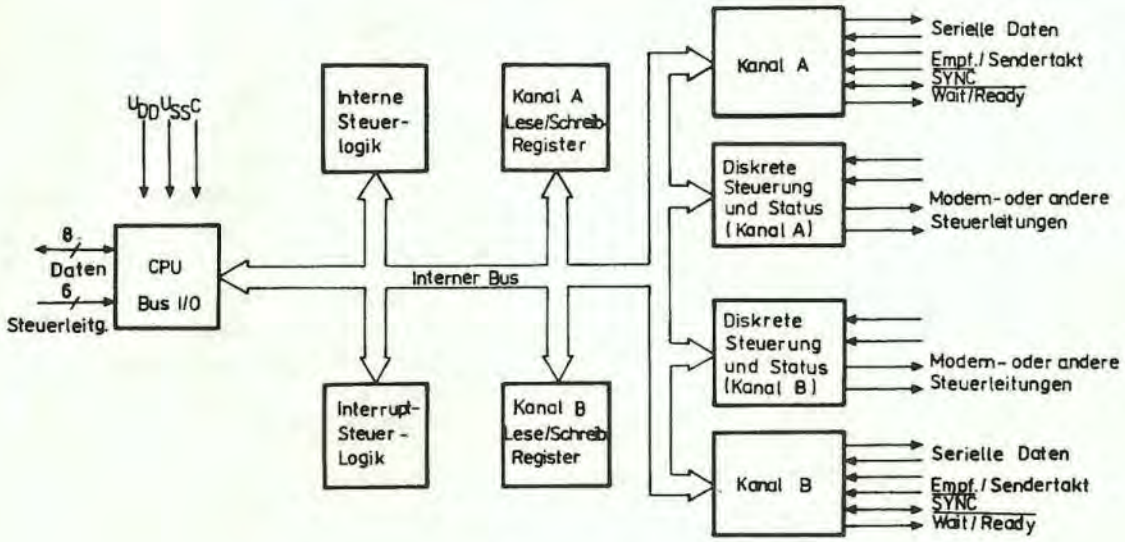


Bild 3: Blockschaltbild der SIO U 856 D

Die Lese- und Schreibregistergruppe umfaßt fünf 8-Bit-Steuerregister, zwei Sync-Zeichen-Register und zwei Statusregister. Der Interrupt-Vektor wird in ein 8-Bit-Zusatzregister (Schreibregister 2) im Kanal B eingeschrieben, der über Leseregister 2 im Kanal B gelesen werden kann.

Die Register für beide Kanäle werden im Text wie folgt bezeichnet:

WR0-WR7	Schreibregister 0 bis 7
RR0-RR2	Leseregister 0 bis 2

Die Bitzuordnung und funktionelle Gruppierung jedes Registers ist so ausgelegt, daß die Programmierarbeit erleichtert wird.
Tabelle 1 veranschaulicht die Funktionen, die jedem Lese- oder Schreibregister zugeordnet sind.

WR0	Registerzeiger, CRC-Initialisierung, Initialisierungsbefehle für die verschiedenen Betriebsarten, usw.
WR1	Sende/Empfangs-Interrupt und Festlegung der Datenübertragungsbetriebsart
WR2	Interrupt-Vektor (nur Kanal B)
WR3	Empfangsparameter und Steuerbits
WR4	Parameter für Senden und Empfang und die dazugehörigen Betriebsarten
WR5	Sendeparameter und Steuerbits
WR6	SYNC-Zeichen oder SDLC-Adressenfeld
WR7	SYNC-Zeichen oder SDLC-Flag

a) Schreibregisterfunktionen

RR0	Pufferstatus Senden/Empfangen, Interruptstatus und externer Status
RR1	Status für spezielle Empfangsbedingungen
RR2	Modifizierter Interrupt-Vektor (nur Kanal B)

b) Leseregisterfunktionen

Tabelle 1: Funktionen der Lese- und Schreibregister

Die Logik beider Kanäle liefert Formate, Synchronisation und gültige Daten, die zum und vom Kanalinterface übertragen werden. Die Modemsteuereingänge Clear to Send (CTS) und Data Carrier Detect (DCD) werden durch die diskrete Steuerlogik unter Programmkontrolle überwacht. Alle Modemsteuersignale sind dem Wesen nach Mehrzwecksignale und können auch für andere Funktionen verwendet werden.
Für die automatische Interrupt-Vektorbildung legt die Interruptsteuerlogik fest, welcher Kanal die höchste Priorität hat.
Die Priorität ist so festgelegt, daß Kanal A eine höhere Priorität hat als Kanal B; Empfänger-, Sender- und externe bzw. Status-Interrupts sind in jedem Kanal in dieser Reihenfolge priorisiert.

Datenweg

Der Sende- und Empfangsdatenweg für jeden Kanal ist in Bild 4 dargestellt. Der Empfänger hat drei 8-Bit-Pufferregister in einer F/FO-Anordnung (um eine 3-Byte-Verzögerung zu erzeugen) zusätzlich zu dem 8-Bit-Empfangs-Schieberegister. Auf diese Weise erhält die CPU zusätzlich Zeit, um einen Interrupt am Anfang eines schnellen Datenblocks zu bedienen.
Der Empfangsfehler - FIFO speichert Paritäts- und Rahmenfehler und andere Arten von Statusinformationen für jedes der drei Bytes in dem Empfangsdaten - FIFO.

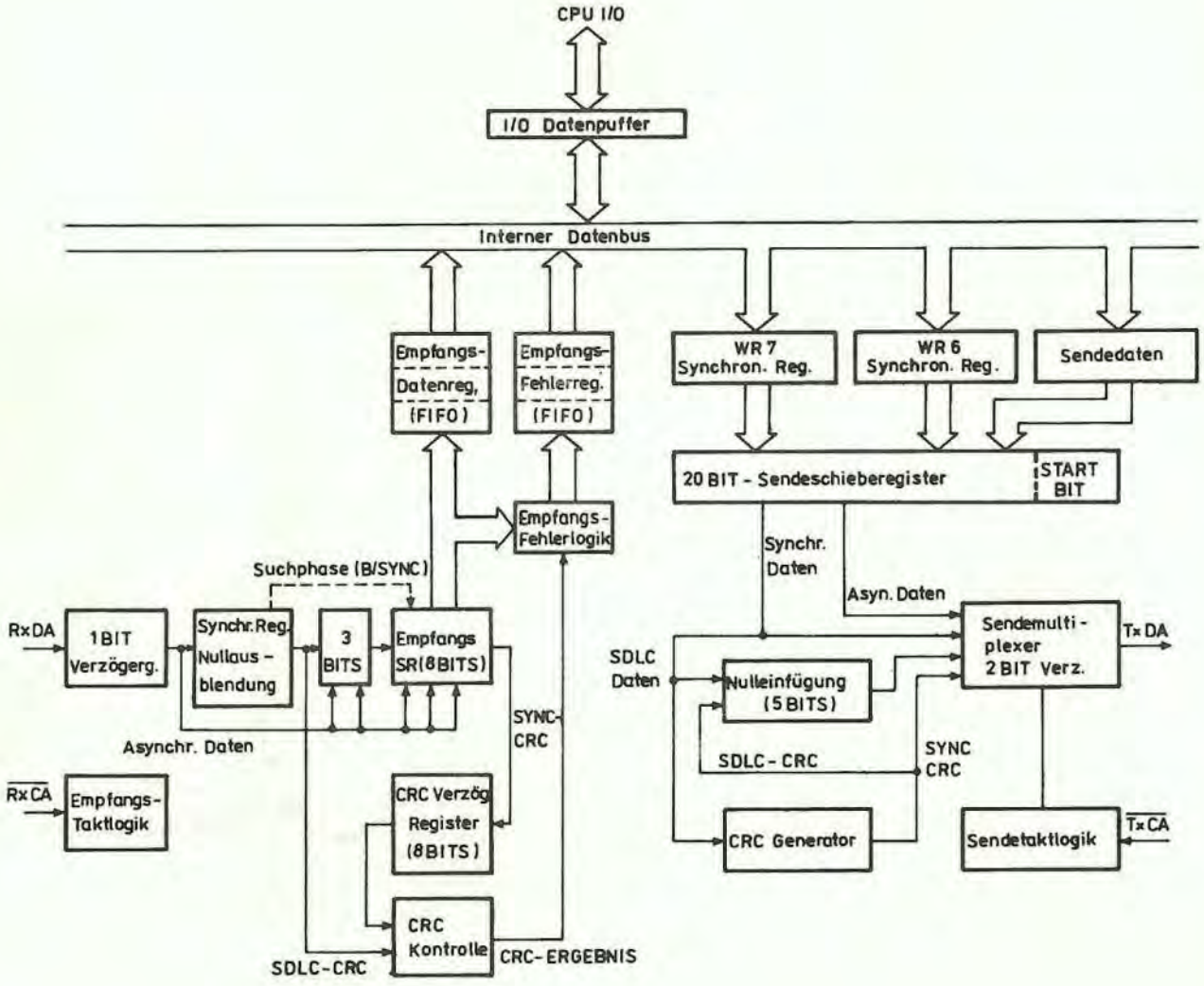


Bild 4: Sende- und Empfangsdatenweg

Ankommende Daten werden je nach Betriebsart und Zeichenlänge über einen von mehreren Wegen geleitet. In der asynchronen Betriebsart werden die seriellen Daten in einen 3-Bit-Puffer eingeschrieben, wenn ihre Zeichenlänge 7 oder 8 Bits beträgt, oder sie gelangen direkt in das 8-Bit-Schieberegister, wenn sie 5 oder 6 Bits lang sind.
Bei der synchronen Betriebsart wird der Datenweg durch die Phase des Empfangsprozesses bestimmt, die gerade abläuft.

Eine synchrone Empfangsoperation beginnt der Empfänger mit der Suchphase (Hunt-Phase), dabei wird der ankommende Datenstrom auf ein Bitmuster abgesucht, welches den vorprogrammierten Sync-Zeichen (oder Flags in der SDLC-Betriebsart) entspricht. Ist der Baustein für Monosync programmiert, erfolgt ein Vergleich mit einem einzelnen Sync-Zeichen, das in WR7 gespeichert ist. Im Bisync-Betrieb wird mit einem Doppelsynchronisationszeichen verglichen (gespeichert in WR6 und WR7). In jedem Fall passieren die ankommenden Daten das Empfangs-Sync-Register und werden mit dem programmierten Sync-Zeichen in WR6 oder WR7 verglichen. In der SDLC-Betriebsart durchlaufen die ankommenden Daten zuerst das Empfangs-Sync-Register, welches ständig den Empfangsdatenstrom überwacht und die Nullausblendung ausführt, wenn sie notwendig ist. Beim Empfang von fünf aufeinanderfolgenden Einsen wird das sechste Bit geprüft. Ist das sechste Bit eine Null, wird es aus dem Datenstrom gestrichen. Wenn das sechste Bit eine 1 ist, wird das siebente Bit geprüft. Ist dieses Bit eine Null, so wurde eine Flag-Folge empfangen, ist es eine Eins, so lag eine Abbruch (Abort)-Folge vor. Die so geänderten Daten gelangen in den 3-Bit-Puffer und werden zum Empfangsschieberegister weitergeleitet. Die SDLC-Empfangsoperation beginnt ebenfalls mit der Suchphase, in der der SIO versucht, das Zeichen im Empfangsschieberegister mit dem Flagmuster in WR7 zu vergleichen. Ist das erste Flagzeichen einmal erkannt, werden alle darauffolgenden Daten über denselben Weg geführt, unabhängig von der Zeichenlänge. Der CRC-Prüfer wird sowohl für die SDLC- als auch die synchronen Daten verwendet, jedoch ist der eingeschlagene Datenweg für jede Betriebsart unterschiedlich. Im Bisync-Format beinhaltet die byte-orientierte Arbeitsweise, daß die CPU entscheidet, ob das Datenzeichen in die CRC-Berechnung einzubeziehen ist. Um der CPU genügend Zeit zu geben, diese Entscheidung zu treffen, besitzt der SIO eine 8-Bit-Verzögerung für synchrone Daten. In der SDLC-Betriebsart ist keine Verzögerung vorgesehen, da der SIO eine Logik enthält, welche die Bytes bestimmt, bei denen CRC berechnet wird. Der Sender besitzt ein 8-Bit-Datenregister, das vom internen Datenbus geladen wird und ein 20-Bit-Sendeschieberegister, das seine Informationen von WR6, WR7 und dem Sendedatenregister erhält. WR6 und WR7 enthalten Sync-Zeichen in den Betriebsarten Monosync oder Bisync oder ein Adressenfeld (im Zeichen lang) bzw. ein Flag in der SDLC-Betriebsart. Während der synchronen Betriebsarten werden die in WR6 und WR7 enthaltenen Informationen in das Sendeschieberegister am Anfang der Nachricht geladen oder als Zeitfüller in der Mitte der Nachricht, wenn eine Transmit-Underrun (Sender - leer)-Bedingung auftritt. In der SDLC-Betriebsart werden die Flags in das Sendeschieberegister am Beginn und Ende der Nachricht eingeschrieben. Die asynchronen Daten im Sende-Schieberegister werden mit Start- und Stop-Bits versehen und mit der ausgewählten Taktrate zum Sende-Multiplexer geschoben. Synchrone (Monosync oder Bisync) - Daten gelangen zum Sendemultiplexer und auch zum CRC-Generator bei einfacher Taktrate.
SDLC/HDLC-Daten werden zur Nulleinfügungslogik weitergeleitet, die unwirksam ist, wenn die Flags gesendet werden. Für alle anderen Felder (Adresse, Steuerbits und Rahmenkontrolle) wird eine Null eingefügt, die auf fünf aufeinanderfolgende Einsen im Datenstrom folgt. Das Ergebnis des CRC-Generators für die SDLC-Daten wird auch über die Nulleinfügungslogik geleitet.

3. SIO-Zeitabläufe

Read-Zyklus (CPU-E/A-Lesezyklus)

Diese Signalfolge (Bild 5) tritt auf, wenn die CPU Daten oder Statusregisterinhalte vom SIO liest. Bei aktivierter WAIT-Funktion kann der SIO weitere WAIT-Zyklen anfordern. U 880-Eingabebefehle entsprechen diesem Zeitablauf.

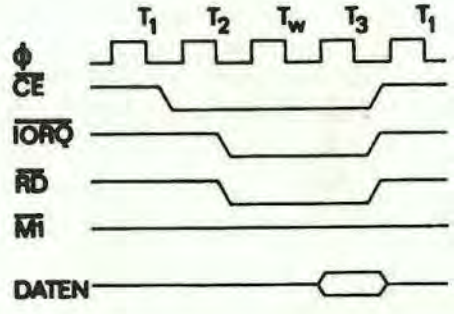


Bild 5: Read-Zyklus

WRITE-Zyklus (CPU-E/A-Schreibzyklus)

Beim Schreiben eines Daten- oder Steuerbytes von der CPU zum SIO tritt diese Signalfolge (Bild 6) auf. Bei aktivierter WAIT-Funktion kann der SIO weitere WAIT-Zyklen anfordern.
U 880-Ausgabebefehle entsprechen diesem Zeitablauf.

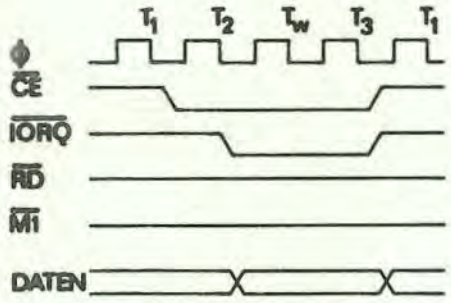


Bild 6: Write-Zyklus

Interrupt-Bestätigungs-Zyklus

Nach Erhalt eines Interrupt-Request-Signales ($\overline{INT}$ wird auf Low gezogen) sendet die CPU ein Interrupt-Bestätigungssignal (Bild 7) ($\overline{M1}$ und $\overline{IORQ}$ beide Low).

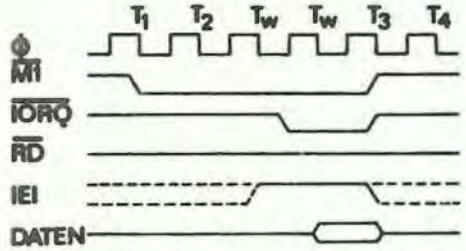
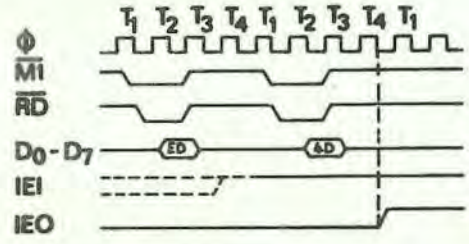


Bild 7: Interrupt-Bestätigungs-Zyklus

Die Daisy-Chain-Interrupt-Kette bestimmt den Interrupt-Anforderer mit der höchsten Priorität. Das IEI des peripheren Gerätes mit der höchsten Priorität ist auf High zu legen. Bei jedem peripheren Gerät, das keinen Interrupt anstehen hat oder gerade abarbeitet, ist IEO=IEI. Jedes periphere Gerät, das einen Interrupt anstehen oder in Bearbeitung hat, bringt seinen IEO auf Low.
Um das Einschwingen der Prioritätskette zu garantieren, können die Kanäle während $\overline{M1}$ ihren Interrupt-Aktivierungszustand nicht ändern. Ist der SIO der höchstpriorisierte Schaltkreis, plaziert er den entsprechenden Interruptvektor während $\overline{IORQ}$ auf den Datenbus.

Rückkehr vom Interruptzyklus

Normalerweise gibt die CPU einen RETI-(Return from Interrupt) Befehl am Ende einer Interrupt-Service-Routine aus. RETI ist ein 2-Byte-Operationskode (ED4D), welcher die Interrupt-Under-Service-Verriegelung zurücksetzt, um den Interrupt abzuschließen, der gerade bearbeitet wurde. Dies geschieht

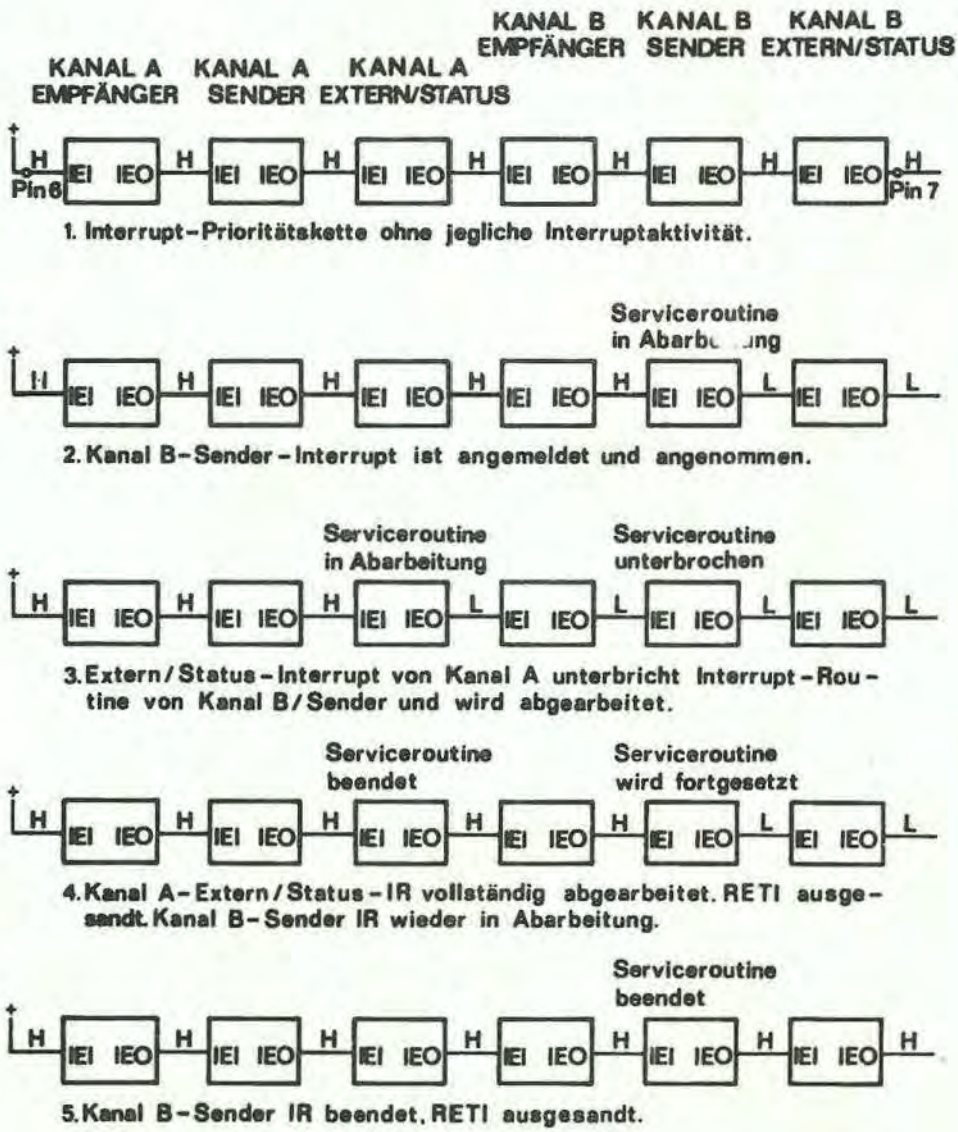


unter Benutzung der Daisy Chain auf die folgende Art und Weise. Der normale Daisy-Chain-Betrieb kann verwendet werden, um einen anstehenden Interrupt zu ermitteln; aber er kann nicht zwischen einem Interrupt in Abarbeitung und einem anstehenden nicht bestätigten Interrupt von höherer Priorität unterscheiden. Falls ein Interrupt angemeldet, aber noch nicht bestätigt ist, liegt IEO auf "Low" bis der Befehlskode "ED" auf dem Datenbus dekodiert wird. Zu diesem Zeitpunkt liegt IEO des angemeldeten Schaltkreises auf "High" bis zur Dekodierung des nächsten Datenbytes, woraufhin IEO wieder auf "Low" geht. Ist das auf "ED" folgende Befehlsbyte ein "4D", entsprach der Befehlskode einem "RETI"-Befehl. Nach der Dekodierung des Befehlsbytes "ED" hat nur der Peripherieschaltkreis IEI auf "High"

und IEO auf "Low", dessen Interrupt in Abarbeitung befindlich ist (momentan höchstpriorisierte in Abarbeitung befindliche IR). Alle anderen Peripherieschaltkreise haben IEI=IEO. Folgt auf "ED" ein "4D"-Befehlscode, setzt eben dieser Peripherieschaltkreis seine Interruptaktivierung zurück. WAIT-Zyklen sind während der $\overline{M1}$-Zyklen erlaubt, können aber nicht zur Verkürzung der High-Low-Durchlaufzeit genutzt werden.

Interrupt-Priorisierung innerhalb des SIO

Bild 9 veranschaulicht die Daisy-Chain-Anordnung von Interruptketten und ihr Verhalten bei geschachtelten Interrupts (ein Interrupt, der durch einen anderen mit einer höheren Priorität unterbrochen wird).



1. Interrupt-Prioritätskette ohne jegliche Interruptaktivität.

2. Kanal B-Sender-Interrupt ist angemeldet und angenommen.

3. Extern/Status-Interrupt von Kanal A unterbricht Interrupt-Routine von Kanal B/Sender und wird abgearbeitet.

4. Kanal A-Extern/Status-IR vollständig abgearbeitet. RETI ausgesandt. Kanal B-Sender IR wieder in Abarbeitung.

5. Kanal B-Sender IR beendet, RETI ausgesandt.

Bild 9: Typische Interrupt-Folge

Jedes Kästchen in der Darstellung könnte eine gesonderte externe periphere Schaltung sein, mit einem vom Nutzer angegebenen Reihenfolge der Interrupt-Prioritäten. Eine ähnliche Daisy-Chain-Struktur existiert auch im SIO, der sechs Interruptebenen mit einer festgesetzten Prioritätsreihenfolge hat. Der dargestellte Fall tritt ein, wenn der Sender des Kanals B ein Interrupt auslöst. Während dieser Interrupt bedient wird, wird er durch einen Interrupt höherer Priorität von Kanal A unterbrochen. Der zweite Interrupt wird bedient und - nach Beendigung - durch einen RETI-Befehl abgeschlossen. Jetzt wird die Service-Routine für Kanal B fortgeführt. Ist diese Routine beendet, bewirkt ein anderer RETI-Befehl den Abschluß des Interrupt-Service.

4. E/A-Betriebsarten

4.1. Funktionsbeschreibung

Die Funktionen des SIO können von zwei unterschiedlichen Standpunkten beschrieben werden: als Gerät zur Datenübertragung sendet und empfängt der SIO-Baustein Daten und erfüllt die Forderungen der verschiedenen Datenübertragungsformate; als peripherer Schaltkreis tritt er mit der CPU und anderen peripheren Schaltkreisen in Wechselwirkung und ist an den Daten-, Adreß- und Steuerbus angeschlossen, ebenso wie der SIO auch ein Glied in der Interruptsstruktur des Systems ist. Als peripherer Schaltkreis zu anderen Mikroprozessoren hat der SIO wertvolle Eigenschaften, wie z. B. Interrupts (ohne Vektorbildung), Polling- und einfacher Handshake-Betrieb.

Ein-Ausgabe-Möglichkeiten

Der SIO bietet die Auswahl zwischen den Betriebsarten Polling (Abfragen), Interrupts (mit und ohne Vektoren) und Blockübertragung, um Daten-, Status- und Steuerinformationen zur und von der CPU zu übertragen. Die Betriebsart Blockübertragung kann unter CPU- oder DMA-Steuerung durchgeführt werden.

Polling

In dieser Betriebsart gibt es keine Interrupts. Die Statusregister RR0 und RR1 werden zu bestimmten Zeiten für jede ausgeführte Funktion aktualisiert (z. B. CRC-Fehlerstatus wird am Ende der Nachricht bestätigt). Alle Interrupt-Arten des SIO müssen unwirksam gemacht werden, um den Schaltkreis im Abfragebetrieb zu betreiben. Im Polling-Betrieb untersucht die CPU den Status, der für jeden Kanal in RR0 enthalten ist, die RR0-Status-Bits dienen als eine Bestätigung der Abrufanfrage. Die zwei RR0-Status-Bits D_0 und D_2 zeigen an, daß das Senden oder Empfangen von Daten erforderlich ist. Der Status zeigt auch Fehler oder andere besondere Statusbedingungen an (siehe SIO-Programmierung). Der in RR1 enthaltene spezielle Empfangsbedingungszustand muß nicht abgerufen werden, da die Statusbits in RR1 mit einem gültigen Empfangszeichen-Zustand in RR0 einhergehen.

Interrupts

Der SIO bietet eine komplexe Interrupt-Struktur, um eine schnelle Interrupt-Bearbeitung in Echtzeitanwendungen zu gewährleisten. Wie bereits erwähnt, enthalten die Register WR2 und RR2 von Kanal B den Interrupt-Vektor, der auf eine Interrupt-Service-Routine im Speicher zeigt. Um Operationen in beiden Kanälen auszuführen, und um die Notwendigkeit einer Status-Analyse-Routine auszuschalten, kann der SIO den Interrupt-Vektor in RR2 verändern und somit zwischen acht verschiedenen Interrupt-Service-Routinen wählen. Dies geschieht unter Programmsteuerung durch Setzen eines Bits (WR1, D_2) in Kanal B, das als "Status Affects Vector" bezeichnet wird.

Wenn dieses Bit gesetzt ist, wird der Interruptvektor in WR2 verändert gemäß der zugeordneten Priorität der verschiedenen Interrupt-Bedingungen. Die Tabelle in der Beschreibung des Schreibregisters 1 zeigt die Einzelheiten der Veränderung.

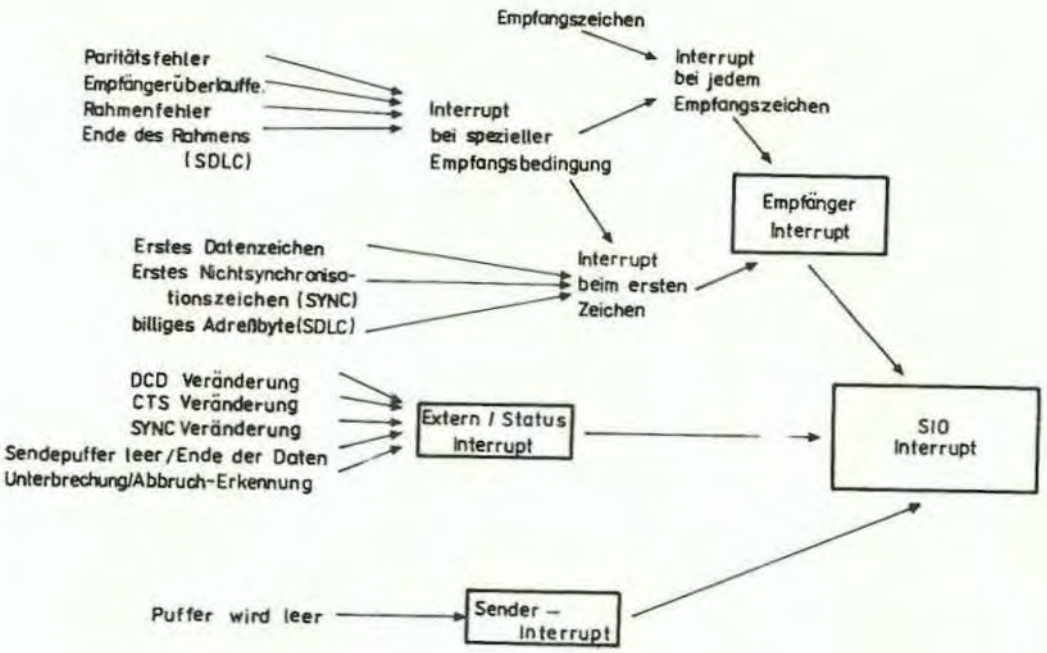


Bild 10: Interruptstruktur

Sende-Interrupts, Empfangs-Interrupts und Extern/Status-Interrupts sind die Hauptquellen für Interrupts (Bild 10). Jede Interrupt-Möglichkeit wird unter Programmsteuerung gewählt, wobei Kanal A eine höhere Priorität hat als Kanal B, und Empfänger-, Sender- und Extern/Status-Interrupts sind in dieser Reihenfolge in jedem Kanal priorisiert. Bei aktiviertem Sende-Interrupt wird die CPU durch den Sendepuffer unterbrochen, der leer wird. (Dies bedeutet, daß der Sender zuvor ein Datenzeichen ausgegeben hat, um die Bedingung des Leerwerdens zu erfüllen.)

Bei Empfangsbetrieb kann die CPU auf 3 verschiedenen Wegen unterbrochen werden:

- Interrupt bei dem ersten empfangenen Zeichen
- Interrupt bei jedem empfangenen Zeichen
- Interrupt bei einer speziellen Empfangsbedingung

Interrupt beim ersten empfangenen Zeichen wird gewöhnlich bei Blockübertragung verwendet. Interrupt bei jedem Empfangszeichen ermöglicht die Modifizierung des Interruptvektors im Falle eines Paritätsfehlers. Der Interrupt bei einer speziellen Empfangsbedingung kann auf Grundlage eines Zeichens oder von Daten erfolgen (z. B. Rahmenende-Interrupt in SDLC). Die spezielle Empfangsbedingung kann einen Interrupt nur verursachen, wenn entweder der Interrupt beim ersten Empfangszeichen oder der Interrupt bei jedem Empfangszeichen ausgewählt wurde. Im Falle des Interrupts beim ersten Empfangszeichen kann ein Interrupt bei speziellen Empfangsbedingungen (ausgenommen Paritätsfehler) nach dem ersten Empfangszeichen-Interrupt auftreten (Beispiel: Empfänger-Überlauf-Interrupt).
Die Hauptfunktion des Extern/Status-Interrupts besteht darin, die Signalübergänge der Pins $\overline{\text{CTS}}$, $\overline{\text{DCD}}$ und $\overline{\text{SYNC}}$ zu überwachen; aber ein Extern/Status-Interrupt wird ebenso durch eine Sender-leer-Bedingung oder durch die Feststellung einer Break-(Asynchrone Betriebsart) oder Abort-Folge (SDLC-Betriebsart) im Datenstrom verursacht. Der durch die Break/Abort-Folge verursachte Interrupt hat ein besonderes Merkmal, das dem SIO zu unterbrechen erlaubt, wenn die Break/Abort-Folge festgestellt wird oder beendet ist. Diese Eigenschaft erleichtert den richtigen Abschluß der laufenden Datenübertragung, korrekte Initialisierung der nächsten Daten und die genaue Zeitgebung der Break/Abort-Bedingung in der externen Logik.

CPU/DMA-Blockübertragung

Der SIO bietet eine Block-Transfer-Betriebsart, um die CPU-Blockübertragungsfunktionen und die DMA-Steuerung zu ermöglichen. Die Block-Transfer-Betriebsart benutzt den $\overline{\text{WAIT/READY}}$-Ausgang in Verbindung mit den Wait/Ready-Bits des Schreibregisters 1. Der $\overline{\text{WAIT/READY}}$-Ausgang kann unter Software-Kontrolle als Warte-Leitung in der CPU-Block-Transfer-Betriebsart oder als READY-Leitung in der DMA-Block-Transfer-Betriebsart festgelegt werden.
Der DMA-Steuerung zeigt der READY-Ausgang der SIO an, daß der SIO bereit ist, Daten zum oder vom Speicher zu übertragen. Für die CPU zeigt der WAIT-Ausgang an, daß der SIO nicht bereit ist, Daten zu übertragen, wobei die CPU aufgerufen wird, den I/O-Zyklus zu verlängern. Die Programmierung der Bits 5, 6 und 7 des Schreibregisters 1 und die logischen Zustände der $\overline{\text{WAIT-READY}}$-Leitung sind in der Beschreibung des Schreibregisters 1 festgelegt (Abschnitt über Programmierung der SIO).

Datenübertragungsmöglichkeiten

Zusätzlich zu den Ein-/Ausgabemöglichkeiten, die bereits erörtert worden sind, hat der SIO zwei unabhängige Vollduplexkanäle, welche in den Betriebsarten Asynchron, Synchron und SDLC (HDLC) programmiert werden können.
Diese unterschiedlichen Betriebsarten geben die Möglichkeit, die gewöhnlich verwendeten Datenübertragungsformate zu implementieren.
Die besonderen Merkmale dieser Betriebsarten werden in den folgenden Abschnitten beschrieben. Um die Unabhängigkeit und Vollständigkeit jedes Abschnittes zu erhalten, werden einige Informationen wiederholt, die allen Betriebsarten gemeinsam sind.

4.2. Asynchrone Betriebsart

Für den Empfang oder das Senden von Daten in asynchroner Betriebsart benötigt der SIO folgende Parameter:
Zeichenlänge, Taktrate, Anzahl der Stoppbits, gerade oder ungerade Parität, Interrupt-Betriebsart, WAIT/READY-Mode und Empfänger- oder Senderfreigabe. Durch das Systemprogramm werden diese Parameter in die entsprechenden Schreibregister geladen. Die WR4-Parameter müssen vor den WR1-, WR3- und WR5-Parametern oder Befehlen eingeschrieben werden.
Wenn die Daten über ein Modem oder ein RS232C-Interface übertragen werden, so müssen die Ausgänge $\overline{\text{REQUEST TO SEND}}$ ($\overline{\text{RTS}}$) und $\overline{\text{DATA TERMINAL READY}}$ ($\overline{\text{DTR}}$) gemeinsam mit dem Senderfreigabe-Bit gesetzt sein. Die Übertragung beginnt erst, wenn das Senderfreigabe-Bit gesetzt ist.
Die Auto-Enable-Betriebsart gestattet dem Programmierer, das erste Datenzeichen zur SIO zu senden ohne auf $\overline{\text{CTS}}$ zu warten. Ist das Auto-Enable-Bit gesetzt, wartet der SIO bis das Signal am $\overline{\text{CTS}}$-Pin Low wird, bevor er mit der Signalübertragung beginnt. $\overline{\text{CTS}}$, $\overline{\text{DCD}}$ und $\overline{\text{SYNC}}$ sind im allgemeinen Ein-/Ausgabe-Leitungen, die für andere Funktionen außer dem bezeichneten Zweck verwendet werden können. Wird $\overline{\text{CTS}}$ anderweitig genutzt, so muß das Auto-Enable-Bit auf "0" programmiert werden.

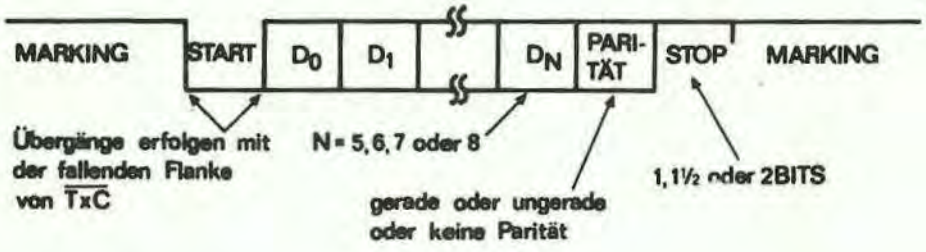


Bild 11: Asynchrones Datenformat

Bild 11 zeigt die asynchronen Übertragungsformate; Tabelle 2 enthält die WR3-, WR4- und WR5-Schreibregister mit den dazugehörigen Bits, die gesetzt werden müssen, um verwendete Betriebsarten, Parameter und Befehle im asynchronen Betrieb zu kennzeichnen. WR2 (nur Kanal B) speichert den Interruptvektor; WR1 definiert die Interrupt- und Datenübertragungsarten. WR6 und WR7 werden in der asynchronen Betriebsart nicht verwendet. Tabelle 3 zeigt die typischen Programmschritte, die zu einem Vollduplex-Empfangs-/Sendebetrieb in einem der Kanäle gehören.

4.2.1. Asynchrones Senden

Der Datenausgang des Senders (TxD) wird auf High-Pegel gehalten, solange der Sender keine Daten zum Senden hat. Unter Programmsteuerung kann der Break-Befehl (WR5, D4) erteilt werden, um den Ausgang auf Low zu halten bis der Befehl wieder gelöscht ist.
Der SIO faßt automatisch das Startbit, das programmierte Paritätsbit (ungerade, gerade oder keine Parität) und die programmierte Anzahl der Stoppbits für die zu übertragenden Datenzeichen zusammen. Wenn die Zeichenlänge sechs oder sieben Bits beträgt, werden die nichtbenutzten automatisch vom SIO ignoriert. Für den Fall, daß die Zeichenlänge fünf oder weniger Bits beträgt, verweisen wir auf die Beschreibung des Schreibregisters 5 (Abschnitt Programmierung der SIO).
Serielle Daten werden von TxD entsprechend der Taktrate mit dem Faktor 1, 1/16, 1/32 oder 1/64 der Taktfrequenz übertragen, die an den Takteingang des Senders ($\overline{\text{TxC}}$) angelegt wird. Serielle Daten werden mit der fallenden Flanke von $\overline{\text{TxC}}$ weitergeschoben.

In der Betriebsart Extern/Status-Interrupt wird der Zustand von $\overline{DCD}$, $\overline{CTS}$ und $\overline{SYNC}$ während des Sendens einer Nachricht überwacht. Verändern sich diese Signale während eines Zeitraumes, der größer ist als die angegebene Mindestimpulsbreite, so wird ein Interrupt hervorgerufen. Im Sendebetrieb wird diese Eigenschaft zur Überwachung des Modemsteuersignales $\overline{CTS}$ verwendet.

4.2.2. Asynchroner Empfang

Eine asynchrone Empfangsoperation beginnt, wenn das Empfangs-Enable-Bit gesetzt ist. Wurde der Auto-Enable-Betrieb ausgewählt, so muß $\overline{DCD}$ ebenfalls Low sein. Ein Low (Spacing) am Dateneingang des Empfängers (RxD) kennzeichnet ein Startbit. Wenn dieses Low mindestens eine halbe Bitdauer lang ist, wird das Startbit als gültig angesehen und das Dateneingangssignal wird in der Bitmitte abgetastet bis das vollständige Zeichen empfangen ist. Diese Erkennungsmethode für ein Startbit verbessert die Fehlerunterdrückung, wenn Störspitzen auf der Leitung vorhanden sind.
Wenn die Betriebsart Taktx1 gewählt wird, muß die Bitsynchronisation extern erfolgen. Die empfangenen Daten werden mit der steigenden Flanke von RxC abgetastet. Der Empfänger fügt Einsen ein, wenn eine andere Zeichenlänge als 8 Bit benutzt wird. Wurde die Parität gebildet, so wird das Paritätsbit nicht vom empfangenen Zeichen abgetrennt (außer für eine Zeichenlänge von 8 Bit). Für Zeichenlängen kleiner 8 Bit wird die erforderliche Zahl Datenbits empfangen zuzüglich eines Paritätsbits und Einsen für jedes nichtbenutzte Bit.
Ein 5-Bit-Zeichen hätte folgendes Format:

$$1\ 1\ P\ D_4\ D_3\ D_2\ D_1\ D_0.$$

Da der Empfänger durch 3 8-Bit-Register zusätzlich zum Empfangsschieberegister gepuffert ist, hat die CPU genügend Zeit, einen Interrupt zu bedienen und das durch die SIO empfangene Datenzeichen zu übernehmen. Der Empfänger besitzt ebenfalls 3 Puffer, die Fehler-Flags für jedes Datenzeichen im Empfänger-Puffer speichern. Diese Fehler-Flags werden zur gleichen Zeit wie die Datenzeichen geladen.

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
WR3	00=Rx5 Bits/Zeich. 10=Rx6 Bits/Zeich. 01=Rx7 Bits/Zeich. 11=Rx8 Bits/Zeich.		Auto Enables	0	0	0	0	Empfänger-Freigabe
WR4	00=x1 Taktrate 01=x16 Taktrate 10=x32 Taktrate 11=x64 Taktrate		0	0	00=keine 01=1 Stoppbit/Zeichen 10=1½ Stoppbit/Zeichen 11=2 Stoppbit/Zeichen		gerade, ungerade Parität	Parität
WR5 DTR	00=Tx5 Bits(od.weniger)/Zeichen 10=Tx6 Bits/Zeichen 01=Tx7 Bits/Zeichen 11=Tx8 Bits/Zeichen			Break-erzeu-gung	Sender-Freigabe	0	RTS	0

Tabelle 2: Inhalt der Schreibregister 3, 4 und 5 in der asynchronen Betriebsart

Nachdem ein Zeichen empfangen wurde, wird es auf folgende Fehlerbedingungen geprüft:

- Wurde die Parität gebildet, so wird das Paritätsfehlerbit (RR1,D_4) immer dann gesetzt, wenn das Paritätsbit des Zeichens nicht mit der programmierten Parität übereinstimmt. Dieses Bit bleibt so lange gesetzt, bis der Fehler-Rücksetz-Befehl (WRO) gegeben wird.
- Das Rahmen-Fehler-Bit (RR1,D_6) wird gesetzt, wenn das empfangene Zeichen keine Stoppbits enthält, d. h. an der Stelle des Stoppbits wurde Low-Pegel erkannt. Im Gegensatz zum Paritätsfehler-Bit wird dieses Bit nur für das Zeichen gesetzt, bei dem der Fehler aufgetreten ist. Bei Erkennen des Rahmenfehlers fügt die SIO eine weitere halbe Bitzeit zur Zeichendauer hinzu; somit wird der Rahmenfehler nicht als neues Startbit angesehen.
- Wenn die CPU ein Datenzeichen nicht lesen kann, nachdem mehr als drei Zeichen empfangen wurden, wird das Empfänger-Überlauf (Receive-Overrun)-Bit (RR1,D_5) gesetzt. Wenn dies auftritt, ersetzt das vierte empfangene Zeichen das dritte Zeichen in den Empfängerpuffern. In diesem Fall wird nur das Zeichen, welches überschrieben wurde, mit dem Empfänger-Überlauffehler-Bit gekennzeichnet. Dieses Bit kann genau wie das Paritätsfehlerbit durch den Error-Reset-Befehl von der CPU zurückgesetzt werden. Sowohl der Rahmenfehler als auch der Empfänger-Überlauffehler rufen einen Interrupt hervor, mit einem Interruptvektor, der eine spezielle Empfangsbedingung anzeigt (Voraussetzung: der statusabhängige Int-Vektor wurde ausgewählt).

Da die Paritätsfehler- und Empfängerüberlauffehler-Flags gespeichert werden, wiederspiegelt der gelesene Fehlerstatus einen Fehler im laufenden Wort im Empfängerpuffer und irgendwelche Paritäts- oder Überlauffehler, die nach dem letzten Fehlerrücksetz-Befehl aufgetreten sind. Um eine Korrespondenz zwischen dem Zustand der Fehlerpuffer und dem Inhalt der Empfangsdatenpuffer zu erhalten, muß das Fehlerstatusregister vor den Daten gelesen werden.

Tabelle 3: Asynchrone Betriebsart

Funktion	Typische Programmschritte		Bemerkungen
	Register: geladene Information		
	WRO	Kanal-RESET	Reset SIO
	WRO	Zeiger 2	
	WR2	Interruptvektor	nur Kanal B
	WRO	Zeiger 4, RESET Extern/Status-Interrupt	
	WR4	Asynchrone Betriebsart, Paritätsinformation, Anzahl der Stoppbits, Taktrate	Parameterausgabe
	WRO	Zeiger 3	
Initialisierung	WR3	Empfängerfreigabe, Auto Enables, Empfängerzeichenlänge	
	WRO	Zeiger 5	
	WR5	REQUEST TO SEND, Senderfreigabe, Senderzeichenlänge, DATA TERMINAL READY	Empfänger und Sender voll initialisiert, AUTO ENABLES gibt den Sender frei, wenn $\overline{CTS}$ aktiv ist und den Empfänger, wenn $\overline{DCD}$ aktiv ist.

	WR0 Zeiger 1, RESET Extern/Status-Interrupt	
	WR1 Sendeinterruptfreigabe, statusabhängiger Vektor, Interrupt bei jedem Empfangszeichen, Sperrung der WAIT/READY-Funktion, Extern-Interrupt-Freigabe	Sende/Empfangs-Interruptarten ausgewählt. Externer Interrupt überwacht den Status von den $\overline{CTS}$, $\overline{DCD}$ und $\overline{SYNC}$-Eingängen und erkennt die Breakfolge. Statusabhängiger Vektor nur in Kanal B.
	Übertrage erstes Datenbyte an die SIO	Dieses Datenbyte muß übertragen werden, da sonst keine Interrupts auftreten
Wartezustand	Ausführung des Haltebefehls oder irgendeines anderen Programmes	Das Programm wartet auf einen Interrupt der SIO
	Während des Interrupt-Anforderungs-/Annahme-Zyklus wird RR2 zur CPU übertragen.	Sobald der Interruptvektor erscheint, wird er modifiziert durch:
	Wenn ein Zeichen empfangen wird: - Übertragung des Datenzeichens zur CPU - Aktualisierung der Zeiger und Parameter - Rückkehr vom Interrupt	1. Verfügbare Empfangszeichen 2. Leerer Sendepuffer 3. Veränderung Extern/Status 4. Spezielle Empfangsbedingungen
Datenübertragung und Fehlerüberwachung	Wenn der Sendepuffer leer ist: - Übertragung eines Datenzeichens zur SIO - Aktualisierung der Zeiger und Parameter - Rückkehr vom Interrupt	Programmsteuerung wird an eine der acht Interrupt-Service-Routinen übertragen.
	Wenn der externe Status sich ändert: - Übertragung von RR0 zur CPU - Ausführen von Fehlerroutinen (einschließlich Breakerkennung) - Rückkehr vom Interrupt	Wenn andere Prozessoren als der U 880 verwendet werden, sollte der modifizierte Interruptvektor (RR2) im Interrupt-Anforderungs-/Annahme-Zyklus zur CPU gelangen.
	Wenn eine spezielle Empfangsbedingung auftritt: - Übertragung von RR1 zur CPU - Ausführung einer speziellen Fehlerroutine (z. B. Rahmenfehler) - Rückkehr vom Interrupt	
	Neubestimmung der Empfänger/Sender-Interruptarten	Wenn die Sender- oder Empfänger-Datenübertragung vollständig ist.
Ende der Datenübertragung	Sperrung der Sende-/Empfangs-Betriebsarten	

Aktualisierung der Modemsteuerausgänge (z. B. RTS OFF)	In der Sendebetriebsart zeigt das "Sender leer" - Statusbit D_0,RR1 an, daß der Sendevorgang abgeschlossen ist.

Diese Aufgabe kann leicht ausgeführt werden, indem man den Vektorinterrupt nutzt, da ein spezieller Interruptvektor für diese Bedingungen erzeugt wird.
Ist der Extern/Status-Interrupt freigegeben, dann verursacht die Breakerkennung einen Interrupt und das Breakerkennungs-Statusbit (RR0,D_7) wird gesetzt.
Als Reaktion zum ersten Breakerkennungsinterrupt, welcher einen Breakstatus von 1 (RR0,D_7) besitzt, sollte der "Reset Extern/Status Interrupt"-Befehl zur SIO ausgesendet werden. Der SIO überwacht den Empfänger-Daten-Eingang, um das Ende der Break-Folge zu erkennen und die CPU nach dem Nullstellen des Breakstatus zu unterbrechen. Die CPU muß in ihrer Interruptserviceroutine wieder den Reset-Befehl für den Extern/Status-Interrupt aussenden, um die Breakerkennungslogik neu zu initialisieren.
Der Extern/Status-Interrupt überwacht auch den Status von $\overline{DCD}$. Wenn das $\overline{DCD}$-Pin für einen Zeitraum inaktiv wird, der länger ist als die minimal spezifizierte Impulsbreite, dann wird ein Interrupt hervorgerufen, wobei das DCD-Statusbit (RR0,D_3) auf 1 gesetzt wird. Es ist zu beachten, daß der $\overline{DCD}$-Eingang im RR0-Statusregister invertiert wird. Wird der Status nach den Daten gelesen, sind die Fehlerdaten für das nächste Wort auch mit eingeschlossen, wenn es im Puffer gespeichert worden ist. Wenn die Operationen schnell genug ausgeführt werden, so daß das nächste Zeichen noch nicht vollständig empfangen wurde, bleibt das Statusregister für das letzte empfangene Byte gültig. Eine Ausnahme tritt auf bei der Betriebsart "Interrupt nur beim ersten Zeichen". Ein spezieller Interrupt in dieser Betriebsart speichert die Fehlerdaten und das Zeichen (selbst wenn sie vom Puffer gelesen werden) bis der Fehlerrücksetz-Befehl erteilt wird. Dies verhindert, daß weitere Daten im Empfänger zur Verfügung gestellt werden, bis der Reset-Befehl erteilt ist. Gleichzeitig kann die CPU auf das fehlerhafte Zeichen reagieren, auch wenn DMA- oder Blockübertragungstechnik angewendet wird.
Wenn die Betriebsart "Interrupt bei jedem Zeichen" ausgewählt wurde, dann ist der Interruptvektor unterschiedlich, falls in RR1 ein Fehlerstatus auftritt. Tritt ein Empfängerüberlauf auf, wird das neueste der erhaltenen Zeichen in den Puffer geladen; das vorausgehende Zeichen geht verloren. Wenn das über die anderen Zeichen geschriebene Zeichen gelesen wird, wird das Empfängerüberlaufbit gesetzt und der Vektor für die spezielle Empfangsbedingung wird ausgesandt, unter der Voraussetzung, daß der statusabhängige Vektor freigegeben ist.
Im Polling-Betrieb muß D_0 von RR0 (Zeichen im Empfängerpuffer) überwacht werden, damit die CPU weiß, wann sie ein Zeichen lesen kann. Dieses Bit wird automatisch zurückgesetzt, wenn die Empfängerpuffer gelesen sind. Um das Überschreiben von Daten im Polling-Betrieb zu vermeiden, muß der Senderpufferstatus überprüft werden, bevor in den Sender eingeschrieben wird. Das Senderpuffer-leer-Bit wird immer dann auf 1 gesetzt, wenn der Sendepuffer leer ist.

4.3. Synchrone Betriebsart

Bevor das synchrone Senden und Empfangen näher beschrieben wird, sollen die 3 Arten der Zeichensynchronisation-Monosync, Bisync und Extern-Sync- näher erläutert werden. Diese Betriebsarten benutzen den x1 Takt für Sende- und Empfangsoperationen. Die Daten werden an der steigenden Flanke des Empfangstaktes ($\overline{RxC}$) abgetastet. Die Übergänge der Sendedaten erfolgen jeweils an der fallenden Flanke des Sendetaktes ($\overline{TxC}$).
Monosync, Bisync und Extern-Sync unterscheiden sich im wesentlichen durch die Art des Erreichens der Anfangssynchronisation. Die Betriebsart muß vor dem Laden der Sync-Zeichen

ausgewählt werden, da die Register in den verschiedenen Betriebsarten unterschiedlich verwendet werden. Bild 12 zeigt die Formate für alle drei synchronen Betriebsarten.

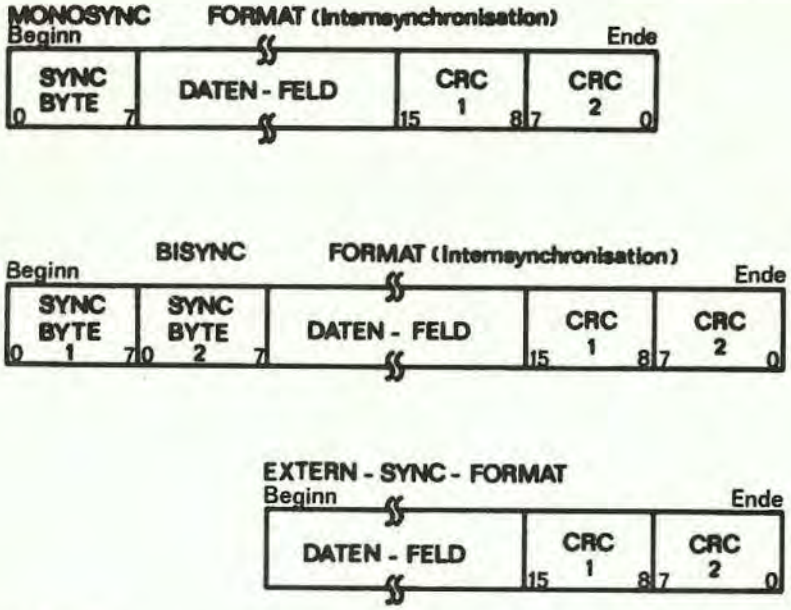


Bild 12: Synchrone Formate

Monosync:

Das Übereinstimmen eines einzelnen Sync-Zeichens (8-Bit-Sync-Betriebsart) mit dem programmierten Sync-Zeichen, das in WR7 gespeichert ist, bewirkt die Zeichensynchronisation und ermöglicht die Datenübertragung.

Bisync:

Die Zeichensynchronisation erfolgt bei Übereinstimmung von zwei aufeinanderfolgenden Sync-Zeichen (16-Bit-Sync-Betriebsart) mit den programmierten Sync-Zeichen, die in WR6 und WR7 gespeichert sind. Sowohl bei der Monosync- als auch der Bisync-Betriebsart wird $\overline{\text{SYNC}}$ als Ausgang verwendet. Dieser Ausgang ist aktiv für den Teil des Empfangstaktes, bei dem die Sync-Zeichen erkannt werden.

Mit Ausnahme von mit "01..." oder "001..." beginnenden Sync-Zeichen ist in WR7 jede beliebige Bit-Kombination verwendbar.

Extern Sync:

In dieser Betriebsart erfolgt die Zeichensynchronisation extern; SYNC ist ein Eingang, an den ein Signal zur externen Zeichensynchronisation angelegt wird. Nachdem das Sync-Muster erkannt wurde, muß die externe Logik zwei volle Empfangstakt-Zyklen warten, um den Sync-Eingang zu aktivieren. Der Sync-Eingang muß Low gehalten werden, bis die Zeichensynchronisation beendet ist. Der Zeichenempfang beginnt mit der steigenden Flanke von $\overline{\text{RxC}}$, die der fallenden Flanke von $\overline{\text{SYNC}}$ folgt.

In allen Fällen ist der Empfänger nach dem Rücksetzen in der Suchphase, während dieser Zeit versucht der SIO die Zeichensynchronisation zu erlangen. Die Suche kann nur beginnen, wenn der Empfänger freigegeben ist, und die Datenübertragung beginnt erst dann, wenn die Zeichensynchronisation vorhanden ist.

Geht die Zeichensynchronisation verloren, kann die Suchphase (Hunt Phase) von neuem beginnen. Dazu ist ein Steuerwort zum SIO zu senden, in dem das Bit D_4, WR3 (Enter Hunt Phase) gesetzt ist. Im Senderbetrieb wird immer die programmierte Anzahl von Sync-Bits (8 oder 16) ausgesandt. In der Monosync-Betriebsart benutzt der Sender das Sync-Byte aus WR6, jedoch der Empfänger vergleicht mit WR7.
In den Betriebsarten Monosync, Bisync und Extern Sync werden Daten so lange empfangen, bis der SIO zurückgesetzt wird, oder bis der Empfänger gesperrt wird (durch Befehl oder durch $\overline{DCD}$ in der Auto Enables-Betriebsart), oder bis die CPU das Enter-Hunt-Phase-Bit setzt. Nachdem erst einmal die Synchronisation erreicht worden ist, sind die Operationen der Betriebsarten Monosync, Bisync und Extern Sync im wesentlichen gleich. Die verbleibenden Unterschiede werden im folgenden näher erläutert.

Tabelle 4 zeigt, wie WR3, WR4 und WR5 in den synchronen Empfangs- und Sendeoperationen verwendet werden. WR0 dient als Zeiger für die anderen Register und enthält noch verschiedene Befehle. WR1 definiert die Interruptarten, WR2 speichert den Interruptvektor und WR6 und WR7 enthalten die Sync-Zeichen. Tabelle 5 zeigt die typischen Programmschritte für eine Halb-Duplex-Bisync-Sendeoperation.

4.3.1. Synchrones Senden

Initialisierung

Das Systemprogramm muß den Sender mit den folgenden Parametern initialisieren: ungerade oder gerade Parität, x1 Taktrate, 8- oder 16-Bit-Sync-Zeichen, CRC-Polynom, Senderfreigabe, Sendeanforderung (Request To Send), Terminal-Bereitschaft (Data Terminal Ready), Interruptarten und Sendezeichenlänge. Die WR4-Parameter müssen vor den WR1, WR3, WR5, WR6 und WR7-Parametern oder Befehlen ausgesandt werden.

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
WR3	00=Rx5 Bits/Zeichen 10=Rx6 Bits/Zeichen 01=Rx7 Bits/Zeichen 11=Rx8 Bits/Zeichen		Auto Enables	Beginne Such-Betrieb	Empfänger CRC-Freigabe	0	Sync-Ladesperre	Empfänger-Freigabe
WR4	0	0	00=8 Bit Sync-Zeich. 01=16 Bit Sync-Zeich. 10=SDLC-Mode 11=Ext Sync Mode		0	0	gerade, ungerade Parität	Parität
WR5	DTR	00=Tx5 Bits(od.wen.) /Zeichen 10=Tx6 Bits/Zeichen 01=Tx7 Bits/Zeichen 11=Tx8 Bits/Zeichen		Break-Senden	Sender-Freigabe	1 CRC-16	RTS	Sender-CRC-Freigabe

Tabelle 4: Inhalt der Schreibregister 3, 4 und 5 in den synchronen Betriebsarten

Eins der zwei Polynome - CRC-16 ($X^{16}+X^{15}+X^2+1$) oder SDLC ($X^{16}+X^{12}+X^5-1$) - kann mit synchronen Betriebsarten verwendet werden. In beiden Fällen (SDLC-Betriebsart nicht ausgewählt) werden der CRC-Generator und Tester auf Null zurückgesetzt. Im Sender-Initialisierungsprozeß wird der CRC-Generator initialisiert durch Setzen des Befehls (WR0) Reset Transmit CRC-Generator. Sowohl der Sender als auch der Empfänger benutzen das gleiche Polynom.
Sendeinterruptfreigabe oder Wait/Ready-Freigabe können für die Übertragung der Daten ausgewählt werden. Der Extern/Status-Interrupt wird zur Überwachung des Status des Clear-To-Send-Einganges verwendet, sowie für das Sender-leer/EOM-Catch. Wahlweise kann die Auto-Enable-Betriebsart verwendet werden, um den Sender freizugeben, wenn $\overline{CTS}$ aktiv ist. Die erste Datenübertragung zum SIO kann beginnen, wenn der Extern/Status-Interrupt

auftritt (CTS-Status-Bit gesetzt)oder unmittelbar nach dem Senderfreigabe-Befehl (wenn Auto-Enable-Mode vorliegt). Wenn der Sender nicht freigegeben ist, wird der Senderausgang auf High gehalten. Ein Break kann programmiert werden, um ein Low (Spacing) zu erzeugen, welches beginnt, sobald das Send-Break-Bit gesetzt ist. Werden einem voll initialisierten und freigegebenem Sender keine Daten zugeführt (default condition) so sendet er ständig 8- oder 16-Bit-Sync-Zeichen.

Datenübertragung und Statusüberwachung

In dieser Phase gibt es mehrere Kombinationen von Interrupts und Wait/Ready.

Datenübertragung mit Interruptbetrieb

Wenn das Transmit-Interrupt-Enable-Bit (WR1,D_1) gesetzt ist, wird jedesmal ein Interrupt erzeugt, wenn der Senderpuffer leer wird. Dieser Interrupt kann entweder beantwortet werden durch Einschreiben eines anderen Zeichens in den Sender oder durch Rücksetzen des Transmitter Interrupt Pending Catch mit dem Befehl (WR0,CMD5) Reset Transmitter Pending. Wenn dem Interrupt mit diesem Befehl Genüge getan wurde und nichts mehr in den Sender eingeschrieben wird, kann es keine weiteren Senderpuffer-leer-Interrupts (Transmit Buffer Empty) mehr geben, weil gerade der Vorgang, daß der Puffer leer wird, die Interrupts hervorruft und der Puffer nicht leer werden kann, wenn er bereits leer ist. Diese Situation verursacht eine Sender-leer-(Transmit-Underrun)-Bedingung, welche im Abschnitt "Bisync Transmit Underrun" erläutert wird.

Datenübertragung unter Benutzung von WAIT/READY

Der CPU zeigt die Aktivierung von $\overline{\text{WAIT}}$ an, daß der SIO nicht bereit ist, Daten anzunehmen und daß die CPU den Ausgabezyklus verlängern muß. Für eine DMA-Steuerung zeigt READY an, daß der Senderpuffer leer ist, und daß der SIO bereit ist, das nächste Datenzeichen anzunehmen. Wenn das Datenzeichen nicht rechtzeitig in den SIO geladen wird, wird das Sendeschieberegister leer und der SIO tritt in die Transmit-Underrun-Bedingung ein.

Bisync Transmit Underrun

Im Bisync-Format werden Füllzeichen eingesetzt, um die Synchronisation aufrecht zu erhalten, wenn der Sender keine Daten zum Senden hat (Transmit Underrun Condition). Der SIO hat zwei programmierbare Möglichkeiten für die Lösung dieser Situation: er kann Sync-Zeichen einfügen, oder er kann die bisher erzeugten CRC-Zeichen senden, gefolgt von den Sync-Zeichen. Diese Möglichkeiten werden durch den Befehl Reset Transmit Underrun/EOM in WR0 gesteuert. Nach einem Baustein- oder Kanal-Reset ist das Transmit Underrun/EOM Status-Bit (RR0,D_6) gesetzt und erlaubt das Einfügen von Sync-Zeichen, wenn es keine Daten zu senden gibt. CRC wird für die automatisch eingefügten Sync-Zeichen nicht berechnet. Wenn die CPU das Ende der Nachricht feststellt, kann ein Reset Transmit Underrun/EOM-Befehl ausgegeben werden. Das ermöglicht, daß CRC gesendet wird, wenn der Sender keine Daten hat. In diesem Fall sendet der SIO CRC, gefolgt von Sync-Zeichen, um die Nachricht abzuschließen.
Es gibt keine Einschränkung dafür, wann in der Nachricht das Transmit-Underrun/EOM-Bit zurückgesetzt werden kann. Wenn EOM-Reset ausgegeben wird, nachdem das erste Datenzeichen geladen worden ist, wird das 16-Bit-CRC gesendet, gefolgt von Sync-Zeichen zu dem Zeitpunkt, wenn der Sender keine Daten zum Senden hat. Auf Grund der Transmit Underrun-Bedingung wird ein Extern/Status-Interrupt erzeugt, wenn das Transmit Underrun/EOM-Bit gesetzt wird.
Im Falle der Sync-Zeichen-Einfügung wird ein Interrupt erst erzeugt, nachdem das erste automatisch eingefügte Sync-Zeichen geladen worden ist. Der Status zeigt an, daß das Transmit-Underrun/EOM-Bit und das Transmit-Buffer-Empty-Bit gesetzt sind.
Im Falle der CRC-Einfügung, wird das Transmit Underrun/EOM-Bit gesetzt und das Transmit-Buffer-Empty-Bit ist rückgesetzt, während CRC gesendet wird. Wenn CRC vollständig gesendet worden ist, wird das Transmit-Buffer-Empty-Status-Bit gesetzt und ein Interrupt erzeugt, um der CPU anzuzeigen, daß ein anderer Datenstrom beginnen kann (dieser Interrupt

tritt auf, weil CRC gesendet und Sync geladen worden ist). Wenn keine weiteren Daten mehr zu senden sind, kann das Programm die Übertragung abschließen, indem RTS rückgesetzt und der Sender gesperrt (WR5,D_3) wird.
Auffüllzeichen können gesendet werden, indem der SIO auf 8 Bits/Zeichen programmiert wird und FF in den Sender eingeschrieben wird, während CRC am Ausgang des Senders erscheint. Als Alternative können die Sync-Zeichen während dieser Zeit als Auffüllzeichen (pad characters) umdefiniert werden.
Das folgende Beispiel wurde angeführt, um diesen Punkt zu verdeutlichen.
Der SIO erzeugt einen Interrupt (durch das Setzen des Transmit-Buffer-Empty-Bits).
Die CPU erkennt, daß das letzte Zeichen (ETX) der Nachricht bereits zur SIO gesendet wurde, indem sie den internen Programmstatus überprüft.

Um den SIO zum Senden von CRC zu veranlassen, sendet die CPU den Reset Transmit Underrun/EOM Latch-Befehl (WRO) aus und genügt dem Interrupt mit dem Reset-Transmit-Interrupt-Pending-Befehl. (Dieser Befehl verhindert, daß der SIO mehr Daten anfordert.) Da durch diesen Befehl ein Transmit-Underrun hervorgerufen wird, beginnt der SIO CRC zu senden. Der SIO ruft auch einen Extern/Status-Interrupt hervor durch das Setzen des Transmit-Underrun/EOM Latch.
Die CPU genügt diesem Interrupt, indem sie Füllzeichen (pad characters) in den Senderpuffer lädt und den Befehl Reset Extern/Status-Interrupt aussendet.
Bei diesem Ablauf folgen dem CRC Füllzeichen an der Stelle von Sync-Zeichen. Man beachte, daß der SIO mit einem Transmit-Buffer-Empty-Interrupt reagiert, wenn CRC komplett gesendet wurde und die Füllzeichen in das Senderschieberegister geladen wurden.
Von diesem Punkt an kann die CPU mehr Füllzeichen oder Sync-Zeichen senden.

Bisync CRC-Erzeugung

Durch Setzen des Sender-CRC-Freigabe-Bits (WR5,D_0) beginnt die CRC-Berechnung, wenn das Programm das erste Datenzeichen zur SIO sendet. Der SIO sendet zwar automatisch bis zu zwei Sync-Zeichen (16-Bit-Sync), dennoch ist es ratsam, ein paar Sync-Zeichen mehr der Nachricht vorauszuschicken (bevor Sende-CRC wirksam gemacht wird), um die Synchronisation an der Empfangsseite zu sichern.
Das Sender-CRC-Freigabe-Bit kann jederzeit während der Datenübertragung verändert werden, um bestimmte Datenzeichen in die CRC-Berechnung einzubeziehen oder sie auszuschließen. Das Sender-CRC-Freigabe-Bit muß in dem gewünschten Zustand sein, wenn das Datenzeichen vom Sende-Daten-Puffer in das Sendeschieberegister geladen wird. Um zu sichern, daß sich dieses Bit im richtigen Zustand befindet, muß zuerst das Sender-CRC-Freigabe-Bit ausgesendet werden, bevor das Datenzeichen zur SIO gelangt.

Sender-Transparent-Betriebsart

Die Transparent-Betriebsart (Bisync-Protokoll) wird ermöglicht durch die Fähigkeit, die Sender-CRC-Freigabe fliegend zu ändern und durch die zusätzliche Möglichkeit 16-Bit-Sync-Zeichen einzufügen. Der Ausschluß von DLE Zeichen aus der CRC-Berechnung kann durch Sperrung der CRC-Berechnung erreicht werden, bevor die DLE-Zeichen zur SIO übertragen werden. Im Fall einer Transmit-Underrun-Bedingung im Transparent-Mode werden ein paar DLE-SYN-Zeichen gesendet. Der SIO kann zum Senden einer DLE-SYN-Folge programmiert werden, indem ein DLE-Zeichen in WR6 und ein Sync-Zeichen in WR7 geladen wird.

Sendeabschluß

Der SIO besitzt eine spezielle Abschlußbehandlung, die die Datenzusammengehörigkeit und -gültigkeit bewahrt. Wenn während der Aussendung eines Daten- oder Sync-Zeichens der Sender gesperrt wird, so wird dieses Zeichen wie gewöhnlich gesendet, aber es folgt dann kein CRC- oder Sync-Zeichen, sondern der Ausgang geht auf High (marking line).
Bei der Sperrung des Senders bleibt ein im Puffer befindliches Zeichen erhalten.
Wenn während der CRC-Aussendung der Sender gesperrt wird, wird die 16-Bit Übertragung abgeschlossen, aber es werden Sync-Zeichen gesendet anstatt CRC.

Ein programmierter Abbruch (Break) wird wirksam, sobald er in das Befehlsregister eingeschrieben worden ist; Zeichen im Sendepuffer und im Schieberegister gehen verloren.
In allen Betriebsarten werden Zeichen mit dem niedrigsten Bit zuerst gesendet. Das erfordert ein rechtsbündiges Ausrichten der zu sendenden Daten, wenn die Wortlänge weniger als 8 Bits beträgt. Ist die Wortlänge fünf Bits oder weniger, muß das Sonderverfahren, das in der Diskussion des Schreibregisters 5 beschrieben ist (Programmierung des SIO), für das Datenformat angewendet werden. Die Zustände der unbenutzten Bits in einem Datenzeichen sind irrelevant, ausgenommen bei einem fünf-Bit- oder weniger als fünf Bit-Mode.
Wenn das Extern/Status-Interrupt-Enable-Bit gesetzt ist, verursachen Senderbedingungen wie "beginne CRC-Zeichen zu senden", "beginne Sync-Zeichen zu senden" und ein $\overline{\text{CTS}}$ verändernder Zustand Interrupts, die einen einheitlichen Vektor haben, wenn der statusabhängige Vektor gesetzt ist. Diese Interrupt-Betriebsart kann während Blockübertragungen verwendet werden.
Alle Interrupts können gesperrt werden (für Operationen im Polling-Betrieb, oder um Interrupts zu unpassenden Zeiten während der Abarbeitung eines Programms zu unterbinden).

Tabelle 5: Bisync-Sende-Betriebsart

Funktion	Typische Programmschritte	Bemerkungen
Register:	geladene Information	
WR0	Kanal Reset, Reset Sender-CRC-Generator	
WR0	Zeiger 2	
WR2	Interruptvektor	nur Kanal B
WR0	Zeiger 3	
WR3	Auto Enables	Übertragung beginnt nur nachdem $\overline{\text{CTS}}$ erkannt wurde
WR0	Zeiger 4	
WR4	Paritätsinformation, Information über Sync-Modes, x1-Taktrate	Senderparameter ausgegeben
Initialisierung		
WR0	Zeiger 6	
WR6	Sync-Zeichen 1	
WR0	Zeiger 7, Reset Extern/Status-Interrupts	
WR7	Sync-Zeichen 2	
WR0	Zeiger 1, Reset Extern/Status-Interrupts	
WR1	Statusabh. Vektor, Extern-Interrupt-Freigabe, Sender-Interrupt-Freigabe oder WAIT/READY-Mode freigegeben	Die Extern-Interrupt-Betriebsart überwacht den Status der $\overline{\text{CTS}}$- und $\overline{\text{DCD}}$-Eingänge sowie den Status der Tx-Underrun/EOM-Speicherzelle. Der Senderinterrupt wird wirksam, wenn der Senderpuffer leer wird; die Wait/Ready-Betriebsart kann zum Datentransfer mit CPU oder DMA genutzt werden.

Funktion	Typische Programmschritte	Bemerkungen
WR0	Zeiger 5	Statusabh. Vektor (nur Kanal B)
WR5	Request To Send, Senderfreigabe, Bisync CRC, Sendezeichenlänge	Die Sender-CRC-Freigabe sollte erfolgen, wenn das erste Nicht-Sync-Zeichen zur SIO gesendet wird.
	Erstes Sync-Byte zur SIO	Benutze mehrere Sync-Zeichen am Beginn der Datenübertragung. Der Sender ist voll initialisiert.
Wartezustand	Ausführung einer Halt-Operation oder eines anderen Programmteils	Es wird auf einen Interrupt gewartet oder den Wait/Ready-Ausgang, um Daten zu übertragen.
Datentransfer und Statusüberwachung	Wenn ein Interrupt (Wait/Ready) auftritt: - Einbeziehung/Ausschließung eines Datenbytes von der CRC-Berechnung. - Übertragung eines Datenbytes von der CPU (oder Speicher) zur SIO - Erkennung und Setzen zugeordneter Flags für Steuerzeichen (in der CPU). - Reset Tx Underrun/EOM Latch (WR0), wenn das letzte Zeichen der Daten erkannt wurde - Aktualisierung der Zeiger und Parameter (CPU). - Rückkehr vom Interrupt.	Ein Interrupt tritt auf (Wait/Ready wird aktiv), wenn das erste Datenbyte gesendet wurde. Die Wait-Betriebsart erlaubt der CPU einen Blocktransfer vom Speicher zur SIO, die Ready-Betriebsart erlaubt der DMA einen Blocktransfer vom Speicher zur SIO. Der DMA-Baustein kann auf das Erkennen spezieller Steuerzeichen programmiert werden (es werden nur die Bits überprüft, die ASCII- oder EBCDIC-Steuerzeichen spezifizieren). Das Vorhandensein der Zeichen löst einen Interrupt aus.
	Wenn Fehlerbedingungen oder Status-Veränderungen auftreten: - Übertragung von RR0 zur CPU - Ausführung einer Fehlerroutine - Rückkehr vom Interrupt	Tx Underrun/EOM zeigt entweder eine Sender-leer-Bedingung an (Sync-Zeichen wurden gesendet), oder das Ende der Datenübertragung (CRC-16 wurde gesendet).
Ende der Datenübertragung	Neubestimmung der Interruptarten Aktualisierung der Modem-Steuer-Ausgänge (z. B. RTS ausschalten) Sperrung Sendebetriebsart	Das Programm soll die Datenübertragung ordnungsgemäß abschließen.

4.3.2. Synchroner Empfang

Initialisierung

Das Systemprogramm initialisiert die synchronen Empfangsoperationen mit den folgenden Parametern: ungerade oder gerade Parität, 8- oder 16-Bit-Sync-Zeichen, x1-Takt-Betriebsart, CRC-Polynom, Empfangszeichenlänge usw.. Sync-Zeichen müssen in die Register WR6 und WR7 geladen werden. Die Empfänger können nur freigegeben werden, nachdem alle Empfangsparameter gesetzt worden sind. Die WR4-Parameter müssen vor den WR1-, WR3-, WR5-, WR6- und WR7-Parametern oder Befehlen ausgegeben werden.
Nach der Freigabe befindet sich der Empfänger in der Suchphase (Hunt-Phase). Er bleibt in dieser Phase, bis die Zeichensynchronisation erreicht worden ist. Es ist zu beachten, daß unter Programmsteuerung alle führenden Synczeichen der Daten vom Laden in die Empfangspuffer ausgeschlossen werden können, indem das Sync-Character-Load- Bit in WR3 gesetzt wird.

Datenübertragung und Statusüberwachung

Nachdem die Zeichensynchronisation erreicht worden ist, werden die empfangenen Zeichen in den Empfangsdaten-F/FP übertragen. Die folgenden vier Interrupt-Betriebsarten stehen für die Übertragung der Daten und des dazugehörigen Statusses an die CPU zur Verfügung.

Keine Interrupts freigeben

Diese Betriebsart wird für Polling-Operationen oder für off-line-Bedingungen benutzt.

Interrupt nur beim ersten Zeichen

Diese Betriebsart wird normalerweise benutzt zum Starten einer Polling-Schleife oder eines Block-Transfer-Befehls, indem $\overline{\text{Wait/Ready}}$ genutzt wird, um die CPU oder die DMA auf die ankommende Datenrate zu synchronisieren. In dieser Betriebsart unterbricht der SIO beim ersten Zeichen und danach entsteht nur ein Interrupt, wenn eine spezielle Empfangsbedingung erkannt wurde. Die Betriebsart wird von neuem mit dem Befehl "Enable-Interrupt-On-Next-Receive-Character" initialisiert, damit das nächste Zeichen empfangen werden kann, um einen Interrupt zu erzeugen. Paritätsfehler rufen in dieser Betriebsart keine Interrupts hervor. Dies ist nicht der Fall bei Rahmenende (SDLC-Betriebsart) und Empfangsüberlauf.

In der Betriebsart Extern/Status-Interrupt können $\overline{\text{DCD}}$-Status-Veränderungen jederzeit einen Interrupt hervorrufen.

Interrupt bei jedem Zeichen

Immer wenn ein Zeichen in den Empfangspuffer gelangt, wird ein Interrupt erzeugt. Fehler und spezielle Empfangsbedingungen erzeugen einen speziellen Vektor, wenn der statusabhängige Vektor ausgewählt wurde. Wahlweise kann eine Betriebsart benutzt werden, wo der Paritätsfehler keinen speziellen Interruptvektor erzeugt.

Interrupts bei speziellen Empfangsbedingungen

Der Interrupt bei spezieller Empfangsbedingung kann nur dann auftreten, wenn entweder die Betriebsart "Interrupt nur beim ersten Zeichen" oder "Interrupt bei jedem Empfangszeichen" ausgewählt wurde. Der Interrupt bei spezieller Empfangsbedingung wird durch die Empfängerüberlaufbedingung hervorgerufen. Da die Empfangsüberlauf- und Paritätsfehlerstatusbits gespeichert sind, wiederspiegelt der Fehlerstatus - wenn er gelesen wird - einen Fehler im laufenden Wort im Empfangspuffer zusätzlich zu allen Paritäts- oder Überlauffehler, die seit dem letzten Fehler-Rücksetz-Befehl empfangen wurden. Diese Status-Bits können nur durch den Fehler-Rücksetz-Befehl zurückgesetzt werden.

CRC-Fehlerkontrolle und Abschluß

Eine CRC-Fehlerkontrolle der Empfangsnachricht kann unter Programmsteuerung pro Zeichen erfolgen. Das Empfangs-CRC-Freigabe-Bit (WR3,D_3) muß durch das Programm gesetzt/rückgesetzt werden, bevor das nächste Zeichen vom Empfangsschieberegister in das Empfangs-Puffer-Register übertragen wird. Dies sichert das richtige Ein- oder Ausschließen von Datenzeichen in die CRC-Kontrolle.
Um der CPU genügend Zeit zu lassen, die CRC-Kontrolle an einem besonderen Zeichen wirksam oder unwirksam zu machen, berechnet der SIO-CRC acht Bit-Zeiten nachdem das Zeichen zum Empfangspuffer übertragen worden ist. Wenn die CRC-Berechnung freigegeben ist, bevor das nächste Zeichen übertragen wird, dann wird CRC für das übertragene Zeichen berechnet. Wenn CRC gesperrt wird vor der nächsten Übertragung, dann wird die Berechnung für das anstehende Wort fortgesetzt, aber das Wort, das gerade in den Puffer übertragen wird, wird nicht einbezogen. Wenn diese Forderungen erfüllt werden müssen, kann der 3 Byte-Empfangsdatenpuffer für eine byteorientierte Bisync-Operation nicht in voller Tiefe genutzt werden. CRC kann freigegeben oder gesperrt werden so oft es für eine gegebene Berechnung notwendig ist.
In den Monosync-, Bisync- und Extern-Sync-Betriebsarten enthält das CRC/Rahmenfehlerbit (RR1,D_6) das Vergleichsresultat der CRC-Kontrolle 16 Bitzeiten (acht Bits Verzögerung und acht Bits Verschiebung für CRC) nachdem das Zeichen vom Empfangsschieberegister in den Puffer übertragen wurde. Für eine fehlerfreie Übertragung sollte das Resultat Null sein. (Beachte, daß das Resultat nur am Ende der CRC-Berechnung gültig ist. Wenn das Ergebnis vor diesem Zeitpunkt überprüft wird, zeigt es gewöhnlich einen Fehler an.) Der Vergleich wird mit jeder Übertragung hergestellt und ist nur so lange gültig, so lange das Zeichen im Empfangs-F/FO verbleibt.
Es folgt ein Beispiel für die CRC-Kontroll-Operation, wenn 4 Zeichen (A, B, C und D) in folgender Ordnung empfangen werden.

Zeichen A wird in den Puffer geladen.
Zeichen B wird in den Puffer geladen.

Wenn CRC gesperrt ist, bevor C sich im Puffer befindet, dann wird CRC nicht für B berechnet.

Zeichen C wird in den Puffer geladen.

Nachdem C geladen ist, zeigt das CRC-Rahmenfehlerbit das Ergebnis des Vergleiches durch Zeichen A.

Zeichen D wird in den Puffer geladen.

Nachdem D im Puffer ist, zeigt das CRC-Fehlerbit das Ergebnis des Vergleiches durch Zeichen B, ob nun B in die CRC-Berechnung einbezogen wurde oder nicht.
Infolge der seriellen Natur der CRC-Berechnung muß der Empfangstakt ($\overline{RxC}$) 16 mal (8-Bit-Verzögerung plus 8 Bit CRC-Verschiebung) erscheinen, nachdem das zweite CRC-Zeichen in den Empfangspuffer geladen wurde, oder 20 mal (die erwähnten 16 plus 3 Bit Puffer-Verzögerung und 1 Bit Eingangsverzögerung) nachdem das letzte Bit im RxD Eingang ist, bevor die CRC-Berechnung komplett ist. Ein schneller externer Takt kann an dem Empfangstakt-Eingang angeschlossen werden, um die geforderten 16 Zyklen zu liefern.

Die typischen Programmschritte für einen Halb-Duplex-Bisync-Empfangsbetrieb sind in Tabelle 6 dargestellt.

Tabelle 6: Bisync-Betriebsart

Funktion		Typische Programmschritte	Bemerkungen
	Register	geladene Information	
	WR0	Kanal Reset, Reset Empfangs-CRC-Prüfer	Reset SIO; Initialisierung Empfangs-CRC-Prüfer
	WR0	Zeiger 2	
	WR2	Interruptvektor	nur Kanal B
	WR0	Zeiger 4	
	WR4	Paritätsinformation, Sync-Betriebsart, x1 Taktrate	Ausgabe Empfangsparameter
	WR0	Zeiger 5, Reset Extern Status Interrupt	
	WR5	Bisync CRC-16, Data Terminal Ready	
	WR0	Zeiger 3	
Initialisierung	WR3	Sync-Ladesperre, Empfänger-CRC-Freigabe, Beginne Suchbetrieb (Hunt Mode), Empfängerzeichenlänge	Die Sync-Ladesperre verhindert das Laden der führenden Sync-Zeichen am Beginn der Nachricht. Im Auto-Enables-Mode werden die Daten erst dann angenommen, wenn der $\overline{DCD}$-Eingang aktiv ist.
	WR0	Zeiger 6	
	WR6	Sync-Zeichen 1	
	WR0	Zeiger 7	
	WR7	Sync-Zeichen 2	
	WR0	Zeiger 1, Reset Extern/Status-Interrupt	
	WR1	Statusabh. Vektor, Extern-Interrupt-Freigabe, Empfangsinterrupt nur beim ersten Zeichen	In dieser Interrupt-Betriebsart wird nur das erste Datenzeichen zur CPU übertragen. Alle folgenden Daten werden auf der Basis eines DMA übertragen; jedoch unterbrechen Interrupts bei einer speziellen Empfangsbedingung die CPU. Der statusabhängige Vektor wird nur in Kanal B genutzt.
	WR0	Zeiger 3, Interruptfreigabe für das nächste Empfangszeichen	Das Rücksetzen dieser Interrupt-Betriebsart liefert eine einfache Programmschleife für den Beginn weiterer Datenübertragungen
	WR3	Empfängerfreigabe, Sync-Ladesperre, Beginn Suchbetrieb (Hunt Mode), Auto Enable, Empfängerwortlänge	WR3 wird erneut ausgegeben, um den Empfänger freizugeben. Die Empfänger-CRC-Freigabe muß gesetzt werden, nachdem SOH- oder STX-Zeichen empfangen wurden.

Funktion	Typische Programmschritte	Bemerkungen
Wartezustand	Ausführung des Haltbefehls oder anderer Programmteile	Die Empfänger-Betriebsart ist voll initialisiert und das System wartet auf einen Interrupt beim ersten Zeichen
	Wenn ein Interrupt beim ersten Zeichen auftritt, führt die CPU folgendes aus: - Übertragung des Datenbytes zur CPU - Erkennen und Setzen der zugehörigen Flags für die Steuerzeichen (in der CPU) - Ein-/Ausschließen von Datenbytes in die CRC-Kontrolle - Aktualisierung der Zeiger und anderen Parameter - Freigabe von Wait/Ready für die DMA-Operation - Freigabe des DMA-Controllers - Rückkehr vom Interrupt	Während der Suchphase (Hunt Mode) erkennt der SIO zwei aufeinanderfolgende Zeichen für die Herstellung der Synchronisation. Die CPU stellt die DMA-Betriebsart her und alle nachfolgenden Datenzeichen werden durch die DMA übertragen.
Datenübertragung und Statusüberwachung		Das DMA-Steuergerät wird auch dazu programmiert, spezielle Zeichen zu erkennen (durch Überprüfung nur solcher Bits die ASCII- oder EBCDIC-Steuerzeichen entsprechen) und die CPU bei ihrer Feststellung zu unterbrechen. Als Reaktion überprüft die CPU den Status oder die Steuerzeichen und leitet entsprechende Handlungen ein (z. B. CRC-Freigabe aktualisieren).
	Wenn Wait/Ready aktiv wird, dann führt der DMA-Controller folgendes aus: - Übertragung des Datenbytes zum Speicher - Unterbrechung der CPU, wenn ein spezielles Zeichen durch den DMA-Controller erkannt wurde - Unterbrechung der CPU, wenn das letzte Zeichen der Nachricht erkannt wurde	
	Zur Beendigung der Datenübertragung führt die CPU folgendes aus: - Übertragung von RR1 zur CPU - Setzen des ACK/NAK-Antwort-Flags basierend auf dem CRC-Resultat - Aktualisierung der Zeiger und Parameter - Rückkehr vom Interrupt	Der SIO ruft bei einer Fehlerbedingung in der CPU einen Interrupt hervor und die Fehlerroutine unterbricht die laufende Nachrichtenübertragung, klärt die Fehlerbedingung und wiederholt die Operation.
Ende der Datenübertragung	Neubestimmung der Interrupt- und Sync-Betriebsarten Aktualisierung der Modem-Steuersignale Sperrung der Empfangsbetriebsart	

4.4. SDLC (HDLC) Betriebsart

Der SIO kann sowohl das Format "High-level Synchronous Data link Control" (HDLC) als auch das Format "IBM Synchrons Data Link Control" (SDLC) übertragen. Da sich SDLC und HDLC sehr ähnlich sind, wird in dem folgenden Text nur auf SDLC Bezug genommen.

Die SDLC-Betriebsart unterscheidet sich beträchtlich von dem synchronen Bisync-Format, weil es bitorientiert und nicht zeichenorientiert ist, wodurch natürlich auch eine transparente Operation ausgeführt werden kann. Die Bitorientierung macht SDLC zu einem flexiblen Format hinsichtlich Nachrichtenlänge und Bitmuster. Der SIO besitzt mehrere Einrichtungen für die Bearbeitung verschiedener Nachrichtenlängen.
Die SDLC-Nachricht, die man auch als Rahmen (Bild 13) bezeichnet, wird durch Flags eröffnet und beendet ähnlich den Sync-Zeichen im Bisync-Protokoll. Der SIO überträgt und erkennt Flagzeichen, die den Anfang und das Ende des Rahmens kennzeichnen.

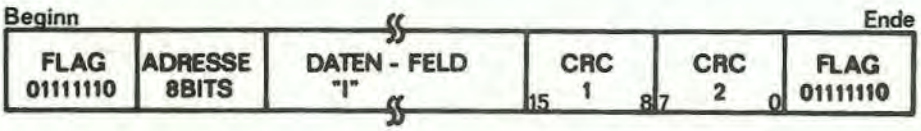


Bild 13: Sende/Empfangs-SDLC/HDLC-Datenformat

Es ist zu beachten, daß der SIO Null-Flags empfangen kann, aber nicht aussenden. Das 8-Bit-Adressenfeld eines SDLC-Rahmens enthält die sekundäre Stationsadresse. Der SIO besitzt eine Adressen-Such-Betriebsart, die die sekundäre Stationsadresse erkennt und so den Rahmen annehmen oder ablehnen kann.
Da das Steuerfeld des SDLC-Rahmens für den SIO transparent ist, ist es einfach, ihn zur CPU zu übertragen. Der SIO bearbeitet die Rahmen-Kontroll-Folge so, daß das Programm vereinfacht wird durch die Einbeziehung von folgenden Arbeitsweisen,wie die Initialisierung des CRC-Generators auf alles Einsen, das Rücksetzen des CRC-Prüfers, wenn das Anfangsflag in der Empfangsbetriebsart erkannt wurde und das Senden der Rahmen-Kontroll/ Flag-Folge in der Sendebetriebsart. Die Hardware für die Steuerung wird dadurch vereinfacht, daß eine automatische Null-Einfügungs- und Ausblendlogik im SIO enthalten ist. Tabelle 7 zeigt den Inhalt von WR3, WR4 und WR5 während der SDLC-Empfangs- und Sendebetriebsarten. WR0 dient als Zeiger für die anderen Register und enthält verschiedene Befehle. WR1 definiert die Interrupt-Betriebsarten. WR2 enthält den Interruptvektor. WR7 speichert das Flag-Zeichen und WR6 die sekundäre Adresse.

4.4.1. SDLC-Senden

Initialisierung

Wie im synchronen Betrieb muß die SDLC-Sendebetriebsart mit den folgenden Parametern initialisiert werden: SDLC-Betriebsart, SDLC-Polynom, Sendeanforderung (Request To Send), Terminalbereitschaft (Data Terminal Ready) Senderzeichenlänge, Senderinterrupt-Betriebsarten (oder Wait/Ready-Funktion), Senderfreigabe, Selbstfreigabe (Auto Enables) und Extern/Status-Interrupt.
Die Auswahl der SDLC-Betriebsart und des SDLC-Polynoms veranlaßt den SIO, den CRC-Generator auf lauter Einsen zu initialisieren. Dies geschieht durch Ausgabe des Befehls "Reset-Sende-CRC-Generator" (WR0). Ausführliche Informationen über die Interruptbetriebsarten sind im Abschnitt "Synchrone Betriebsart" zu finden.
Nach einem Reset oder wenn der Sender nicht freigegeben ist, befindet sich der Senderausgang auf "High" (Marking). Ein Break kann programmiert werden, um am Ausgang ein ständiges "Low" (Spacing) zu erzeugen. Bei vollständig initialisiertem und freigegebenem Sender erscheinen am Senderdatenausgang ständig Flags.

Eine Abbruch-(Abort)Folge kann gesendet werden, indem der Send-Abort-Befehl (WRO,CMD_1) ausgegeben wird. Dies bewirkt, daß mindestens acht, aber weniger als vierzehn Einsen gesendet werden, ehe wieder ständig Flags erscheinen. Es ist möglich, daß die Abortfolge (bestehend aus acht Einsen) auf bis zu 5 aufeinanderfolgende Einsen (gestattet durch die Nulleinfügungslogik) folgen kann und folglich bis zu 13 Einsen gesendet werden. Alle Daten, die gesendet werden und alle Daten im Sendepuffer gehen verloren, wenn ein Abort ausgegeben wird.
Eine zusätzliche Null wird automatisch eingefügt, wenn im Datenstrom fünf aufeinanderfolgende Einsen auftreten. Dies gilt nicht für Aborts oder Flags.

Datenübertragung und Statusüberwachung

Es gibt mehrere Kombinationen von Interrupts und der Wait/Ready-Funktion in der SDLC-Betriebsart.

Datenübertragung mittels Interrupts

Wenn das Sende-Interrupt-Freigabe-Bit gesetzt ist, wird jedesmal ein Interrupt erzeugt, wenn der Puffer leer wird. Dem Interrupt kann Genüge getan werden, entweder durch das Einschreiben eines anderen Zeichens in den Sender oder durch Rücksetzen des Transmit Interrupt Pending-Speichers mit einem Reset Transmitter-Pending-Befehl (WRO,CMD_5). Wurde der Interrupt mit diesem Befehl beantwortet und nichts mehr in den Sender eingeschrieben, dann treten keine weiteren Senderinterrupts auf. Das Ergebnis ist eine Sender-leer-Bedingung (Transmit Underrun). Wenn ein anderes Zeichen eingeschrieben und ausgesendet wird, dann kann der Sender erneut leer werden und die CPU unterbrechen. Im Anschluß an die Flags in einer SDLC-Operation können unter Benutzung der Sende-Interrupt-Betriebsart das 8-Bit-Adressenfeld, das Steuerfeld und das Informationsfeld zum SIO gesendet werden. Der SIO sendet die Rahmenkontrollfolge (Frame check), indem die Transmit Underrun-Bedingung verwendet wird. Wenn der Sender das erste Mal freigegeben wird, ist er bereits leer und kann dann nicht mehr leer werden. Daher können keine Transmit-Buffer-Empty-Interrupts auftreten, bevor das erste Datenzeichen geschrieben ist.

Datenübertragung unter Benutzung von Wait/Ready

Wenn die Wait/Ready-Funktion ausgewählt worden ist, zeigt Wait der CPU an, daß der SIO nicht zu einer Datenübertragung von oder zur CPU bereit ist und die CPU den I/O-Zyklus verlängern muß. Einem DMA-Steuergerät zeigt Ready an, daß der Senderpuffer leer ist, und daß der SIO bereit ist, das nächste Zeichen aufzunehmen. Wenn das Datenzeichen nicht rechtzeitig in den SIO geladen wird, dann wird das Sendeschieberegister leer und der SIO tritt in die Transmit-Underrun-Bedingung ein. Adreß-, Steuer- und Informationsfeld können in dieser Betriebsart zur SIO übertragen werden, indem die Wait/Ready-Funktion benutzt wird. Der SIO sendet die Rahmen-Kontroll-Folge mittels der Transmit-Underrun-Bedingung.

SDLC Transmit Underrun/Ende der Nachricht

SDLC-ähnliche Formate haben keine Einrichtungen für das Auffüllen von Zeichen in einer Nachricht. Der SIO schließt daher einen SDLC-Rahmen automatisch ab, wenn der Sendedatenpuffer und das Ausgangsschieberegister keine Bits mehr zum Senden haben. Zuerst werden die zwei CRC-Bytes ausgesendet, gefolgt von einem oder mehreren Flags. Dieses Verfahren erlaubt eine sehr schnelle Übertragung unter DMA- oder CPU-Block-I/O-Steuerung,wobei es nicht notwendig ist, daß die CPU schnell auf das Ende der Datenübertragung reagiert.
Die Arbeitsweise des SIO in der Underrun-Situation hängt von dem Zustand des Transmit-Underrun/EOM-Befehls ab. Nach einem Reset befindet sich das Transmit-Underrun/EOM-Statusbit im gesetzten Zustand und verhindert das Einfügen von CRC-Zeichen in der Zeit, in der keine Daten zu senden sind. Folglich werden Flagzeichen gesendet. Der SIO beginnt, den Rahmen zu senden, sobald die Daten in den Senderpuffer eingeschrieben worden sind. In der Zeit zwischen dem Einschreiben des ersten Datenbytes und dem Ende der Nachricht muß der Reset-Transmit-Underrun/EOM-Befehl ausgegeben werden.

Folglich ist das Transmit-Underrun/EOM-Statusbit am Ende der Datenübertragung (wenn Underrun auftritt) im rückgesetzten Zustand, wo die CRC-Zeichen automatisch gesendet werden. Das Senden von CRC setzt wieder das Transmit Underrun/EOM-Status-Bit. Obgleich es keine Einschränkung gibt, wann das Transmit-Underrun/EOM-Bit während einer Datenübertragung zurückgesetzt werden kann, geschieht es gewöhnlich, nachdem das erste Datenzeichen (Sekundäradresse) zum SIO gesendet wird. Das Zurücksetzen dieses Bits erlaubt das Senden von CRC und Flags, wenn keine Daten vorhanden sind. Dies gibt der CPU zusätzliche Zeit für das Erkennen des Fehlers und der Reaktion mit einem Abort-Befehl. Bei einem frühzeitigen Rücksetzen in der Datenübertragung hat die CPU maximal Zeit, auf eine unbeabsichtigte Transmit-Underrun-Situation zu reagieren.

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
WR3	00=Rx5 Bits/Zeichen 10=Rx6 Bits/Zeichen 01=Rx7 Bits/Zeichen 11=Rx8 Bits/Zeichen		Auto Enables	Beginne Such-betrieb	RxCRC-Freigabe	Adreß-Suchbe-triebsart	0	Empfänger-Freigabe
WR4	0	0	1 (wählt SDLC-Betriebsart aus)	0	0	0	0	0
WR5	DTR	00=Tx5 Bits(oder weniger)/Zeich. 10=Tx6 Bits/Zeich. 01=Tx7 Bits/Zeich. 11=Tx8 Bits/Zeich.		0	Sender-Freigabe	0 (SDLC-CRC)	RTS	Sender-CRC-Freigabe

Tabelle 7: Inhalt der Schreibregister 3, 4 und 5 in den SDLC-Betriebsarten

Ist der Extern/Status-Interrupt freigegeben, wird während des Sendens von CRC das Transmit Underrun/EOM-Bit gesetzt und das Transmit-Buffer-Empty-Bit wird rückgesetzt, um anzuzeigen, daß das Senderegister mit CRC-Daten gefüllt ist. Wurde CRC vollständig gesendet, dann wird das Transmit-Buffer-Empty-Status-Bit gesetzt und ein Interrupt erzeugt, um der CPU anzuzeigen, daß eine andere Datenübertragung beginnen kann. Dieser Interrupt erfolgt, wenn CRC gesendet und das Flag geladen wurde. Wenn keine Daten mehr zu senden sind, kann das Programm die Übertragung durch Rücksetzen von RTS beenden und den Sender sperren.
In der SDLC-Betriebsart ist es üblich, das Transmit-Underrun/EOM-Statusbit sofort zurückzusetzen, nachdem das erste Zeichen zum SIO gesendet worden ist. Wenn ein Transmit-Underrun erkannt wird, garantiert dies, daß die folgende Übertragungszeit mit CRC-Zeichen gefüllt wird, um der CPU genügend Zeit zu geben, Abort-Befehl zu senden, falls dies notwendig ist. Dies verhindert, daß die Flags vorzeitig ausgegeben werden und schaltet die Möglichkeit aus, daß der Empfänger den Rahmen als gültige Daten akzeptiert. Die Situation kann auftreten, weil es möglich ist, daß am Empfangsende das Datenmuster, welches der automatischen Flageinfügung unmittelbar vorausgeht, auf die CRC-Kontrolle paßt und ein falsches CRC-Kontrollergebnis ergibt. Der Extern/Status-Interrupt wird immer dann hervorgerufen, wenn das Transmit-Underrun/EOM-Bit gesetzt wird, weil eine Transmit-Underrun-Bedingung existiert.
Die Transmit-Underrun-Logik liefert einen zusätzlichen Schutz gegen die vorzeitige Flag-Einfügung, wenn durch die CPU-Interrupt-Service-Routine eine entsprechende Antwort zur SIO gegeben wird. Das folgende Beispiel erläutert dieses Problem:

Der SIO erzeugt einen Interrupt mit dem Setzen des Transmit-Buffer-Empty-Statusbits

Die CPU antwortet nicht rechtzeitig und ruft eine Transmit-Underrun-Bedingung hervor.

Der SIO beginnt CRC-Zeichen (zwei Bytes) zu setzen.

Die CPU befriedigt schließlich den Transmit-Buffer-Empty-Interrupt mit einem Datenzeichen, welches den gesendeten CRC-Zeichen folgt.

Der SIO erzeugt einen Extern/Status-Interrupt mit dem Setzen des Transmit Underrun/EOM-Statusbits.

Die CPU erkennt den Transmit Underrun/EOM-Status und entscheidet aufgrund ihres internen Programmstatus, daß der Interrupt nicht für das "Ende der Datenübertragung" erzeugt wurde.

Die CPU gibt sofort den "Sende-Abort-Befehl"(WRO) zur SIO

Der SIO sendet die Abort-Folge, dadurch gehen alle Daten (CRC, Daten oder Flags), die gesendet werden, verloren.

Dieser Ablauf zeigt, daß die CPU einen Schutz von minimal 32 und maximal 30 Sendertaktzyklen besitzt.

SDLC-CRC-Erzeugung

Der CRC-Generator muß am Beginn jedes Rahmens, bevor die CRC-Berechnung beginnen kann, auf alles Einsen zurückgesetzt werden. Die CRC-Berechnung beginnt, wenn das Programm das Adressenfeld (acht Bits) zur SIO sendet. Obwohl der SIO automatisch nach der Senderfreigabe ein Flagzeichen sendet, ist es günstig zu Beginn der Datenübertragung ein paar Flags mehr auszugeben, um die Zeichensynchronisation am Empfangsende zu sichern. Dies kann durch eine externe Zeitsteuerung nach der Freigabe des Senders und vor dem Laden des ersten Zeichens erfolgen.
Die Sender-CRC-Freigabe (WR5,D_0) sollte vor dem Aussenden des Adreßfeldes erfolgen. In der SDLC-Betriebsart werden alle Zeichen zwischen dem Anfangs- und Endflag in die CRC-Berechnung einbezogen und das erzeugte CRC wird invertiert ausgesendet.

Abschluß des Sendevorgangs

Erfolgt die Sperrung des Senders während der Ausgabe eines Zeichens, dann wird dieses Zeichen in der üblichen Form gesendet, aber es folgt ein "High" (marking line) und nicht CRC oder Flag-Zeichen.
Im Puffer befindliche Zeichen bleiben dort, wenn der Sender gesperrt wird, jedoch wird eine programmierte Abort-Folge sofort wirksam, sobald sie in das Steuerregister eingeschrieben wird.
Zeichen, die gerade gesendet werden, gehen verloren. Im Fall des CRC wird die 16-Bit-Übertragung abgeschlossen, wenn der Sender gesperrt ist, aber anstelle von CRC werden Flags gesendet.
In allen Betriebsarten werden die Zeichen mit den niedrigststelligen Bits zuerst gesendet. Das erfordert ein rechtsbündiges Ausrichten der zu übertragenden Daten, wenn die Wortlänge weniger als 8 Bits beträgt. Bei einer Wortlänge von fünf oder weniger Bits muß ein Sonderverfahren angewendet werden (siehe Abschnitt Programmierung).
Da die Anzahl der Bits/Zeichen fliegend geändert werden kann, können die Daten mit jeder Anzahl von Bits aufgefüllt werden. Der SIO kann in Verbindung mit dem Empfänger-Residuen-Kode eine Nachricht empfangen, die ein variables I-Feld hat und sie exakt zurücksenden ohne vorherige Information über die Zeichenstruktur des I-Feldes. Eine Veränderung in der Anzahl der Bits hat keinen Einfluß auf das Zeichen, das gerade ausgegeben wird. Die Zeichen werden mit der Anzahl der Bit gesendet, die in der Zeit programmiert werden, in der das Zeichen vom Senderpuffer in den Sender geladen wird.
Wenn der Extern/Status-Interrupt freigegeben ist, dann rufen solche Senderbedingungen wie "Beginn CRC-Zeichen zu senden", "Beginne Flagzeichen zu senden" und $\overline{CTS}$-Zustandsänderung einen einheitlichen Vektor hervor, wenn der statusabhängige Vektor gesetzt ist.
Alle Interrupts können gesperrt werden für Operationen im Pollingbetrieb.

Tabelle 8 zeigt die typischen Programmschritte für eine Halb-Duplex-SDLC-Sendebetriebsart.

Tabelle 8: SDLC-Sendebetriebsart

Funktion	Typische Programmschritte	Bemerkungen
	Register: geladene Information:	
	WR0 Kanal Reset	Reset SIO
	WR0 Zeiger 2	
	WR2 **Interruptvektor**	nur Kanal B
	WR0 Zeiger 3	
	WR3 Auto Enables	Sender gibt nur Daten aus, nachdem $\overline{CTS}$ erkannt wurde.
	WR0 Zeiger 4, Reset Extern/Status-Interrupts	
	WR4 Paritätsinformation, SDLC-Betriebsart, x1 Takt-Betriebsart,	
	WR0 Zeiger 1, Reset Extern/Status-Interrupts	
	WR1 Extern-Interrupt-Freigabe, statusabh. Vektor, Sender-Interrupt-Freigabe oder Wait/Ready-Betriebsart-Freigabe	Die Extern-Interrupt-Interrupt-Betriebsart überwacht den Status der $\overline{CTS}$ und $\overline{DCD}$-Eingänge sowie den Status des Tx-Underrun/EOM-Speichers. Sendeinterrupts unterbrechen, wenn der Sendepuffer leer wird; die Wait/Ready-Betriebsart kann zur Datenübertragung auf DMA- oder Blocktransferbasis genutzt werden. Der erste Interrupt erscheint, wenn $\overline{CTS}$ aktiv wird, zu diesem Zeitpunkt werden die Flags durch die SIO gesendet. Das erste Datenbyte (Adreßfeld) kann nach diesem Interrupt in den SIO geladen werden. Flags können nicht als Daten zum SIO gesendet werden. Der statusabhängige Vektor wird nur in Kanal B genutzt.
Initialisierung		
	WR0 Zeiger 5	
	WR5 Sender-CRC-Freigabe, Sendeanforderung, SDLC-CRC, Senderfreigabe, Senderwortlänge, Terminalbereitschaft	Die SDLC-CRC-Betriebsart muß definiert werden, bevor der Sender CRC-Generator initialisiert wird.
	WR0 Reset Sender-CRC-Generator	CRC-Generator wird auf alles "1" initialisiert.
Wartezustand	Ausführung des Haltebefehls oder anderer Programmschritte	Es wird auf einen Interrupt oder den Wait/Ready-Ausgang **gewartet, um Daten zu übertragen.**

	Wenn ein Interrupt (Wait/Ready) auftritt, führt die CPU folgendes aus: - Verändert die Senderwortlänge (wenn notwendig) - Überträgt ein Datenbyte von der CPU (Speicher) zur SIO - Reset Tx Underrun/EOM-Latch (WR0)	Die Flags werden durch die SIO gesendet, sobald die Senderfreigabe erfolgt und $\overline{CTS}$ aktiv wird. Die $\overline{CTS}$-Statusänderung ist der erste Interrupt, der auftritt und ihm folgt ein Transmit-Buffer-Empty-Interrupt für nachfolgende Datenübertragungen.
	Wenn das letzte Zeichen des I-Feldes gesendet wurde, führt die SIO folgendes aus: - Sendet CRC - Sendet das Endeflag - Unterbricht die CPU mit einem Buffer-Empty-Status	Die Wortlänge kann laufend für eine variable I-Feldlänge verändert werden. Das Datenbyte kann Adreß-, Steuer- oder I-Feld-Information (niemals ein Flag) enthalten. In der Praxis ist es üblich, den Tx-Underrun/EOM-Speicherplatz am Anfang der Nachricht zurückzustellen, um eine falsche Rahmenende-Feststellung am Empfangsende zu vermeiden. Wenn ein Underrun auftritt, ist damit gesichert, daß CRC gesendet wird und ein Underrun-Interrupt (Tx Underrun/EOM-Speicher wird aktiv) auftritt. Es ist zu beachten, daß der Befehl "Send Abort" als Antwort auf jeden Interrupt zur SIO ausgegeben werden kann, um die Übertragung abzubrechen.
	Die CPU führt folgendes aus: - Aussenden des Reset-Tx-Interrupt-Pending-Befehls zur SIO - Aktualisierung des NS-Zählers - Wiederholung des Prozesses für die nächste Übertragung usw.	
	Wenn der Vektor einen Fehler anzeigt, führt die CPU folgendes aus: - Sendet Abort-Folge - Ausführung einer Fehlerroutine - Aktualisierung der Parameter, Betriebsarten usw. - Rückkehr vom Interrupt	
Ende der Datenübertragung	Neubestimmung der Interruptbetriebsarten Aktualisierung der Modem-Steuersignale Sperrung der Sendebetriebsart	

4.4.2. SDLC-Empfang

Initialisierung

Die SDLC-Empfangs-Betriebsart wird durch das System mit den folgenden Parametern initialisiert: SDLC-Betriebsart, x1 Takt-Betriebsart, SDLC-Polynom, Empfänger-Wortlänge, usw. Das Flag muß ebenfalls in WR7 und das sekundäre Adreßfeld in WR6 geladen werden. Der Empfänger wird nur freigegeben, nachdem alle Empfangsparameter gesendet worden sind. Nachdem das geschehen ist, befindet sich der Empfänger in der Suchphase und bleibt in dieser Phase bis das erste Flag empfangen wird. Während des SDLC-Betriebs kommt der Empfänger niemals erneut in die Such-Phase (Hunt-Phase), sofern das nicht durch das Programm angewiesen wurde. Die WR4-Parameter müssen vor den WR1, WR3, WR5, WR6 und WR7-Parametern ausgegeben werden. Unter Programmsteuerung kann der Empfänger in die Adreß-Such-Betriebsart gelangen. Wenn das Adreß-Such-Bit (WR3,D_2) gesetzt ist, wird ein Zeichen, welches dem Flag folgt (erstes Zeichen, welches kein Flag ist), mit der programmierten Adresse in WR6 und der globalen Adresse (1111 1111) verglichen.

Wenn das SDLC-Rahmen-Adressenfeld zu irgendeiner der Adressen paßt, beginnt die Datenübertragung.
Da der SIO nur ein Adressenzeichen auf Übereinstimmung prüfen kann, muß die erweiterte Adressenfelderkennung von der CPU vorgenommen werden. In diesem Fall überträgt der SIO einfach die zusätzlichen Adressenbytes zur CPU, als wären sie Datenzeichen. Wenn die CPU feststellt, daß der Rahmen nicht das richtige Adressenfeld hat, kann er das Such-Bit (Hunt) setzen. Der SIO beendet den Empfang und sucht nach einer neuen Nachricht, die von einem Flag angeführt wird. Da das Steuerfeld des Rahmens gegenüber dem SIO transparent ist, wird es an die CPU als Datenzeichen übertragen.

Datenübertragung und Statusüberwachung

Nach dem Erhalt eines gültigen Flags werden die empfangenen Zeichen zum Empfangsdaten-FIFO übertragen. Die folgenden vier Interrupt-Betriebsarten stehen für die Übertragung dieser Daten und ihres entsprechenden Statusses zur Verfügung.

Keine Interrupts freigegeben

Diese Betriebsart wird für reine Polling-Operationen oder für "off-line" -Bedingungen verwendet.

Interrupt nur beim ersten Zeichen

Diese Betriebsart wird normalerweise zum Start einer Software-Polling-Schleife benutzt oder zu Beginn eines Blocktransfers, der $\overline{\text{Wait/Ready}}$ nutzt, um die CPU oder DMA auf die ankommende Datenrate zu synchronisieren. In dieser Betriebsart unterbricht der SIO nur beim ersten Zeichen und später entstehen nur noch Interrupts, wenn eine spezielle Empfangsbedingung erkannt wurde. Die Betriebsart wird erneut initialisiert mit dem Enable-Interrupt-On-Next-Receive-Character-Befehl.

Das erste empfangene Zeichen, welches nach diesem Befehl ausgesandt wurde, ruft einen Interrupt hervor. Wenn der Extern/Status-Interrupt freigegeben ist, kann dieser jederzeit unterbrechen, wenn der $\overline{\text{DCD}}$-Eingang den Zustand ändert. Spezielle Empfangsbedingungen wie "das Ende des Rahmens" und "Empfängerüberlauf" rufen auch Interrupts hervor. Der "Rahmen-Ende"-Interrupt kann benutzt werden, um die Block Transfer-Betriebsart zu beenden.

Interrupt bei jedem Zeichen

Es wird ein Interrupt erzeugt, immer wenn der Empfangs-FIFO ein Datenwort enthält. Fehler und spezielle Empfangsbedingungen erzeugen einen speziellen Vektor, wenn der statusabhängige Vektor ausgewählt wurde.

Interrupts bei spezieller Empfangsbedingung

Der Interrupt bei spezieller Empfangsbedingung als solcher,ist keine gesonderte Interrupt-Betriebsart. Ehe die spezielle Empfangsbedingung einen Interrupt hervorrufen kann, muß entweder ein Interrupt beim ersten Empfangszeichen oder ein Interrupt bei jedem Empfangszeichen ausgewählt worden sein. Der Interrupt bei spezieller Empfangsbedingung wird durch einen Empfängerüberlauf oder durch das Erkennen des Rahmenendes hervorgerufen. Da das Empfängerüberlauf-Statusbit gespeichert wird, spiegelt der gelesene Fehlerstatus einen Fehler im laufenden Wort im Empfangspuffer wieder, zusätzlich zu den Fehlern, die seit dem letzten Error-Reset-Befehl empfangen wurden. Das Empfänger-Überlauf-Status-Bit kann nur durch den Error-Reset-Befehl zurückgesetzt werden. Das Rahmenende-Status-Bit zeigt an, daß ein gültiges Endflag empfangen wurde und daß der CRC-Fehler und der Residuen-Kode auch gültig sind.

Die Zeichenlänge kann laufend verändert werden. Wenn die Adreß- und Steuerbytes als 8-Bit-Zeichen verarbeitet werden, kann der Empfänger während der Zeit, in der das erste Informations-Zeichen empfangen wird, auf eine kürzere Zeichenlänge umschalten. Diese Änderung muß so schnell erfolgen, daß sie wirksam wird, ehe die Anzahl der Bits, die für die Zeichenlänge festgelegt wurde, empfangen worden ist. Wenn z. B. die Änderung vom 8-Bit-Steuerfeld zu einem 7-Bit-Informationsfeld erfolgt, muß die Änderung vorgenommen werden,

ehe die ersten sieben Bits des I-Feldes empfangen wurden.

SDLC-Empfangs-CRC-Prüfung

Die Steuerung des Empfangs-CRC-Prüfers erfolgt automatisch. Er wird durch das führende Flag zurückgesetzt und CRC wird bis zum Endflag berechnet. Das Byte, bei dem das Rahmenende-Bit gesetzt ist, ist das Byte, welches das Ergebnis der CRC-Kontrolle enthält. Wenn das CRC/Rahmen-Fehler-Bit nicht gesetzt ist, liegt eine gültige Nachricht vor. Eine spezielle Prüffolge wird für die SDLC-Prüfung benutzt, da das gesendete CRC-Ergebnis invertiert ist. Die Endprüfung muß 0001110100001111 ergeben. Die 2-Byte CRC-Prüfzeichen müssen durch die CPU gelesen und abgelegt werden, da der SIO, während er sie für die CRC-Prüfung benutzt, sie als gewöhnliche Daten behandelt.

Abschluß des SDLC-Empfanges

Es wird ein spezieller Vektor erzeugt, wenn das Endflag empfangen wurde. Dies signalisiert, daß das Byte mit dem gesetzten Rahmenende-Bit empfangen worden ist. Zusätzlich zu den Ergebnissen der CRC-Prüfung enthält RR1 drei Bits des Residuen-Kodes, die zu diesem Zeitpunkt gültig sind. Für solche Fälle, in denen die Anzahl der I-Feld-Bits kein ganzzahliges Vielfaches der Zeichenlänge ist, zeigen diese Bits die Grenze zwischen den CRC-Prüf-Bits und den I-Feld-Bits an. Eine detaillierte Beschreibung dieser Bits findet man im Abschnitt "SIO-Programmierung".

Jeder Rahmen kann vorzeitig durch eine Abortfolge abgebrochen werden. Aborts werden festgestellt, wenn sieben oder mehr Einsen auftreten und einen Extern/Status-Interrupt verursachen (sofern gewählt), wobei das Break/Abort-Bit in RR0 gesetzt wird. Nachdem der Reset-Befehl für den Extern/Status-Interrupt ausgegeben worden ist, tritt ein zweiter Interrupt auf, wenn das Zustandekommen der aufeinanderfolgenden Einsen gekl[illegible] worden ist. Dieser kann verwendet werden, um zwischen den Abort- und Idle-line-Bedingungen zu unterscheiden. Im Gegensatz zur synchronen Betriebsart hat die CRC-Berechnung in SDLC keine 8-Bit-Verzögerung, da alle Zeichen in der CRC-Berechnung enthalten sind. Wenn das zweite CRC-Zeichen in den Empfangspuffer geladen worden ist, ist die CRC-Berechnung a[illegible]geschlossen.

Tabelle 9: SDLC-Empfangsbetriebsart

Funktion	Typische Programmschritte	Bemerkungen
	Register: geladene Information:	
Initialisierung	WR0 Kanal Reset	Reset SIO
	WR0 Zeiger 2	
	WR2 Interruptvektor	nur Kanal B
	WR0 Zeiger 4	
	WR4 Paritätsinformation, Sync-Betriebsart, SDLC-Betriebsart, x1 Taktrate,	
	WR0 Zeiger 5, Reset Extern/Status-Interrupts	
	WR5 SDLC-CRC, Terminalbereitschaft	
	WR0 Zeiger 3	
	WR3 Empfänger-CRC-Freigabe, Beginn Suchphase, Auto Enables, Empfänger-Zeichenlänge, Adreßsuch-Betriebsart	Auto Enables gibt den Empfänger frei, um Daten zu empfangen nur nachdem $\overline{DCD}$ aktiv wird. Die Adreßsuch-Betriebsart gibt den SIO frei, um die Datenadresse mit der programmierten Adres-

		se oder der globalen Adresse zu vergleichen.
	WR0 Zeiger 6	
	WR6 Sekundäres Adressenfeld	Diese Adresse wird mit der Datenadresse in einer SDLC-Abfrageoperation verglichen.
	WR0 Zeiger 7	
	WR7 SDLC-Flag 01111110	Dieses Flag kennzeichnet den Start und das Ende des Rahmens in einer SDLC-Operation.
	WR0 Zeiger 1, Reset Extern/Status-Interrupts WR1 Statusabhängiger Vektor, Extern-Interrupt-Freigabe, Empfangsinterrupt nur beim ersten Zeichen	In dieser Interrupt-Betriebsart wird nur das Adressenfeld (nur 1 Zeichen) zur CPU übertragen. Alle nachfolgenden Felder (Steuerinformation, usw.) werden auf DMA-Basis übertragen. Der statusabhängige Vektor befindet sich nur in Kanal B.
	WR0 Zeiger 3, Freigabe des Interrupts beim nächsten Empfangszeichen	Dieser Befehl liefert eine einfache Rückschleife zum Anfang für die nächste Übertragung.
	WR3 Empfänger-Freigabe, Empfänger-CRC-Freigabe, Beginne Suchphase, Auto Enables, Empfänger-Zeichenlänge, Adreß-Such-Betriebsart	Es erfolgt erneut die Ausgabe von WR3, um den Empfänger freizugeben.
Wartezustand	Ausführung des Haltbefehls oder anderer Programmteile	Die SDLC-Empfangsbetriebsart ist voll initialisiert und der SIO wartet auf das Anfangsflag, gefolgt von einem Vergleichsadressenfeld, um die CPU zu unterbrechen.
	Wenn ein Interrupt beim ersten Zeichen auftritt, führt die CPU folgendes aus: - Übertragung des Datenbytes (Adreßbyte) zur CPU - Erkennen und Setzen des zugeordneten Flags für das erweiterte Adressenfeld - Aktualisierung der Zeiger und Parameter - Freigabe des DMA-Steuergerätes - Freigabe der Wait/Ready-Funktion im SIO - Rückkehr vom Interrupt	Während der Suchphase unterbricht der SIO die CPU, wenn die programmierte Adresse mit der Datenadresse übereinstimmt. Die CPU legt die DMA-Betriebsart fest und alle nachfolgenden Datenzeichen werden durch die DMA zum Speicher übertragen.
Datenübertragung und Statusüberwachung	Wenn der Ready-Ausgang aktiv wird, führt die DMA folgendes aus: - Übertragung des Datenbytes zum Speicher - Aktualisierung der Zeiger	Während der DMA-Operation überwacht der SIO den $\overline{DCD}$-Eingang und die Abortfolge im Datenstrom, um die CPU mit einem externen Statusfehler zu unterbrechen. Der Interrupt bei spezieller Empfangsbedingung wird durch einen Empfängerüberlauffehler hervorgerufen.

	Bei Rahmenende erscheint ein Interrupt, wobei die CPU folgendes ausführt: - Beendet DMA-Betriebsart (Sperrung Wait/Ready) - Übertragung von RR1 zur CPU - Überprüfung des CRC-Fehlerbit-Statusses und des Residuen-Kodes	Das Erkennen des Rahmenendes (Flag) ruft einen Interrupt hervor und sperrt die Wait/Ready-Funktion. Die Residuen-Kodes kennzeichnen die Bitstruktur der letzten zwei Bytes der Daten, die zum Speicher unter DMA-Kontrolle übertragen wurden. Der "Error Reset"-Befehl wird ausgegeben, um die spezielle Bedingung zu löschen.
	Wenn die Abort-Folge erkannt wurde, wird ein Interrupt hervorgerufen, wobei die CPU folgendes tut: - Übertragung von RR0 zur CPU - Beendigung der DMA-Betriebsart - Ausgabe des "Reset Extern Status-Interrupt"-Befehls zur SIO	Eine Abort-Folge wird erkannt, wenn sieben oder mehr Einsen im Datenstrom gefunden werden.
	- Eintritt in den Wartezustand	CPU wartet auf eine Abortfolge, um die Übertragung zu beenden. Die Beendigung löscht das Break/Abort-Statusbit und ruft einen Interrupt hervor.
	Wenn ein Interrupt aufgrund einer zweiten Abortfolge auftritt, führt die CPU folgendes aus: - Aussenden des "Reset-Extern-Status-Interrupt"-Befehls zur SIO	Zu diesem Zeitpunkt geht das Programm weiter bis zur Beendigung dieser Datenübertragung.
Abschluß der Datenübertragung	Neubestimmung der Interruptbetriebsarten, Sync- und SDLC-Betriebsarten. Sperrung der Empfangsbetriebsart	

5. Programmierung des Schaltkreises U 856 D

Zur Programmierung des SIO gibt das Systemprogramm zuerst eine Serie von Befehlen aus, welche die Hauptbetriebsart initialisieren und dann andere Befehle, welche die Bedingungen in der ausgewählten Betriebsart näher bestimmen. Zum Beispiel werden asynchrone Betriebsart, Zeichenlänge, Taktrate, Anzahl der Stoppbits, gerade oder ungerade Parität zuerst gesetzt, dann die Interrupt-Betriebsart und schließlich Empfänger- oder Sender-Freigabe. Die WR4-Parameter müssen vor allen anderen Parametern innerhalb der **Initialisierungsroutine** ausgegeben werden. Beide Kanäle enthalten Befehlsregister, welche über das Systemprogramm vor dem Betrieb programmiert werden. Erst nach Programmierung aller Befehlsregister (Schreibregister) ist der Inhalt der Statusregister (Leseregister) definiert. Durch den CPU-Adressenbus werden die entsprechenden Kanaleingänge (B/$\overline{A}$) bzw. Steuer/Daten-Eingänge (C/$\overline{D}$) vorgewählt.

C/$\overline{D}$	B/$\overline{A}$	Funktion
0	0	Kanal A Daten
0	1	Kanal B Daten
1	0	Kanal A Befehle/Status
1	1	Kanal B Befehle/Status

5.1. Schreibregister

Der SIO enthält acht Register (WRO-WR7) in jedem Kanal, die getrennt durch das Systemprogramm programmiert werden können, um die funktionelle Besonderheit des Kanals herzustellen. Mit Ausnahme von WRO erfordert die Programmierung der Schreibregister zwei Bytes. Das erste Byte enthält drei Bits (DO-D2), die als Zeiger für das auszuwählende Register dienen; das zweite Byte ist das aktuelle Steuerwort, das in das Register eingeschrieben wird, um den SIO zu programmieren.

Es ist zu beachten, daß der Programmierer völlige Freiheit hat, nach der Auswahl des Registers, dieses zu lesen (z. B. Test eines Leseregisters) oder zu schreiben (Laden eines Schreibregisters für die Initialisierung). Bei der Software-Erstellung zur Initialisierung des SIO in einer modularen oder strukturierten Form, kann der Programmierer die leistungsfähigen Block-I/O-Befehle verwenden.

WRO ist ein Spezialfall, bei dem alle Basis-Befehle (CMD_0- CMD_2) mit einem einfachen Byte ausgegeben werden können. Ein Reset (intern oder extern) initialisiert die Zeigerbits DO-D2 auf WRO zu weisen.

Die Grundbefehle (CMD_0-CMD_2) und die CRC-Steuerbits (CRC_0, CRC_1) sind im ersten Byte jedes Schreibregisterzugriffs enthalten. Dies erlaubt eine maximale Flexibilität und Systemsteuerung. Jeder Kanal enthält die folgenden Steuerregister. Diese Register werden als Befehle (keine Daten) adressiert.

Schreibregister 0

WRO ist ein Befehlsregister; es wird jedoch auch für den CRC-Reset-Kode und als Zeiger für die anderen Register benutzt. Wenn dieses Schreibregister erstmals programmiert wird, ist stets der Befehl CMD 1 zu programmieren (D5=0, D4=0, D3=1).

D7	D6	D5	D4	D3	D2	D1	DO
CRC Reset Code 1	CRC Reset Code 0	CMD 2	CMD 1	CMD 0	PTR 2	PTR 1	PTR 0

Zeiger Bits (D_0-D_2)

Die Bits D_0-D_2 sind Zeigerbits, die festlegen, in welches andere Schreibregister das nächste Byte eingeschrieben oder aus welchem Leseregister das nächste Byte ausgelesen wird. Das erste Byte, das in jedem Kanal nach einem Reset (entweder durch einen Reset-Befehl oder durch externe Reset-Eingabe) eingeschrieben wird, geht in WRO. Nach einem Schreiben oder Lesen zu irgendeinem Register (außer WRO), weist der Zeiger auf WRO.

Befehlsbits (D_3-D_5)

Die drei Bits D_3-D_5 werden genutzt, um die sieben Grundbefehle auszugeben.

Befehl	CMD_2	CMD_1	CMD_0	
0	0	0	0	Null-Befehl (keine Wirkung)
1	0	0	1	Sende Abortfolge (SDLC-Betriebsart)
2	0	1	0	Reset Extern/Status-Interrupt
3	0	1	1	Kanal-Reset
4	1	0	0	Reset Interrupt beim nächsten Empfangszeichen
5	1	0	1	Reset Sender-Interrupt-Pending
6	1	1	0	Error-Reset
7	1	1	1	RETI (nur Kanal A)

Befehl 0 (Null)
Der Null-Befehl hat keine Wirkung. Seine Verwendung besteht darin, den Zeiger für das folgende Byte zu setzen.

Befehl 1 (Sende Abortfolge)
Dieser Befehl wird nur im SDLC-Betrieb verwendet, um eine Folge von acht bis dreizehn Einsen zu erzeugen.

Befehl 2 (Reset Extern/Status-Interrupt)
Nach einem Extern/Status-Interrupt (z. B. eine Änderung auf einer Modemleitung oder eine Break-Bedingung) werden die Statusbits von RR0 gespeichert. Dieser Befehl gibt sie erneut frei und ermöglicht, daß erneut Interrupts auftreten. Durch das Zwischenspeichern der Statusbits werden auch kurze Impulse eingefangen, so daß die CPU genügend Zeit hat, die Änderung zu lesen.

Befehl 3 (Kanal-Reset)
Dieser Befehl führt die gleiche Funktion aus wie ein externes Reset, aber nur in einem Kanal. Ein Reset auf Kanal A setzt außerdem die Interrupt-Prioritätslogik zurück. Alle Steuerregister des entsprechenden Kanals müssen neu initialisiert werden. Ein CPU-Zugriff zum rückgesetzten Kanal darf frühestens 4 Systemtaktzyklen nach einem Reset erfolgen. Dies kann normalerweise die Zeit sein, die eine CPU für einen OP-Kode-Fetch benötigt.

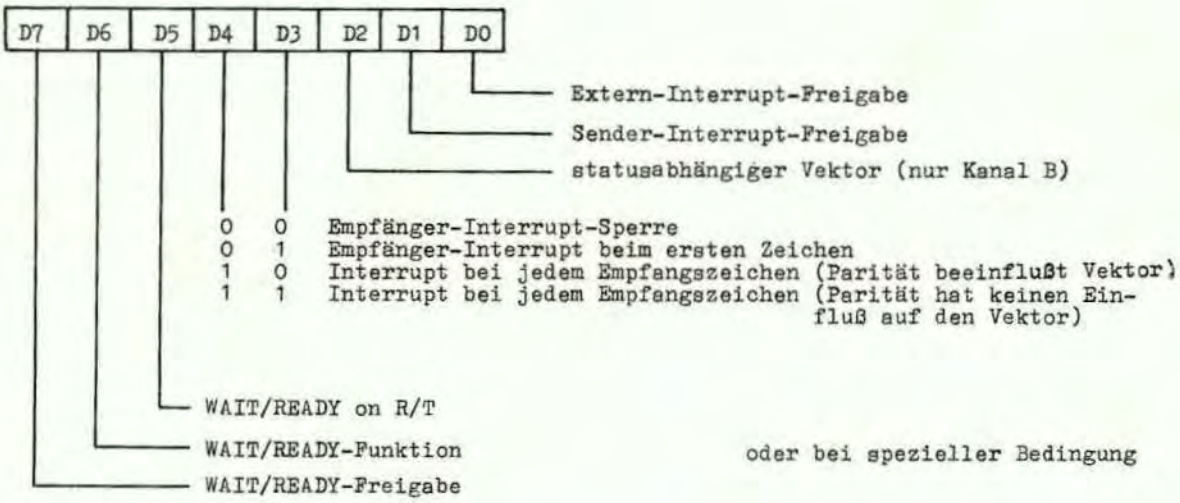
Befehl 4 (Interruptfreigabe beim nächsten Empfangszeichen)
Wenn die Betriebsart "Interrupt beim ersten empfangenen Zeichen" ausgewählt wurde, reaktiviert dieser Befehl diese Betriebsart, nachdem eine vollständige Nachricht empfangen wurde, um den SIO für die nächste Übertragung vorzubereiten.

Befehl 5 (Reset Sender-Interrupt-Pending)
Bei freigegebenem Senderinterrupt unterbricht dieser, wenn der Senderpuffer leer wird. In den Fällen, wo keine Zeichen mehr gesendet werden (z. B. Ende der Datenübertragung), verhindert die Ausgabe dieses Befehls weitere Senderinterrupts bis das nächste Zeichen in den Senderpuffer geladen oder CRC vollständig gesendet wurde.

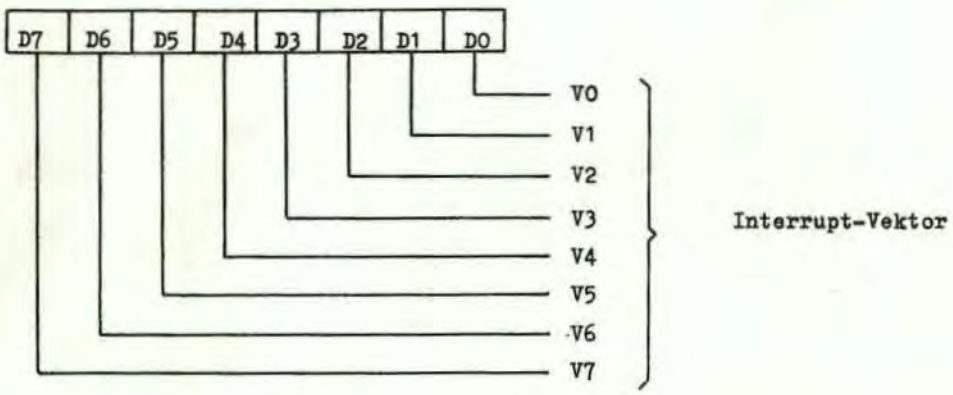
Bild 14: Schreibregister - Bitfunktionen
Schreibregister 0 (Write-Register 0)

D7	D6	D5	D4	D3	D2	D1	D0	
					0	0	0	Register 0
					0	0	1	Register 1
					0	1	0	Register 2
					0	1	1	Register 3
					1	0	0	Register 4
					1	0	1	Register 5
					1	1	0	Register 6
					1	1	1	Register 7
		0	0	0				Null-Kode
		0	0	1				Sende Abortfolge (SDLC)
		0	0	0				Reset Ext/Status-Interrupt
		0	1	1				Kanal-Reset
		1	0	0				Interruptfreigabe für nächstes Empfangszeichen
		1	0	1				Reset Sender-Int. Pending
		1	1	0				Error Reset
		1	1	1				RETI (nur Kanal A)
0	0							Null-Kode
0	1							Reset Empfänger CRC-Prüfer
1	0							Reset Sender - CRC-Generator
1	1							Reset Tx (Sender) Underrun/EOM Latch

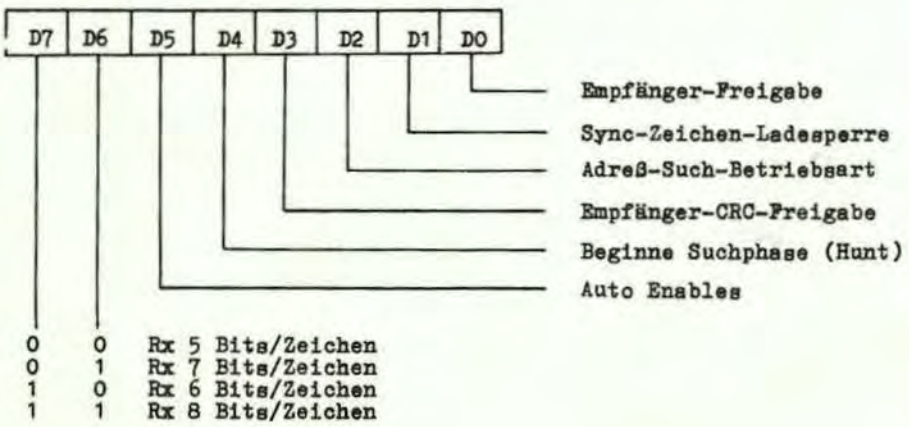
Schreibregister 1 (Write-Register 1)



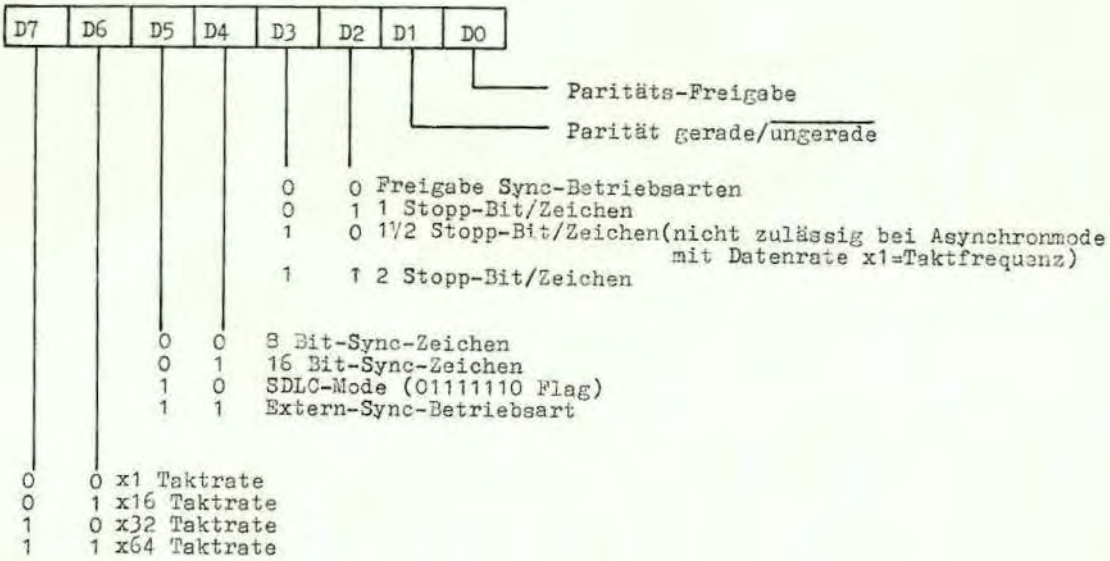
Schreibregister 2 (nur Kanal B) (Write-Register 2)



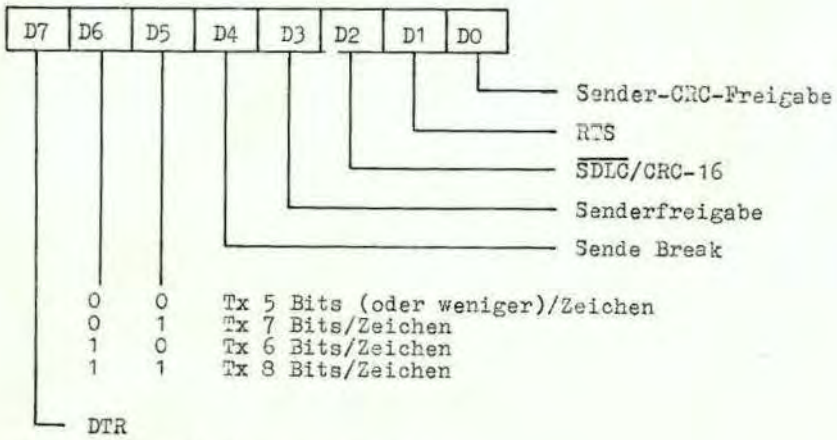
Schreibregister 3 (Write-Register 3)



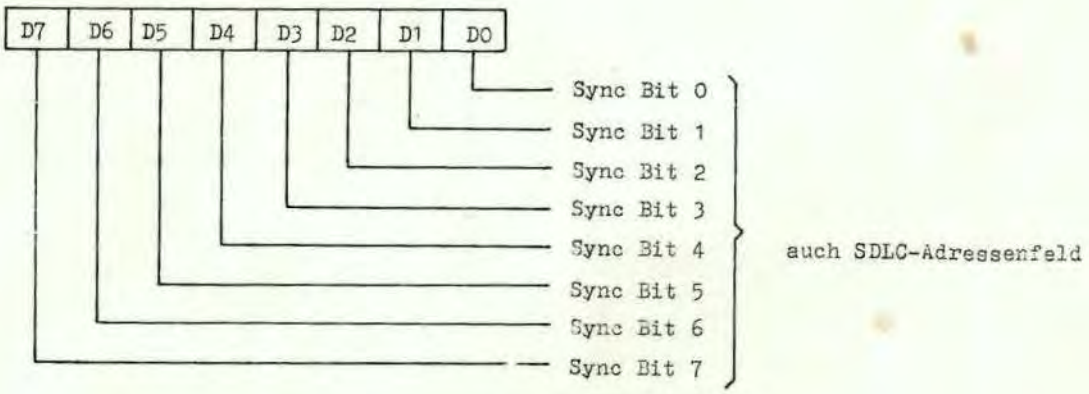
Schreibregister 4 (Write-Register 4)



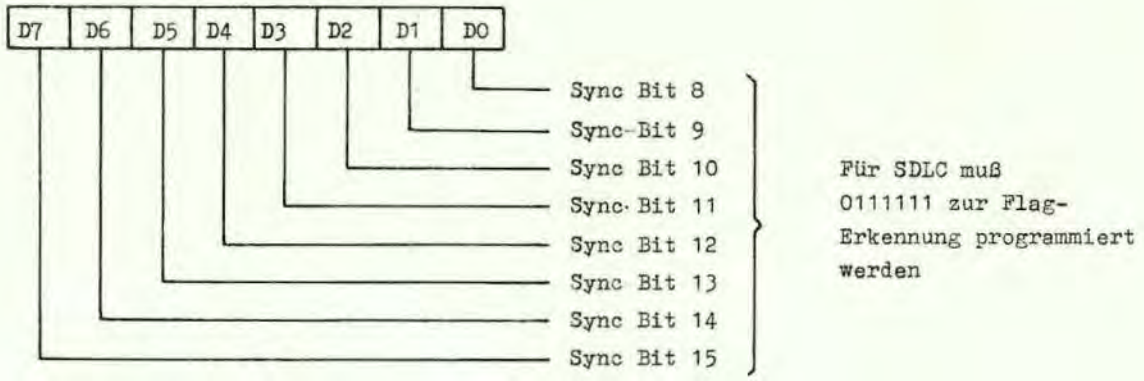
Schreibregister 5 (Write-Register 5)



Schreibregister 6 (Write-Register 6)



Schreibregister 7 (Write-Register 7)



Befehl 6 (Error Reset)
Dieser Befehl setzt die Fehlerspeicher (Error latches) zurück. Paritäts- und Überlauffehler werden in RR1 solange gespeichert, bis sie mit diesem Befehl rückgesetzt werden. Damit können Paritätsfehler, die in Blockübertragungen auftreten, am Ende des Blockes geprüft werden.

Befehl 7 (Rückkehr vom Interrupt)

Dieser Befehl muß in Kanal A ausgegeben werden und wird von dem SIO in genau derselben Art und Weise interpretiert wie ein RETI-Befehl auf dem Datenbus. Er setzt das Interrupt-Under-Service-Latch der am höchsten priorisierten Interruptquelle zurück und ermöglicht einem niedriger priorisierten Gerät (Sender, Empfänger) über die Daisy Chain zu unterbrechen. Dieser Befehl erlaubt die Anwendung der internen Daisy Chain sogar in Systemen ohne externe Daisy Chain oder RETI-Befehl.

CRC-Reset-Codes 0 und 1 (D_6 und D_7)

Mit diesen Bits können die folgenden drei Reset-Befehle ausgewählt werden.

CRC Reset Code 1	CRC Reset Code 0	
0	0	Null-Code (keine Wirkung)
0	1	Reset Empfangs-CRC-Prüfer
1	0	Reset Sender-CRC-Generator
1	1	Reset Tx Underrun/End of Message latch

Der "Reset-Sender-CRC-Generator"-Befehl initialisiert normalerweise den CRC-Generator auf alles Nullen. In der SDLC-Betriebsart wird der Generator auf alles "1" initialisiert. Der Empfänger-CRC-Prüfer wird auch auf alles "1" für die SDLC-Betriebsart initialisiert.

Schreibregister 1

WR1 enthält die Steuerbits für die verschiedenen Interrupt- und Wait/Ready-Betriebsarten.

D_7	D_6	D_5	D_4
Wait/Reday Freigabe	$\overline{\text{Wait}}$ oder Ready Funktion	Wait/Ready bei Empfang /Senden	Empfangs-interrupt Mode 1

D_3	D_2	D_1	D_0
Empfangs-Interrupt Mode 0	Statusabhängiger Vektor	Sender-Interrupt-Freigabe	Extern-Interrupt-Freigabe

Extern/Status Interrupt Freigabe (D_0)

Die Extern/Status-Interrupt-Freigabe gestattet Interrupts als Ergebnis von Änderungen auf den $\overline{DCD}$, $\overline{CTS}$ oder $\overline{Sync}$-Eingängen, sowie als Ergebnis einer Break/Abort-Erkennung und Beendigung oder am Anfang der CRC oder Sync-Zeichen-Übertragung, wenn das Sender **Underrun/EOM** latch gesetzt wird.

Sender-Interrupt-Freigabe (D_1)

Bei Leerwerden des Sendepuffers wird ein Interrupt ausgelöst.

Statusabhängiger Vektor (D_2)

Dieses Bit wirkt nur im Kanal B. Bei nicht gesetzten Bit wird der programmierte Interruptvektor unverändert an die CPU gegeben. Bei gesetztem Bit wird der Interruptvektor in Abhängigkeit der folgenden Bedingungen verändert:

	V_3	V_2	V_1		
Kanal B	0	0	0	Kanal B	Senderpuffer leer
	0	0	1	Kanal B	Extern/Status-Veränderung
	0	1	0	Kanal B	Empfangszeichen vorhanden
	0	1	1	Kanal B	Spezielle Empfangsbedingung *
Kanal A	1	0	0	Kanal A	Senderpuffer leer
	1	0	1	Kanal A	Extern/Status-Veränderung
	1	1	0	Kanal A	Empfangszeichen vorhanden
	1	1	1	**Kanal A**	**Spezielle Empfangsbedingung** *

* Spezielle Empfangsbedingungen: Paritätsfehler, Rx Überlauffehler, Rahmenfehler, Rahmenende (SDLC)

Empfangs-Interrupt-Mode 0 und 1 (D_2 und D_4)

Diese beiden Bits spezifizieren die verschiedenen Empfangszeichen-Bedingungen. Im Empfangs-Interrupt-Mode 1, 2 und 3 kann eine spezielle Empfangs-Bedingung einen Interrupt hervorrufen und den Interruptvektor modifizieren.

D_4 Empfangs-Interrupt Mode 1	D_3 Empfangs-Interrupt Mode 0		
0	0	0.	Empfangsinterrupts gesperrt
0	1	1.	Empfangsinterrupt nur beim ersten Zeichen
1	0	2.	Interrupt bei jedem Empfangszeichen - Paritätsfehler ist eine spezielle Empfangs-Bedingung
1	1	3.	Interrupt bei jedem Empfangszeichen - Paritätsfehler ist keine spezielle Empfangsbedingung

Wait/Ready Funktionsauswahl (D_5-D_7)

Die Wait- und Ready-Funktionen werden durch die Steuerbits D_5, D_6 und D_7 festgelegt. Die Wait/Ready-Funktion wird freigegeben durch Setzen von Wait/Ready Enable (WR1, D_7) auf 1. Die Ready-Funktion wird durch Setzen von D_6 (Wait/Ready-Funktion) auf 1 ausgewählt. Bei gesetztem Bit (D_6) geht der $\overline{Wait/Ready}$-Ausgang von High auf Low, wenn der SIO zur Datenübertragung bereit ist. Die Wait-Funktion wird durch Setzen von D_6 auf 0 ausgewählt. In diesem Fall ist der $\overline{Wait/Ready}$-Ausgang im Open-Drain-Zustand. Im aktiven Zustand wird er Low

Sowohl die Wait- als auch die Ready-Funktion kann entweder im Sender- oder Empfängerbetrieb genutzt werden, aber nicht gleichzeitig. Wenn D_5 (Wait/Ready on Recive/Transmit) auf 1 gesetzt ist, entspricht die Wait/Ready-Funktion dem Zustand des Empfangspuffers (leer oder voll). Ist D_5 auf 0 gesetzt, entspricht die Wait/Ready-Funktion dem Zustand des Sendepuffers (leer oder voll).
Die logischen Zustände des Wait/Ready-Ausganges hängen, wenn sie aktiv oder inaktiv sind, von der Kombination der ausgewählten Betriebsarten ab. Es folgt eine Zusammenfassung dieser Kombinationen:

Bei $D_7 = 0$

und $D_6 = 1$	und $D_6 = 0$
$\overline{\text{Ready}}$ ist High	$\overline{\text{Wait}}$ ist floatend

Bei $D_7 = 1$

und $D_5 = 0$	und $D_5 = 1$
$\overline{\text{Ready}}$ ist High, wenn der Sendepuffer voll ist.	$\overline{\text{Ready}}$ ist High, wenn der Empfangspuffer leer ist.
$\overline{\text{Wait}}$ ist Low, wenn der Sendepuffer voll ist und ein SIO-Datenport ausgewählt wurde.	$\overline{\text{Wait}}$ ist Low, wenn der Empfangspuffer leer ist und ein SIO-Datenport ausgewählt wurde.
$\overline{\text{Ready}}$ ist Low, wenn der Sendepuffer leer ist.	$\overline{\text{Ready}}$ ist Low, wenn der Empfangspuffer voll ist.
$\overline{\text{Wait}}$ ist floatend, wenn der Sendepuffer leer ist.	$\overline{\text{Wait}}$ ist floatend, wenn der Empfangspuffer voll ist.

Der High-Low-Übergang des Wait-Ausganges erscheint mit der Verzögerungszeit t_DIC (WR) nach dem I/O-Request. Der Low-High-Übergang tritt mit der Verzögerung t_DHO (WR) nach der fallenden Flanke von O auf. Der High-Low-Übergang des Ready-Ausganges erscheint mit der Verzögerung t_DLO (WR) nach der steigenden Flanke von O. Der Low-High-Übergang kommt dagegen mit der Verzögerung t_DIC (WR) nachdem $\overline{\text{IORQ}}$ Low wird.
Die Ready-Funktion kann jederzeit auftreten, auch wenn der SIO nicht angesprochen wird. Wenn der $\overline{\text{Ready}}$-Ausgang aktiv wird (Low), sendet das DMA-Steuergerät $\overline{\text{IORQ}}$ und die entsprechenden B/$\overline{\text{A}}$ und C/$\overline{\text{D}}$ Eingänge zur SIO, um Daten zu übertragen. Der $\overline{\text{Ready}}$-Ausgang wird inaktiv, sobald $\overline{\text{IORQ}}$ und $\overline{\text{CS}}$ aktiv werden. Da die Ready-Funktion intern im SIO auftreten kann, egal ob er adressiert ist oder nicht, wird der $\overline{\text{Ready}}$-Ausgang inaktiv, wenn eine Übertragung von CPU-Daten oder Befehlen erfolgt. Das verursacht keine Probleme, weil das DMA-Steuergerät nicht freigegeben ist, wenn die CPU sendet.
Die Wait-Funktion andererseits ist nur aktiv, wenn die CPU versucht, SIO-Daten zu lesen, die noch nicht empfangen worden sind, was häufig auftritt, wenn Blockübertragungsbefehle verwendet werden. Die Wait-Funktion kann auch aktiv werden (unter Programmsteuerung), wenn die CPU versucht, Daten zu schreiben, während der Senderpuffer noch voll ist.
Die Tatsache, daß der $\overline{\text{Wait}}$-Ausgang eines Kanals aktiv werden kann, wenn der andere Kanal adressiert wird, beeinflußt nicht die Softwareschleifen oder Blockbewegungsbefehle.

Schreibregister 2

WR2 ist das Interrupt-Vektor-Register, es existiert nur in Kanal B. V_4-V_7 und V_0 werden durch den statusabhängigen Vektor nicht beeinflußt. V_1-V_3 werden jedoch bei gesetztem Steuerbit (WR1, D_2) modifiziert (siehe Beschreibung WR1).

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
V_7	V_6	V_5	V_4	V_3	V_2	V_1	V_0

Schreibregister 3

WR3 enthält die Empfangs-Logik-Steuerbits und Parameter.

D_7	D_6	D_5	D_4
Empfänger-Bits/ Zeichen 1	Empfänger-Bits/ Zeichen 0	Auto Enables	Beginne Such-Betrieb

D_3	D_2	D_1	D_0
Empfänger CRC Freigabe	Adreß-Such-Betrieb	Sync-Zeichen-Lade-sperre	Empfänger-Freigabe

Empfänger-Freigabe (D_0)

Eine programmierte "1" erlaubt den Beginn der Empfangsoperationen. Dieses Bit darf erst gesetzt werden, nachdem alle anderen Empfangsparameter gesetzt worden sind und der Empfänger vollständig initialisiert ist.

Sync-Zeichen-Ladesperre (D_1)

Sync-Zeichen, die der Nachricht vorangehen (führende Sync-Zeichen), werden nicht in den Empfangspuffer geladen, wenn dieses Bit gesetzt ist. Da die CRC-Berechnungen durch das Unterdrücken der Sync-Zeichen nicht gestoppt werden, darf diese Betriebsart nur am Beginn der Nachricht freigegeben werden.

Adreß-Such-Betriebsart (D_2)

Im SDLC-Betrieb bewirkt diese Betriebsart, daß Nachrichten mit Adressen, die mit der programmierten Adresse in WR6 oder der globalen Adresse (11111111) nicht übereinstimmen, abgelehnt werden. Mit anderen Worten, in der Adreß-Such-Betriebsart können erst Empfängerinterrupts auftreten, wenn eine Adressenübereinstimmung vorhanden ist.

Empfänger-CRC-Freigabe (D_3)

Bei gesetztem Bit beginnt die CRC-Berechnung (oder beginnt erneut) am Anfang des letzten Zeichens, das vom Empfangsschieberegister zum Puffer-Stack übertragen wurde, unabhängig von der Anzahl der Zeichen im Stack. (Zur näheren Erläuterung siehe folgende Abschnitte: "SDLC-Empfangs-CRC-Prüfung" und "CRC-Fehler-Prüfung" im Abschnitt über den synchronen Empfang.)

Beginn Such-Phase (Hunt Phase) (D_4)

Nach einem Reset beginnt der SIO automatisch mit der Hunt-Phase; er kann jedoch auch erneut beginnen, wenn aus irgendeinem Grunde die Zeichensynchronisation verlorengegangen ist (Synchrone Betriebsart) oder wenn der Inhalt einer ankommenden Nachricht nicht benötigt wird (SDLC-Betriebsart). Die Hunt-Phase wird durch Setzen von Bit D_4 erneut initialisiert. Dieses setzt das Sync/Hunt-Bit (D_4) in RR0.

Auto Enables (D_5)

In dieser Betriebsart werden $\overline{DCD}$ und $\overline{CTS}$ dem Empfänger und Sender entsprechend freigegeben. Ist dieses Bit nicht gesetzt, sind $\overline{DCD}$ und $\overline{CTS}$ einfache Eingänge für ihre zugehörigen Statusbits in RR0.

Empfängerbits/Zeichen 1 und 0 (D_7 und D_6)

Diese Bits bestimmen die Anzahl der seriellen Empfangsbits, die zu einem Zeichen gehören. Beide Bits können während der Zeit geändert werden, in der ein Zeichen empfangen wird. Die Umprogrammierung muß jedoch abgeschlossen sein, bevor die ausgewählte Anzahl Bits erreicht ist.

D_7	D_6	Bits/Zeichen
0	0	5
0	1	6
1	0	7
1	1	8

Schreibregister 4

WR4 enthält die Steuerbits, welche sowohl den Empfänger als auch den Sender beeinflußen. In der Sender- und Empfängerinitialisierungsroutine sollten diese Bits gesetzt sein, bevor WR1, WR3, WR5, WR6 und WR7 ausgegeben werden.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Takt Rate 1	Takt Rate 0	Sync Mode 1	Sync Mode 0	Stopp Bits 1	Stopp Bits 0	Parität gerade/ $\overline{\text{ungerade}}$	Parität

Parität (D_0)

Wenn dieses Bit gesetzt ist, wird eine zusätzliche Bitposition (zusätzlich zu denjenigen, die in der Bits/Zeichen-Steuerung angegeben sind) zu den gesendeten Daten hinzugefügt und bei den Empfangsdaten erwartet. In der Betriebsart Empfangen wird das erhaltene Paritätsbit als Teil des Zeichens zur CPU übertragen, falls nicht 8 Bits/Zeichen programmiert sind.

gerade, $\overline{\text{ungerade}}$ Parität (D_1)

Dieses Bit legt fest, ob die Daten mit gerader oder ungerader Parität gesendet und geprüft werden. (D_1 = 1 entspricht gerader Parität)

Stopp-Bits 0 und 1 (D_2 und D_3)

Diese Bits bestimmen die Anzahl der Stopp-Bits, die jedem gesendeten asynchronen Zeichen beigefügt werden. Der Empfänger prüft immer auf ein Stopp-Bit. Werden beide Bits auf Null gesetzt, so wird die synchrone Betriebsart ausgewählt.

D_3 Stopp-Bits 1	D_2 Stopp-Bits 0	
0	0	Synchrone Betriebsart
0	1	1 Stopp-Bit pro Zeichen
1	0	1 1/2 Stopp-Bit pro Zeichen
1	1	2 Stopp-Bits pro Zeichen

Synchrone Betriebsarten 0 und 1 (D_4 und D_5)

Mit diesen Bits kann zwischen den verschiedenen Synchronmodes ausgewählt werden.

Sync Mode 1	Sync Mode 0	
0	0	8-Bit Sync-Betriebsart (Monosync)
0	1	16-Bit Sync-Betriebsart (Bisync)
1	0	SDLC-Mode (01111110-Flag)
1	1	Extern-Sync-Mode

Takt-Rate 0 und 1 (D_6 und D_7)

Diese Bits bestimmen den Multiplikator zwischen dem Takt ($\overline{TxC}$ und $\overline{RxC}$) und der Datenrate. Für die synchronen Betriebsarten muß die x1-Taktrate benutzt werden. Für die asynchronen Betriebsarten kann jede Rate verwendet werden, jedoch muß sie für Empfänger und Sender die gleiche sein. Der Systemtakt muß in allen Betriebsarten mindestens 4,5x größer als die Datenrate sein. Wenn die x1-Taktrate ausgewählt worden ist, muß die Bitsynchronisation extern erfolgen.

Taktrate 1	Taktrate 0	
0	0	Datenrate x1 = Taktfrequenz
0	1	Datenrate x16 = Taktfrequenz
1	0	Datenrate x32 = Taktfrequenz
1	1	Datenrate x64 = Taktfrequenz

Schreibregister 5

WR5 enthält Steuerbits, die die Senderoperation beeinflussen, mit Ausnahme von D_2, das sowohl für Empfänger als auch Sender gilt.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
DTR	Tx Bits/ Zeich.1	Tx Bits/ Zeich.0	Sende Break	Sender Freigabe	CRC-16/ $\overline{SDLC}$	RTS	Sender-CRC-Freigabe

Sender-CRC-Freigabe (D_0)

Dieses Bit bestimmt, ob CRC für ein spezielles Sender-Zeichen berechnet wird. Wenn es in der Zeit gesetzt wird, wo das Zeichen vom Sendepuffer in das Sendeschieberegister geladen wird, so erfolgt für dieses Zeichen die CRC-Berechnung. CRC wird nicht automatisch gesendet, es sei denn, daß das Bit gesetzt ist und die Transmit-Underrun-Bedingung existiert.

Request to Send (D_1) (Sendeanforderung)

Dies ist das Steuerbit für das $\overline{RTS}$-Pin. Wenn das $\overline{RTS}$-Bit gesetzt ist, geht das $\overline{RTS}$-Pin auf Low; wenn es rückgesetzt ist, geht $\overline{RTS}$ auf High. In der asynchronen Betriebsart wird $\overline{RTS}$ nur High, nachdem alle Bits des Zeichens übertragen worden sind und der Senderpuffer leer ist. In synchronen Betriebsarten folgt das Pin direkt dem Zustand des Bits.

CRC-16/$\overline{SDLC}$ (D_2)

Dieses Bit wählt das CRC-Polynom aus, das sowohl vom Sender, als auch vom Empfänger verwendet wird. Bei **gesetztem** Bit wird das CRC-16 Polynom ($X^{16} + X^{15} + X^2 + 1$) benutzt, ist es rückgesetzt, so kommt das SDLC-Polynom ($X^{16} + X^{12} + X^5 + 1$) zur Anwendung.

In der SDLC-Betriebsart wird der CRC-Generator und Prüfer auf alles "1" gesetzt und eine spezielle Prüffolge benutzt. In der SDLC-Betriebsart muß das SDLC-CRC-Polynom ausgewählt werden. Wurde die SDLC-Betriebsart nicht ausgewählt, so werden der CRC-Generator und Prüfer auf alles "0" gesetzt (für beide Polynome).

Sender-Freigabe (D_3)

Die Datenausgabe bleibt blockiert und der Datenausgang auf High (Marking) bis D_3 gesetzt wird. Wenn beim Senden von Daten oder Sync-Zeichen das Bit rückgesetzt wird, werden diese Zeichen noch voll ausgesandt. Wenn der Sender während der Übertragung von CRC-Zeichen gesperrt wird, so werden Sync- oder Flag-Zeichen anstatt von CRC ausgesandt.

Sende Break (D_4)

Das Setzen dieses Bits bewirkt das sofortige Aussenden von Low (Spacing) am Datenausgang, unabhängig davon, ob Daten gesendet wurden. Beim Rücksetzen geht TxD auf High (Marking).

Sender-Bits/Zeichen 0 und 1 (D_5 und D_6)

D_6 und D_5 steuern die Anzahl der Bits in jedem Byte, das zum Sendepuffer übertragen wird.

D_6 Sender Bits/ Zeichen 1	D_5 Sender Bits/ Zeichen 0	Bits/Zeichen
0	0	5 oder weniger
0	1	7
1	0	6
1	1	8

Die zu sendenden Bits müssen rechtsbündig ausgerichtet sein; D_0 wird zuerst gesendet. Die Betriebsart "5 Bits oder weniger" erlaubt die Übertragung von einem bis fünf Bits pro Zeichen, wobei die Daten wie folgt formatiert sein müssen:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	1	1	1	0	0	0	0	Sendet 1 Bit
1	1	1	0	0	0	0	0	Sendet 2 Bits
1	1	0	0	0	0	0	0	Sendet 3 Bits
1	0	0	0	0	0	0	0	Sendet 4 Bits
0	0	0	0	0	0	0	0	Sendet 5 Bits

Data Terminal Ready (D_7) (Terminalbereitschaft)

Dies ist das Steuerbit für das DTR-Pin. Wenn es gesetzt ist, ist $\overline{DTR}$ aktiv (Low), bei $D_7 = 0$ ist $\overline{DTR}$ inaktiv (High).

Schreibregister 6

Dieses Register enthält entweder das Sende-Sync-Zeichen in der Monosync-Betriebsart, die ersten acht Bits des 16-Bit-Sync-Zeichens in der Bisync-Betriebsart oder ein Sende-Sync-Zeichen in der Extern-Sync-Betriebsart. Im SDLC-Mode enthält dieses Register das sekundäre Adressenfeld, welches mit dem Adressenfeld des SDLC-Rahmens verglichen wird.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Sync 7	Sync 6	Sync 5	Sync 4	Sync 3	Sync 2	Sync 1	Sync 0

Schreibregister 7

Dieses Register enthält entweder das Empfangs-Sync-Zeichen in der Monosync-Betriebsart, das zweite Byte des 16 Bit-Sync-Zeichens in der Bisync-Betriebsart oder ein Flag-Zeichen (01111110) im SDLC-Mode. **WR 7 wird im Extern-Sync-Mode nicht benutzt.**

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Sync 7	Sync 6	Sync 5	Sync 4	Sync 3	Sync 2	Sync 1	Sync 0

5.2. Leseregister

Der SIO enthält drei Register RR0-RR2 (Bild 15), die gelesen werden können, um die Statusinformation für jeden Kanal (ausgenommen für RR2 - nur Kanal B) zu erhalten. Die Statusinformation umfaßt **Fehlerbedingungen,** Interrupt-Vektor und Standard-Kommunikations-Interface-Signale.
Um den Inhalt eines ausgewählten Leseregisters außer RR0 zu lesen, muß das Systemprogramm zuerst das Zeigerbyte für WR0 in genau derselben Art und Weise wie bei einer Schreibregisteroperation schreiben. Dann kann der Inhalt des adressierten Leseregisters durch einen Eingabebefehl der CPU gelesen werden.

Leseregister 0

Dieses Register enthält den Status des Empfänger- und Senderpuffers, die $\overline{DCD}$-, $\overline{CTS}$- und $\overline{SYNC}$-Eingänge, den Transmit-**Underrun**/EOM-Latch und den Break/Abort-Latch.

D_7	D_6	D_5	D_4	D_3	D_2	I_1	D_0
Break/ Abort	Transmit Underrun/ EOM	CTS	Sync/ Hunt	DCD	Sender-Puffer leer	Interrupt Pending (nur Kanal A)	Zeichen im Empfänger-Puffer

Zeichen im Empfängerpuffer (D_0)

Dieses Bit wird gesetzt, wenn mindestens ein Zeichen im Empfangspuffer ist; es wird rückgesetzt, wenn der Empfangs-FIFO vollständig leer ist.

Interrupt Pending (D_1)

Jede Interrupt-Bedingung im SIO veranlaßt, daß dieses Bit zu setzen ist; jedoch ist dieses Bit nur in Kanal A lesbar. Dieses Bit wird hauptsächlich dort verwendet, wo keine Vektorinterrupts möglich sind. Während der Interrupt-Service-Routine zeigt dieses Bit in diesen Anwendungen an, ob irgendwelche Interrupt-Bedingungen im SIO vorhanden sind. Dadurch müssen nicht alle Bits von RR0 sowohl in Kanal A als auch B analysiert werden. Bit D_1 wird rückgesetzt, wenn alle Interrupt-Bedingungen erfüllt sind. Dieses Bit ist immer 0 in Kanal B.

Transmit Buffer Empty (D_2) (Senderpuffer leer)

Dieses Bit wird immer gesetzt, wenn der Senderpuffer leer wird, ausgenommen, wenn ein CRC-Zeichen in einer synchronen oder SDLC-Betriebsart gesendet wird. Dieses Bit wird rückgesetzt, wenn ein Zeichen in den Sender-Puffer geladen wird. Dieses Bit ist nach einem Reset gesetzt.

Data Carrier Detect (D_3) (Datenträgererkennung)

Das DCD-Bit zeigt den Zustand des DCD-Einganges während der letzten Änderung eines der fünf externen/Status-Bits (DCD, CTS, Sync/Hunt, Break/Abort oder Transmit Underrun/EOM). Jede Veränderung des DCD-Einganges bewirkt, daß das DCD-Bit gespeichert wird und verursacht einen Extern/Status-Interrupt. Um den laufenden Zustand des DCD-Bits zu lesen, muß dieses Bit sofort nach einem Reset-Befehl für Extern/Status-Interrupt gelesen werden.

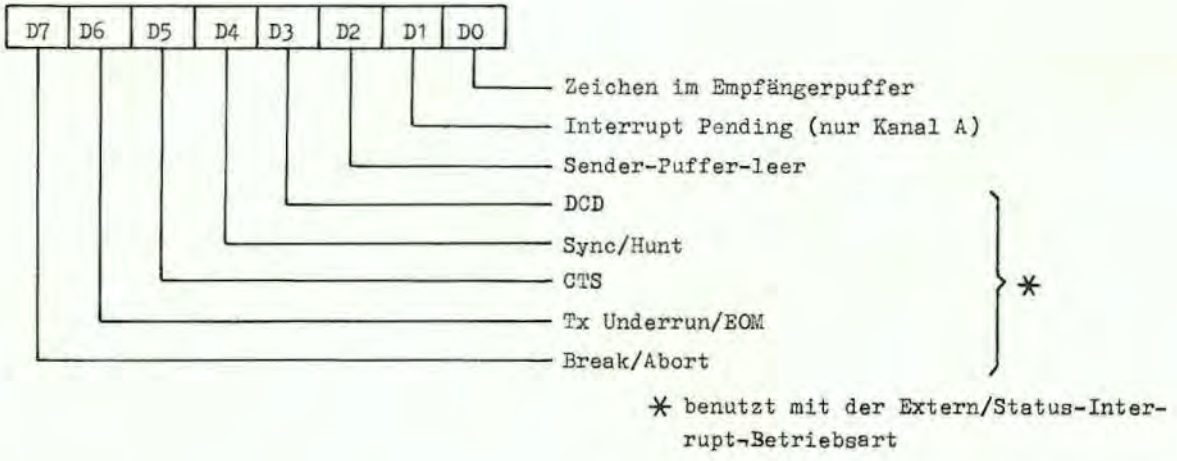
Sync/Hunt (D_4)

Da dieses Bit in der asynchronen, synchronen und SDLC-Betriebsart unterschiedlich benutzt wird, ist eine umfangreiche Erläuterung der Funktion notwendig.

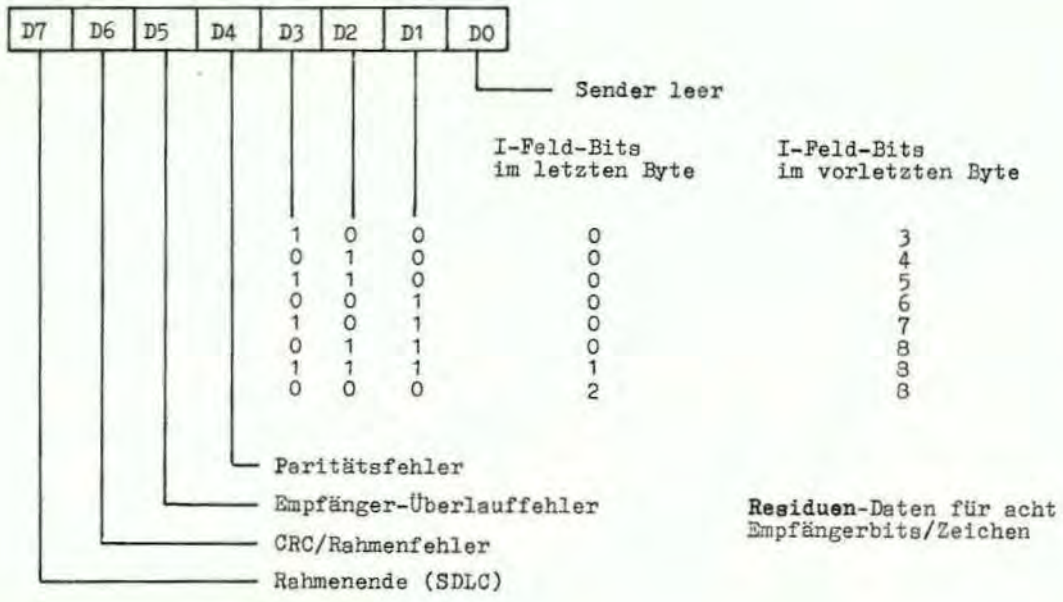
In den asynchronen Betriebsarten ist die Funktion dieses Bits dem DCD-Statusbit ähnlich, ausgenommen, daß Sync/Hunt den Zustand des $\overline{\text{Sync}}$-Einganges anzeigt. Jeder High-Low-Übergang an dem $\overline{\text{Sync}}$-Pin setzt dieses Bit und verursacht einen Extern/Status-Interrupt. Der Reset-Befehl für den Extern/Status-Interrupt wird ausgegeben, um den Interrupt zu löschen. Ein Low-High-Übergang löscht dieses Bit und setzt den Extern/Status-Interrupt. Wenn der Extern/Status-Interrupt durch die Zustandsänderung irgendeines anderen Eingangs oder einer Bedingung gesetzt wird, zeigt dieses Bit den invertierten Zustand des Sync-Pins zur Zeit der Änderung an. Dieses Bit muß sofort nach einem Reset-Befehl für einen Extern/Status-Interrupt gelesen werden, um den momentanen Zustand des $\overline{\text{Sync}}$-Einganges zu lesen.
Im externen - Sync - Betrieb wirkt das Sync-Hunt-Bit in einer der asynchronen Betriebsart ähnlichen Weise, ausgenommen, daß Enter-Hunt-Mode-Steuerbit gibt die externe Synchronisationserkennungslogik frei. Wenn die Bits für Extern-Sync- und Enter-Hunt-Betrieb gesetzt sind (z. B. wenn der Empfänger nach einem Reset freigegeben ist), muß der Sync-Eingang durch die externe Logik High gehalten werden, bis die externe Zeichensynchronisation erreicht ist. Ein High am $\overline{\text{Sync}}$-Eingang hält das Sync/Hunt-Status-Bit in der Reset-Bedingung.
Wenn die externe Synchronisation erreicht ist, muß $\overline{\text{Sync}}$ mit der zweiten steigenden Flanke von $\overline{\text{RxC}}$ Low werden, und zwar nach der steigenden Flanke von RxC, mit der das letzte Bit des Sync-Zeichens empfangen wurde. Mit anderen Worten, nachdem das Syncmuster ermittelt worden ist, muß die externe Logik zwei volle Empfänger-Takt-Zyklen warten, um den $\overline{\text{Sync}}$-Eingang zu aktivieren.
Wenn $\overline{\text{SYNC}}$ einmal Low geworden ist, wird es üblicherweise Low gehalten, bis die CPU die externe Logik informiert, daß die Synchronisation verlorengegangen ist oder eine neue Nachricht im Begriff ist zu starten. Der High-Low-Übergang des $\overline{\text{Sync}}$-Einganges setzt das Sync/Hunt-Bit, welches seinerseits den Extern/Status-Interrupt setzt. Die CPU muß den Interrupt löschen, indem sie den Reset-Befehl für den Extern/Status-Interrupt ausgibt.

Bild 15 Leseregister - Bitfunktionen

Leseregister 0 (Read Register 0)

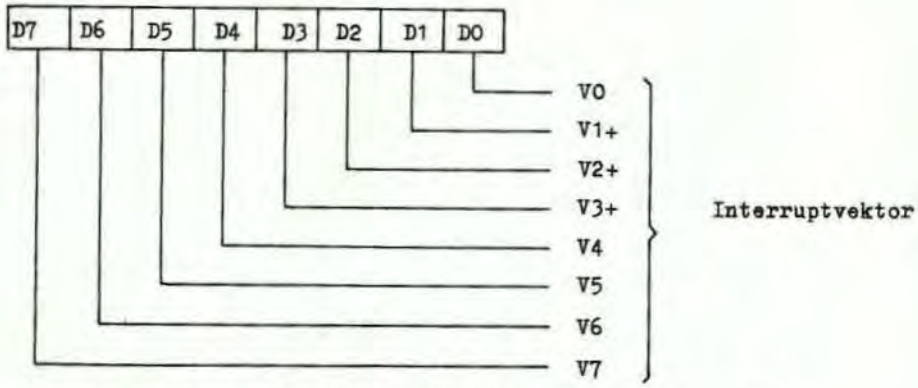


Leseregister 1 (Read Register 1) +



+ wird benutzt mit der speziellen Empfangsbedingungs-Betriebsart

Leseregister 2 (Read Register 2)



+ variabel, wenn der statusabhängige Vektor programmiert wurde

Wenn der $\overline{\text{Sync}}$-Eingang wieder High wird, wird ein anderer Extern/Status-Interrupt erzeugt, welcher auch gelöscht werden muß. Das Enter-Hunt-Mode-Steuerbit wird immer dann gesetzt, wenn die Zeichensynchronisation verlorengegangen ist oder das Ende der Nachricht festgestellt wurde. In diesem Fall wartet der SIO wieder auf einen High-Low-Übergang am $\overline{\text{Sync}}$-Eingang und die Operation wiederholt sich, wie vorher erklärt worden ist. Das bedeutet, daß die CPU die externe Logik auch darüber informieren muß, daß die Zeichensynchronisation verlorengegangen ist und daß der SIO darauf wartet, daß $\overline{\text{Sync}}$ aktiv wird.
In den Betriebsarten Monosync- und Bisync-Empfang wird das Sync/Hunt-Status-Bit anfangs durch das Enter-Mode-Bit auf 1 gesetzt. Das Sync/Hunt_Bit wird zurückgesetzt, wenn der SIO die Zeichensynchronisation herstellt. Der High-Low-Übergang des Sync/Hunt-Bits verursacht ein Extern/Status-Interrupt, der von der CPU gelöscht werden muß, indem sie den Reset-Extern/Status-Interrupt-Befehl ausgibt. Dies gibt den SIO frei, den nächsten Übergang eines anderen Extern/Status-Bits zu erkennen.
Wenn die CPU das Ende der Nachricht erkennt oder erkennt, daß die Zeichensynchronisation verlorengegangen ist, setzt sie das Enter-Hunt-Mode-Steuer-Bit, das seinerseits das Sync/Hunt-bit auf 1 setzt. Der Low-High-Übergang des Sync/Hunt-Bits bewirkt einen Extern/Sta-

tus-Interrupt, der auch durch den Reset-Befehl für den Extern/Status-Interrupt gelöscht werden muß. Es ist zu beachten, daß das $\overline{\text{Sync}}$-Pin in dieser Betriebsart als Ausgang wirkt und jedesmal auf Low geht, wenn im Datenstrom ein Sync-Muster erkannt wurde.
In der SDLC-Betriebsart wird das Sync/Hunt-Bit anfangs durch das Enter-Hunt-Bit gesetzt oder wenn der Empfänger gesperrt ist. In jedem Fall wird es auf 0 zurückgesetzt, wenn das Anfangsflag des ersten Rahmens vom SIO erkannt wird. Ein Extern/Status-Interrupt wird ebenfalls erzeugt, der in der bereits erwähnten Weise behandelt wird.
Im Gegensatz zum Monosync- und Bisync-Betrieb braucht das Sync/Hunt-Bit, wenn es einmal im SDLC-Betrieb zurückgesetzt worden ist, nicht gesetzt zu werden, wenn das Ende der Nachricht erkannt worden ist. Der SIO erhält die Synchronisation automatisch aufrecht. Der einzige Weg, daß das Sync/Hunt-Bit wieder gesetzt werden kann, ist durch das Enter-Hunt-Mode-Bit oder durch die Sperrung des Empfängers.

Clear to Send (D_5)

Dieses Bit ist dem DCD-Bit ähnlich, nur daß es den invertierten Zustand des $\overline{\text{CTS}}$-Pins anzeigt.

Transmit Underrun/End of Message (D_6)

Nach einem Reset (intern oder extern) ist dieses Bit in einem gesetzten Zustand. Der einzige Befehl, der dieses Bit zurücksetzen kann, ist der Reset-Befehl für das Transmit-Underrun/EOM-Latch (WRO, D_6 und D_7). Wenn die Transmit-Underrun-Bedingung auftritt, wird dieses Bit gesetzt, sein Setzen verursacht einen Extern/Status-Interrupt, der durch Ausgabe des Reset-Extern/Status-Interrupt-Befehls (WRO) zurückgesetzt werden muß. **Dieses Statusbit spielt eine wichtige Rolle in Verbindung mit anderen Steuerbits bei der Steuerung einer Senderoperation. Zusätzliche Informationen sind den Abschnitten "Bisync Transmit Underrun" und "SDLC Transmit Underrun" zu entnehmen.**

Break/Abort (D_7)

In der Betriebsart "Asynchroner Empfang" wird dieses Bit gesetzt, wenn im Datenstrom eine Breakfolge (Null-Zeichen plus Rahmenfehler) erkannt wird. Der freigegebene Extern/Status-Interrupt wird gesetzt, wenn ein Break erkannt wird. Die Interrupt-Service-Routine muß den Reset-Befehl für den Extern/Status-Interrupt (WRO, CMD_2) zur Break-Erkennungs-Logik ausgeben, damit das Break-Folgen-Ende erkannt werden kann.
Das Break/Abort-Bit wird zurückgesetzt, wenn das Ende der Breakfolge im ankommenden Datenstrom ermittelt worden ist. Das Ende der Breakfolge bewirkt auch, daß ein Extern/Status-Interrupt hervorgerufen wird. Der Reset-Befehl für den Extern/Status-Interrupt muß ausgegeben werden, um die Break-Erkennungslogik zu veranlassen, auf die nächste Breakfolge zu warten. Nach dem Ende eines Break ist im Empfänger ein einzelnes äußeres Nullzeichen vorhanden; es sollte gelesen und abgelegt werden.
Im SDLC-Empfangsbetrieb wird dieses Statusbit durch die Erkennung einer Abortfolge gesetzt (sieben oder mehr Einsen). Der Extern/Status-Interrupt wird auf die gleiche Weise behandelt wie im Fall eines Break. Das Break/Abort-Bit wird im synchronen Empfangsbetrieb nicht verwendet.

Leseregister 1 (Read-Register 1)

Dieses Register enthält die Status-Bits für die spezielle Empfangsbedingung und die Residuen-Codes für das I-Feld im SDLC-Empfangs-Betrieb.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
End of Frame (Rahmenende) (SDLC)	CRC/ Rahmen-fehler	Empfänger-überlauf-fehler	Paritäts-fehler	Residuen Code 2	Residuen Code 1	Residuen Code 0	All Sent (Sender leer)

All Sent (D_0) (Sender leer)

In den asynchronen Betriebsarten wird dieses Bit gesetzt, wenn alle Zeichen den Sender vollständig verlassen haben. Änderungen dieses Bits verursachen keine Interrupts. In der synchronen Betriebsart ist es immer gesetzt.

Residuen Codes 0, 1 und 2 (D_1-D_3)

In Fällen des SDLC-Empfang-Betriebes, in denen das I-Feld nicht ein ganzzahliges Vielfaches der Zeichenlänge ist, zeigen diese drei Bits die Länge des I-Feldes an. Diese Kodes sind nur für die Übertragung bedeutungsvoll, in der das Rahmenende-Bit gesetzt ist (SDLC). Für eine Empfangszeichenlänge von acht Bits pro Zeichen bedeuten die Kodes folgendes:

Residuen Code 2	Residuen Code 1	Residuen Code 0	I-Feld-Bits im letzten Byte	I-Feld-Bits im vorletzten Byte
1	0	0	0	3
0	1	0	0	4
1	1	0	0	5
0	0	1	0	6
1	0	1	0	7
0	1	1	0	8
1	1	1	1	8
0	0	0	2	8

Die I-Feld-Bits sind in allen Fällen rechtsbündig

Wenn eine Empfangs-Zeichenlänge abweichend von acht Bit für das I-Feld benutzt wird, kann eine ähnliche Tafel für jede unterschiedliche Zeichenlänge aufgestellt werden. Wenn kein Reset vorhanden ist (d. h. die letzte Zeichengrenze stimmt mit der Grenze des I-Feldes und des CRC-Feldes überein) ergeben sich folgende Residuen-Kodes:

Bits pro Zeichen	Residuen Code 2	Residuen Code 1	Residuen Code 0
8 Bits pro Zeichen	0	1	1
7 Bits pro Zeichen	0	0	0
6 Bits pro Zeichen	0	1	0
5 Bits pro Zeichen	0	0	1

Paritäts-Fehler (D_4)

Bei freigegebener Parität wird dieses Bit für die Zeichen gesetzt, deren Parität nicht mit der programmierten (gerade/ungerade) übereinstimmt. Das Bit wird gespeichert, so daß wenn ein Fehler auftritt, es gesetzt bleibt, bis der Fehler-Rücksetz-Befehl (WRO) gegeben wird.

Empfänger-Überlauffehler (D_5)

Dieses Bit zeigt an, daß mehr als drei Zeichen empfangen wurden, ohne daß sie von der CPU gelesen worden sind. Nur das Zeichen, welches überschrieben worden ist, wird als fehlerhaft gekennzeichnet, aber wenn dieses Zeichen gelesen wird, bleibt die Fehlerbedingung gespeichert, bis eine Rücksetzung durch den Error-Reset-Befehl erfolgt. Wenn der statusabhängige Vektor freigegeben ist, unterbricht das Zeichen, das übergelaufen ist, mit einem speziellen Empfangsbedingungs-Vektor.

CRC/Rahmenfehler (D_6)

Wenn ein Rahmenfehler auftritt (asynchrone Betriebsart), wird dieses Bit für das Empfangszeichen gesetzt (aber nicht gespeichert), in welchem der Rahmenfehler aufgetreten ist. Die Erkennung eines Rahmenfehlers bringt eine zusätzliche halbe Bitzeit zur Zeichenzeit. so **wird der Rahmenfehler nicht als neues Startbit interpretiert. In den synchronen und SDLC-Betriebsarten zeigt dieses Bit das Ergebnis des Vergleiches zwischen CRC-Prüfer und dazu**gehörigem Prüfwert an. Dieses Bit wird durch Ausgabe eines Error-Reset-Befehles zurückgesetzt. Das Bit wird nicht gespeichert, so enthält es immer den aktuellen Wert, wenn das nächste Zeichen empfangen wird. Wenn es für CRC-Fehler und Status in den synchronen Betriebsarten benutzt wird, ist es gewöhnlich gesetzt, da die meisten Bitkombinationen zu einem Nicht-Null-CRC führen, ausgenommen für eine korrekte vollständige Nachricht.

Rahmenende (D_7)

Dieses Bit wird nur im SDLC-Betrieb verwendet und zeigt an, daß ein gültiges Endflag empfangen worden ist, und daß CRC-Fehler und Residuen-Code gültig sind. Dieses Bit kann durch Ausgabe des Error-Reset-Befehls zurückgesetzt werden. Es wird auch durch das erste Zeichen des folgenden Rahmens aktualisiert.

Leseregister 2 (Read Register 2) (nur Kanal B)

Dieses Register enthält den Interrupt-Vektor, der in WR2 eingeschrieben ist, wenn das statusabhängige Vektor-Steuerbit nicht gesetzt worden ist. Wenn das Steuerbit gesetzt ist, enthält es den modifizierten Vektor (siehe Abschnitt über statusabhängigen Vektor, WR1). Beim Lesen dieses Registers, wird der Interrupt-Vektor ausgegeben, der durch die zur Zeit höchstpriorisierte Interruptbedingung modifiziert wurde. Liegt keine Interruptbedingung vor, so wird der Vektor mit V_3=0, V_2=1 und V_1=1 modifiziert. Dieses Register kann nur durch Kanal B gelesen werden.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
V_7	V_6	V_5	V_4	V_3	V_2	V_1	V_0
				variabel, wenn statusabhängiger Vektor freigegeben ist			

6. Technische Daten

6.1. Elektrische Kennwerte

GRENZWERTE

bei ϑ_a = 0 bis 70 °C, alle Spannungen bezogen auf U_{ss} = 0 V

Kenngröße	Einheit	Kleinstwert	Größtwert	Bemerkung
Betriebsspannung U_{CC}	V	-0,5	7	
Eingangsspannung U_I	V	-0,5	7	
Betriebstemperaturbereich ϑ_a	°C	0 bis	70	
Lagerungstemperaturbereich ϑ_{stg}	°C	-55 bis	125	
Verlustleistung P	W	-	1,1	bei ϑ_a=25 °C

ZUVERLÄSSIGKEITSWERTE

Prüfausfallrate: $\lambda_{PO,6} \leq 5 \cdot 10^{-5} \cdot h^{-1}$

Betriebszuverlässigkeit (garantiert)

$$\lambda_{BG} \leq 5 \cdot 10^{-6} \cdot h^{-1}$$

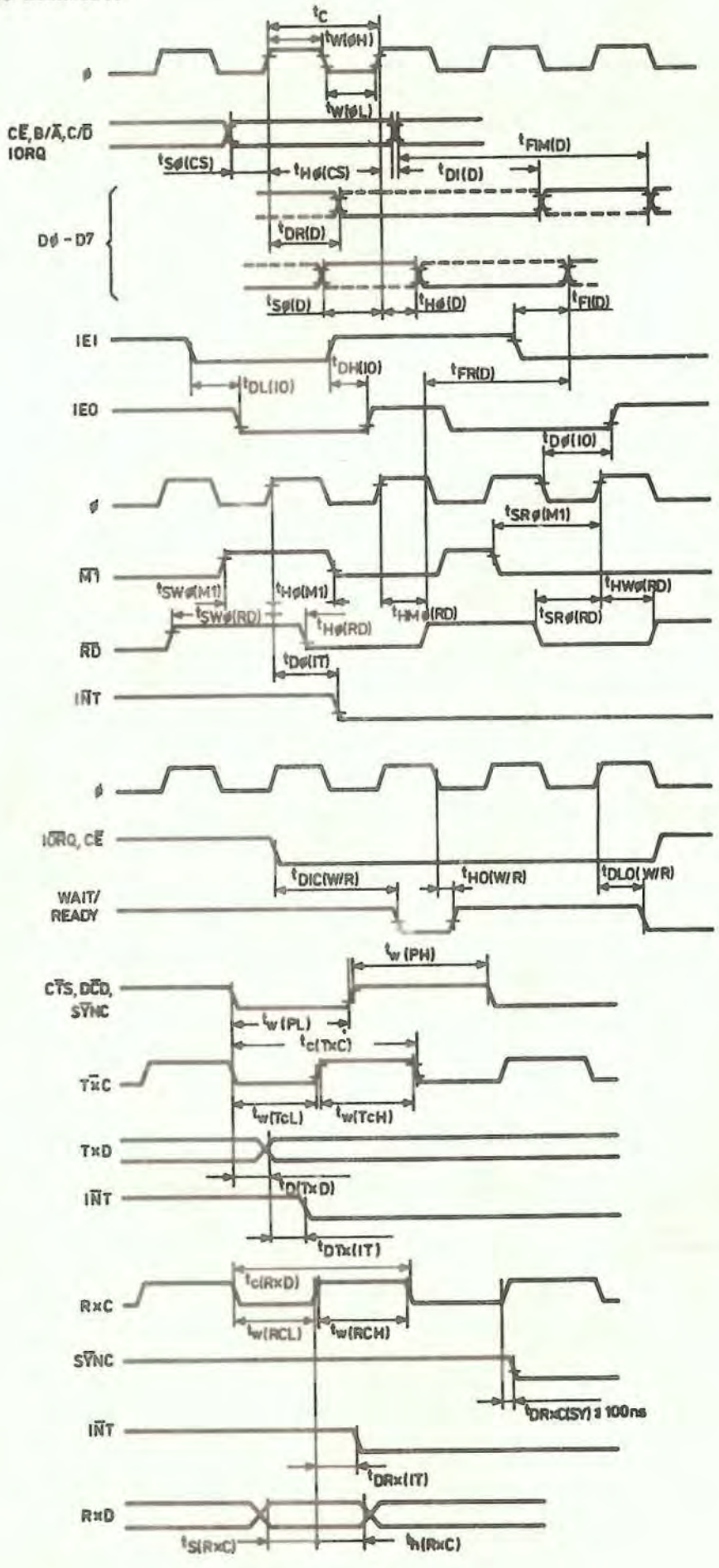
Bei mittlerer elektrischer Belastung
(V_{CC} = 4,75...5,25 V; $\vartheta_a \leq 50$ °C),
normaler klimatischer Beanspruchung (TGL 24 951) und vernachlässigbarer mechanischer Beanspruchung.

STATISCHE KENNWERTE

bei ϑ_a = 0 bis 70 °C

Kenngröße	Kurzzeichen	Einheit	Kleinstwert	Größtwert	Bemerkung
Betriebsspannung	U_{CC}	V	4,75	5,25	-
Eingangsspannung	U_{IL}	V	-0,5	0,8	-
	U_{IH}	V	2	U_{CC}	-
Takteingangsspannung	U_{ILC}	V	-0,5	0,45	-
	U_{IHC}	V	U_{CC}-0,2	U_{CC}	-
Ausgangsspannung	U_{OL}	V	-	0,4	bei I_{OL} = 1,8 mA
	U_{OH}	V	2,4	-	bei $I_{OH} \equiv$ -0,25 mA
Eingangsreststrom	I_{LI}	µA	-	10	
Reststrom der SYNC-Anschlüsse	I_{LSY}	µA	-40	10	
Reststrom INT im hochohmigen Zustand	I_{LINT}	µA	-	10	
Reststrom W/RDY im hochohmigen Zustand	$I_{L\ W/R}$	µA	-	40	
Reststrom des Datenbusses bei Eingabe	I_{LD}	µA	-	10	
Taktkapazität	C_{CP}	pF	-	50	bei ϑ_a = 25 °C
Eingangskapazität	C_I	pF	-	5	und f = 0,5...2MHz
Ausgangskapazität	C_O	pF	-	10	
Stromaufnahme	I_{CC}	mA	-	100	bei U_{CC} = 5,25 V

6.2. Dynamische Kennwerte



Signal	Symbol	Parameter	Einheit	Kleinstwert	Größtwert
"C"	t_c (C)	Taktperiode	ns	400	
	t_w (CH)	Taktimpulsweite High	ns	170	2000
	t_w (CL)	Taktimpulsweite Low	ns	170	2000
	t_r, t_f	Taktanstiegs- und Abfallzeiten	ns	0	30
$\overline{CE}$, B/$\overline{A}$, C/$\overline{D}$, $\overline{IORQ}$	t_H (CS)	Steuersignal-Haltezeit von steigender **C-Flanke**	ns	0	
	t_{SC} (CE)	Steuersignal-Voreinstellzeit **vor** steigender **C-Flanke**	ns	160	
D0-D7	t_{DR} (D)	Daten-Ausgangsverzögerung von steigender **C-Flanke** während **READ-Zyklus**	ns		490
	t_{SC} (D)	Daten-Voreinstellzeit **vor** steigender **C-Flanke** während **WRITE-** oder M1-Zyklus	ns	50	
	t_{HC} (D)	Daten-Haltezeit von steigender **C-Flanke** während **WRITE-** oder $\overline{M1}$-Zyklus	ns	0	
	t_{DI} (D)	Daten-Ausgangsverzögerung von fallender $\overline{IORQ}$-Flanke während INTA-Zyklus	ns		350
	t_{FIM} (D)	Verzögerung bis zum floatenden Bus von steigender $\overline{IORQ}$-Flanke während INTA-Zyklus	ns		240
	t_{FR} (D)	Verzögerung bis zum floatenden Bus von steigender $\overline{RD}$-Flanke während INTA-Zyklus	ns		240
	t_{FI} (D)	Verzögerung bis zum floatenden Bus von fallender IEI-Flanke während INTA-Zyklus	ns		240
IEO	t_{DL} (IO)	IEO-Verzögerungszeit von fallender IEI-Flanke	ns		210
	t_{DH} (IO)	IEO-Verzögerungszeit von steigender IEI-Flanke	ns		210
	t_{DO} (IO)	IEO-Verzögerungszeit von fallender $\overline{M1}$-Flanke (wenn Interrupt genau vor $\overline{M1}$-Flanke erfolgte)	ns		310
M1	t_{SWC} (M1)	$\overline{M1}$-Voreinstellzeit **vor** steigender **C-Flanke** wänrend READ- oder WRITE-Zyklus	ns	210	
	t_{SRC} (M1)	$\overline{M1}$-Voreinstellzeit **vor** steigender **C-Flanke** während INTA- oder M1-Zyklus	ns	210	
	t_{HC} (M1)	$\overline{M1}$-Haltezeit von steigender Flanke	ns	0	
RD	t_{SRC}	**RD-Voreinstellzeit vor steigender C-Flanke**	ns	240	

Signal	Symbol	Parameter	Einheit	Kleinstwert	Größtwert
$\overline{INT}$	$t_{DRx(IT)}$	$\overline{INT}$-Verzögerungszeit von der Mitte des Empfangsdatenbits	C-Perioden	10	13
	$t_{DTx(IT)}$	$\overline{INT}$-Verzögerungszeit von der Mitte des Sendedatenbits	C-Perioden	5	9
	$t_{DC\ (IT)}$	$\overline{INT}$-Verzögerung von der steigenden **C-Flanke**	ns		210
$\overline{WAIT}$/ READY	$t_{DIC(W/R)}$	$\overline{W}$/R-Verzögerungszeit von $\overline{IORQ}$ oder $\overline{CE}$ im $\overline{WAIT}$-Mode	ns		190
	$t_{DHC\ (W/R)}$	$\overline{W}$/R-Verzögerungszeit von der fallenden **C-Flanke**, $\overline{WAIT}$/READY-High, $\overline{WAIT}$-Mode	ns		160
1)	$t_{DRx(W/R)}$	$\overline{W}$/R-Verzögerungszeit von der Mitte des Empfangsdatenbits, **READY-Mode**	C-Perioden	10	13
	$t_{DTx(W/R)}$	$\overline{W}$/R-Verzögerungszeit von der Mitte des Sendedatenbits, **READY-Mode**	C-Perioden	5	9
	$t_{DLC\ (W/R)}$	$\overline{W}$/R-Verzögerungszeit von der steigenden **C-Flanke**, $\overline{WAIT}$/ READY-Low, **READY-Mode**	ns		130
$\overline{CTSX}$, $\overline{DCDX}$, $\overline{SYNCX}$	$t_{W\ (RH)}$	Minimale Impulslänge zum Einspeichern des Status im Register und Erzeugen eines Interrupts	ns	200	
	$t_{W\ (PL)}$	Minimale Low-Impulslänge zum Einspeichern des Status im Register und Erzeugen eines Interrupts	ns	200	
2)					
$\overline{SYNCX}$	$t_{DL\ (SY)}$	Sync-Impuls-Verzögerungszeit von der Mitte des Empfangsdatenbits, Ausgang	C-Perioden	4	7
	$t_{SL\ (SY)}$	Sync-Impuls-Voreinstellzeit von der steigenden RxC-Flanke, Eingang (Ext.-Sync-Mode)	ns	100	
2)	$t_{W\ (SY)}$	SYNC-Impulslänge (Ext.-Sync-Mode)	RxC-Perioden	1	
$\overline{TxCX}$	$t_{C\ (TxC)}$	Sendertaktperiode	ns	400 3)	∞
	$t_{W\ (TCH)}$	Sendertaktimpulsweise High	ns	180	∞
2)	$t_{W\ (TCL)}$	Sendertaktimpulsweite Low	ns	180	∞
$\overline{TxDX}$	$t_{D\ (TxD)}$	TxD-Ausgangsverzögerung von fallender TxC-Flanke (x1-Taktfaktor)	ns		410
$\overline{RxCX}$	$t_{C\ (RxC)}$	Empfänger-Taktperiode	ns	400 3)	∞
	$t_{W\ (RCH)}$	Empfängertaktimpulsweite High	ns	180	∞
2)	$t_{W\ (RCL)}$	Empfängertaktimpulsweite Low	ns	180	∞

1) Wird die SIO-WAIT-Funktion genutzt, muß $\overline{CE}$, $\overline{IORQ}$, C/$\overline{D}$ und $\overline{M1}$ während der gesamten WAIT-Dauer gültig sein.

2) X = A oder B

3) In allen Modes muß der Systemtakt mindestens das 4,5fache der maximalen Datenrate sein.

Das RESET-Signal muß mindestens einen kompletten System-Taktzyklus aktivieren.

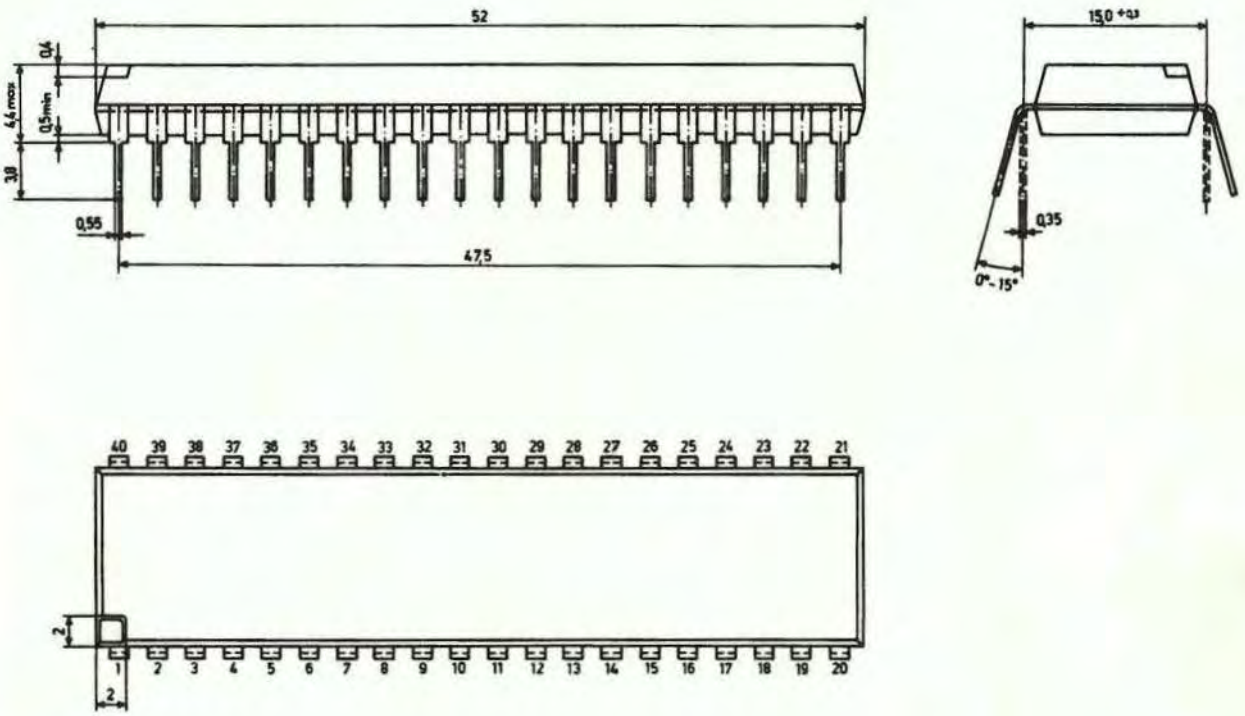
Für alle Verzögerungszeiten gilt: $\vartheta_a = 70\ ^{\circ}C$; $U_{cc} = 4{,}75$ V;
$U_{IL} = 0{,}8$ V; $U_{IH} = 2$ V;
$U_{ILC} = 0{,}45$ V; $U_{IHC} = 4{,}55$ V

Für alle übrigen dynamischen Kennwerte gilt: $\vartheta_a = 0...70\ ^{\circ}C$;
$U_{cc} = 4{,}75...5{,}25$ V

6.3. Gehäuse

Bauform 21.2.3.2.40
nach TGL 26 713

Masse ca. 5,4 g



Rs 01 82–53 V 7 1 2358 KO

elektronik
export·import
VOLKSEIGENER AUSSENHANDELSBETRIEB DER
DEUTSCHEN DEMOKRATISCHEN REPUBLIK
DDR 1026 BERLIN ALEXANDERPLATZ
HAUS DER ELEKTROINDUSTRIE

VEB FUNKWERK ERFURT
im VEB Kombinat Mikroelektronik
DDR - 5010 Erfurt, Rudolfstraße 47
Telefon: 5 80
Telex: 061 306
Kabel: FUNKWERK ERFURT

Rs 01-82-51 W-V-2-1