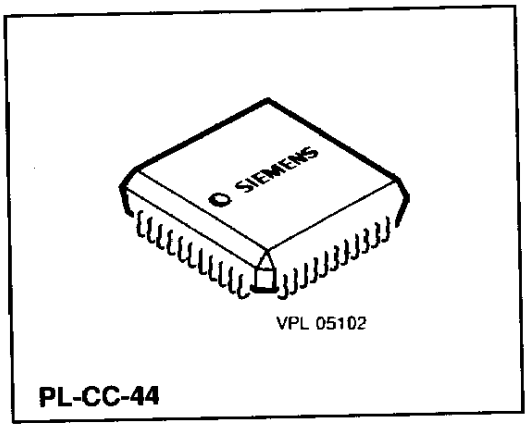
SIEMENS

ISDN Terminal Adapter Circuit (ITAC®)

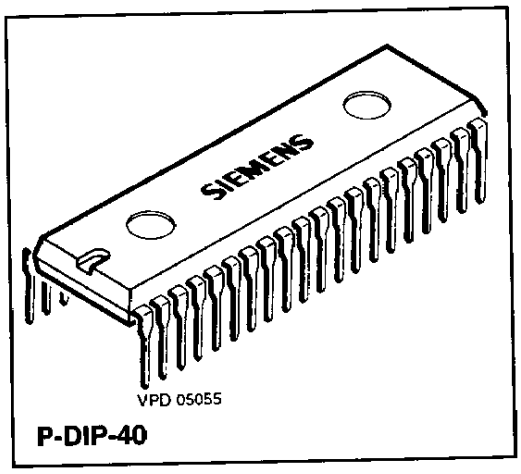
PSB 2110

CMOS IC

1 Features



PL-CC-44



P-DIP-40

DTE Interface

- Support of async and sync interfaces (X.21, X.21 bis, V.24, RS232C, V.35)
- Modem control lines
- Programmable baud rates

Protocol Support

- Bit rate adaptation according to X.30, V.110, ECMA.102
- USART and HDLC controller to support V.120 and DMI applications
- Support of V.120 protocol sensetive and bit transparent mode
- Hayes modem protocol support
- In-band parameter exchange and signaling support

Synchronous Network Interface

- Supports SSI- and IOM®-2-interface for Basic Rate applications
- Supports time-slot oriented interface (up to64-time slots)
- Supports PCM30- and PCM24-interface for Primary Rate applications
- IOM-2 MONITOR channel controller

Microprocessor Interface

- Siemens/Intel multiplexed microprocessor interface
- DMA support for USART or HDLC controller

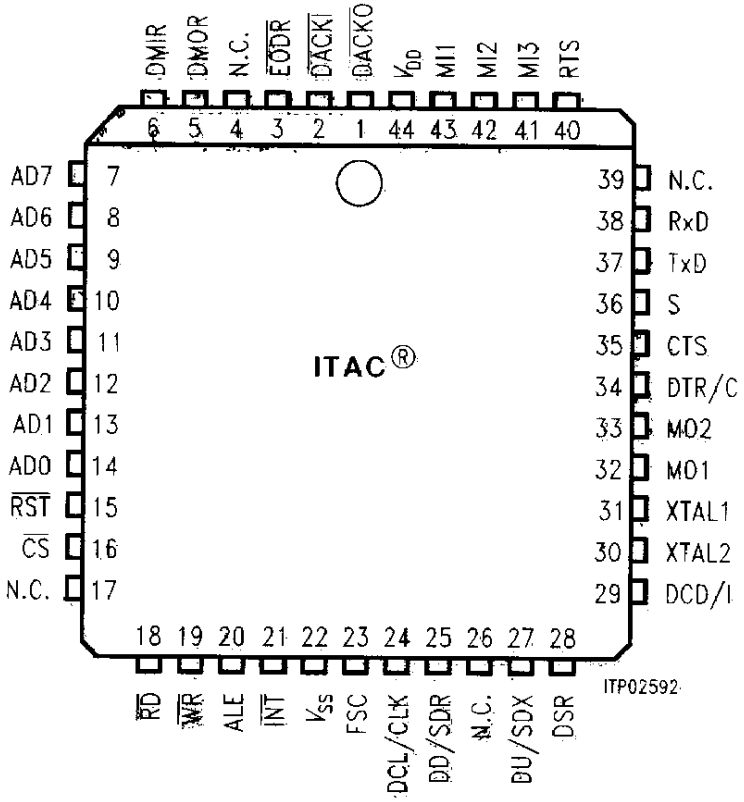
Power Supply

- Single +5 V supply
- Low power CMOS technology
- Power down (standby) mode

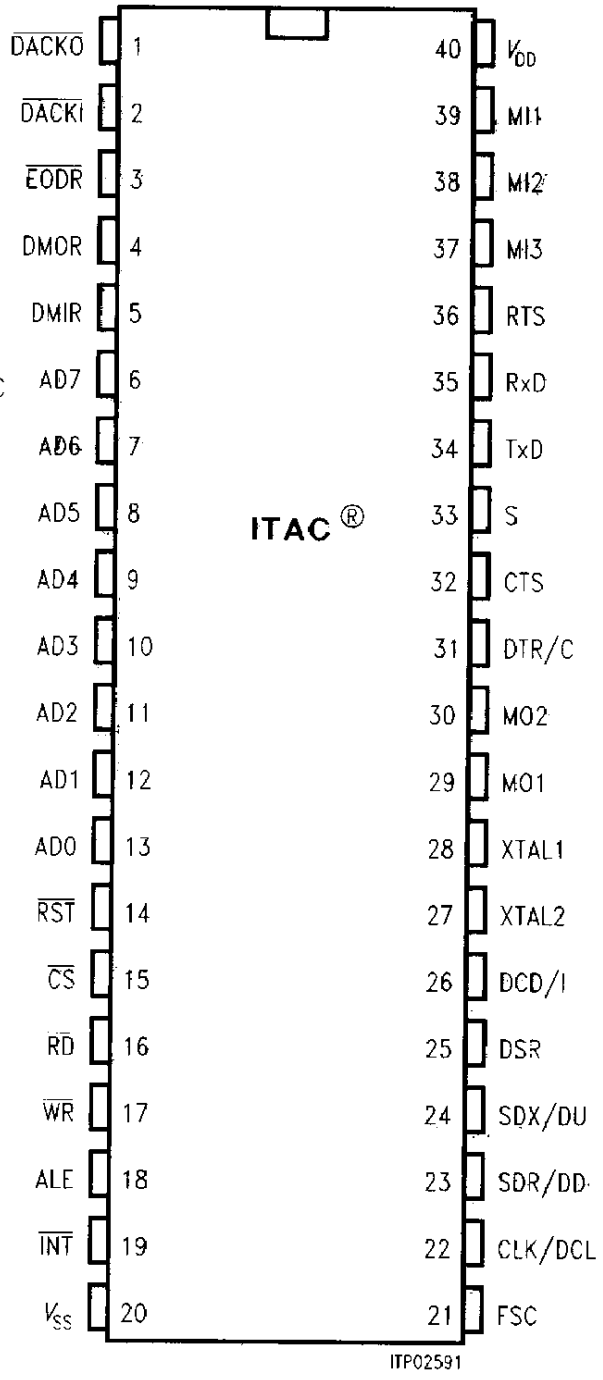
Type	Version	Ordering Code	Package
PSB 2110-N	V2.2	Q67100-H6293	P-LCC-44 (SMD)
PSB 2110-P	V2.2	Q67100-H6294	P-DIP-40

Pin Configurations
(top view)

P-LCC-44



P-DIP-40



1.1 Pin Definitions and Functions

Pin No. P-LCC	Pin No. P-DIP	Symbol	Input (I) Output (O) Open Drain (OD)	Function
14	13	AD0	I/O	**Multiplexed Address/Data Bus.** Transfers addresses and data between the microprocessor and the ITAC.
13	12	AD1	I/O	
12	11	AD2	I/O	
11	10	AD3	I/O	
10	9	AD4	I/O	
9	8	AD5	I/O	
8	7	AD6	I/O	
7	6	AD7	I/O	
16	15	$\overline{CS}$	I	**Chip Select.** A low level on this line selects the ITAC for a read/ write operation.
19	17	$\overline{WR}$	I	**Write.** A low level on this line indicates a write operation to the ITAC.
18	16	$\overline{RD}$	I	**Read.** A low level on this line indicates a read operation from the ITAC.
21	19	$\overline{INT}$	OD	**Interrupt Request.** This signal is active (low) while the ITAC indicates an interrupt to the microprocessor. It is an open drain output and level active.
20	18	ALE	I	**Address Latch Enable.** A transition from high to low latches the contents of the AD1-AD6 lines and use them as the address.
6	5	DMIR	O	**DMA Input Request.** A high level on this line indicates a request for a DMA transfer from memory to the ITAC. It is released by a write operation to one of the corresponding FIFO-addresses.
5	4	DMOR	O	**DMA Output Request.** A high level on this line indicates a request for a DMA transfer from the ITAC to memory. It is released by a read operation to one of the corresponding FIFO-addresses.

Pin Definitions and Functions (cont'd)

Pin No. P-LCC	Pin No. P-DIP	Symbol	Input (I) Output (O) Open Drain (OD)	Function
2	2	$\overline{\text{DACKI}}$	I	**DMA Input Acknowledge.** A low level on this line indicates an acknowledgement to a DMA Input Request. It will select the corresponding FIFO address-space. It will also return the DMIR output to a low level.
1	1	$\overline{\text{DACKO}}$	I	**DMA Output Acknowledge.** A low level on this line indicates an acknowledgement to a DMA Output Request. It will select the corresponding FIFO address-space. It will also return the DMOR output to a low level.
3	3	$\overline{\text{EODR}}$	O	**End of DMA Request.** A low level on this line indicates the end of a DMA transfer. It will go active only for HDLC reception after the last byte of a frame has been read.
15	14	$\overline{\text{RST}}$	I	**Reset.** A low level of at least 2 µs will reset the ITAC.
27	24	SDX/DU	O/OD	**Synchronous Data Transmit/ Data Upstream.** Transmit data output to the network interface.
25	23	SDR/DD	I	**Synchronous Data Receive/Data Downstream.** Receive data input from the network interface.
24	22	CLK/DCL	I	**Clock/Data Clock.** Clock signal on the network interface.
23	21	FSC	I	**Frame Sync.** 8-kHz signal which indicates the start of a new frame on the network interface.
37	34	TxD	I	**Transmit Data from the Terminal.** This line is the data input to the ITAC.
38	35	RxD	O	**Receive Data to the Terminal.** This line is the data output from the ITAC.

Pin Definitions and Functions (cont'd)

Pin No. P-LCC	Pin No. P-DIP	Symbol	Input (I) Output (O) Open Drain (OD)	Function
36	33	S	O/OD	**S Clock.** Bit element timing for the synchronous DCE interface. Data is clocked off on the RxD pin on the falling edge of S. Data on the TxD pin is latched on the rising edge of S. It is tristate for async DCE interface selection.
34	31	DTR/C	I	**Data Terminal Ready/Control.** Data Terminal Ready (108) interchange circuit if V.24 interface is selected. Control interchange circuit if X.21 interface is selected.
29	26	DCD/I	O/OD	**Data Channel Received Line Signal Detector Carrier Detect/Indicate.** DCD (109) interchange circuit if V.24 interface is selected. Indicate interchange line if X.21 interface is selected.
40	36	RTS	I	**Request to Send.** RTS (105) interchange circuit if V.24 interface is selected.
35	32	CTS	O/OD	**Clear to Send.** CTS (106) interchange circuit if V.24 interface is selected.
28	25	DSR	O/OD	**Data Set Ready.** DSR (107) interchange circuit if V.24 interface is selected.
43 42 41	39 38 37	MI1 MI2 MI3	I I I	**Multifunctional Input 1-3.** General purpose inputs if V.24 interface is selected.
32 33	29 30	MO1 MO2	O/OD O/OD	**Multifunctional Output 1-2.** General purpose outputs if V.24 interface is selected.
31	28	XTAL1	I	**Crystal 1.** Connection for the external crystal or input for external clock generator.

Pin Definitions and Functions (cont'd)

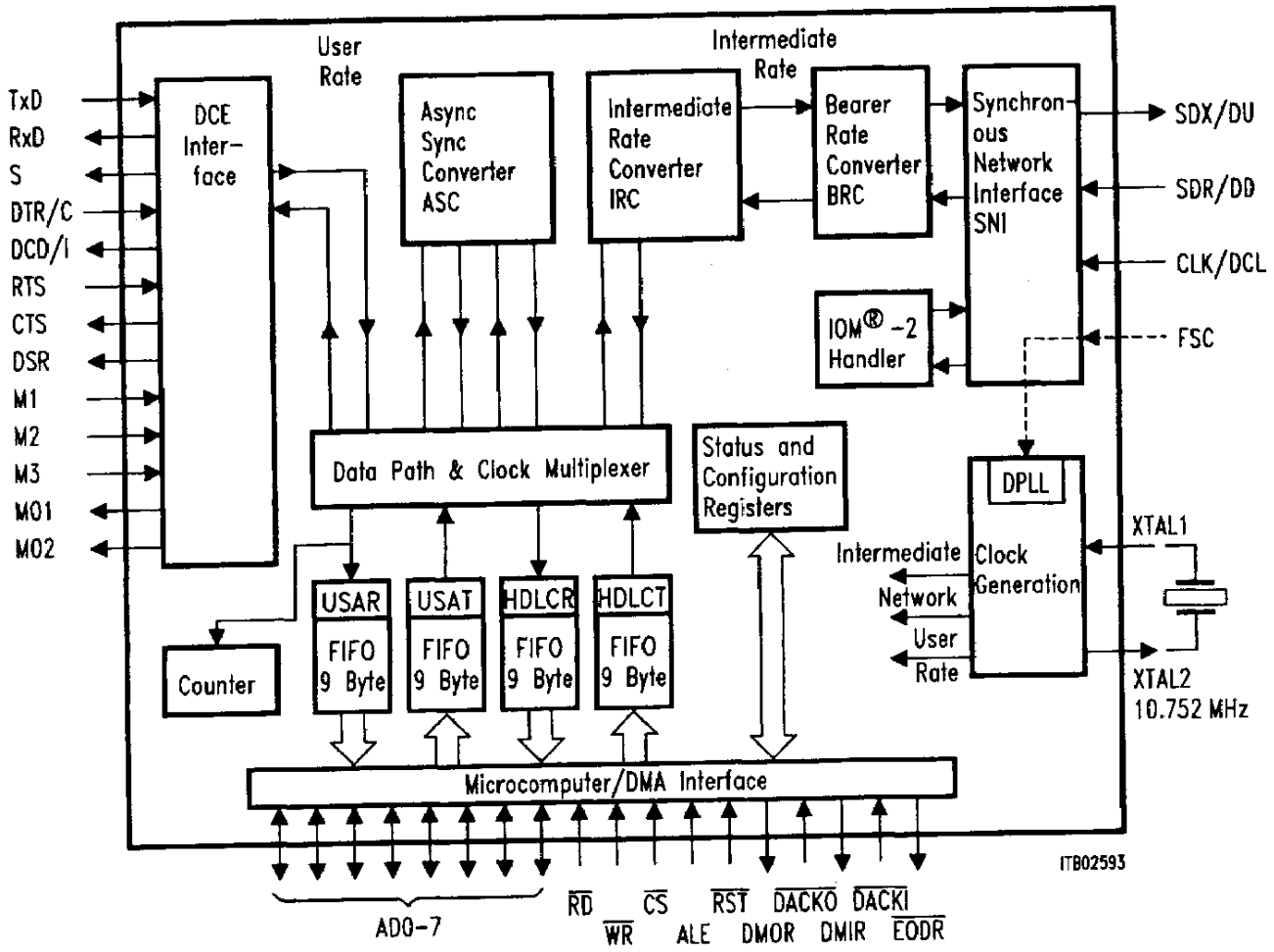
Pin No. P-LCC	Pin No. P-DIP	Symbol	Input (I) Output (O) Open Drain (OD)	Function
30	27	XTAL2	O	**Crystal 2.** Connection for the external crystal. Not connected when external clock generator is used.
44	40	V_{DD}	I	Power supply (+5 V ±5 %)
22	20	V_{SS}	I	Power supply ground (GND)

1.2 Functional Description

1.2.1 General Functions and Device Architecture

Figure 1 shows the block diagram of the ITAC.

Figure 1
Block Diagram



Interfaces

The ITAC has three interfaces. The DCE interface provides all necessary data, interchange and clock signals to form a V.- or X.-series interface. The Synchronous Network Interface is a PCM compatible interface which consists of four lines. It can be configured as IOM-2 compatible and offers additionaly communication paths via the MONITOR channel. The microprocesssor interface is compatible to multiplexed bus systems like Siemens/ Intel CPUs. Control lines may be used to synchronize the data transfer between the serial controllers and the memory via DMA.

Bit Rate Adaptation

Three blocks of the ITAC perform the bit rate adaptation according to the V.110, X.30 or ECMA.102 specification. The first block is the Async/Sync Converter (ASC). It corresponds to the RA0 block and converts an asynchronous data stream received at the user rate into a synchronous data stream at the network rate. It handles overspeed and underspeed conditions and provides support for flow control. The Intermediate Rate Converter (IRC) corresponds to the RA1 block and converts the synchronous data stream into the V.110 frame. The IRC outputs the V.110 frame at the intermediate rate. The Bearer Rate Converter (BRC) corresponds to the RA2 block and converts the data between the IRC and the Synchronous Network Interface (SNI).

Serial Communications Logic

The serial communications logic consists of an USART transmitter, receiver and an HDLC transmitter and receiver plus a programmable counter. The data path for the receivers and transmitters are selected independently. They connect to the DCE interface or the IRC. Each receiver/transmitter has a 9 byte FIFO which reduces the response time of the microcontroller. The data transfer between the FIFOs and memory is performed by the microprocessor or by a DMA controller. The USART is used for in-band signaling (X.21, V.25, Hayes), speed conversion and flag-stuffing adaptation protocols (V.120, DMI). The HDLC controller is also used for flag-stuffing protocols and for general HDLC transfer over the B channels.

Clock Generation

All internal clock signals are derived from a DPLL inside the ITAC. The clock source for the DPLL is the internal oscillator which requires a 10.752-MHz crystal. The DPLL generates the intermediate rate, the network rate and the user rate. The network rate specifies the speed at which the IRC operates while the user rate determines the speed at the DCE interface.

IOM®-2 Handler

The IOM-2 mode, the ITAC supports access to both B channels or the intercommunication channels (IC) for transfering user data. Additionally it can control the MONITOR channel and two C/I bits of the second IOM channel.

Test Loops

The ITAC provides three test loops. Two loop back the data of the IRC at the network side. The third test loop loops the data of the IRC or ASC at the terminal (DCE) side.

1.3 Operating Modes

The ITAC distinguishes between two basic operating modes. In transparent mode the V.110 framing is done autonomously. Therefore the user rate and the network rate have to be the same. In non-transparent modes the elements of the Serial Communications Logic (USART, HDLC controller) are switched into the data path. The user rate and the network rate may be different.

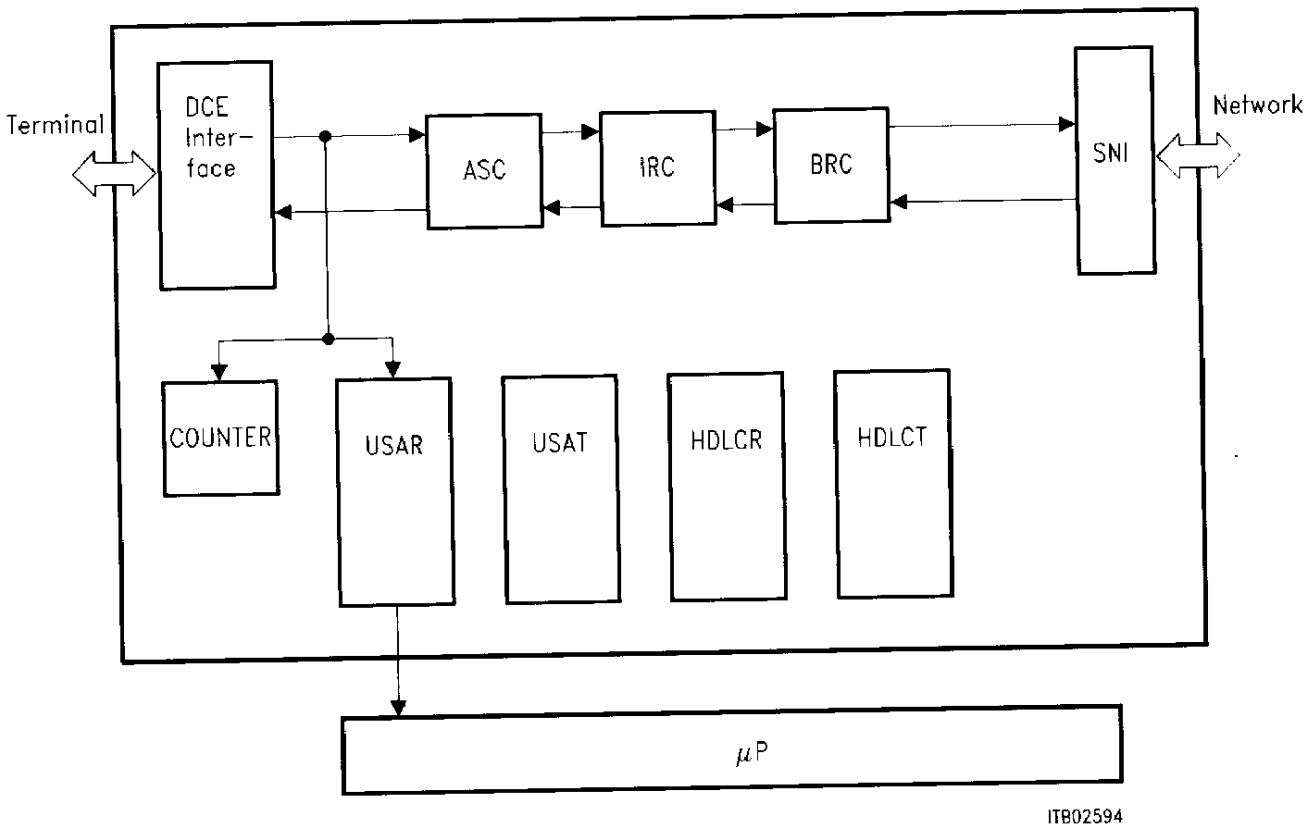
1.3.1 Transparent Mode

In the transparent mode, the ASC (in case of an asynchronous DCE interface), the IRC and the BRC operate autonomously. The microprocessor supervises the status of the IRC and will react only in case of synchronization loss or a disconnect request. To support in-band signaling like the Hayes modem protocol, the USART receiver may be used to monitor the terminal data while the counter is used to supervise the guard time.

The transparent mode is selected by setting the network rate equal to the user rate.

Figure 2 shows the data path for the transparent mode.

Figure 2
Transparent Mode. User Rate = Network Rate



1.3.2 Non - Transparent Modes

All non-transparent modes are characterized by the fact that the user rate and the network rate are not the same or the data source or destination is in memory and not the DCE interface.

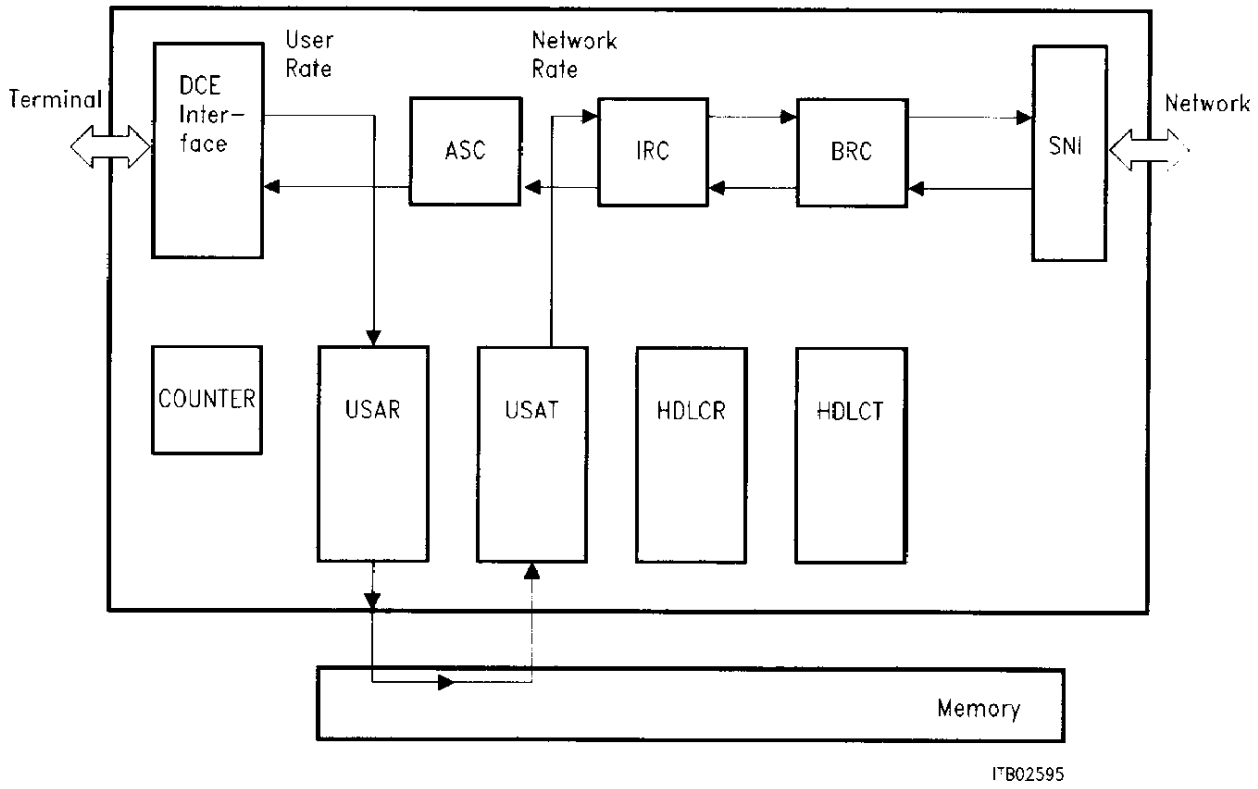
Applications for non-transparent modes are:

- V.110 Adaptation of terminals with unequal speeds
- In-band signaling
- V.120 or DMI Terminal adapters
- Data transfer from or to the local memory

1.3.2.1 Terminal Adapter for Unequal Speeds

Figure 3 shows the data path for a terminal adapter which can handle unequal data rates.

Figure 3
Terminal Adapter for Unequal Speeds



In this figure the user rate is higher than the network rate. In the data path from the lower to the higher data rate, the ASC will add stop bits between the characters to fill the gaps. In the opposite data path, the USART receiver and transmitter are used to convert the data rates. Data from the DCE interface is received by the USART receiver and transfered to memory. The data from memory is than written to the USART transmitter and sent to the IRC at the lower data rate.

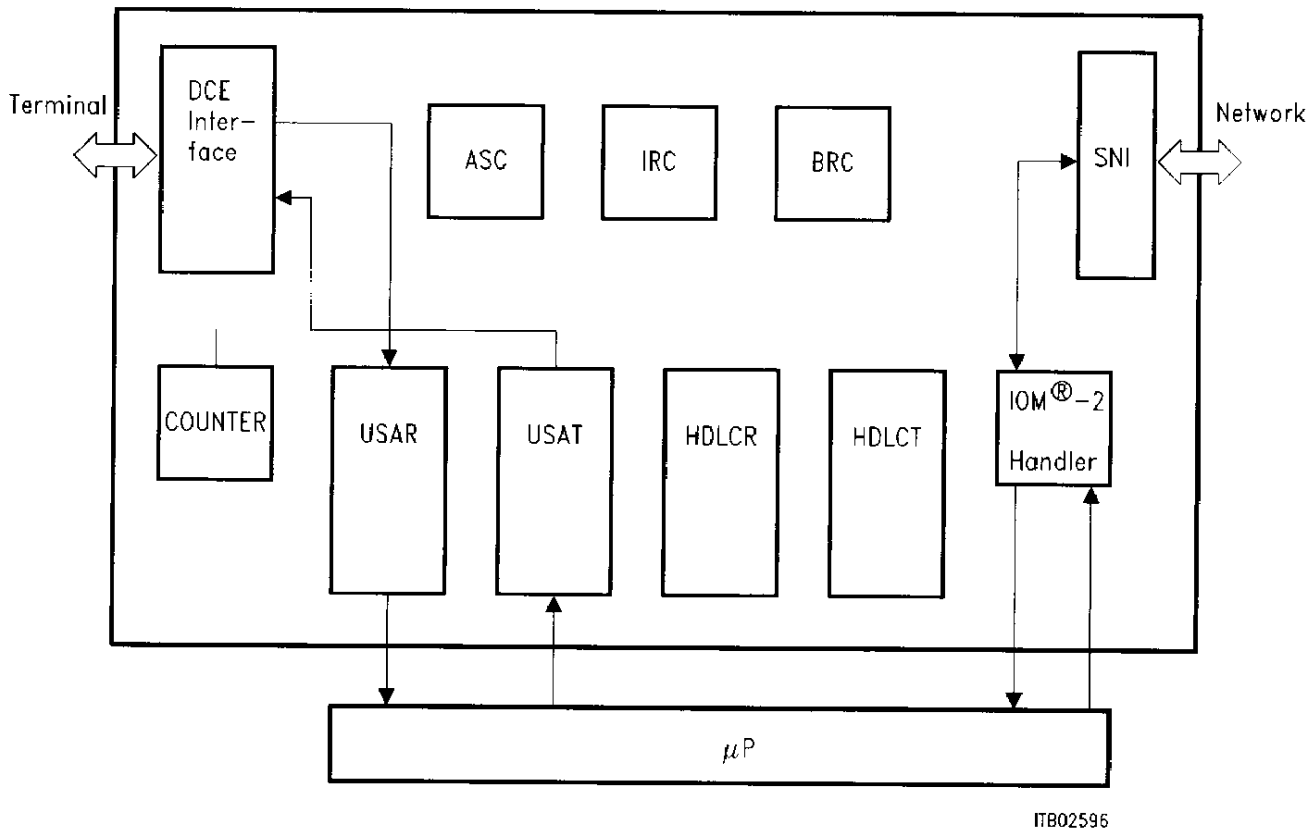
Flow control between both ends is performed by interchange circuits or by using flow control characters. To detect these flow control characters (e.g. XON/XOFF) the ASC can compare each received character with two register values. If a match occurs, an interrupt is generated and the character may be deleted from the data stream.

1.3.2.2 In-Band Signaling

Another non-transparent mode is the handling of in-band signaling information (e.g. X.21, Hayes modem protocol). The USART receiver and transmitter are used during local mode to transfer signaling information between the terminal and the terminal adapter.

Figure 4 shows the data path for in-band signaling.

Figure 4
Data Path for In-Band Signaling (e.g. Hayes modem protocol)

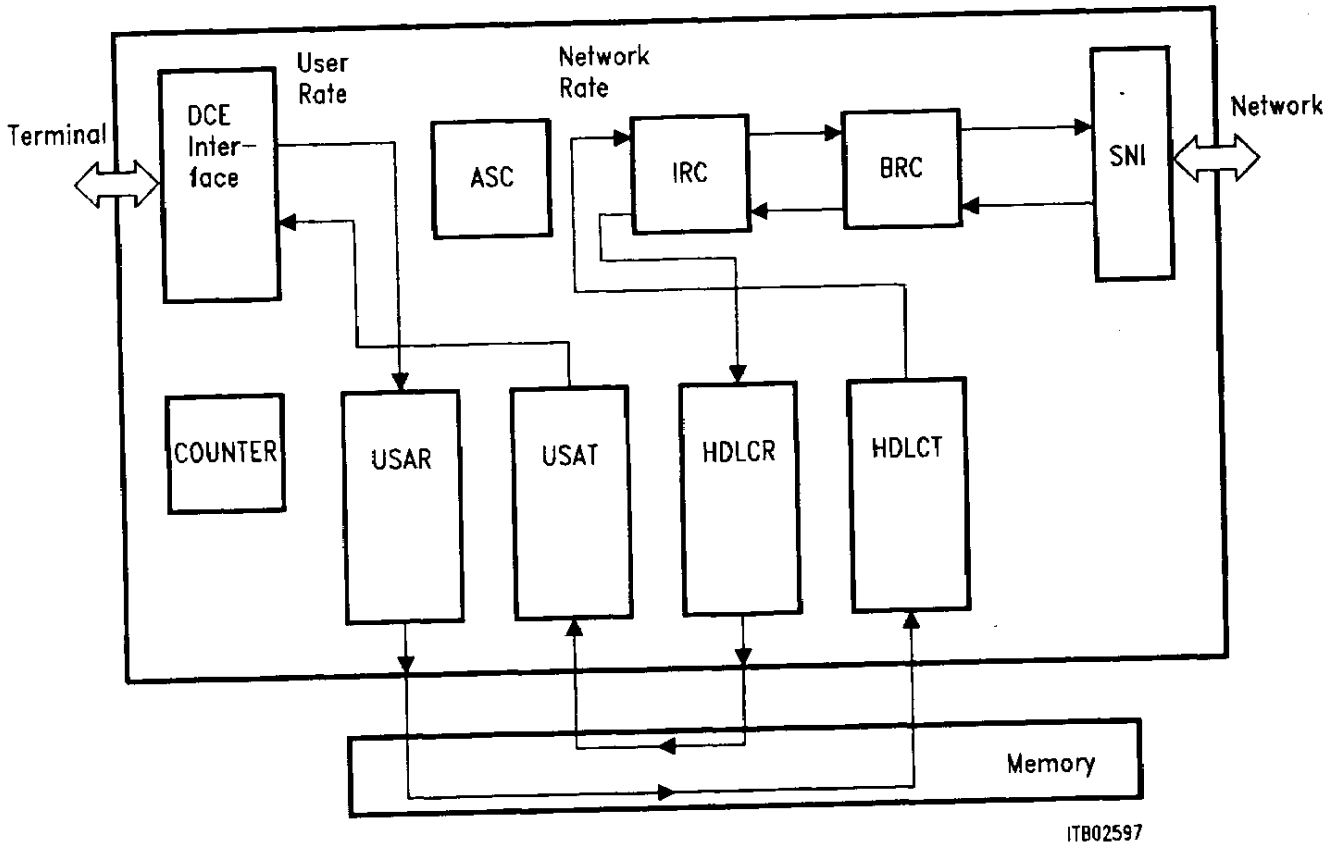


All data from the terminal is received by the USART receiver and transfered to the microprocessor. The microprocessor interprets the data and transmits responses to the terminal using the USART transmitter. If the microprocessor which operates with the ITAC is different from the one which performs the LAPD signaling protocol, the MONITOR channel 1 of the IOM-2 interface can be used for intercommunication.

1.3.2.3 Terminal Adapter for V.120 or DMI

A terminal adapter for V.120 or DMI will also use the non-transparent mode. **Figure 5** shows the data path.

Figure 5
Terminal Adapter for V.120 or DMI

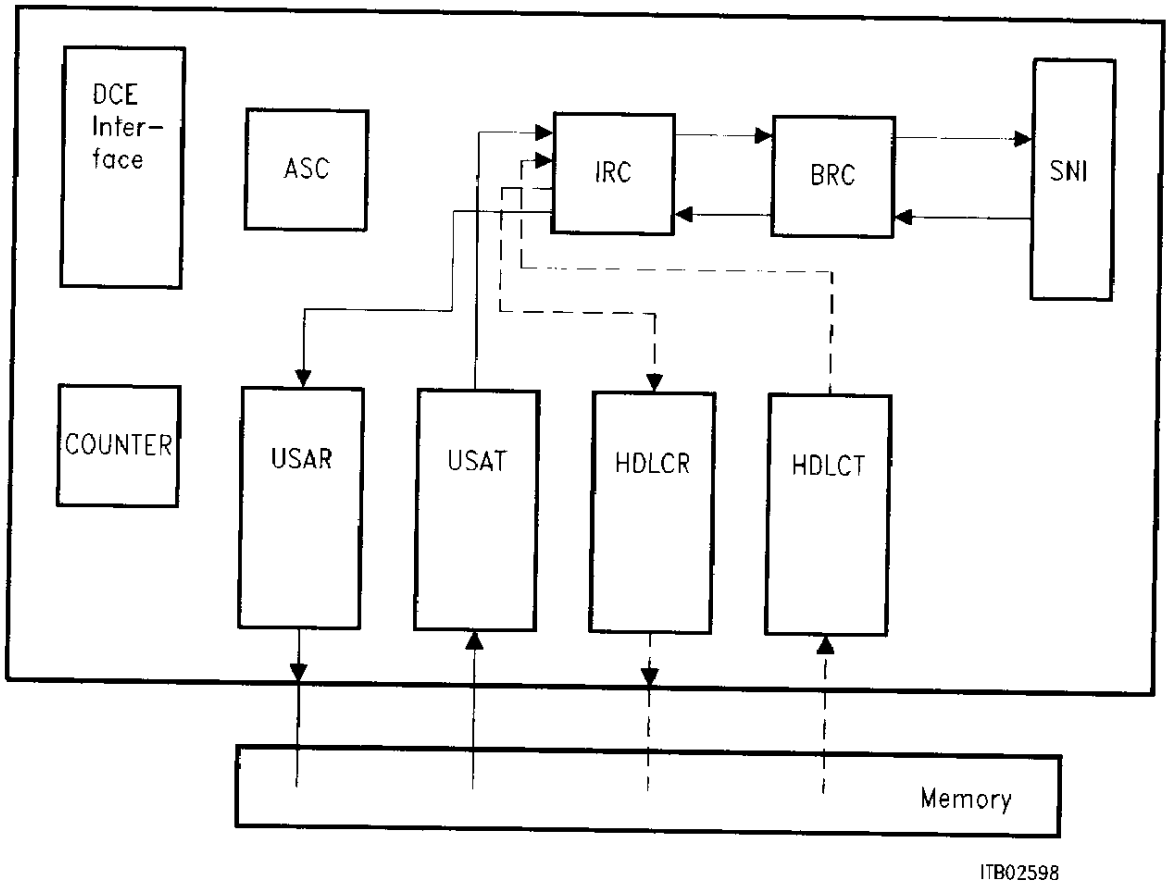


All data from the terminal is received by the USART receiver and transfered to memory. The microprocessor adds the V.120 control elements and writes the data to the HDLC transmitter. The HDLC transmitter formats the HDLC frame and sends it to the IRC. In receive direction, the HDLC receiver receives a frame and transfers the data to memory. The microprocessor extracts the user data and writes it to the USART transmitter FIFO. The USART transmitter sends the characters to the terminal. Depending on the selected network rate, 7 bits (restricted 56 kbit/s) or 8 bits (64 kbit/s) of the B channel are used to transfer the HDLC information.

1.3.2.4 Data Transfer from or to the Local Memory

If the data which has to be transfered over the network doesn't originate from the DCE interface but from local memory, another non-transparent mode is used. **Figure 6** shows the data path.

Figure 6
Data Transfer from or to the Local Memory



The data from memory is written to the USART transmitter which sends it to the IRC. The IRC adapts the data and outputs it to the SNI. The data from the network is received by the IRC. The user data is forwarded to the USART receiver and transfered to memory. The same data path may be selected for the HDLC blocks to transfer HDLC frames formatted in V.110 frames or transparent over the B channel.

1.4 System Integration

The ITAC may be used in a number of applications, where terminals with V- or X-series interfaces are connected to an ISDN.

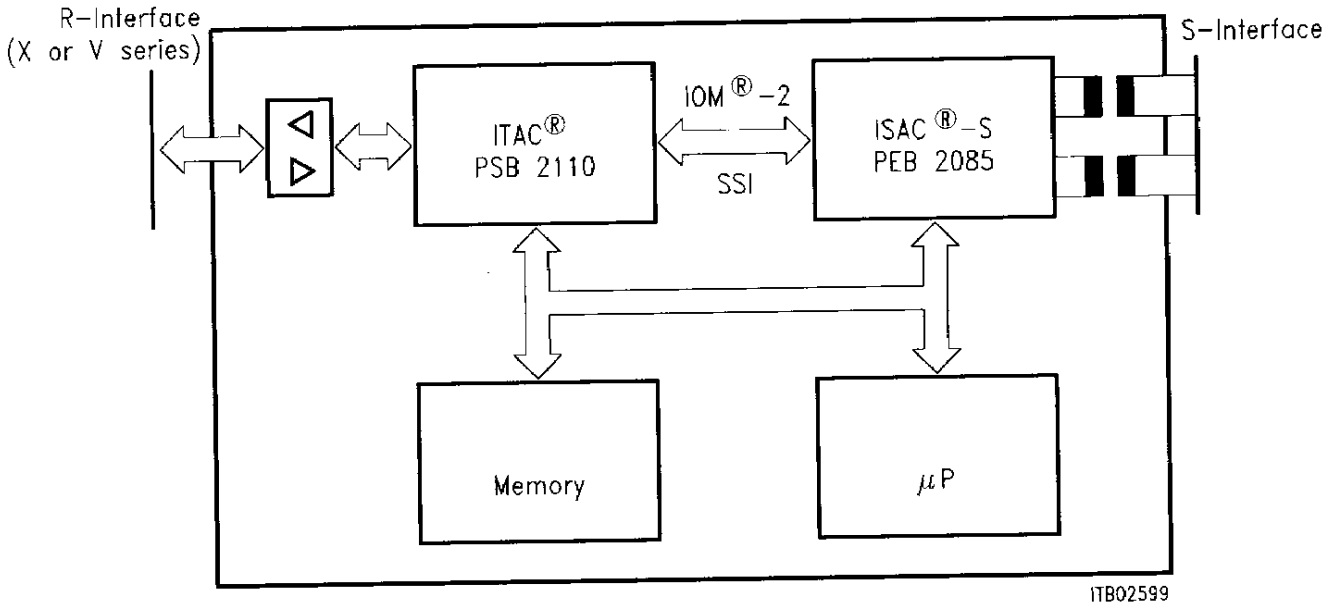
These applications are:

- V- or X- series terminal adapters for basic access
- V- or X- series terminal adapter add-on modules to feature phones
- PC add-on cards as terminal adapters or basis for ISDN terminals
- V- or X- series terminal adapters in exchanges
- Modem pools

1.4.1 Terminal Adapter for ISDN Basic Access

A typical implementation of an ISDN basic access for a conventional X- or V- series terminal using the ITAC is shown in **figure 7**.

Figure 7
Terminal Adapter for ISDN Basic Access



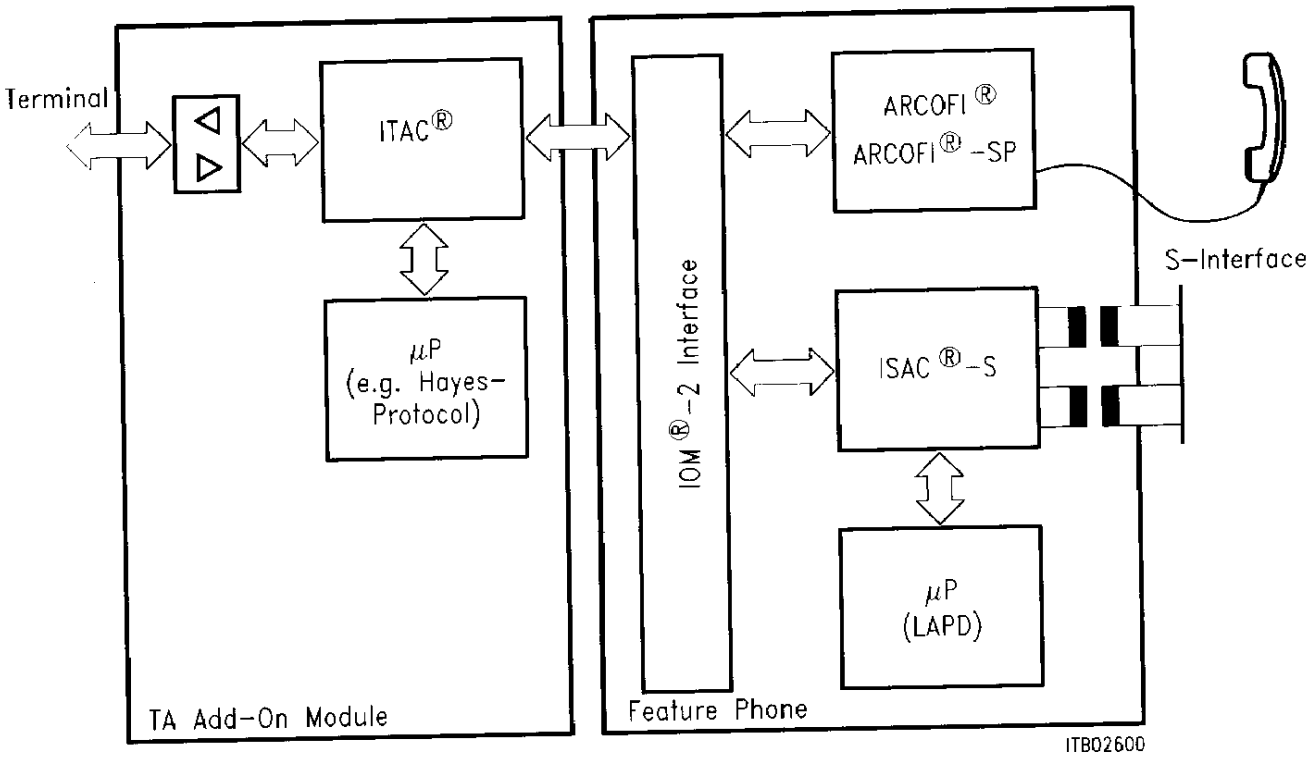
The ITAC can be connected via a serial synchronous interface (e.g. IOM-2, SSI) to an ISDN basic access transceiver/LAPD controller (in this case, the ISDN Subscriber Access Controller for the S interface, ISAC® -S). These two devices, together with a microcontroller, convert V- and X-series interface characteristics to the functional and procedural interface characteristics required by an ISDN at reference point S.

In this configuration, the ITAC will do the rate adaptation according to V.110, X.30 or ECMA.102 autonomously. The ITAC together with the microcontroller is used to receive and transmit signaling information between the terminal and the terminal adapter. The microcontroller together with the ISAC-S will convert the signaling data and transfer them using LAPD on S-interface. In V.120 and DMI terminal adapters the ITAC together with the microprocessor will perform the rate adaptation.

1.4.2 Terminal Adapter Add-On Modules for Feature Phones

A modular design of a feature phone may contain of a basic voice board with a microcontroller and an S-interface circuitry. Via the IOM-2 interface additional modules like encryption modules, card readers or terminal adapter modules may be added. Such a configuration is shown in **figure 8**.

Figure 8
Feature Phone Plus Terminal Adapter Add-On Module Based on IOM®-2

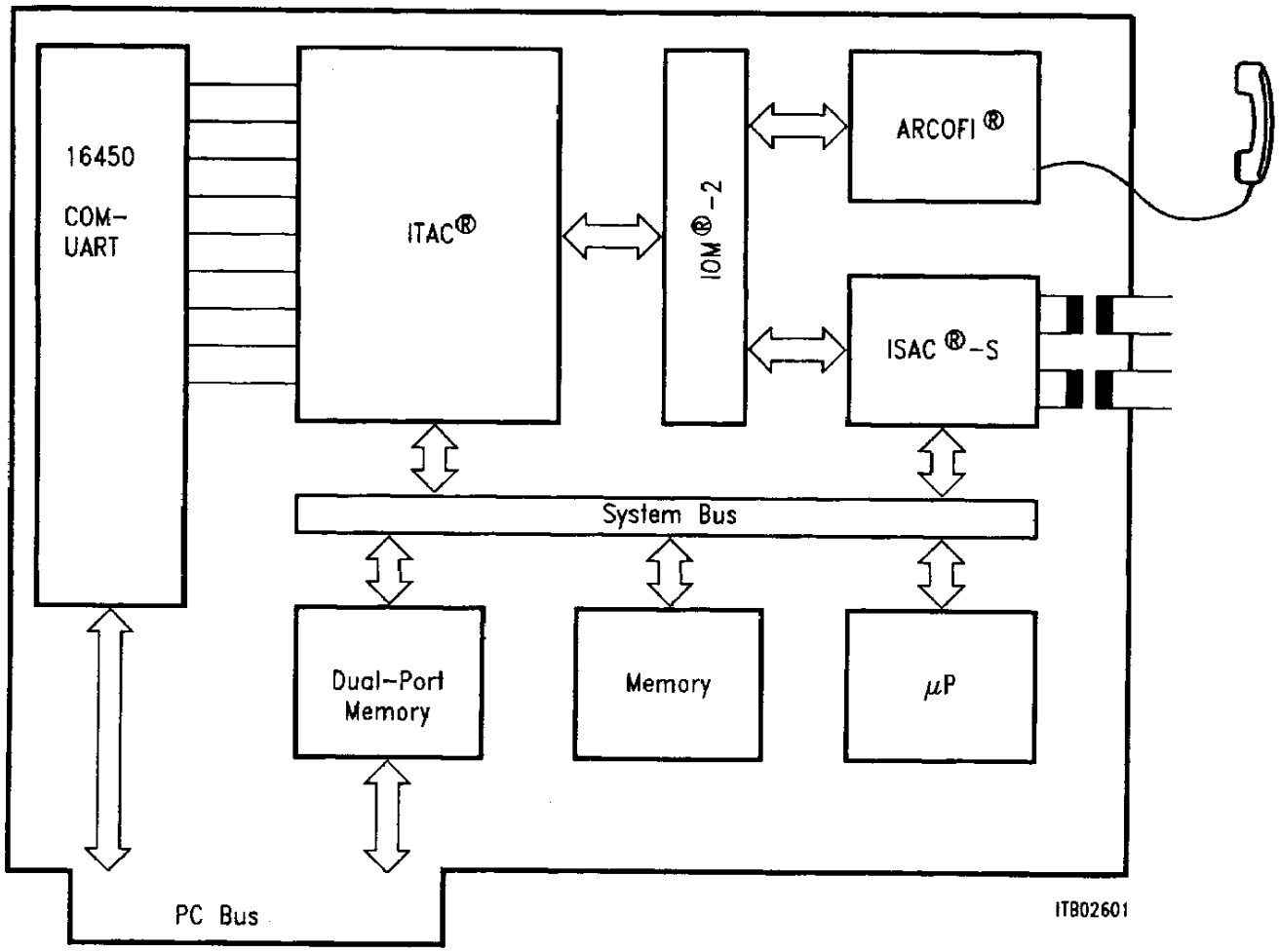


The microcontroller on the feature phone performs the telephone functions plus the processing of signaling information according to the LAPD protocol. The microcontroller on the TA add-on module performs the DTE signaling protocol (V.25 bis, Hayes) and controls the ITAC. Data between both microcontrollers is exchanged via the MONITOR channel 1. This data includes telephone numbers, call status information as well as B-channel assignment.

1.4.3 PC Add-On Cards

Add-on cards for IBM PCs or compatibles based on the ITAC are not just simple terminal adapters but turn the PC into an ISDN terminal. A standard design is shown in **figure 9**.

Figure 9
Add-On Cards for PCs

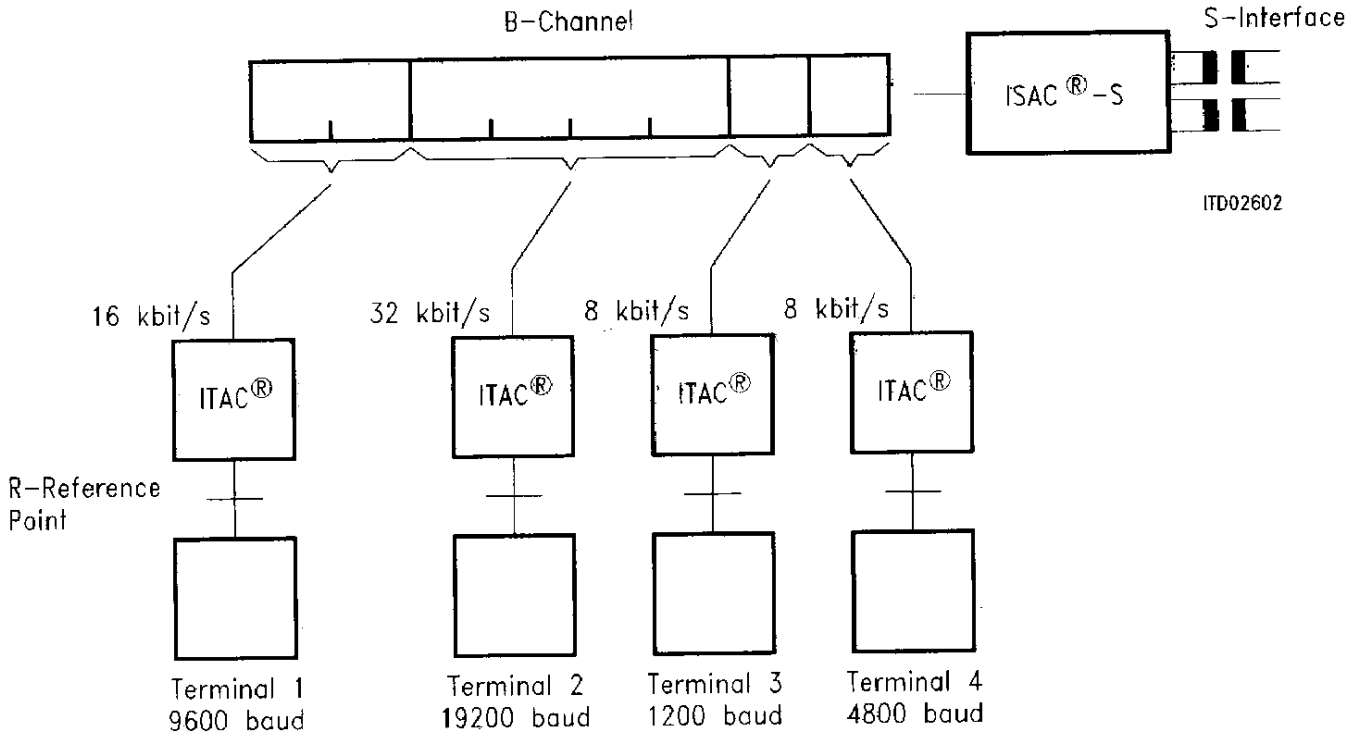


Universal programs use the COM-Port to transfer data. The ITAC operates just like a terminal adapter. For card-specific application programs the Dual-Port Memory is used to transfer data between the PC Bus and the ISDN card. The ISDN card will transfer this data using the build-in communications controllers of the ITAC. A card like this is able to adapt the COM-Port according to the standards V.110, V.120 or DMI mode II. Additionally it can be used to access X.25 packet data services and it can transfer group IV facsimile data.

1.4.4 Modem Pools

The ITAC supports V.110 subchannel multiplexing. This means that the 64 kbit/s bearer channel is shared by up to eight independent terminals. This is illustrated in **figure 10**.

Figure 10
Sharing of Bearer Channel Among Serveral Independent DTEs



1.4.5 Other Applications

The synchronous network interface of the ITAC is compatible to most PCM systems using programmable time slots. Consequently, the circuit in association with all IOM-2 compatible circuits, is ideally suited for applications on PABX line cards and concentrators/ multiplexers.

Other applications of the ITAC include: host computer multiple line communication couplers, primary access/DMI peripheral boards, and Interworking Units (IWUs) between ISDN and analog PSTN.

2 Operational Description

2.1 Microprocessor/DMA Interface

Microprocessor Interface

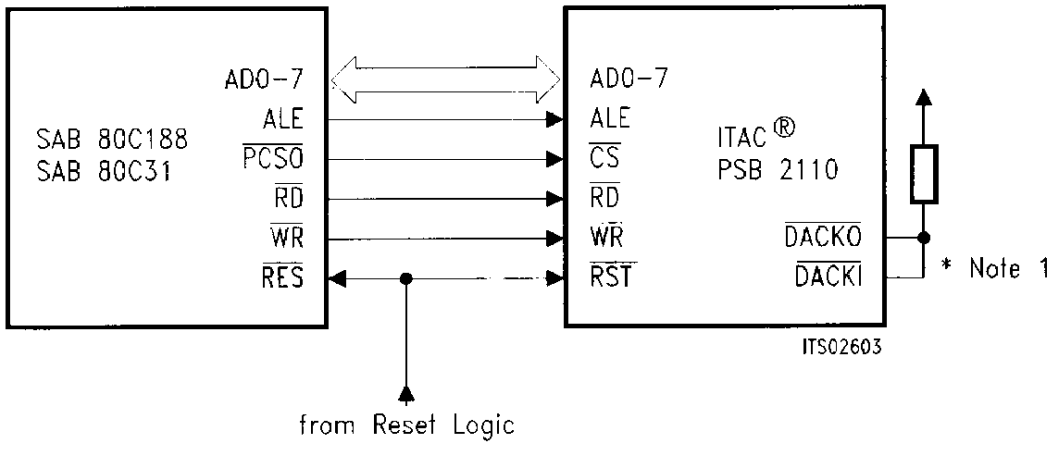
The microprocessor interface of the ITAC is designed for multiplexed address/data bus systems. The contents of the address/data lines is latched on the falling edge of ALE to form the register address. After chip select (CS) becomes active, the read or write control line specifies the register access. If read becomes active, the contents of the selected register is output on the address/data lines. If write becomes active, the contents of the address/data lines is latched into the selected register.

The address decoder uses address lines AD1 to AD6. Address line AD0 is not evaluated to allow a direct interface to 16-bit microprocessors like the SAB 80C186. The SAB 80C186 accesses the ITAC registers via even addresses. Address line AD7 is also not evaluated so that the occupied address space ranges from 00H to 7FH and another device may use the address space from 80H to FFH.

The DMA acknowledge inputs have to be connected to V_{DD} if they are not used.

Figure 11 shows the connection of an ITAC to a SAB 80C188/80C31 microprocessor without DMA support.

Figure 11
Connecting the ITAC® to SAB 80C188/80C31



* Note 1: if DMA acknowledge inputs not used

DMA Interface

The ITAC supports data transfer between the FIFOs and memory via DMA (Direct Memory Access). If the ITAC requires a data transfer, it will indicate this to the DMA controller via a DMA request signal (DMIR, DMOR). The DMA controller will than signal to the CPU that it requires the control over the address and data bus. After the CPU responded by issuing an acknowledge signal, the DMA controller starts to transfer data.

DMA requests generated by the ITAC are acknowledged by the proper read or write operation to the FIFO. The read or write operation can be performed in two ways.
The first way is to perform a standard CPU cycle on one of the FIFO addresses. In this case, both DMA acknowledge inputs have to be connected to V_{DD}.
The second way uses the DMA acknowledge inputs to select the FIFO address instead of a CS-signal. The RD or WR control signal together with the proper DMA acknowledge input selects the data bus operation. This allows single cycle I/O to memory transfers.

Both possible acknowledgements are shown in **table 1**.

Table 1
DMA Acknowledgements

Request	Acknowledgement	Operation
DMIR	$\overline{\text{DACKI}}$ • $\overline{\text{WR}}$ or $\overline{\text{CS}}$ (FIFO) • $\overline{\text{WR}}$	DMA transfer from memory to ITAC FIFO
DMOR	$\overline{\text{DACKO}}$ • $\overline{\text{RD}}$ or $\overline{\text{CS}}$ (FIFO) • $\overline{\text{RD}}$	DMA transfer from ITAC FIFO to memory

The EODR output indicates that the last byte of a frame has been transfered by the DMA controller. It reflects the inverted status of the RME bit if the DMA interface is connected to the HDLC controller.

The generation of DMOR requests is blocked until the RME status bit has been acknowledged regardless whether the RME interrupt enable bit is set or cleared.

2.2 DCE Interface

The DCE interface consists of 13 lines in total. The number of available signals and their meaning depends on the selected clock mode (synchron or asynchron) and on the selected interchange configuration. The DCE interface may either operate synchron or asynchron. The selection is done by the 'ASY' bit in the General Configuration Register (GCR).

2.2.1 Operating Modes

Synchronous Operation (GCR:ASY = 0)

During the synchronous operation, the clock signal which synchronizes the data transfer is outputed on the S-line. The data rate is equal to the selected user rate (BRS:UR0-3). The data on RxD is clocked off on the falling edges of the S-clock while data on TxD is latched on the rising edge of S. The ASC is inactive.

Asynchronous Operation (GCR:ASY = 1)

No clock signal is available on the S-line during asynchronous operation. In receive direction, the internal clock (16 * user rate) is synchronized on every falling edge of a start bit or every falling edge of the received data stream (bit transparent mode). The TxD is sampled in the middle of each bit (after 8 internal clocks). Data on RxD is transmitted with no synchronization to the received data stream. The ASC is active.

2.2.2 Interchange Circuits

The meaning of the interchange circuits and the corresponding status detection logic depends on the selected interface type. If the X.21 interface is selected, only C (Control) and I (Indicate) interchange lines are available. The status detection logic reports the recognized combination of local (TxD,C). If a non-X.21 interface (V-series interface) is selected, 10 interchange circuits are available. Their current status can be read from or written into registers, or controlled by the S-bits of the V.110 frame.

X.21 Interchange Circuits (GCR:V24 = 0)

The status detect logic samples the received data on TxD and the status of the control line (C). It reports the current status combination via the LDS (Local DTE Status) and the ELDS (Extended Local DTE Status) registers. A change in the LDS register is indicated by the LDC bit in the IST (Interrupt Status) register. A change in the ELDS register is indicated by the ELDC bit in the IST register. Both bits may generate an interrupt if their corresponding bit in the ISEN (Interrupt Status Enable) register is set to one.

The status detection logic is in no-state after reset. 'No state' is indicated by a value of zero in both status registers (LDS, ELDS). The status detection logic starts to sample the TxD and C line on every rising edge of the S-clock signal. The sampled values are compared against the different possibilities after 16 samples were taken. If a match is detected, the LDS and ELDS register change to the new value and set the corresponding LDC or ELDC bit to '1'. This may generate an interrupt. If a status is detected, which is different from LONX or LOFX, the status detection logic will sample four bits before the pattern is varified. If the status is no longer true, the LDS and ELDS registers change to 'no-state' (00) and the corresponding LDC or ELDC bit will be set. The next status will be indicated after another 16 bits have been sampled.

If LONX or LOFX or no valid status is detected after 16 samples, the status detection logic will check its last 16 samples after every clock cycle.

The valid status are shown in **table 2**.

Table 2
DTE Status Logic Indications

Status	Status Name	Sampled Data	Register Bit
LON0	Local (0, ON)	TxD 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 C 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	LDS : Bit 7
LOF01	Local (0101, OFF)	TxD 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 C 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	LDS : Bit 6
LL3	Local Loop 3	TxD 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 C 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	LDS : Bit 5
LONX	Local (X, ON)	TxD X X X X X X X X X X X X X X X X C 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	ELDS : Bit 7
LOFX	Local (X, OFF)	TxD X X X X X X X X X X X X X X X X C 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	ELDS : Bit 6
LON1	Local (1, ON)	TxD 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 C 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	ELDS : Bit 5
LOF1	Local (1, OFF)	TxD 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 C 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	ELDS : Bit 4
LOF0	Local (0, OFF)	TxD 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 C 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	ELDS : Bit 3
LL2	Local Loop2	TxD 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 C 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	ELDS : Bit 2

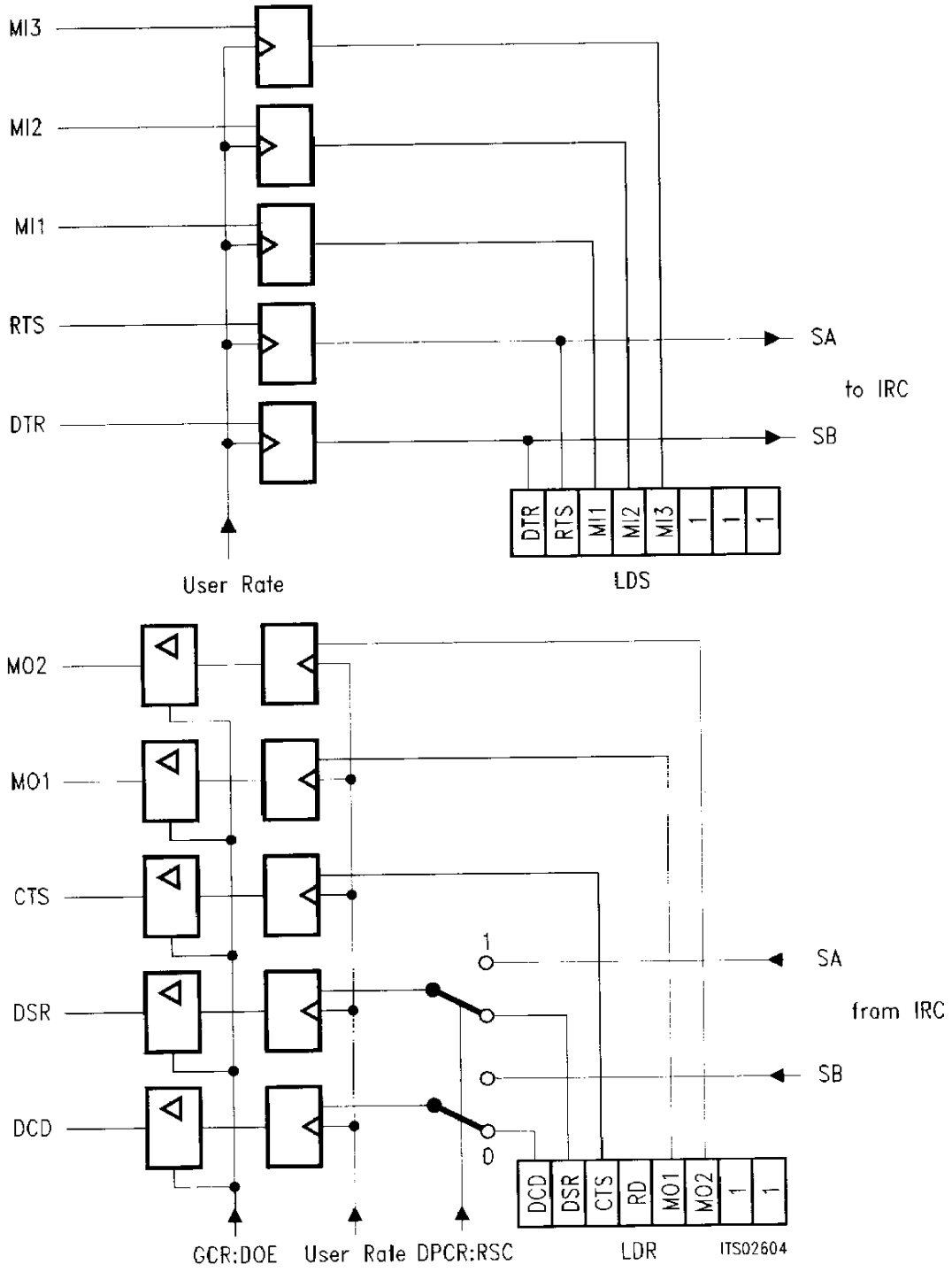
Note: X = 0 or 1

The value of the I interchange circuit depends on the setting of the RCS bit in the DPCR register. If the RCS bit is '0', the value of the I interchange circuit is controlled by the DCD/ I-bit in the LDR register. If the RCS bit is '1', the value of the interchange circuit is controlled by the received S(SA)-bits.

V-Series Interchange Circuits (GCR:V24 = 1)

The configuration of the DCE interchange circuit in V.24 mode are shown in **figure 12.**

Figure 12
DCE Interchange Circuits (GCR:V24 = 1)



The level of the CTS, MO1 and MO2 output is always controlled by the corresponding bits in the LDR register. The level of the DCD and DSR output may be controlled by the corresponding bits of the LDR register. This is selected if the RCS bit in the DPCR register is set to 0. The level of the DCD output is controlled by the received SA bits and the level of the DSR output is controlled by the received SB bit if the RCS bit is set to 1.

The input levels are sampled with the user rate period. The sampled value can always be read from the LDS register and may be forwarded to the S-bits. A change on at least one of the control lines will generate an LDC interrupt. This interrupt is delayed by half a user rate period.

2.2.3 DCE Output and Input Characteristics

Output Drivers

All output drivers of the DCE interface are tri-state drivers. The enable function of the drivers is controlled by the DOE bit in the GCR register. If GCR:DOE (DCE Output Enable) = 0, all output lines are tri-state. If GCR:DOE = 1 all output drivers are active. If it is required to set all output lines to a defined state while DOE = 0, external pull-up resistors must be connected to all output pins.

Connection of Open Inputs

All available input pins of the DCE interface must be set to a defined state in order to avoid interrupts. Unused inputs may be connected to an external pull-up resistor or to GND.

2.3 Synchronous Network Interface

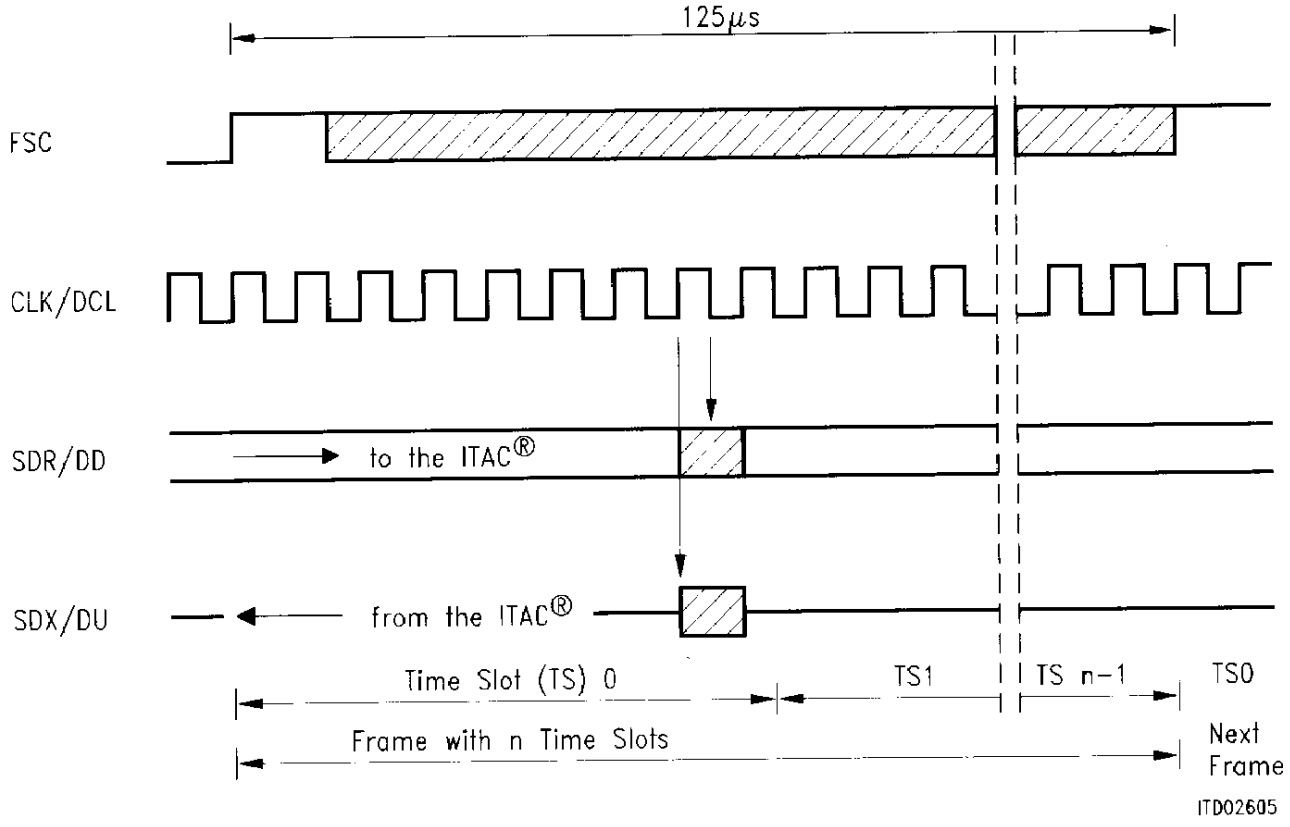
The Synchronous Network Interface (SNI) consists of four lines. Two lines transfer the data signals, one line receives the bit clock signal while the last line receives the frame start signal. The SDX/DU line is used to transmit data to the network side, while SDR/DD line is used to receive data from the network. The bit clock signal (one clock cycle per bit) or the data clock signal (two clock cycles per bit) is connected to the CLK/DCL input. The frame start signal (8 kHz) is connected to the FSC input.

2.3.1 Operating Modes

PCM Interface (single bit clock)

Single bit clock operation is selected, if SCR:DCL = 0. This means that there is one clock cycle per data bit. The transmitted data is clocked off the SDX/DU line by the rising edge of the CLK signal while the received data on the SDR/DD line is latched with the falling edge of the CLK signal.

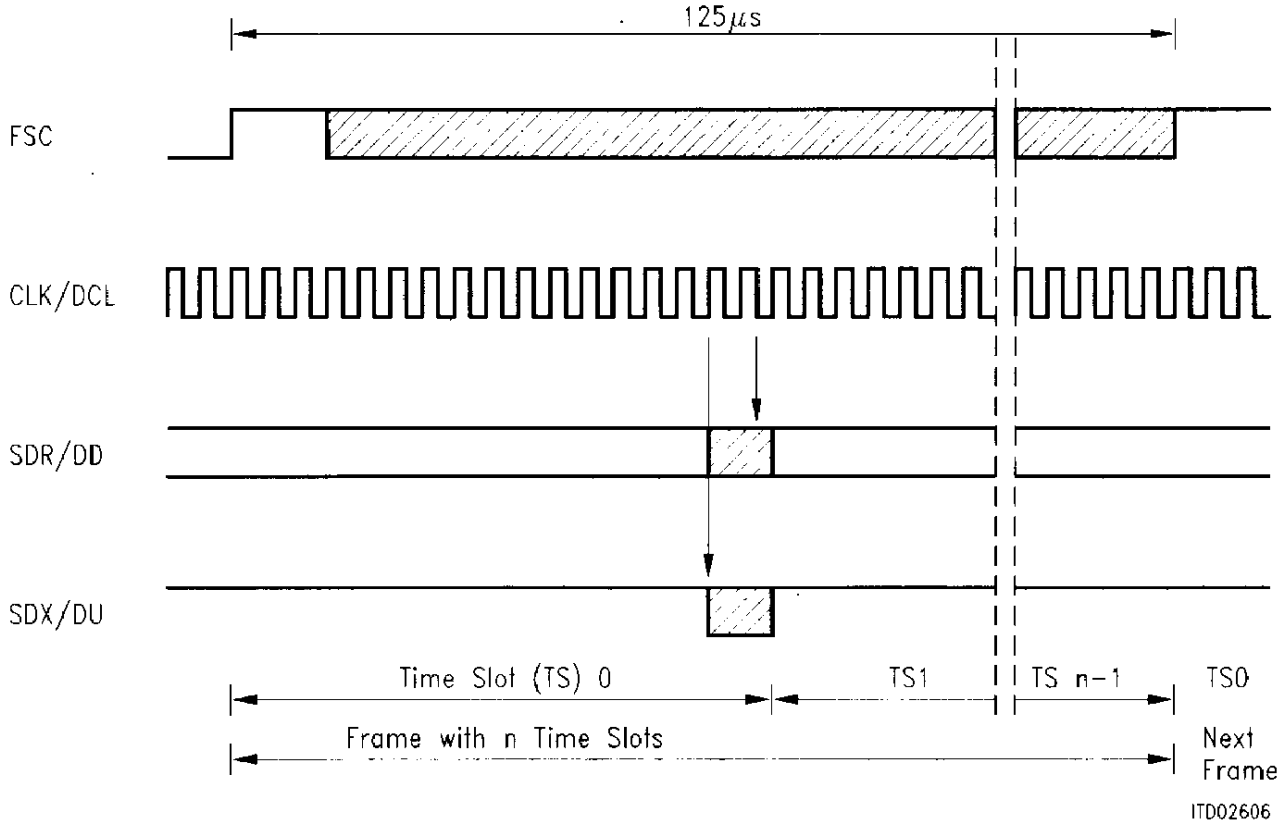
Figure 13
PCM Interface (single bit clock)



IOM®-2 Interface

Double bit clock operation is selected, if SCR:DCL = 1. This means that there are two clock cycles per data bit. The transmitted data is clocked off the SDX/DU line by the first rising edge of the CLK/DCL signal while the received data on the SDR/DD line is latched at the falling edge of the second cycle of the CLK/DCL signal.

Figure 14
IOM®-2 Interface (double bit clock)



SNI Clock Characteristics

In both cases the FSC signal has to fulfill a short setup and hold time. The signal on CLK/DCL may not be symetrical. The data rate on the SNI has to be more or equal to twice the maximum intermediate rate. To transfer 64, 56, 48 or 38.4 kbit/s the minimum data rate on the SNI is 128 kbit/s.

SNI Output Driver Characteristics

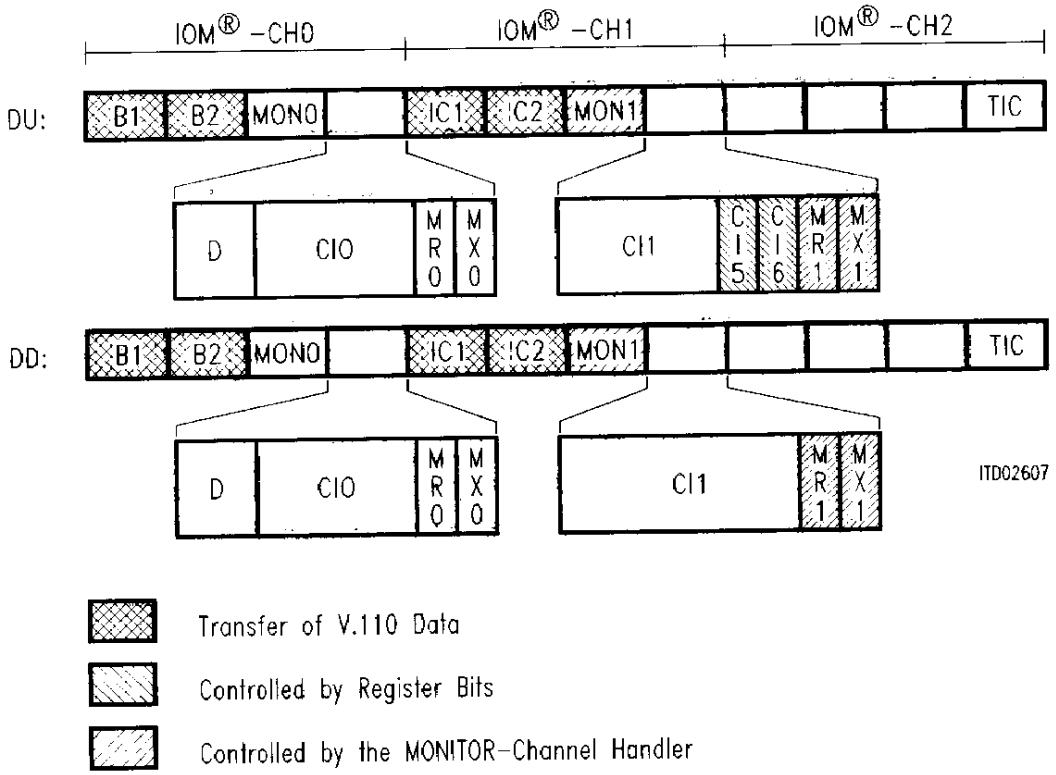
The output driver of the SDX/DU line can be programmed to a push-pull output (SCR:OD = 0) or to an open-drain output (SCR:OD = 1). The output driver is only active during the transmission of the selected bits on the time-slot and tri-state during the other bits of the frame.

If the output driver is set to open-drain output, an external pull-up resistor is required. The value of the resistor depends on the length of the line and the number of connected devices. The minimum value for the resistor is 720 Ω.

2.3.2 IOM®-2 Interface Support

The ITAC offers additional support for the IOM-2 interface mode. This include the protocol support for the MONITOR channel 1 and the control of the CI5 and CI6 bits in the up-stream CI1-channel. It is enabled by setting the MOCR:IOM2 bit to '1'.

Figure 15
IOM®-2 Interface Support



Selection of the V.110/X.30 Data Stream

The V.110/X.30 formated data can be transfered over all time slots of the IOM-2 interface by programming the TSR register. But only four settings are recommended. They are listed in **table 3** and show the programming for B1, B2 and IC1, IC2.

Table 3

Selection of Time -Slots in IOM®-2 Mode

IOM Channel	TSR Value
B1	00000XXX
B2	00001XXX
IC1	00100XXX
IC2	00101XXX

MONITOR Channel Support

The IOM-2 handler of the ITAC includes a MONITOR channel handler for the MONITOR channel 1. The access is hardwired to time slot 7 for the MONITOR data byte and time slot 8 for evaluation of the MR, MX bits.

The MONITOR channel handler performs the MONITOR channel protocol. It provides a receive and transmit register for the MONITOR channel data byte plus control and status bits. Since the ITAC requires a microprocessor interface for operation, there is no automatic comparison of the MONITOR channel address byte. The begin of all MONITOR channel messages is received and the microcontroller accepts a message by setting the MRC control bit to '1'.

More information about the MONITOR channel protocol is available in the IOM-2 reference guide. (Order-No: B115-H6397-X-X-7600)

Control of the MONITOR Channel Handler

The MONITOR channel handler is controlled by four bits of the MONITOR Channel Control Register (MOCR). They are listed in **table 4**.

Table 4
MONITOR Channel Control Bits

Bit Name	Bit No.	Function
MXC	0	**MONITOR Channel Transmit Control** Controls the operation of the transmitter. If set to '0', the MX-bit stays '1' and MOX1 transmits 'FF' (idle). If set to '1', data written into the MOX1 register will be transfered according to the MONITOR channel protocol.
MIE	1	**MONITOR Channel Interrupt Enable** When set to '1', the status bits MDA, MER and MAB may generate an interrupt. Otherwise, the interrupts are masked.
MRC	2	**MONITOR Channel Receive Control** Controls the operation of the receiver. When set to '0', MDR status changes are blocked except for the first byte of a message. When set to '1', received bytes will be acknowledged automatically after they have been read from the MOR1 register.
MRE	3	**MONITOR Channel Receive Enable** When set to '0', the generation of MDR status changes is disabled including the first byte of a message. When set to '1', the MDR status is set for each MONITOR address byte (if MCR = 0) or for each received byte (MRC = 1).

Indications of the MONITOR Channel Handler

Changes in the MONITOR channel handler are indicated by four status bits in the MONITOR Channel Status Register (MOSR). There meaning is shown in **table 5**.

Table 5
MONITOR Channel Indications

Bit Name	Bit No.	Meaning
MAB	0	**MONITOR Channel Abort** If set to '1', the opposite receiver has aborted the transmission.
MDA	1	**MONITOR Channel Data Acknowledge** If set to '1', the opposite receiver has acknowledge the data transfer. A new byte may be entered into the MOX1 register or the MXC-bit may be cleared to indicate the end of message.
MER	2	**MONITOR Channel End of Receiption** If set to '1', the local receiver has detected an end of message condition. The MCR-bit may be cleared to indicate the end of transmission.
MDR	3	**MONITOR Channel Data Received** If set to '1', the local receiver has received a new byte. It can be read from the MOR1 register, which will automatically generate the acknowledgement if MCR = 1.

CI-Channel Control

The IOM-2 handler of the ITAC allows the control of bit 5 and 6 in the upstream direction of Command/Indicate (CI)- Channel 1. The inverted value is written into the CIX5 and CIX6 bits of the MOCR register.

Software Awake

In order to activate a deactivated IOM-2 interface, the SDX/DU pin can be controlled via the SAW-bit. If set to '1', the SDX/DU output is forced to '0' while the following condition is true: FSC*IOM2*SAW=1. Since the FSC signal is high during the first IOM-2 channel and low during the remaining period, MONITOR channel 1 is available to indicate the active condition of the IOM-2 interface and to indicate that the SAW-bit can be reset.

2.4 Clock Generation

The ITAC derives all its clock signal from the on-chip oscillator which requires a 10.752-MHz crystal. The oscillator is controlled by the power-up bit (PU) in the GCR register and in case of an asynchronous interface also from the DTR input.

Oscillator

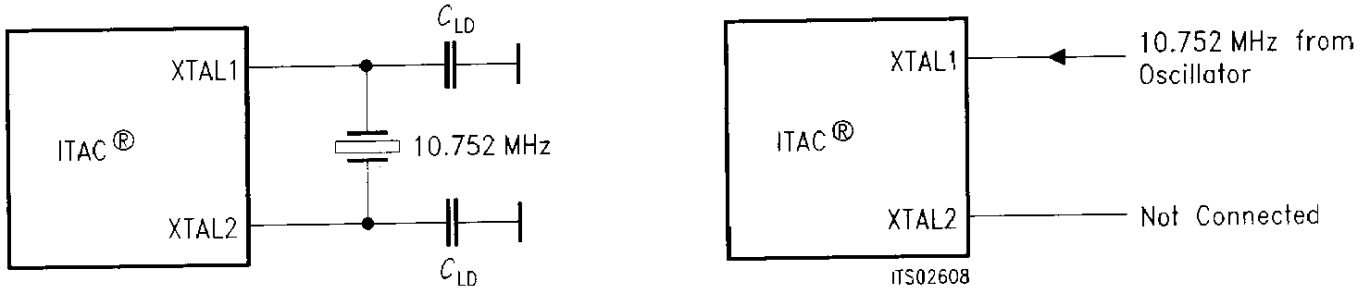
If a synchronous interface is selected (GCR:ASY=0), the oscillator stops if GCR:PU = 0 and starts if GCR:PU = 1.

If an asynchronous interface is selected (GCR:ASY=1), the oscillator stops if GCR:PU = 0 and the level on the DTR input is high (OFF). It is started when either GCR:PU is set to '1' or the level on DTR changes to low (ON). This will automatically set the GCR:PU bit to '1'.

The oscillator needs a maximum period of 10 ms to generate a stable clock signal.

The external crystal is connected between the XTAL1 and XTAL2 pin. XTAL1 is the oscillator input and XTAL2 its output. If an external clock source is used to generate the 10.752-MHz clock, its signal is connected to XTAL1.

Figure 16
Connection of Clock Sources

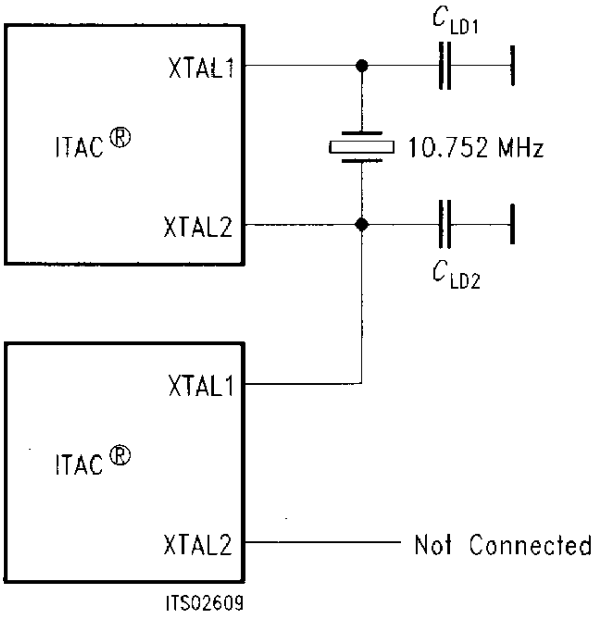


Note: Value of C_{LD} is determined by the XTAL-specification.

Two or more ITAC on the same board may share one crystal. This crystal is connected to XTAL1 and XTAL2 of one ITAC. The XTAL1 input of the other ITACs are connected to the XTAL2 output of the crystal. The XTAL2 output of the other ITACs are left open.
The load capacity of both crystal pins should be same. On XTAL1, there is just the input capacity of one ITAC. On XTAL2, there is the sum of one output capacity plus the input capacity of the other ITACs and the capacities of the line. To calculate the resulting capacities, each input capacity has a maximum value of 7 pF.

Please note that the power up bit (GCR:PU) of the ITAC to which the crystal is connected has to be '1' in order to provide a clock signal on XTAL2.

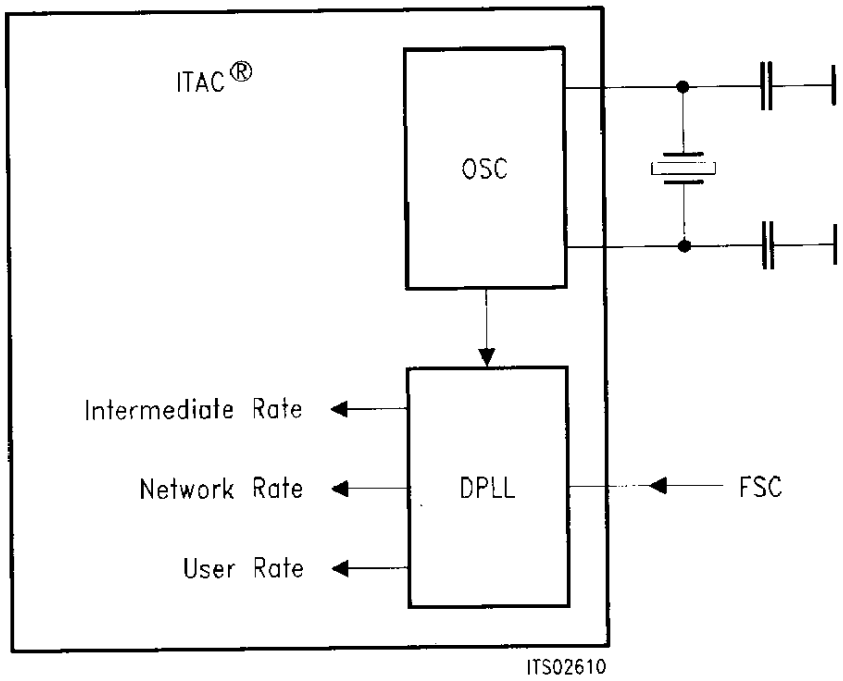
Figure 17
Driving Two ITAC® Oscillators by One Crystal



Operation of the DPLL

The DPLL generates four clock signals which are synchronized to the 8-kHz clock on the FSC input. The four clock signals are the user rate, which specifies the speed of the DCE interface and the DCE side of the ASC, the network rate, which specifies the speed of the IRC and the IRC side of the ASC, the intermediate rate, which specifies the speed between the IRC and the BRC and the 8-kHz reference signal for the DPLL.

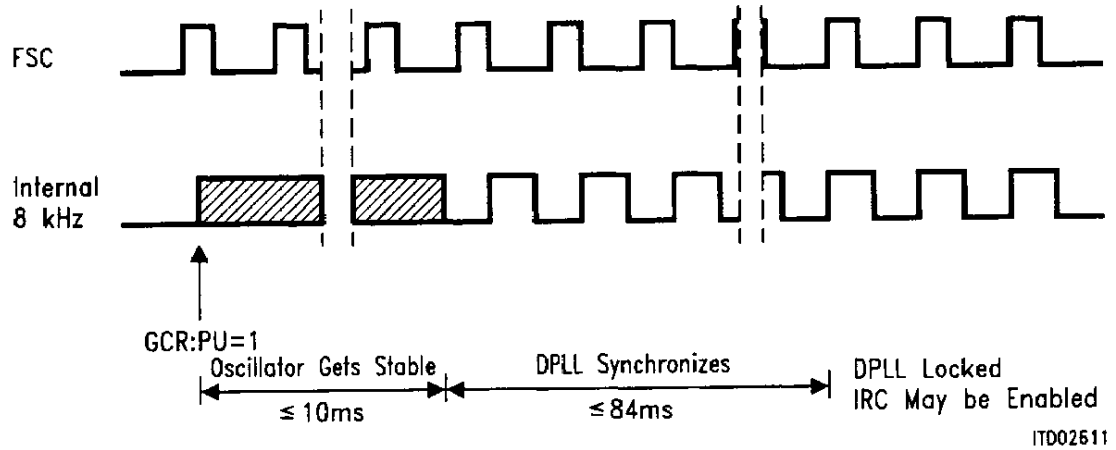
Figure 18
Clock Synchronization



The DPLL evaluates the phase difference between the internal 8-kHz signal and the input signal on the FSC pin and adjusts its clock by one oscillator step (93 ns) every 125 microseconds. The maximum phase difference is one half of the 8-kHz signal which is 62.5 microseconds. If the DPLL starts to adjust in this situation it takes 62.5 us/93 ns = 672 steps which needs 84 ms.

The adjustment of 1 oscillator period per 125 microseconds guarantees a maximum jitter of less than 1 % for synchronous data transmission at 64 kbit/s.

Figure 19
Synchronization of the DPLL



Selection of the User Rate, Network Rate and Intermediate Rate

The user rate and the network rate are programmed in the BRS register. The intermediate rate is automatically selected by the value of the network rate.

Table 6 shows the possible settings for the network rate.

Table 6
Network Rates

NR3	NR2	NR1	NR0	Network Rate	Intermediate Rate	Number of Bits per time-slot	Remarks
0	0	0	0	–	–	–	reserved
0	0	0	1	600 bit/s	8 kbit/s	1	4x80 bit framing
0	0	1	0	1200 bit/s	8 kbit/s	1	2x80 bit framing
0	0	1	1	2400 bit/s	8 kbit/s	1	80 bit framing
0	1	0	0	4800 bit/s	8 kbit/s	1	80 bit framing
0	1	0	1	9600 bit/s	16 kbit/s	2	80 bit framing
0	1	1	0	19200 bit/s	32 kbit/s	4	80 bit framing
0	1	1	1	38400 bit/s	64 kbit/s	8	80 bit framing
1	0	0	0	48000 bit/s	64 kbit/s	8	32 bit framing
1	0	0	1	56000 bit/s	56/64 kbit/s	7/8	7 bit framing 8 bit framing 64 bit framing
1	0	1	0	64000 bit/s	64 kbit/s	8	8 bit transparent
1	0	1	1	–	–	–	reserved
1	1	x	x	–	–	–	reserved

The selectable user rates depend on the selected DCE interface type. If an asynchronous interface (GCR:ASY = 1) is selected, the user rate can be set between 300 and 38400 baud. The user rate may range between 600 and 64000 bit/s if a synchronous interface is selected (GCR:ASY = 0).

Table 7
User Rates

UR3	UR2	UR1	UR0	Sync. Data Rate	Async. Data Rate	Remarks
0	0	0	0	–	300 bit/s	
0	0	0	1	600 bit/s	600 bit/s	
0	0	1	0	1200 bit/s	1200 bit/s	
0	0	1	1	2400 bit/s	2400 bit/s	
0	1	0	0	4800 bit/s	4800 bit/s	
0	1	0	1	9600 bit/s	9600 bit/s	
0	1	1	0	19200 bit/s	19200 bit/s	
0	1	1	1	38400 bit/s	38400 bit/s	
1	0	0	0	48000 bit/s	–	
1	0	0	1	56000 bit/s	–	
1	0	1	0	64000 bit/s	–	
1	0	1	1	–	–	reserved
1	1	x	x	–	–	reserved

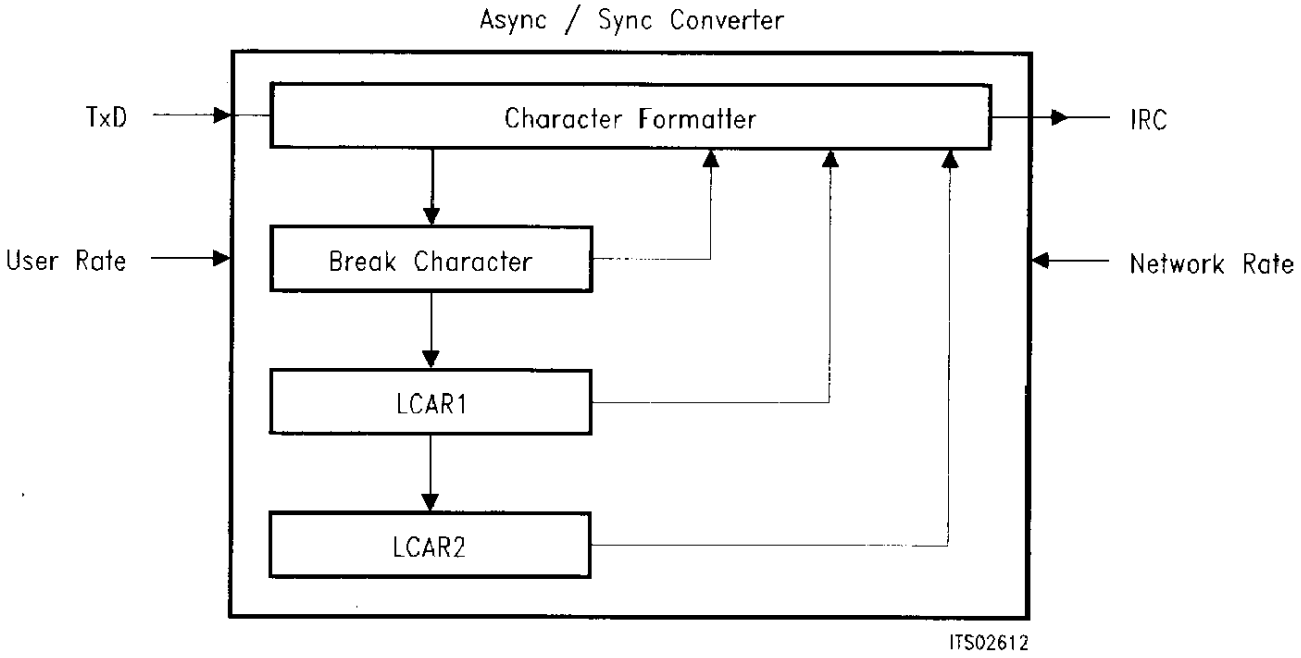
2.5 Async/Sync Converter (ASC)

The Async/Sync Converter transforms a stream of Start/Stop bit formatted characters at the user rate into a synchronous bit stream at the network rate and vice versa. It is active if the DCE interface is set to asynchronous operation (GCR:ASY = 1).

Functions of the ASC in Transmit Direction (to the network)

The ASC block diagram of the transmitter is shown in **figure 20**.

Figure 20
ASC Transmitter

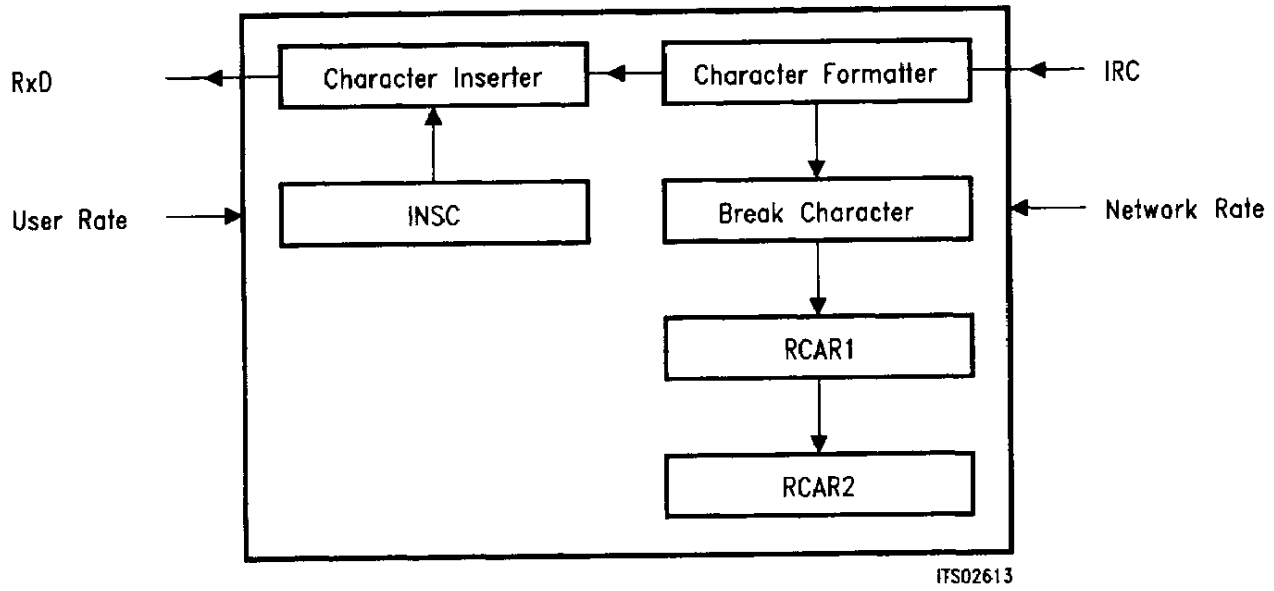


In transmit direction the ASC will frame the incoming characters (from the DCE interface) and transfer them to the IRC at the network rate. If the user rate is less than the network rate, the gaps between characters are padded by stop bits. The ASC will not operate properly if the user rate is higher than the network rate. If the actual data rate at the DCE interface is higher than the nominal user rate, stop bits will be deleted within the limits of the selected tolerance range. If AICR:TR = 0 the tolerance range is 12.5 %. This means that every eighth stop bit may be deleted. If AICR:TR = 1 the tolerance range is set to 25 %. This means that every fourth stop bit may be deleted. The ASC will also recognize a break signal. The break signal is detected when at least M bits of start polarity have been received. M denotes the number of bits per character in the selected format including start and stop bits. If the ASC detects a break condition from the DCE interface, it will transmit at least 2M+1 start bits towards the IRC.

Functions of the ASC in Receive Direction (from the network)

The ASC receiver is shown in **figure 21**.

Figure 21
ASC Receiver

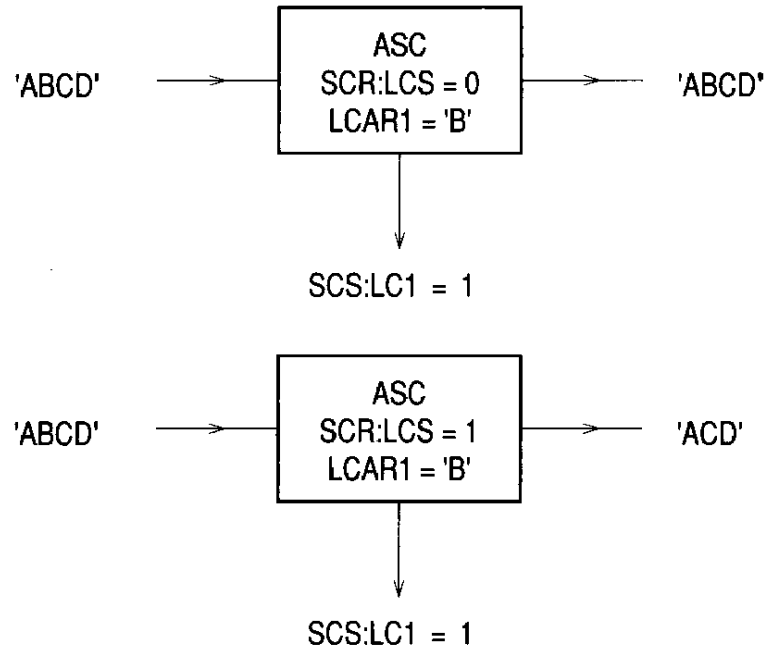


In receive direction the ASC receives the synchronous bit stream at the network rate and transfers it into a character stream at the user rate which is transmitted to the DCE interface. If the network rate is less than the user rate, the gaps between characters are padded by stop bits. The receive direction will not operate properly if the network rate is higher than the user rate. If a missing stop element is detected in the data stream, the length of the stop bits transmitted to the DCE interface is reduced according to the selected tolerance range.

Recognition of Special Characters (flow control)

Two programmable characters can be specified for each direction. If a character is received from the DCE interface, which matches the value of LCAR1 or LCAR2 the corresponding status bit is set in the Special Condition Status register (SCS). If the Local Character Stop bit in the Special Configuration register (SCR:LCS) is set to '1', the received character bits are substituted by stop bits. Thus the character is deleted from the data stream. The same function is available for the received data from the network. The special character registers are RCAR1 and RCAR2. The Remote Character Stop bit (SCR:RCS) controls the substitution of the character bits.

Figure 22
Special Character Recognition



This feature may be used to detect and remove flow control characters (XON/XOFF) from the data stream.

Insertion of Characters

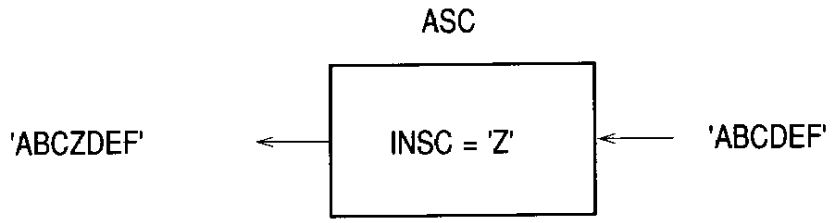
The Character Insert Register INSC allows to insert characters in the receive data stream (from the network). The character which has to be inserted is written into the INSC register and automatically inserted at the next opportunity. The status whether the character has been inserted or not is supervised by the CIS bit in the Status register STR.

This feature may be used to add flow control characters in the data stream towards the DCE interface.

The character written into the INSC register has to contain a possible parity bit and unused bits have to be filled with ones.

If the ASC receives data while a character has to be inserted, it will insert the character after the current character has been completed and it will reduce the length of the stop bits of the following characters. Thus the received characters are not pertubed.

Figure 23
Character Insertion Using INSC



Selecting the character format for the ASC

The character format of the ASC is programmed by the AICR and the UMR register.

If a parity bit is selected, the format of a character is incremented by one bit. The ASC will not check the parity on the received character.

Table 8
Character Format of the ASC

Start	Character Length	Parity	Stop
	AICR:CHL1,0	UMR:PTY	AICR:STP

AICR		
CHL1	**CHL1**	**Character Length (Bit)**
0	0	Eight
0	1	Seven
1	0	Six
1	1	Five

UMR			
PTY	**PY1**	**PY0**	**Parity Bit**
0	X	X	no
1	X	X	yes

AICR:STP	Number of Stop Bits
0	1
1	2

2.6 Intermediate Rate Converter

The intermediate rate converter performs the framing according to V.110 or X.30. The source and destination of the D bits may be the DCE interface, the Serial Communications Logic or register bits. The source and destination of the S bits may be the DCE interface or register bits. The source and destination of the X and E bits are always register bits.

The operation of the IRC is controlled by the ENFR bit (GCR). Before the ENFR bit is set to '1', the network rate, the V.110/X.30 frame format and the time-slot has to be programmed. If one of the register values needs to be changed, the ENFR bit has to be cleared for one period of the previous intermediate rate.

Frame Structures

The frame structure depends on the selected network rate. For network rates from 600 to 38400 bit/s a 80-bit frame is used. For a network rate of 48000 bit/s a 32-bit frame is used. For a data rate of 56000 bit/s one of two frame alternatives may be selected. One uses a 8-bit frame while the second one uses a 64-bit frame. If the 8-bit frame is selected and the status of the IRC is masked, a transparent 56 kbit/s channel is available. No framing is performed if the network rate is 64000 bit/s.

Table 9 to **16** show the framing for the individual data rates.

Table 9
Frame Structure for 600 bit/s

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	1	D1	D1	D1	D1	D1	D1	S1
3	1	D1	D1	D2	D2	D2	D2	X
4	1	D2	D2	D2	D2	D3	D3	S3
5	1	D3	D3	D3	D3	D3	D3	S4
6	1	E1	E2	E3	E4	E5	E6	E7*
7	1	D4	D4	D4	D4	D4	D4	S6
8	1	D4	D4	D5	D5	D5	D5	X
9	1	D5	D5	D5	D5	D6	D6	S8
10	1	D6	D6	D6	D6	D6	D6	S9

Table 10
Frame Structure for 1200 bit/s

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	1	D1	D1	D1	D1	D2	D2	S1
3	1	D2	D2	D3	D3	D3	D3	X
4	1	D4	D4	D4	D4	D5	D5	S3
5	1	D5	D5	D6	D6	D6	D6	S4
6	1	E1	E2	E3	E4	E5	E6	E7
7	1	D7	D7	D7	D7	D8	D8	S6
8	1	D8	D8	D9	D9	D9	D9	X
9	1	D10	D10	D10	D10	D11	D11	S8
10	1	D11	D11	D12	D12	D12	D12	S9

*E7 is controlled automatically and transmits 1011.

Table 11
Frame Structure for 2400 bit/s

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	1	D1	D1	D2	D2	D3	D3	S1
3	1	D4	D4	D5	D5	D6	D6	X
4	1	D7	D7	D8	D8	D9	D9	S3
5	1	D10	D10	D11	D11	D12	D12	S4
6	1	E1	E2	E3	E4	E5	E6	E7
7	1	D13	D13	D14	D14	D15	D15	S6
8	1	D16	D16	D17	D17	D18	D18	X
9	1	D19	D19	D20	D20	D21	D21	S8
10	1	D22	D22	D23	D23	D24	D24	S9

Table 12
Frame Structure for 4800, 9600, 19200, 38400 bit/s

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	1	D1	D2	D3	D4	D5	D6	S1
3	1	D7	D8	D9	D10	D11	D12	X
4	1	D13	D14	D15	D16	D17	D18	S3
5	1	D19	D20	D21	D22	D23	D24	S4
6	1	E1	E2	E3	E4	E5	E6	E7
7	1	D25	D26	D27	D28	D29	D30	S6
8	1	D31	D32	D33	D34	D35	D36	X
9	1	D37	D38	D39	D40	D41	D42	S8
10	1	D43	D44	D45	D46	D47	D48	S9

Table 13
Frame Structure for 56000 bit/s
Frame Alternative 1

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	D1	D2	D3	D4	D5	D6	D7	1
2	D8	D9	D10	D11	D12	D13	D14	1
3	D15	D16	D17	D18	D19	D20	D21	1
4	D22	D23	D24	D25	D26	D27	D28	1
5	D29	D30	D31	D32	D33	D34	D35	1
6	D36	D37	D38	D39	D40	D41	D42	1
7	D43	D44	D45	D46	D47	D48	D49	1
8	D50	D51	D52	D53	D54	D55	D56	1

Table 14
Frame Structure for 56000 bit/s
Frame Alternative 2

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	D1	D2	D3	D4	D5	D6	D7	0
2	D8	D9	D10	D11	D12	D13	D14	X
3	D15	D16	D17	D18	D19	D20	D21	S3
4	D22	D23	D24	D25	D26	D27	D28	S4
5	D29	D30	D31	D32	D33	D34	D35	1
6	D36	D37	D38	D39	D40	D41	D42	1
7	D43	D44	D45	D46	D47	D48	D49	1
8	D50	D51	D52	D53	D54	D55	D56	1

Table 15
Frame Structure for 48000 bit/s

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	1	D1	D2	D3	D4	D5	D6	S1
2	0	D7	D8	D9	D10	D11	D12	X
3	1	D13	D14	D15	D16	D17	D18	S3
4	1	D19	D20	D21	D22	D23	D24	S4

Table 16
Frame Structure for 64000 bit/s

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	D1	D2	D3	D4	D5	D6	D7	D8

Explanation of the Frame Elements

All frames consist of five basic elements. The first element is the frame synchronization pattern. The other four are the D-bits, the S-, X- and E-bits.

Frame Synchronization Pattern

The frame synchronization pattern are shown in **table17** to **20**.

The frame synchronization pattern for a network rate of 600 bit/s uses E7 for super frame synchronization. Network rates of 48000 bit/s or 56000 bit/s use the octet alignment provided by the B channel. Bit 1 of each octet is tested at 48000 bit/s while bit 8 is used at 56000 bit/s.

Table 17
Frame Synchronization Pattern for Network Rates 600 ... 38400 bit/s

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	1							
3	1							
4	1							
5	1							
6	1							(E7)*
7								
8	1							
9	1							
10	1							

Table 18
Frame Synchronization Pattern for a Network Rate of 48000 bit/s

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1	1							
2	0							
3	1							
4	1							

Table 19
Frame Synchronization Pattern for a Network Rate of 56000 bit/s
Frame Alternative 1

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1								1
2								1
3								1
4								1
5								1
6								1
7								1
8								1

Table 20
Frame Synchronization Pattern for a Network Rate of 56000 bit/s
Frame Alternative 2

Octet No.	Frame Structure 1	2	3	4	5	6	7	8
1								0
2								
3								
4								
5								1
6								1
7								1
8								1

*E7 is controlled automatically at 600 bit/s.

D-Bits

The D-bit positions (D1 ... D48) are used to transfer the data stream between both ends. The data stream includes all character, parity, start and stop bits. If the network rate is less or equal to 4800 bit/s each bit of the data stream is transmitted twice, four or eight times to match the intermediate rate.

S-Bits

The S-bits are used to transfer the state of interchange circuits. In a V.110 frame, two S-bit groups SA and SB are available. SA transmits the state of the DTR/C interchange circuit (108) to the DSR interchange circuit (107). SB transmits the state of the RTS interchange circuit (105) to the DCD/I interchange circuit (109).

In an X.30 frame all S-bits are used to transfer the state of the C/DTR (108) interchange circuit onto the I/DCD (109) interchange circuit.

The selection between both S-bit alternatives is done by the V110 bit in the General Configuration Register. If GCR:V110 = 0, the X.30 frame is selected. GCR:V110 = 1 selects the V.110 frame.

Table 21

V.110 frame (GCR:V110 = '1')	X.30 frame (GCR:V110 = '0')
DTR/C (108) → SA → DSR (107) RTS (105) → SB → DCD/I (109)	DTR/C (108) → S → DCD/I (109)

Note: SA = S1, S3, S6, S8
SB = S4, S9

X-Bits

The X-bits are used to perform flow control between both terminal adapters.

E-Bits

The E-bits are used to transfer information between both terminal adapters. In a V.110 frame, E1 to E3 are used to identify the user rate. E7 is automatically controlled if the network rate is set to 600 bit/s.

Sample Points of the S-Bits

The coordination between the sampling of the D- and S-bits for a synchronous DCE interface (GCR:ASY = 0) is shown in **table 22**.

Table 22
Sample points of the S-Bits

S Bit	D Bit	
	Octet No.	Bit No.
S1	2	3 (D8)
S3	3	4 (D16)
S4	4	7 (D24)
S6	7	3 (D32)
S8	8	5 (D40)
S9	9	7 (D48)

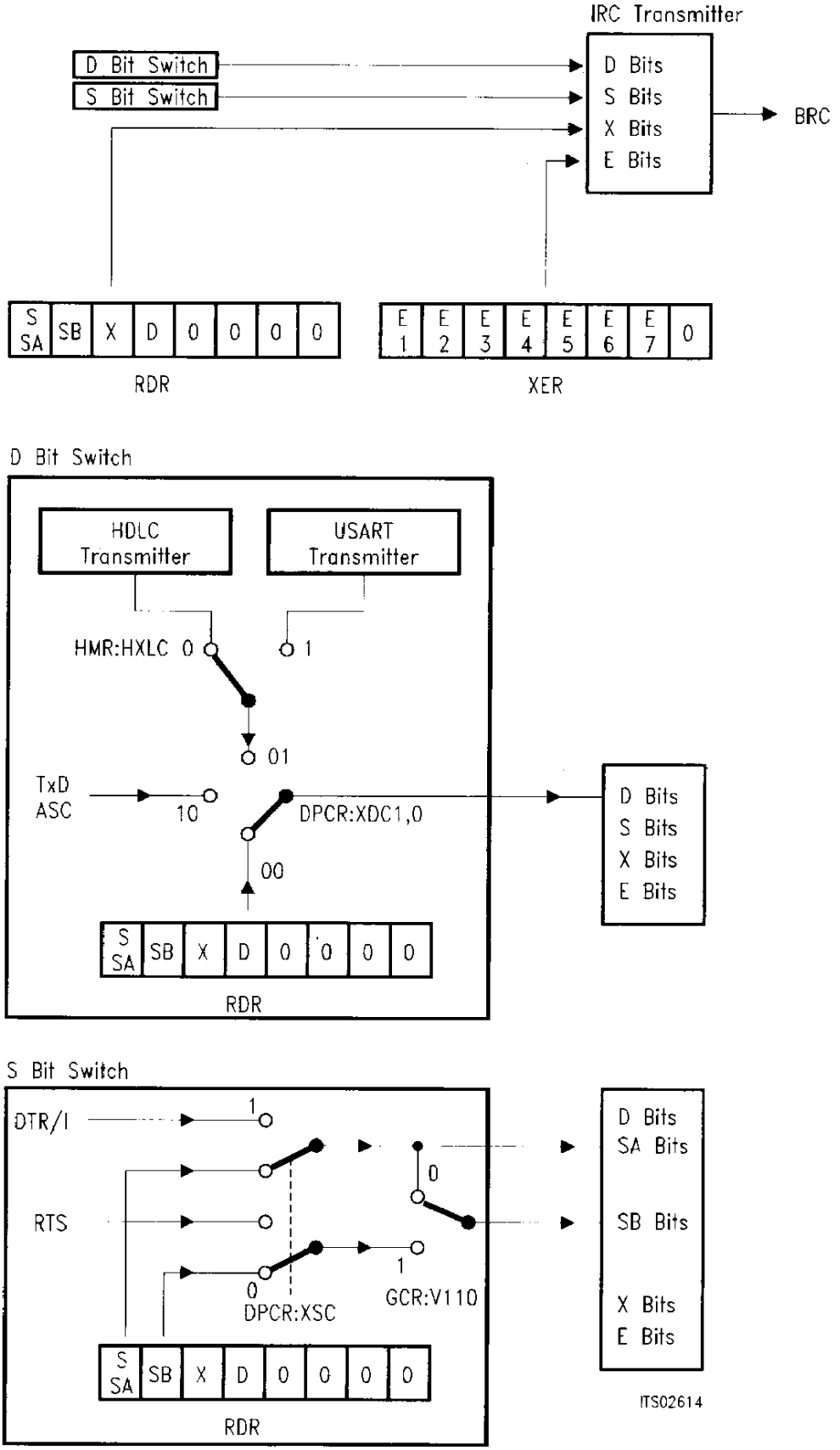
If the DCE interface is asynchronous (GCR:ASY = 1), the S bits are sampled while the stop bit of the character is received. If no character is received, the S-bits are sampled during D8, D16, D24, D32, D40 and D48. During a break condition, the S-bits are not sampled for version VA1, VB1. The version V2.2 samples the S bits every eight D-bits during a break condition.

Function of the Transmitter

The IRC-transmitter is enabled by setting the ENFR bit to '1'.

The transmitter part of the IRC will format the required frame. Therefore it will generate the frame synchronization pattern and insert the D-, S-, X- and E-bits. The source of the D-bits can either be the DCE interface, the Serial Communications Logic or the register bit XD. The source of the S-bits can either be the DCE interface control lines or the register bits S/SA and SB. The source of the X- and E-bits will always be the register bits. **Figure 24** illustrates the possible data paths.

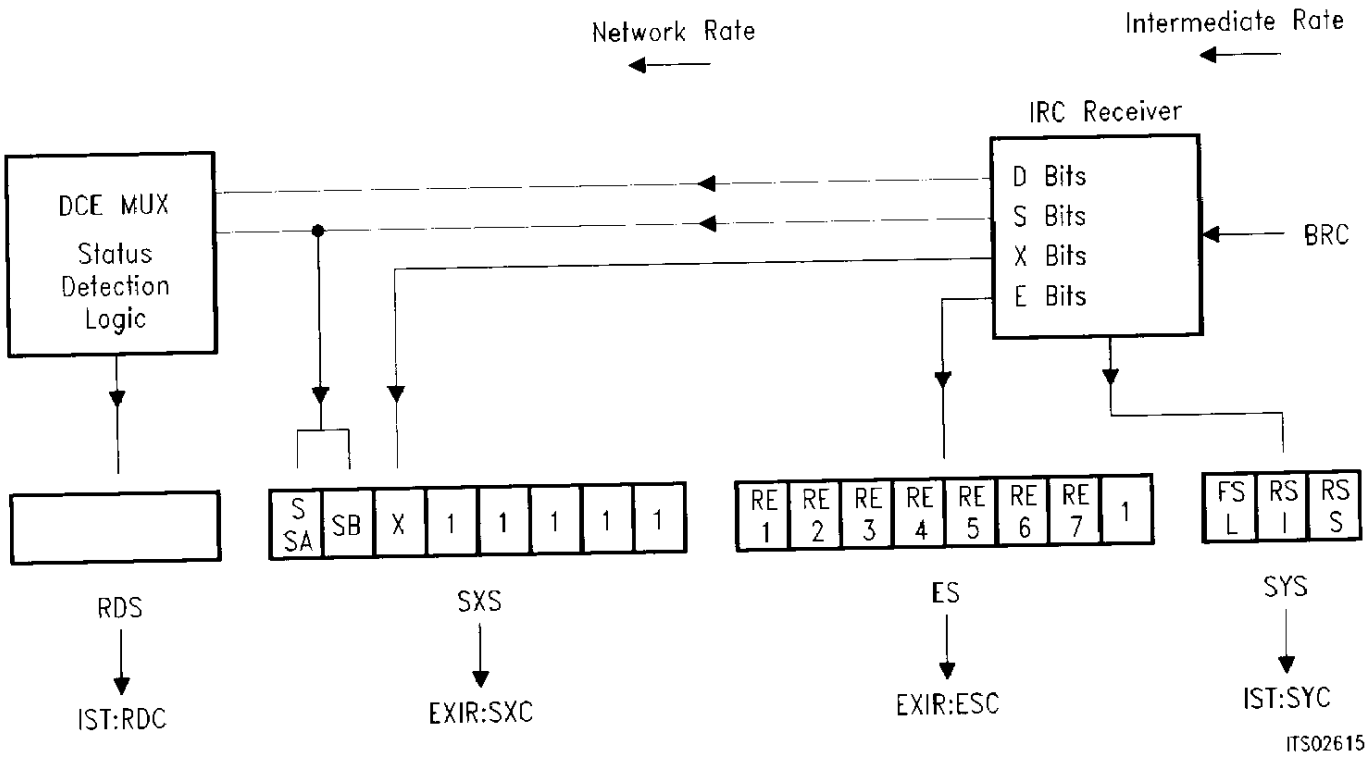
Figure 24
IRC Transmitter



Function of the Receiver

The IRC receiver is enabled by setting the ENFR bit to '1'. The receiver part of the IRC will search for the frame synchronization pattern. A status register tells the actual status of the receiver. The receiver also extracts the D-, S-, X- and E-bits and forwards them to their programmed destination. The received D-bits may be transmitted to the DCE interface. They are always transmitted to the Serial Communication Logic. The received S-bits may also be transmitted to the DCE interface. They are always transmitted to the SXS Status register. An interrupt is generated on every status change. The received X-bits are always transmitted to the SXS Status register and E-bits to the ES register. An interrupt is generated on every X- or E-bit change.

Figure 25
IRC Receiver



Synchronization Status

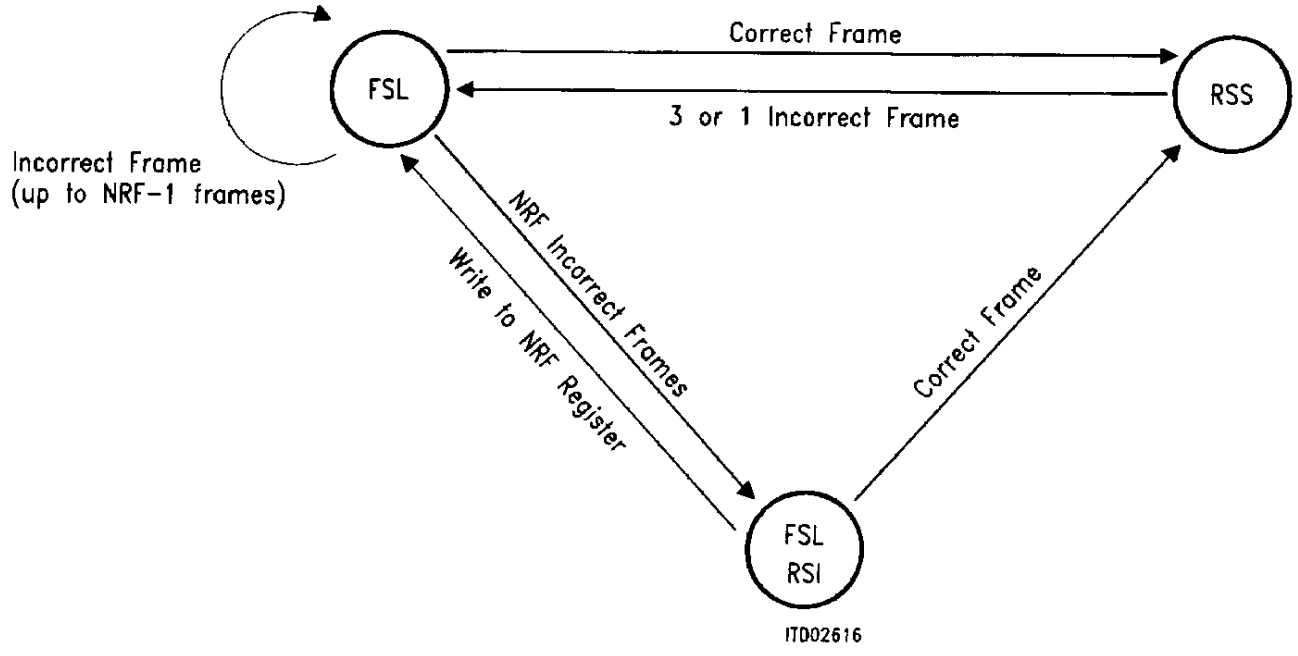
The IRC receiver reports its frame synchronization status in the SYS register. The most significant three bits are used. **Table 23** shows the indicated states.

Table 23
States of the IRC Receiver

SYS			
FSL	**RSI**	**RSS**	**Status**
1	0	0	No synchronization is achieved. Status while ENFR = 0 or after 1 or 3 incorrect frames.
1	1	0	Resynchronization impossible. Status after the number of frames specified in the NFR register were counted down without gaining synchronization.
0	0	0	Synchronization achieved. Status after one correct frame has been received.

If the IRC is enabled, it will always search for the synchronization pattern regardless of its actual state. If it finds the synchronization pattern it will change to RSS. **Figure 25** illustrates the state machine of the IRC receiver.

Figure 26
Synchronization State Machine



The IRC receiver will always output data even if FSL = 1. Therefore the data path should be selected after RSS has been detected.

In case of a network rate of 64 kbit/s, FSL stays '1' and RSI, RSS are '0'.

Remote DTE Status Indications

The received S/SA- and D bits are evaluated by the remote DTE status logic. The detected combinations are reported by the RDS register. Changes in the RDS register set the RDC status bit in the IST register which may activate the INT output.

The remote DTE status logic is in 'no-state' after reset. 'No-state' is indicate by a value of zero in the RDS register. The remote DTE status logic starts to sample the S/SA- and D-bits after ENFR has been set to '1', regardless of the current synchronization status of the IRC receiver. The sampled data is compared against different possibilities after 16 D-bit samples were taken. If a match is detected the RDS register changes to the new value and the RDC status bit is set to '1'. If a status is detected which is different from LONX or LOFX, the remote DTE status detection logic will sample four D-bits before the pattern is varified again. If the status is no longer true, the RDS register changes to 'no-state' and the RDC status bit is set to '1'. The next status indication will be indicated after another 16 D-bits have been sampled.

If LONX or LOFX or no valid status is detected after 16 samples, the status detection logic will check its last 16 samples after every D-bit.

The valid status are shown in **table 24**.

Table 24
Remote DTE Status Indications

Status	Status Name	Sampled Data	Register Bit
RONX	Remote (X, ON)	D XXXXXXXX XXXXXXXX S/SA 00000000 00000000	Bit 7
ROFX	Remote (X, OFF)	D XXXXXXXX XXXXXXXX S/SA 11111111 11111111	Bit 6
RON1	Remote (1, ON)	D 11111111 11111111 S/SA 00000000 00000000	Bit 5
ROF1	Remote (1, OFF)	D 11111111 11111111 S/SA 11111111 11111111	Bit 4
ROF0	Remote (0, OFF)	D 00000000 00000000 S/SA 11111111 11111111	Bit 3
RL2	Remote Loop 2	D 00110011 00110011 S/SA 00000000 00000000	Bit 2

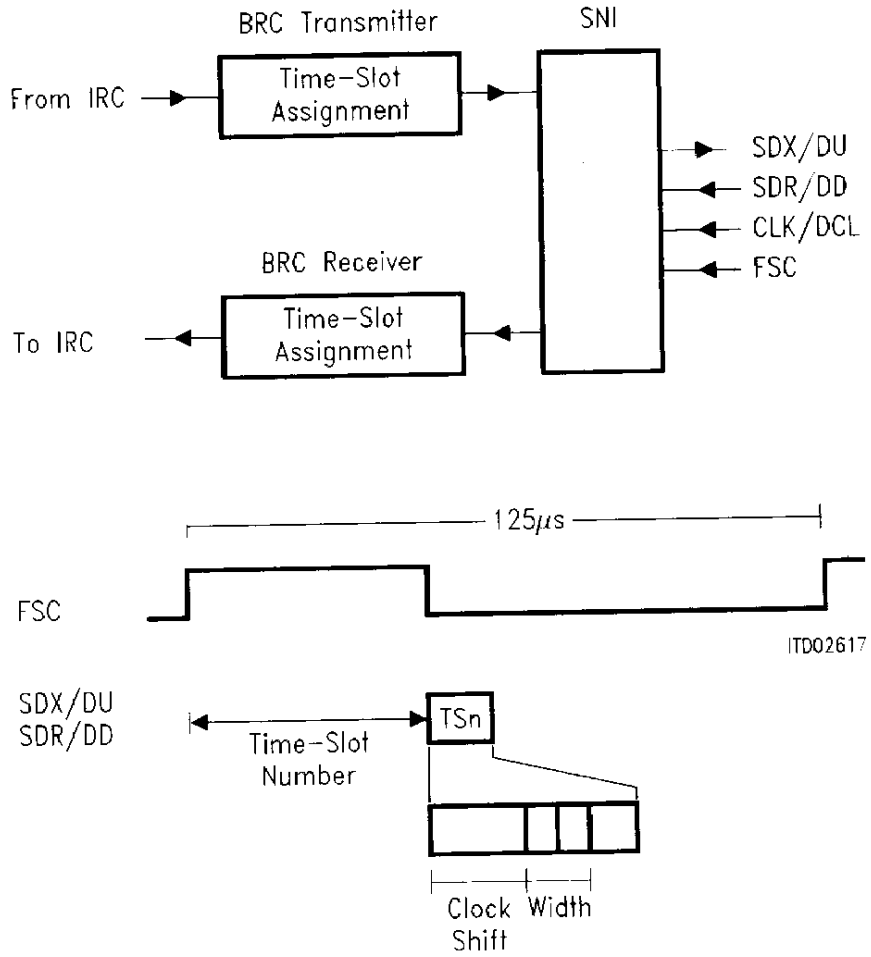
For network rates which do not provide a S-bit (64 kbit/s; 56 kbit/s, F56 = 0), the value of the S/SA-bit in the remote DTE status detection logic is set to '1'. The RDS register will only indicate ROFX, ROF1, ROF0.

The remote DTE status detection may be used to detect the (S/SA = OFF, D = 0) condition which indicates a disconnect request according to V.110.

2.7 Bearer Rate Converter

The Bearer Rate Converter BRC transfers the output of the IRC onto the SNI and vice versa. The BRC is programmed to the first bit position on the SNI frame. The delay from the frame sync signal is programmed in a nine bit register, where the eight least significant bits are located in the TSR register and the most significant bit is located in the SCR register. For convenience the nine bit register is devided in the upper six bits which specify the time-slot number (time-slot width = 8 bit) and the least significant three bits specify the clock shift within the time-slot.

Figure 27



The number of used bits per SNI frame depends on the programmed network rate and the resulting intermediate rate. **Table 25** shows this relationship.

Table 25
Number of Bits per Time Slot

Network Rate bit/s	Intermediate Rate kbit/s	Number of Bit per Time Slot
600	8	1
1200	8	1
2400	8	1
4800	8	1
9600	16	2
19200	32	4
38400	64	8
48000	64	8
56000	64	8
64000	64	8

After the rising edge of the FSC signal has been received, the programmed number of clocks is counted down. During this time, a new edge of the FSC signal will have no effect.

2.8 Serial Communications Logic

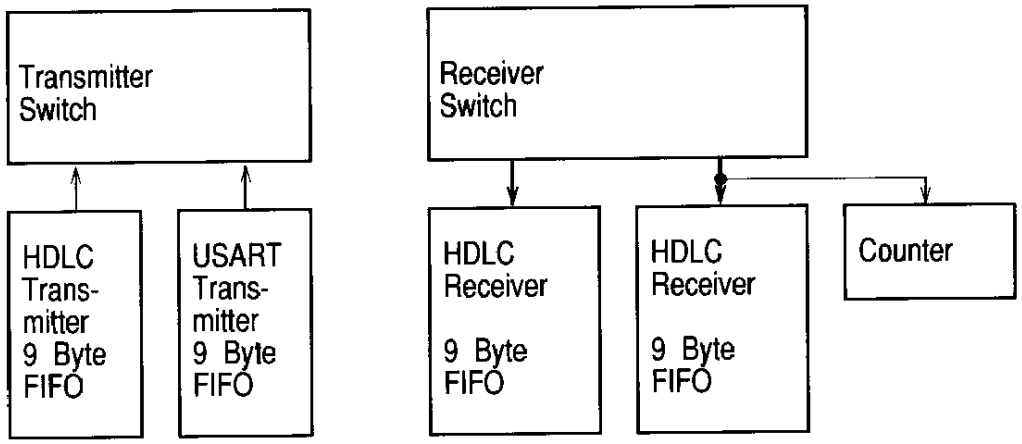
The Serial Communication Logic (SCL) consists of an USART receiver and transmitter plus an HDLC receiver and transmitter. The data path for both transmitters and receivers can be selected independently. They may either be connected to the DCE interface or to the IRC.

Each receiver and transmitter has its 9-byte FIFO to reduce the dynamic load on the microprocessor. Several status bits inform the microprocessor about the FIFO contents and possible errors on the received data.

The FIFOs may be cleared or loaded either via the microprocessor or via direct memory access (DMA). The DMA logic may either be connected to the USART block or the HDLC block.

A 20-bit counter is connected in parallel to the USART receiver. It is used to generated periodical interrupts or to supervise the number of contiguous stop bits. This mode is used to supervise the guard time for the hayes modem protocol application.

Figure 28
Serial Communication Logic



The USART consists of a transmitter and a receiver block. It may either transfer a synchronous data stream (mono or bi-sync), transparent or an asynchronous data stream (start-stop characters).

Synchron Operation

The synchronous operation of the USART is selected by programming the ASYC bit in the USART mode register (UMR) to '0'. All received and transmitted characters have a length of 8 bit including a possible parity bit. The AICR register has to be set to '00'.

Receiver Block

The USART receiver has two modes of operation: transparent mode and hunt mode. In the transparent mode, the receiver, when enabled, immediately starts storing received character without regards of octets synchronization. In the hunt mode, the receiver searches for one (mono) or two (bi-sync) programmable synchronization characters before starting to store the received characters. The hunt mode and the transparent mode may be entered at any time by issuing the corresponding command in the UCC register.

The selection between monosync or bisync operation is done by the SCM bit in the UMR. **Table 26** shows the relationship.

Table 26
USART Synchronous POperating Modes

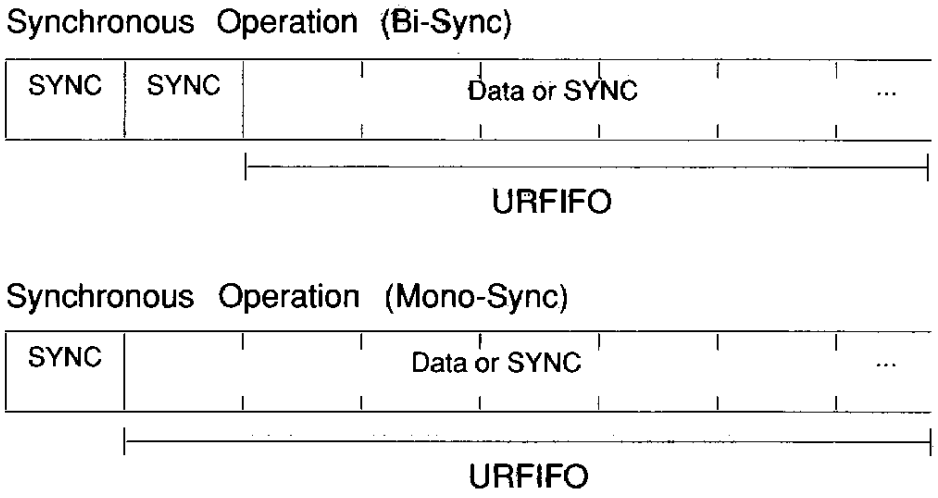
SCM	Operating Mode
0	Bi-Sync
1	Mono-Sync

The synchronization character itself is written to the SYN register. After reset (either hardware reset or reset command) the USART receiver will be in hunt mode.

The hunt mode is left, after the specified number of sync characters have been received. The following characters will be stored in the URFIFO, regardless whether they are sync characters or different ones.

Figure 29 shows the receiver format.

Figure 29
Receiver Data Format



A parity check may also be performed on the received characters. The parity bit will always be the most significant bit.

Table 27 shows the possible parity types. Parity check is enabled by setting the PTY bit in the UMR to '1'.

Table 27
Parity Types

UMR		
PY1	**PY0**	**Parity Type**
0	0	Fixed '0'
0	1	ODD Parity
1	0	EVEN Parity
1	1	Fixed '1'

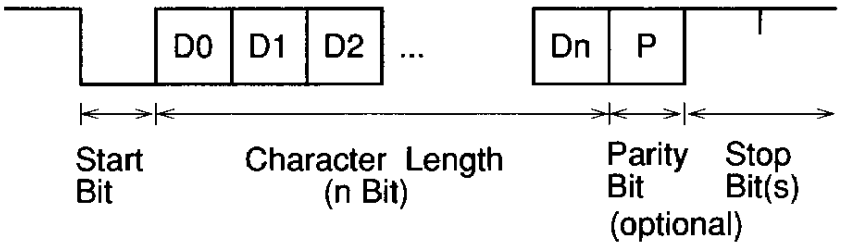
Functions of the Transmitter

When the transmitter is enabled, the transmission of any character entered in the UXFIFO is immediately started. All eight bits of the character are transmitted with the LSB first. A parity bit is not generated by the USART transmitter. If the UXFIFO runs empty, the transmitter will repeatedly transmit the last character.

Asynchronous Operation

The asynchronous operation of the USART is selected by programming the ASYC bit in the USART mode register (UMR) to '1'. The character format and the number of stop bits are programmable in the AICR register. **Figure 30** shows the asynchronous character format.

Figure 30
Asynchronous Character Format



AICR		
CHL1	**CHL0**	**Character Length**
0	0	8
0	1	7
1	0	6
1	1	5

UMR			
PTY	**PY1**	**PY0**	**Parity Bit**
0	X	X	None
1	0	0	Fixed '0'
1	0	1	ODD Parity
1	1	0	Even Parity
1	1	1	Fixed '1'

AICR:STP	No. of Stop Bits
0	1
1	2

The USART receiver will check only the first stop bit. The setting of the STP bit affects just the USART transmitter.

The character format will be latched when the USART block is enabled or a reset command has been issued. To change the character format on the fly it is necessary to issue a reset command after changing the AICR or UMR value. There must be no reset command if the total number of bits between the start and the stop bits remain the same (e.g. 8 bit, no parity → 7 bit, odd parity).

Function of the Receiver

When enabled (UMR:UREN = 1), the receiver searches for valid characters (startbit, character bits, stop bit) and stores the characters in the URFIFO, stripping off the start and stop bit(s). A possible parity bit is stored in the URFIFO if the UFF bit is set to '0'. The parity bit is stored as the bit immediately on the left of the MSB. Thus the parity bit is transfered to the microprocessor. The parity bit is not available for the microprocessor if the character format is 8 data bits + parity. **Table 28** shows the character format in the URFIFO.

Table 28
Character Format in URFIFO

Character Length	Parity	SCR:UFF	Character Format in URFIFO
5	no	X	
	yes	0	0 0 0 D4 D3 D2 D1 D0
	yes	1	0 0 P D4 D3 D2 D1 D0
6	no	X	
	yes	0	0 0 D5 D4 D3 D2 D1 D0
	yes	1	0 P D5 D4 D3 D2 D1 D0
7	no	X	
	yes	0	0 D6 D5 D4 D3 D2 D1 D0
	yes	1	P D6 D5 D4 D3 D2 D1 D0
8	no	X	
	yes	X	D7 D6 D5 D4 D3 D2 D1 D0

If a parity error or framing error has been detected on a received character, the character is marked internally on the URFIFO. The corresponding status bit is set after the character has been read from the URFIFO.

A break signal is received if M contiguous start bits are received if the USART receiver is connected to the DCE interface or after 2M-2 contiguous start bits if the USART receiver is connected to the IRC. The break begin and break end is indicated in two status bits in the Special Condition Status.

Function of the Transmitter

When the transmitter is enabled (UMR:UXEN = 1), the transmission of any character entered in the UXFIFO is immediately started. Characters are transmitted with the LSB first and separated by the selected number of stop bits. Characters may be entered in the UXFIFO even if the transmitter is not enabled. A status bit (USXE) indicates if the transmitter is empty. When the transmitter is empty contiguous '1' is transmitted until a new character is available.

Asynchronous Bit Transparent Operation

The bit transparent mode of the USART is similar to the transparent mode in the synchronous operation. All data is stored as octets without regards of a character format. The programmed character length, parity bit and number of stop bits have no significance. The receive clock is synchronized to every falling edge of the input signal. The bit transparent mode is entered by setting the BTM bit in the UMR register to '1'.

FIFO Handling

URFIFO Handling

The USART receive FIFO (URFIFO) controls three status bits. Two additional status bits are used to indicate a framing error or parity error. The status bits and their meaning are shown in **figure 31** and **table 29**.

Figure 31
USART Receiver

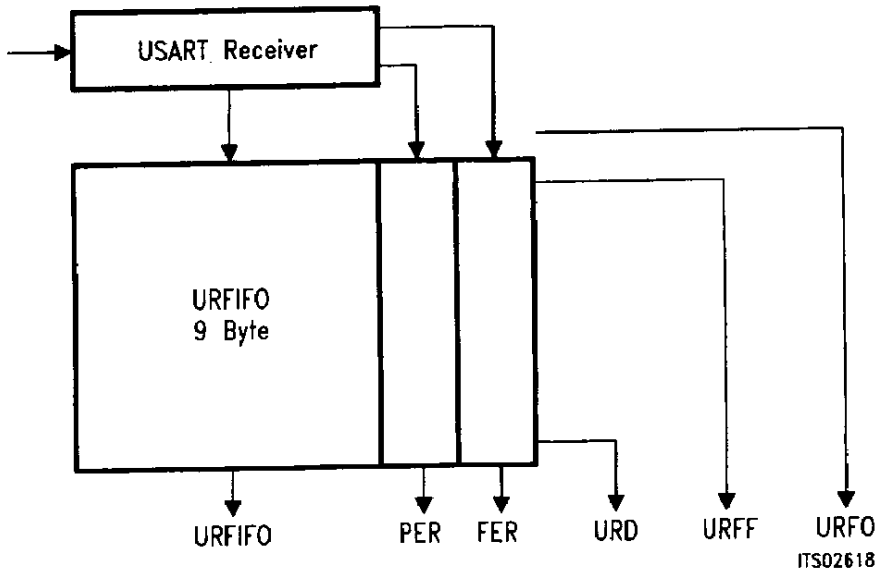


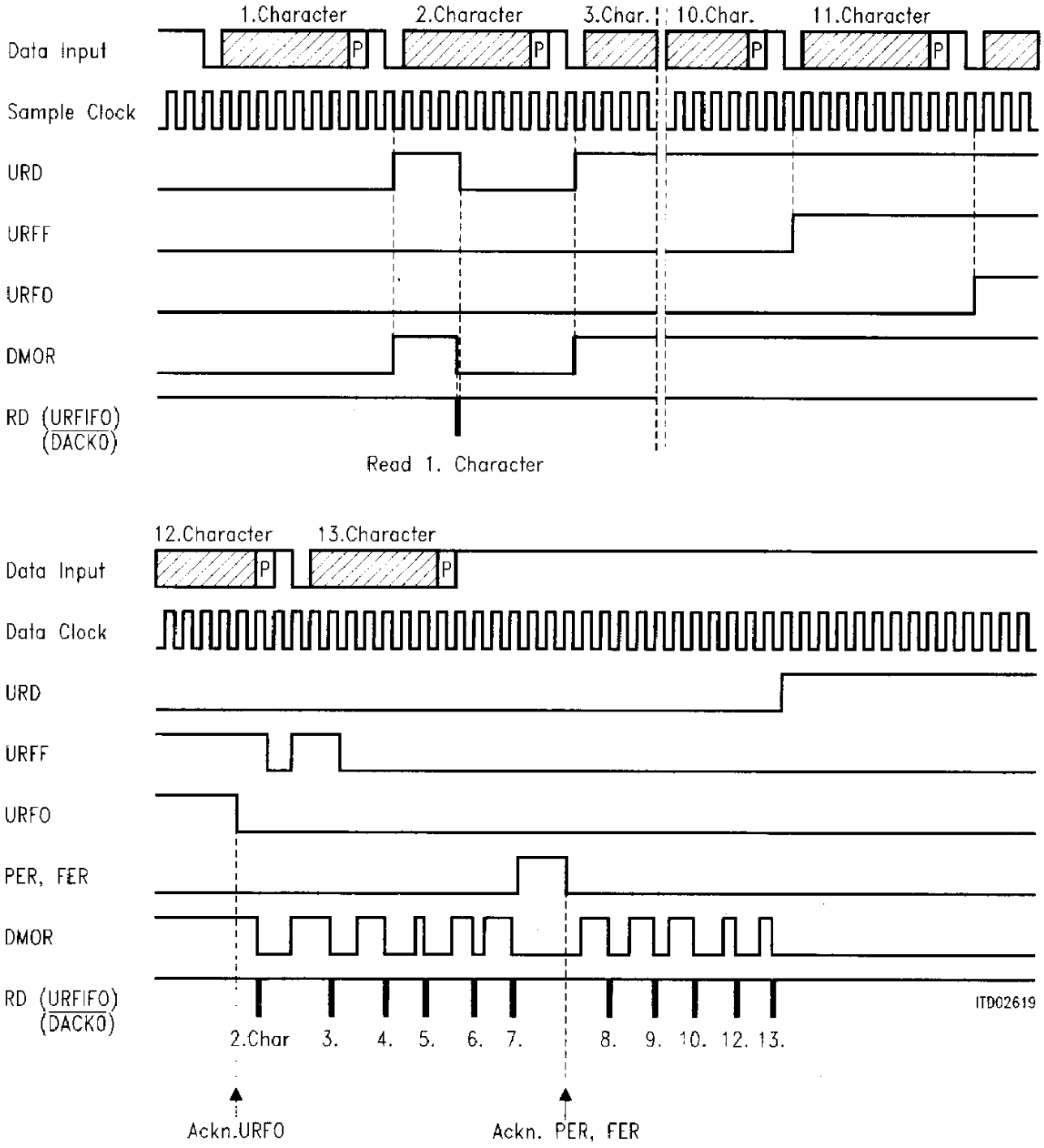
Table 29
USART Receive FIFO Indications

Status Bit	Register	Acknowledgement	Description
URD	RFS	read URFIFO	At least one byte is available in the URFIFO.
URFF	RFS	read URFIFO	All nine bytes in the URFIFO are occupied. A block of nine bytes may read from the URFIFO.
URFO	SCS	write to SCS	A character which was completed in the shift register could not be stored in the URFIFO because the URFIFO is full.
PER	RFS	write to RFS	A parity error has been detected on the last byte read from the URFIFO.
FER	RFS	write to RFS	A framing error has been detected on the last byte read from the URFIFO.

Figure 32 illustrates the URFIFO handling for an asynchronous data format of 7 bit + parity + 1 stopbit. The 11th character got lost because the URFIFO was full.

The DMA output request line DMOR will only go active if DMA is selected for the USART (GCR:DMA = 1 and GCR:DMH = 0). The DMOR line goes inactive after each read operation to the URFIFO. It will go active again with the next rising edge of the internal data clock, if the read operation occured during the first half of the clock cycle or one clock period later if the read operation occured during the second half of the clock cycle. The DMOR line will not go active while the PER or FER status bit is set and the corresponding interrupt is enabled in the RFIE register.

Figure 32
URFIFO Handling



UXFIFO Handling

The USART transmit FIFO (UXFIFO) controls three status bits. The status bits and their meaning are shown in **figure 33** and **table 30**.

Figure 33
USART RTransmitter

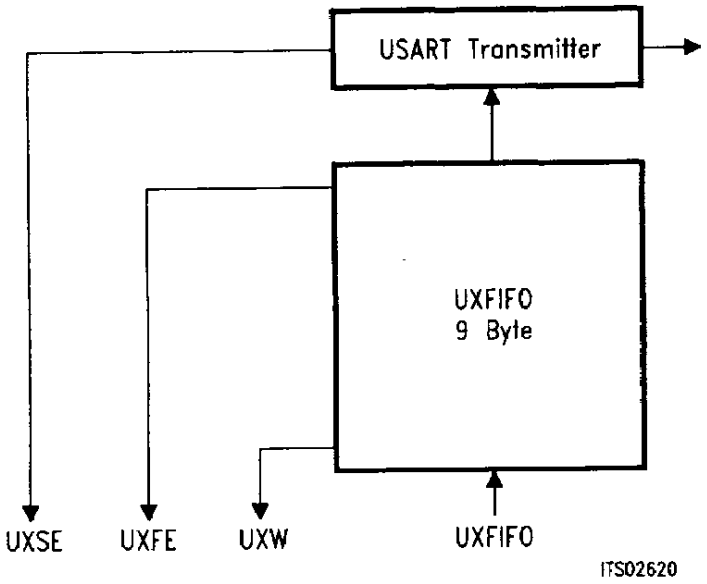


Table 30
USART Transmit FIFO Indications

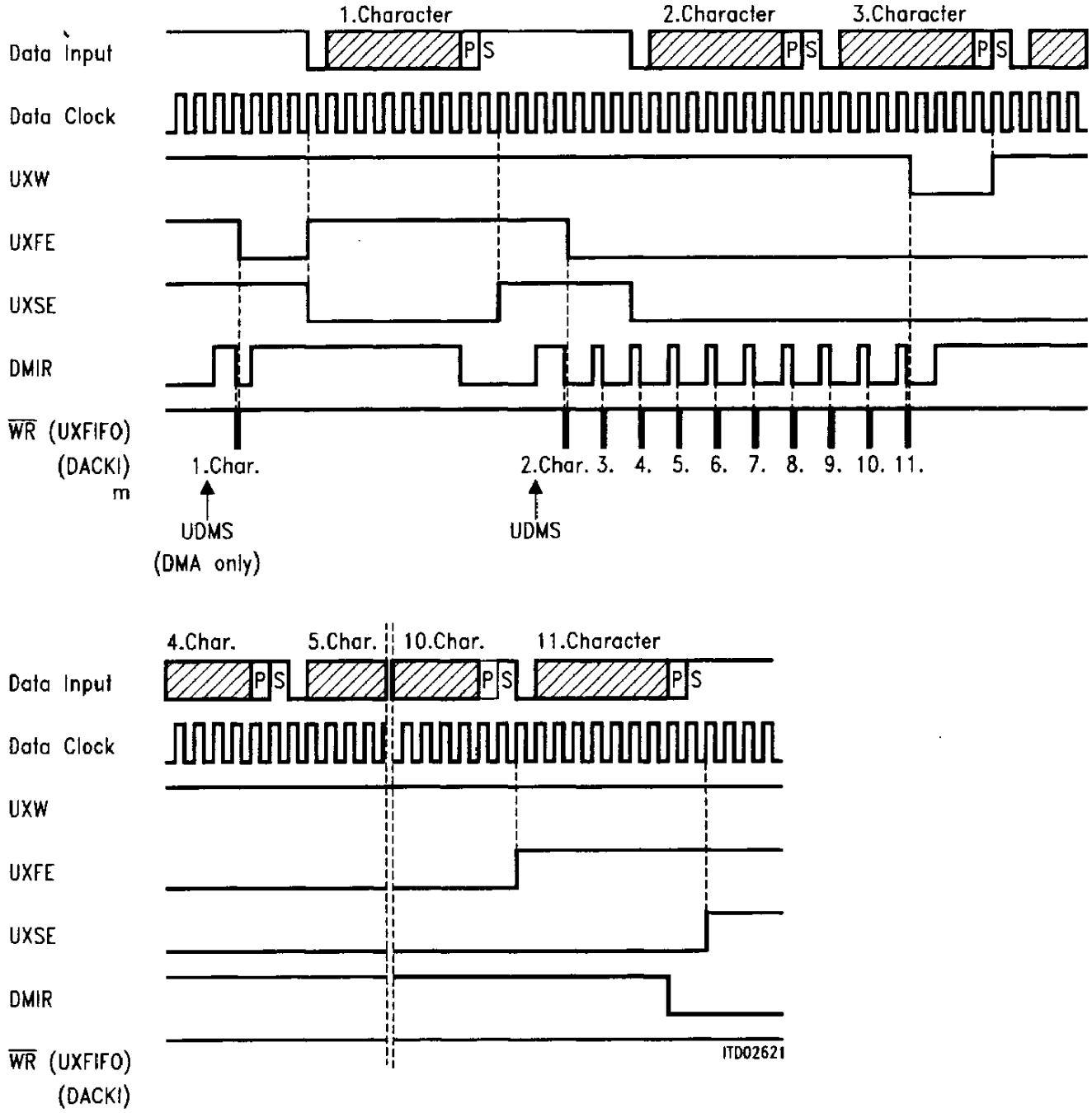
Status Bit	Register	Acknowledgement	Description
UXW	XFS	write UXFIFO	At least one byte can be written to the UXFIFO.
UXFE	XFS	write UXFIFO	The UXFIFO is empty. A block of nine bytes may be entered into the UXFIFO.
UXSE	XFS	write to XFS	The USART Transmit Shifter is empty. This bit is set after the last stop bit has been transmitted.

Figure 34 illustrates the UXFIFO handling for an asynchronous data format of 7 bit + parity + 1 stopbit.

The DMA input request line (DMIR) will only go active after a USART DMA-start command (UDMS) has been issued and DMA is enabled (GCR:DMA = 1 and GCR:DMH = 0). The DMIR line is synchronized to the internal data clock. It will go inactive after a write operation occured. It will go active again with the next clock cycle if the write operation occured in the first half of the clock cycle or one clock cycle later if the write operation occured in the second half of the clock signal.

If DMA operation is selected for the USART but only a small number of bytes has to be transfered, it may not be efficient to setup the DMA-controller. In this case, the UDMS command is also issued but the data is written into the UXFIFO under the control of the microprocessor (block-transfer). The DMA controller is disabled so that no data is transfered via DMA.

Figure 34
UXFIFO Handling

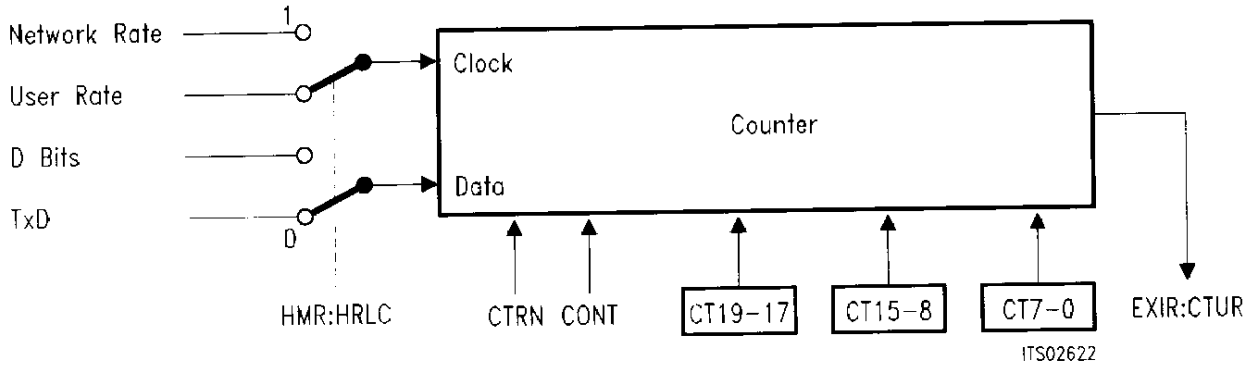


Counter

A programmable counter is connected in parallel to the USART receiver. It is 20 bits wide and has two modes of operation. In the first mode (CTV2:CONT = 1) it will count down the programmed number of clocks regardless of the data signal. This will generate periodical interrupts. In the second mode (CTV2:CONT = 0) it will count down the programmed number of stop bits. It will reload its value if a start bit has been received on the data signal. After the counter underruns (00 → FF) the CTUR status bit is set with the next clock. Thus the programmed value for the CTV0, CTV1, CTV2 registers is equal to the number of clocks – 2.

The CTRN bit in the CTV2 registers controls the counter. The counter is active while the CTRN bit is '1'. If CTRN = 0, the counter is disabled. Every change from CTRN = 0 to CTRN = 1 will reload the counter.

Figure 35
Counter



HDLC Controller

The HDLC controller consists of a transmitter and receiver block. It supports CCITT-CRC (x16+x12+x5+1). All data is transfered with the LSB first except for the CRC field. The CRC field is transfered with the MSB first. The input and output of the HDLC controller can be inverted by setting the HINV bit in the HDLC Mode Register (HMR).

Function of the HDLC Receiver

The functions performed by the HDLC receiver are:

- flag detection
- zero deletion
- CRC checking
- check for abort
- check for idle

When enabled (HMR:HREN = 1), the receiver enters a hunt phase and remains in the hunt phase until a valid opening flag is detected. The status of the receiver at this point is reflected by the RLA and the IDLE bit in the status register (STR).

Once a flag is recognized by the receiver, not followed by another flag or by an abort or idle sequence, all the subsequent eight-bit bytes are stored in the RFIFO up to the CRC field that immediately preceeds the closing flag. When the closing flag or an abort sequence (7 continuous '1') has been received, a receiver status byte is appended to the stored frame. This status byte contains status information pertaining to the received frame:

- Frame abort yes/no
- CRC error yes/no
- Data overflow yes/no
- Number of significant bits in the last received byte

Figure 36 shows the frame structure.

Figure 36
Received HDLC Frame

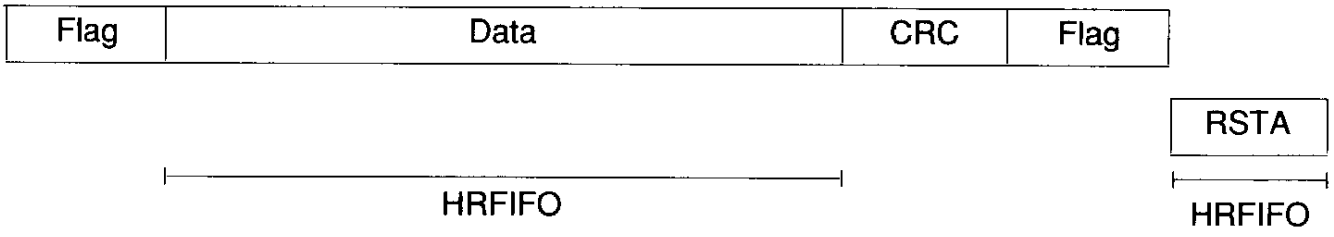


Table 31 shows the RSTA bits and their meaning. The RSTA value is always added to the data and forms the last byte of a frame.

Table 31
RSTA Status Bits

Bit Name	Bit No.	Description
RDO	7	**Receive Data Overflow** At least one byte of the current frame could not be stored due to occupied HRFIFO. The frame must be discard.
CRC	6	**CRC-Check Okay** This bit is set if the CRC-check was correct. If it is cleared the frame must be discard.
RAB	5	**Receive Abort** This bit is set if an abort sequence has been received.
VB2 - 0	2 - 0	**Valid Bit Count** Indicates the valid bits of the last byte before the RSTA value. The actual number of bits is the value of VB2-0 + 1.

Function of the HDLC Transmitter

The functions performed by the HDLC transmitter are:

- Flag generation
- Zero insertion
- CRC generation
- Abort sequence generation
- Interframe time fill generation

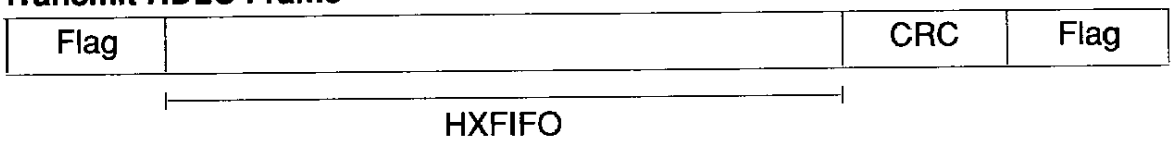
When the transmitter is enabled (HMR:HXEN = 1), the transmission of any bytes entered in the HXFIFO is immediately started, preceeded by an opening flag. To indicate the end of a frame, the microprocessor sets a control bit after having entered the last data byte of that frame in the HXFIFO. Every write operation of the bit serves as a frame delimiter. In DMA operation, this bit is controlled internally. The frame will be completed by the CRC-field and the closing flag if a data underrun condition is detected.

When no data is available in the HXFIFO and the transmitter is enabled, interframe time fill bits are transmitted. Depending on the ITF bit in the HDLC mode register (HMR) the interframe time fill are continuous non-shared flags (HMR:ITF = 1) or continuous '1' (HMR:ITF = 0).

When no DMA operation is selected and the HXFIFO becomes empty and the end-of-frame command was not issued, the transmitted frame is closed by an abort sequence and the HXDU status bit is set.

When DMA operation is selected, the HXDU status bit indicates the end of a frame and the data for the next frame may be entered.

Figure 37
Transmit HDLC Frame



HDLC FIFO Operation

Both HDLC transmitter and receiver have a nine byte FIFO. The FIFO for the HDLC transmitter is called HXFIFO, for the HDLC receiver HRFIFO. The FIFO bytes are not individually addressable. All write operations to any of the XFIFO addresses will be added to the end of the FIFO. All read operations to any of the HRFIFO addresses will read the first byte from the HRFIFO.

HRFIFO Handling

The HDLC receive FIFO (HRFIFO) controls three status bits. Two additional status bits indicate the end of a frame. The status bit and their meaning are shown in **figure 38** and **table 32**.

Figure 38
HDLC Receiver

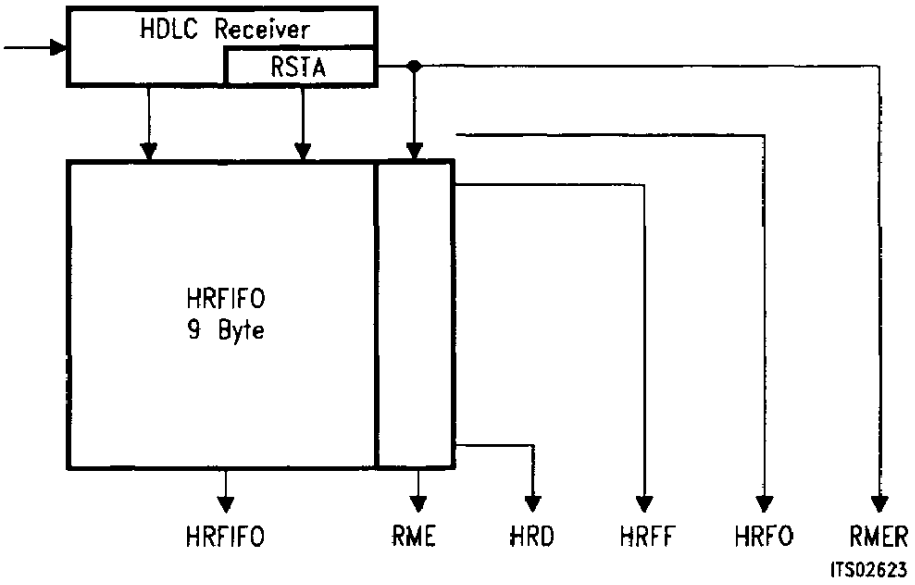


Table 32
HDLC Receive FIFO Indications

Status Bits	Register	Acknowledgement	Description
HRD	RFS	read HRFIFO	At least one byte is available in the HRFIFO.
HRFF	RFS	read HRFIFO	All nine bytes in the URFIFO are occupied. A block of nine bytes may be read from the HRFIFO.
HRFO	SCS	write to SCS	The start of a frame could not be stored in the HRFIFO due occupied HRFIFO. Thus at least one entire frame was lost.
RMER	RFS	write to RFS	Receive Message End Reception This bit is set after the receiver status byte was written into the HRFIFO.
RME	RFS	write to RFS	Receive Message End This bit is set after the last byte of a frame (RSTA) has been read from the HRFIFO.

The DMA output request line DMIR will only go active if DMA is selected for the HDLC controller (GCR:HMR = 1 and GCR:DMH = 1). The DMOR line goes inactive after each read operation to the HRFIFO. It will go active again with the next rising edge of the internal data clock if the read operation occured during the second half of the clock cycle. The DMOR line will not go active while the RME status bit is set.
The EODR pin will go active if DMA is selected.

HXFIFO Handling

The HDLC transmit FIFO (HXFIFO) controls three status bits. The status bits and their meaning are shown in **figure 39** and **table 33**.

Figure 39
HDLC Transmitter

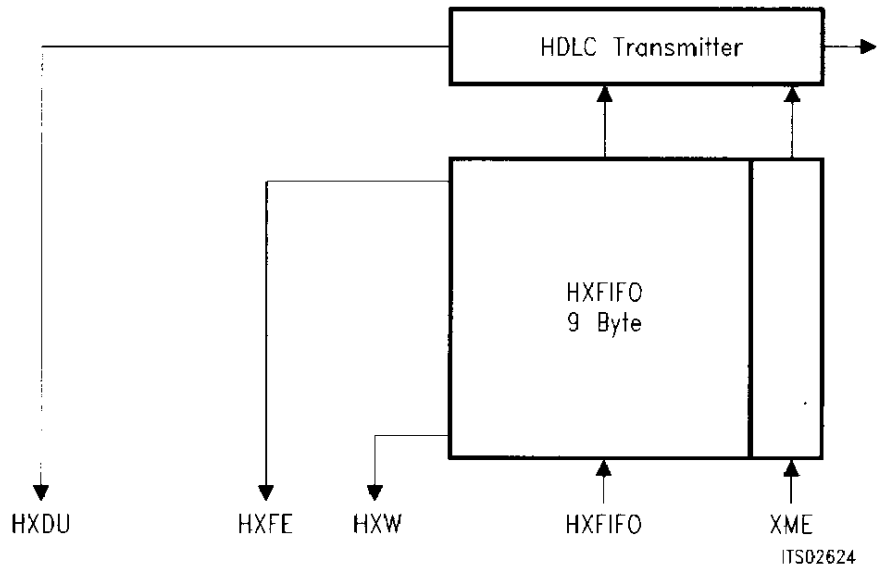


Table 33
HDLC Transmit FIFO Indications

Status Bits	Register	Acknowledgement	Description
HXW	XFS	write HXFIFO	At least one byte can be written to the HXFIFO.
HXFE	XFS	write HXFIFO	The HXFIFO is empty. A block of nine bytes may be entered into the HXFIFO.
HXDU	XFS	write to XFS	non-DMA operation: The HDLC transmitter became empty without an end-of-frame command. The frame was aborted. DMA operation: The HDLC transmitter has completed a frame. A new frame may be written to the HXFIFO after entering a HDMS command. HXDU must be acknowledge before the next HDMS command is issued.

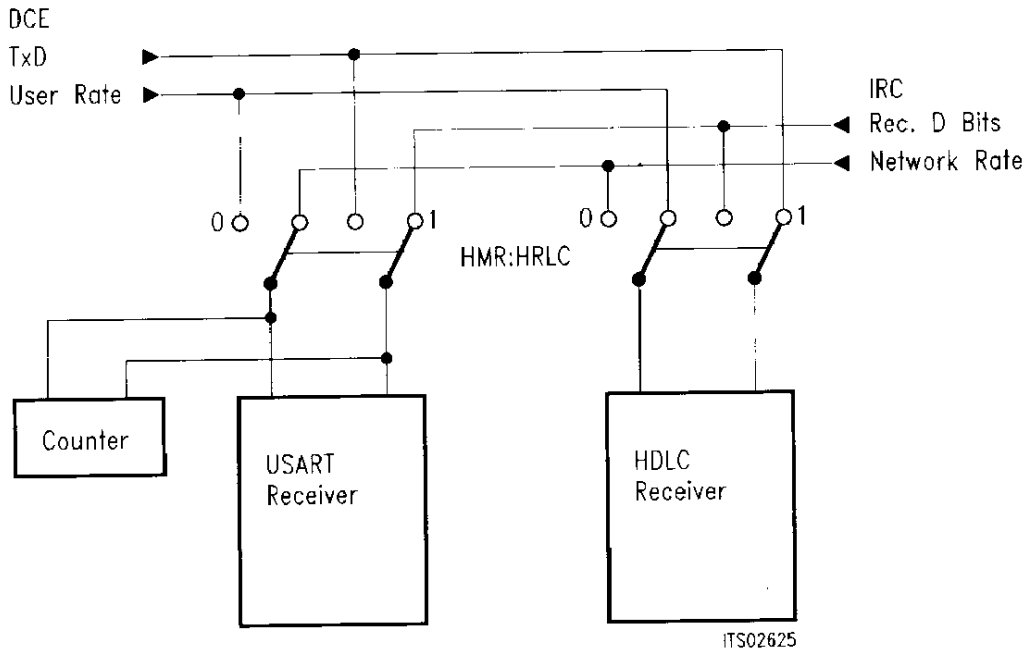
Data Path Selection for the SCL

Each receiver and transmitter block may either be connected to the DCE interface or to the IRC. The selection for the receiver blocks is done by the HMR:HRLC bit. If this bit is set to '0', the USART receiver, including the counter, is connected to the DCE interface. The HDLC receiver is than connected to the IRC receiver. If HMR:HRLC is set to '1', the connections are vice versa.

Table 34
Receiver Data Path Selection

HMR:HRLC	USART Receiver/ Counter	Data Signal	Data Clock	HDLC Receiver	Data Signal	Data Clock
0	DCE interface	TxD	user rate	IRC receiver	received 'D' bits	network rate
1	IRC receiver	received 'D' bits	network rate	DCE interface	TxD	user rate

Figure 40
Data Path for the Receiver Blocks

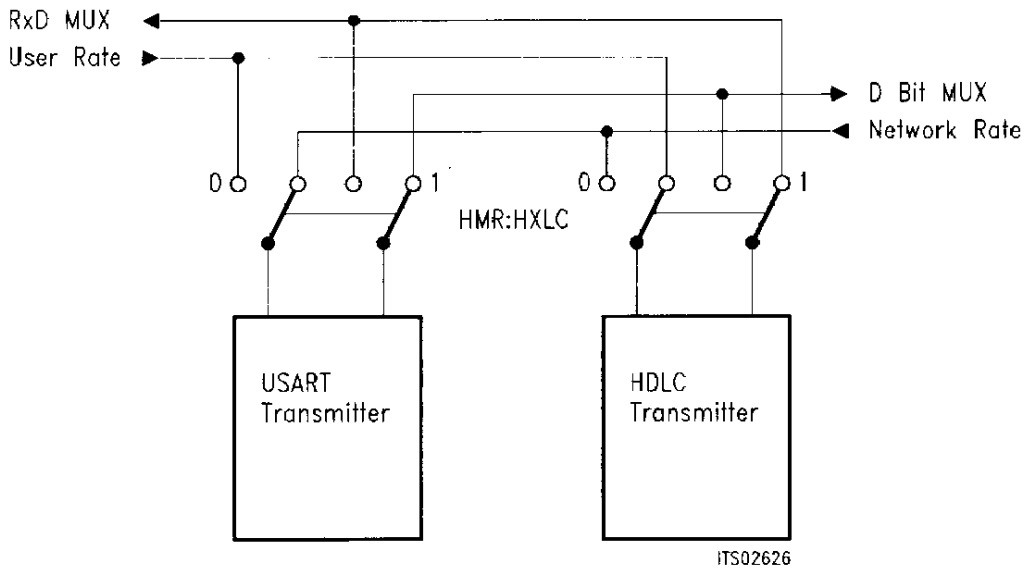


A similar switch is used to connect the transmitter blocks. The transmitter blocks are controlled by the HMR:HXLC bit.

Table 35
Transmitter Data Path Selection

HMR:HXLC	USART Transmitter	Data Clock	HDLC Transmitter	Data Clock
0	DCE	user rate	IRC	network rate
1	IRC	network rate	DCE	user rate

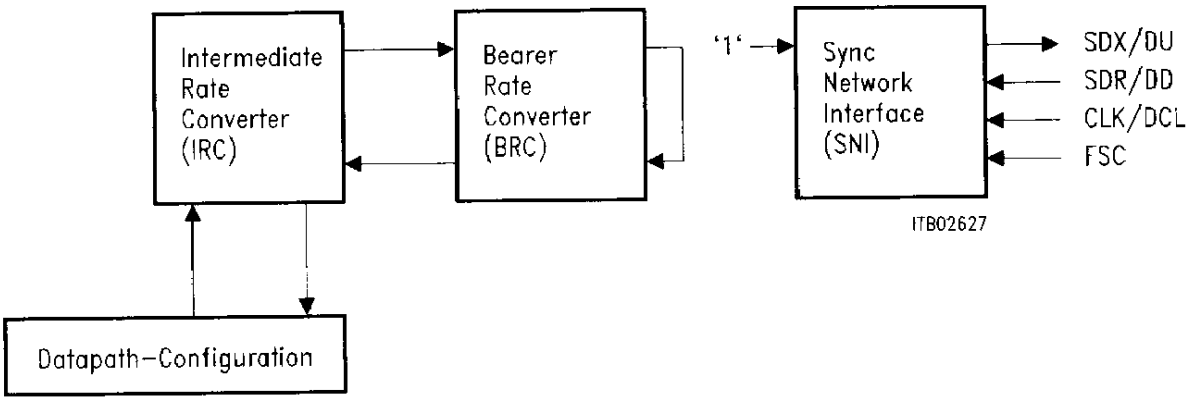
Figure 41
Data Path for the Transmitter Blocks



2.9 Test Loops

The ITAC provides three test loops. Test loop 3b connects the output of the IRC to its input. No data is output to the SDX/DU line. Data on the SDR/DD input has no effect on test loop 3b. To run test loop 3b it is necessary to have a clock and FSC signal on the SNI and the TSR register must be programmed to a valid time-slot within the frame. Test loop 3b is controlled by the TL3 bit in the DPCR register.

Figure 42
Test Loop 3b



Test loop 2b connects the output of the ASC (GCR:ASY=1) or the IRC (GCR:ASY=0) to its input. The signal on the RxD output is specified by the RDC1,0 bits of the DPCR register. Test loop 2b loops back the D-bit information of the received data stream. The S-, X-, E- and frame synchronization bits are not looped.

Figure 43
Test Loop 2b

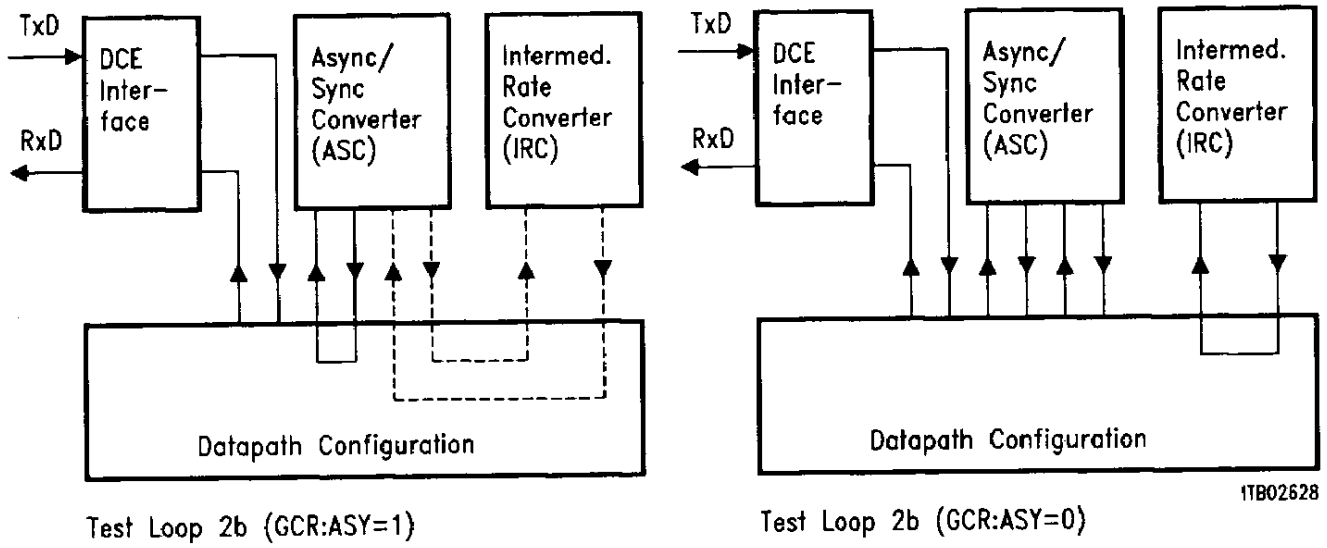


Table 36
Connections of RxD While Test Loop 2 is Active

RDC1	RDC0	Function
0	0	RxD is equal to the value of LDR:RD
0	1	RxD originates from the Serial Communication Logic
1	0	RxD transmits the received D bits (which are looped back)

A third test loop is available via a bit 0 of the test register 1. While bit 0 is '1', the data of the programmed time-slot received from the SNI is looped back after the BRC. At the same time, the output of the IRC is looped back to its receiver.

The BRC receives a continuous data stream from the SNI which has no indication of the start of the time slot. Thus, an octet aligned loop is not guaranteed. It is recommended to use HDLC formatted data to test this loop.

2.10 Interrupt Logic

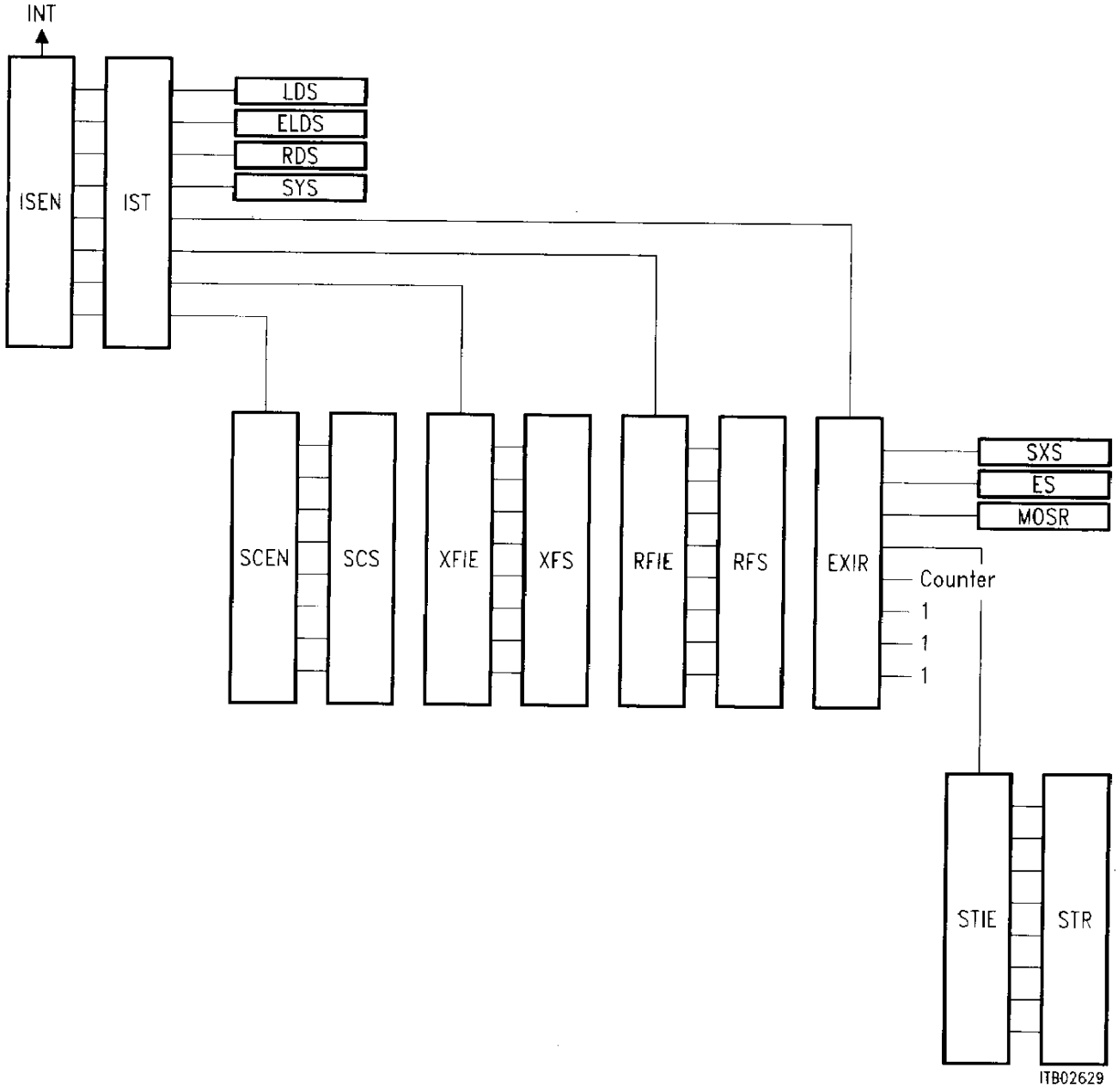
The status of a certain block or control lines are reported to the microprocessor via status registers. A change of any status bit can be used to activate the interrupt output (INT). This indicates to the microprocessor that it has to service the ITAC. The INT output is level active. If an interrupt is pending, the INT line is low. The INT output is open drain which allows to connect more than one interrupt sources to the input of the microprocessor.

Interrupt Structure

Figure 44 shows the interrupt structure. The INT output is controlled by the Interrupt status enable register (ISEN). A '1' in a bit position of the ISEN enables the corresponding status bit of the IST to generate an interrupt.

The four most significant bits of the IST register are controlled by their own enable registers. These bits are only set if a bit in the corresponding status register is active and cleared after the corresponding status bit has been cleared or the interrupt has been disabled.

Figure 44
Interrupt Structure

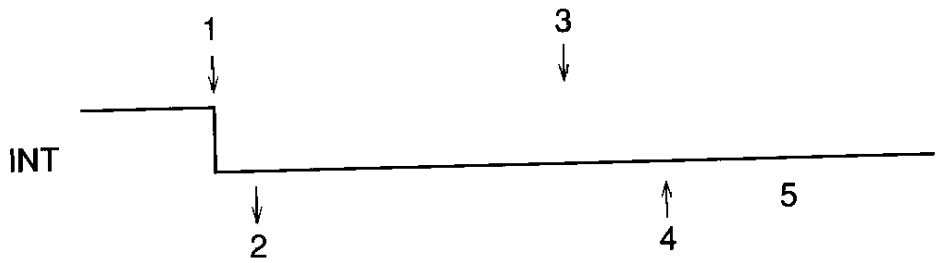


Control of Edge -Triggered Interrupt Controllers

The INT output is level active. It stays active until all interrupt sources have been serviced. If a new status bit is set while an interrupt is serviced, the INT line stays active. This may cause problems if the ITAC is connected to edge-triggered interrupt controller (**figure 45**).

To avoid these problems, it is recommended to mask all interrupts at the end of the interrupt service program and to enable the interrupts again. This is done by writing '00' to the ISEN register and to write back the old value of the ISEN register (**figure 46**).

Figure 45
INT Handling



1 A status bit is set. This causes an interrupt.

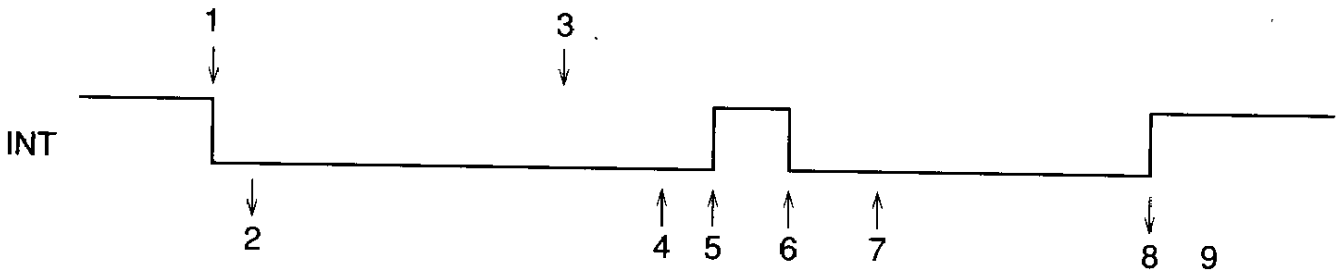
2 The microprocessor starts its service routine and reads the status registers.

3 A new status bit is set before the first status bit has been acknowledged.

4 The first status bit is acknowledged.

5 The INT output stays active but the interrupt controller will not serve the interrupt edge (edge triggered).

Figure 46
Service Program for Edge -Triggered Interrupt Controllers



1 to 4 see above.

5 00 is written to the ISEN register. This masks all interrupts and returns the INT output to its inactive state.

6 The old value is written to the ISEN register. This will activate the INT output if an interrupt source is still active.

7 The microprocessor starts a new interrupt service program.

8 The last status bit is acknowledged.

9 The INT output is inactive.

2.11 Reset State

The ITAC is in the reset state after the application of a reset pulse on RST. The minimum pulse width is 2 microseconds.

In the reset state, all configuration registers are zeroed and all interrupts are disabled. The synchronous network interface and the DCE interface outputs are tri-state.

2.12 Standby State

The ITAC may be set in a standby state to save unnecessary power consumption when idle. This is controlled by the power-up bit in the GCR register. If GCR:PU = 1 the ITAC is in normal operation. All blocks are active. If GCR:PU = 0 the ITAC is in standby mode. Depending on the selected DCE interface mode, two cases are distinguished.

The characteristics of the standby status are shown in **table 37**.

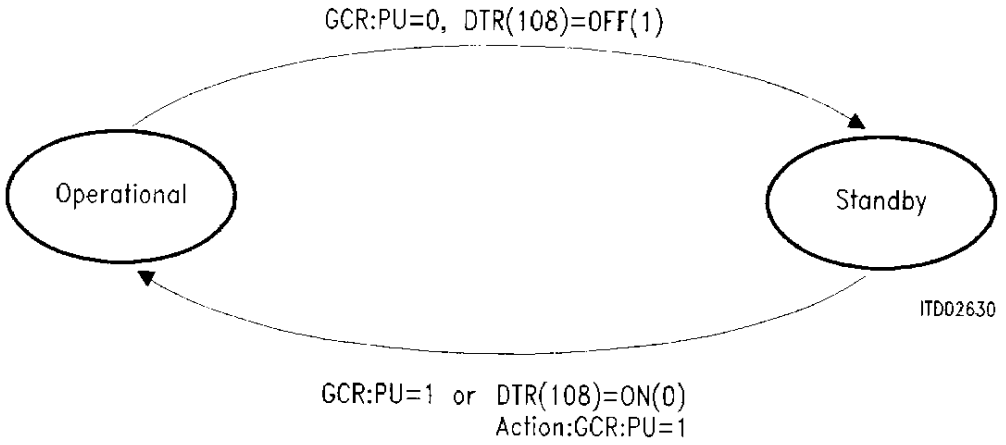
Table 37
Standby Status

Output or Function	X.21 (GCR:V24 = 0)	V.24 (GCR:V24 = 1)
Microprocessor Interface	Operational	Operational
Oscillator	Disabled	Disabled
Other Logic	Disabled	DTR detection enabled
RxD	Previous state	Previous state
S	Logical 1 or tri-state	Logical 1 or tri-state
Other DCE outputs	Previous state	Previous state
SDX	Tri-state	Tri-state

For an X.21 interface, switching between standby and operational is only subject to the state of the control bit PU (Power Up). The oscillator reaches a stable state within 10 ms after the PU bit is set to one.

In the V.24 case, standby is reached when PU is set to zero and DTR (108) interchange circuit is OFF. When DTR is switched ON, the standby state is left, an interrupt is generated and the PU bit is set to one by internal logic (**figure 47**).

Figure 47
Standby State Diagram (GCR:V24 = 1)



2.13 Initialization

After reset, the user has to write a minimum number of registers to set the ITAC into an operational state. The most important among these are:

- General Configuration Register (GCR)
- Bit Rate Selection (BRS)
- Data Path Selection Register (DPCR)
- Local DCE Interface Register (LDR)

Interrupts are enabled via the ISEN register.

The microprocessor may switch the ITAC into operational state at any time, not necessarily at initialization - unless a clock is required by the local DTE - but say, upon detecting an incomming data call. Also the automatic wake-up feature can be used in the case of an outgoing call. Note that switching between standby and operational does not affect the contents of the registers.

2.14 Internal Delays

The delay through the different block of the ITAC depends on there FIFO structures and functions. **Table 38** and **39** shows the delays.

Table 38
Transmit Delays (DCE → SNI)

Block	No. of Clock		Relevant Clock Rate
	typ.	max.	
ASC		16	User rate
IRC		9	Network rate
BRC		8	Intermediate rate

Table 39
Receive Delays (SNI → IRC)

Block	No. of Clocks		Relevant Clock Rate
	typ.	max.	
BRC		8	Intermediate rate
IRC		9	Network rate
ASC		2M+16	User rate

3 Detailed Register Description

The parameterization of the ITAC and the transfer of data and control information between the µP and the ITAC is performed through a set of registers.

Table 40 to **48** list the address map of the ITAC registers and the summary according to the type of the registers.

In order to facilitate a direct connection to 16-bit processors, all registers of the ITAC can be accessed using even or odd microprocessor addresses.

Table 40
ITAC Address Map

Address (hex)	Read		Write	
	Name	**Description**	**Name**	**Description**
00 - 0F	HRFIFO	HDLC Receive FIFO	HXFIFO	HDLC Transmit FIFO
10 - 1F	URFIFO	USART Receive FIFO	UXFIFO	USART Transmit FIFO
20, 21	IST	Interrupt Status Register	ISTA	IST Acknowledge Register
22, 23	EXIR	Extended Interrupt Register	EXIA	EXI Acknowledge Register
24 - 2B		reserved		
2C, 2D		reserved	TEST2	Test Register 2
2E, 2F		reserved	TEST1	Test Register 1
30, 31	LDS	Local DTE Status Register	CTV0	Counter Value 0
32, 33	ELDS	Extended Local DTE Status Register	CTV1	Counter Value 1
34, 35	RDS	Remote DTE Status Register	CTV2	Counter Value 2
36, 37	SYS	Synchronization Status Register		reserved
38, 39	SXS	S -and X-Bit Status Register		reserved
3A, 3B	ES	E-Bit Status Register		reserved
3C, 3D	RFS	Receiver and RFIFO Status Register	RFSA	RFS Acknowledge Register
3E, 3F	XFS	Transmitter and XFIFO Status Register	XFSA	XFS Acknowledge Register
40, 41	SCS	Special Condition Status Register	SCSA	SCS Acknowledge Register
42, 43	MOSR	MONITOR Channel Status Register	MOSA	MOS Acknowledge Register
44, 45	STR	Status Register	STRA	STR Acknowledge Register
46, 47	STIE	Status Register Interrupt Enable Register		
48, 49	ISEN	Interrupt Status Enable Register		
4A, 4B	RFIE	Receiver and RFIFO Interrupt Enable Register		
4C, 4D	XFIE	Transmitter and XFIFO Interrupt Enable Register		
4E, 4F	SCIE	Special Condition Interrupt Enable Register		
50, 51	GCR	General Configuration Register		
52, 53	SCR	Special Configuration Register		

Table 40 (cont'd)

Address (hex)	Read		Write	
	Name	**Description**	**Name**	**Description**
54, 55	AICR	Async. Interface Configuration Register		
56, 57	DPCR	Data Path Configuration Register		
58, 59	HMR	HDLC Mode Register		
5A, 5B	UMR	USART Mode Register		
5C, 5D	MOCR	MONITOR Channel Configuration Register		
5E, 5F	BRS	Bit Rate Select		
60, 61	TSR	Time-Slot Register		
62, 63	NFR	Number of Retry Frames Register		
64, 65	LDR	Local DCE Control Register		
66, 67	RDR	Remote DCE Control Register		
68, 69	XER	Transmit E-Bit Register		
6A, 6B		reserved		
6C, 6D		reserved	HCC	HDLC Controller Command Register
6E, 6F		reserved	UCC	USART Controller Command Register
70, 71	INSC	Insert Character Register		
72, 73		reserved		
74, 75	MOR1	MONITOR Channel Receive Register	MOX1	MONITOR Channel Transmit Register
76, 77	SYN	Synchronization Character Register		
78, 79	LCAR1	Local Character 1		
7A, 7B	LCAR2	Local Character 2		
7C, 7D	RCAR1	Remote Character 1		
7E, 7F	RCAR2	Remote Character 2		

Table 41
FIFO and Transfer Register Summary

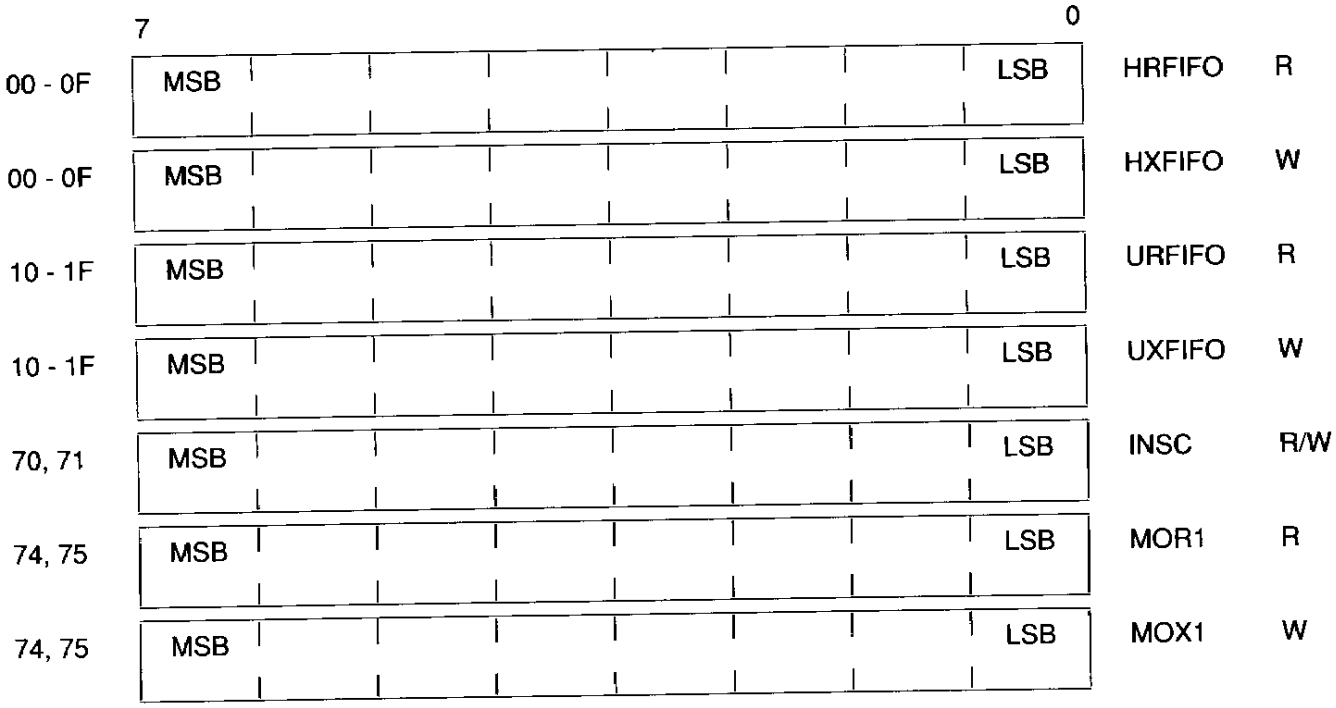


Table 42
Status Register Summary

	7							0		
20, 21	LDC A	ELDC A	RDC A	SYC A	EXI	RF	XF	SC	IST ISTA	R W
22, 23	SXC A	ESC A	MOSC	STRC	CTUR A	1	1	1	EXIR EXIA	R W
30, 31	DTR	RTS	MI1	MI2	MI3	1	1	1	LDS (V24=1)	R
30, 31	LON0	LOF01	LL3	0	0	1	1	1	LDS (V24=0)	R
32, 33	LONX	LOFX	LON1	LOF1	LOF0	LL2	1	1	ELDS	R
34, 35	RONX	ROFX	RON1	ROF1	ROF0	RL2	1	1	RDS	R
36, 37	FSL	RSI	RSS	0	1	VN2	VN1	VN0	SYS	R
38, 39	RS/RSA	RSB	RX	1	1	1	1	1	SXS	R
3A, 3B	RE1	RE2	RE3	RE4	RE5	RE6	RE7	1	ES	R
3C, 3D	HRD	URD	HRFF	URFF	RMER A	RME A	FER A	PER A	RFS RFSA	R W
3E, 3F	HXW	UXW	HXFE	UXFE	HXDU A	UXSE	1	1	XFS XFSA	R W
40, 41	HRFO A	URFO A	BRB A	BRE A	LC1 A	LC2 A	RC1 A	RC2 A	SCS SCSA	R W
42, 43	1	1	1	1	MDR A	MER A	MDA A	MAB A	MOSR MOSA	R W
44, 45	CAC	CIS	RLA	IDLE	RCHR A	RDB A	XDB A	OVS A	STR STRA	R W
– –	RDO	CRC	RAB	0	0	VB2	VB1	VB0	RSTA (HRFIFO)	

Table 43
Interrupt Enable Register Summary

	7							0		
46, 47	Enable CAC	Enable CIS	Enable RLA	Enable IDLE	Enable RCHR	Enable RDB	Enable XDB	Enable OVS	STIE	R/W
48, 49	Enable LDC	Enable ELDC	Enable RDC	Enable SYC	Enable EXI	Enable RF	Enable XF	Enable SC	ISEN	R/W
4A, 4B	Enable HRD	Enable URD	Enable HRFF	Enable URFF	Enable RMER	Enable RME	Enable FER	Enable PER	RFIE	R/W
4C, 4D	Enable HXW	Enable UXW	Enable HXFE	Enable UXFE	Enable HXDU	Enable UXSE			XFIE	R/W
4E, 4F	Enable HRFO	Enable URFO	Enable BRB	Enable BRE	Enable LC1	Enable LC2	Enable RC1	Enable RC2	SCIE	R/W

Table 44
Configuration Register Summary

	7							0		
50, 51	PU	DOE	V24	V110	ASY	ENFR	DMA	DMH	GCR	R/W
52, 53	TS5	DLL	LCS	RCS	F56	DCL	OD	SFS	SCR	R/W
54, 55	CHL1	CHL0	STP	TR	URDF				AICR	R/W
56, 57	RDC1	RDC0	XDC1	XDC0	RSC	XSC	TL2	TL3	DPCR	R/W
58, 59	HRLC	HXLC	HREN	HXEN	HINV	ITF			HMR	R/W
5A, 5B	ASYC	PTY	UREN	UXEN	SCM	PY1	PY0	BTM	UMR	R/W
5C, 5D	IOM2	CIX6	CIX5	SAW	MRE	MRC	MIE	MXC	MOCR	R/W
5E, 5F	UR3	UR2	UR1	UR0	NR3	NR2	NR1	NR0	BRS	R/W
60, 61	TS4	TS3	TS2	TS1	TS0	ICS2	ICS1	ICS0	TSR	R/W
62, 63	N7	N6	N5	N4	N3	N2	N1	N0	NRF	R/W

Table 45
Control Register Summary

	7							0		
64, 65	DCD/I	DSR	CTS	RD	MO1	MO2			LDR	R/W
66, 67	XS/XSA	XSB	XX	XD					RDR	R/W
68, 69	XE1	XE2	XE3	XE4	XE5	XE6	XE7		XER	R/W
30, 31	CT7	CT6	CT5	CT4	CT3	CT2	CT1	CT0	CTV0	W
32, 33	CT15	CT14	CT13	CT12	CT11	CT10	CT9	CT8	CTV1	W
34, 35			CTRN	CONT	CT19	CT18	CT17	CT16	CTV2	W

Table 46
Command Registers

	7							0		
6C, 6D	HRR	HXR	XME	HDMS					HCC	W
6E, 6F	URR	UXR	0	UDMS	SBX	HNT	TRA		UCC	W

Table 47
Constant Registers

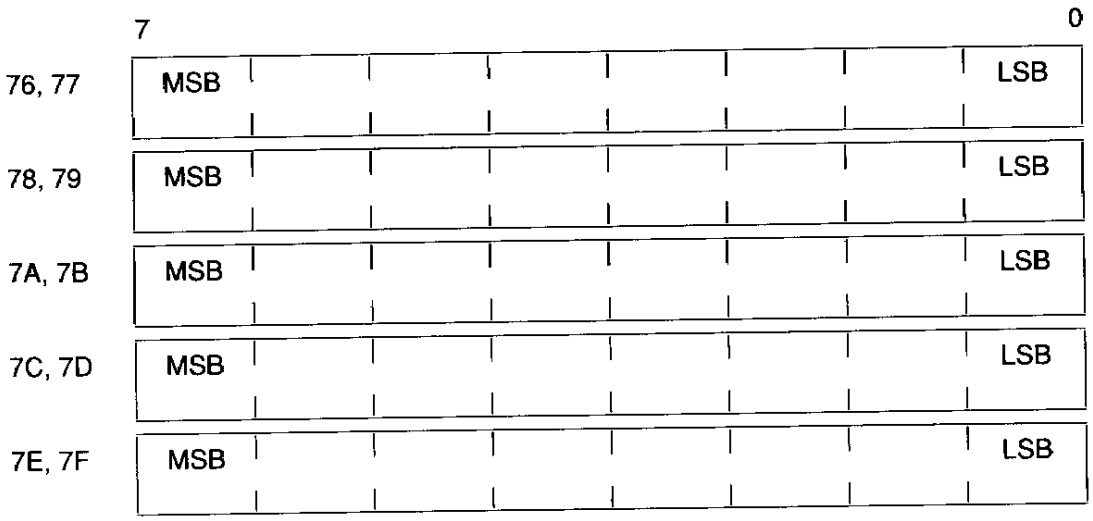
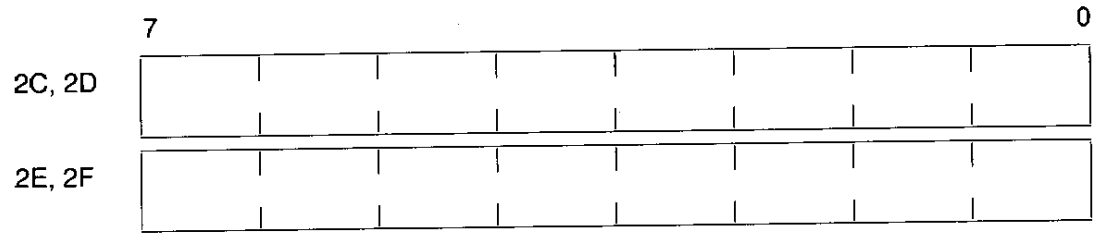


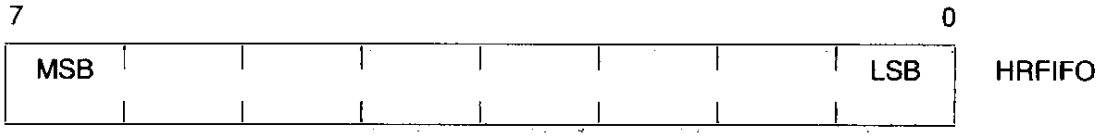
Table 48
Test Registers



3.1 FIFO and Transfer Registers

HDLC Receive FIFO (HRFIFO) Address: 00 - 0FH Read

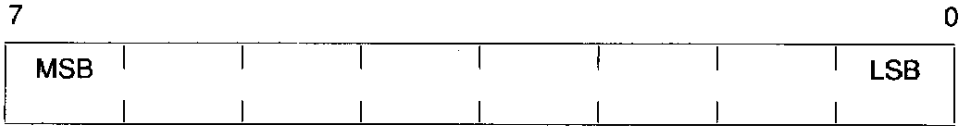
Value after Reset: FFH



Any address combination between 00H and 0FH will read the next byte from the HDLC receive FIFO.

USART Receive FIFO (URFIFO) Address: 10 - 1FH Read

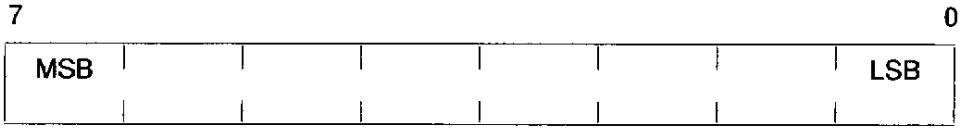
Value after Reset: FFH



Any address combination between 10H and 1FH will read the next byte from the USART receive FIFO.

HDLC Transmit FIFO (HXFIFO) Address: 00 - 0FH Write

Value after Reset: FFH



Any address combination between 00H and 0FH hex will write at the end of the HDLC transmit FIFO.

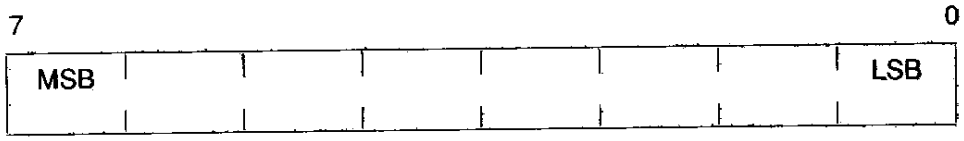
USART Transmit FIFO (UXFIFO) Address: 10 - 1FH Write

Value after Reset: FFH

Any address combination between 10H and 1FH will write at the end of the USART transmit FIFO.

Character Insert Register (INSC) Address: 70, 71H Read/Write

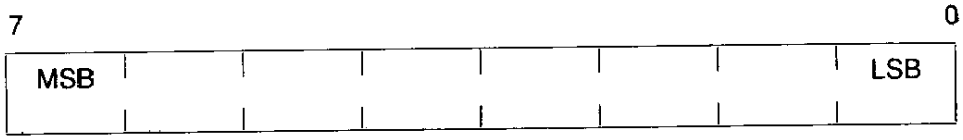
Value after Reset: 00H



The character written into the INSC register will be inserted in the data stream towards the DCE interface at the next opportunity. Received characters are not pertubed. The status of inserting the character is indicated by the STR:CIS status bit.

MONITOR Data Receive Register (MOR1) Address: 74, 75H Read

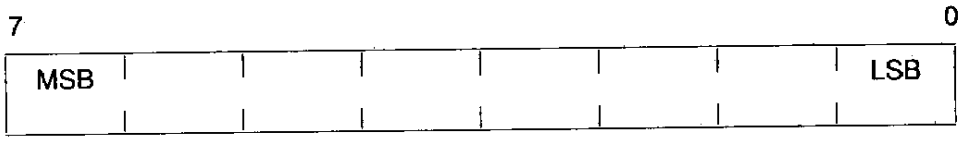
Read Value after Reset: 00H



Contains the last byte received from the MONITOR channel handler.

MONITOR Data Transmit Register (MOX1) Address: 74, 75H Write

Value after Reset: 00H



The value written into the MOX1 register will be transmitted by the MONITOR channel handler if the MOCR:MXC bit is '1'.

3.2 Status Registers

Status bits of the ITAC are not cleared after the corresponding register has been read. Status bits with the indication 'A' have to be acknowledged. This means that a '1' has to be written into the register address to reset the status bit. The acknowledgement of a status bit has to occur before the status is evaluated and the pointed register is read. This procedure asures that no status change gets lost.

Interrupt Status Register (IST) Address: 20, 21H Read

Value after Reset: 00H

7							0
LDC A	ELDC A	RDC A	SYC A	EXI	RF	XF	SC

LDC Local DTE Status Change

Indicates that at least one bit in the LDS register has changed since the last ac knowledgement. This status bit must be acknowledged.

ELDC Extended Local DTE Status Change

Indicates that at least one bit in the ELDS register has changed since the last acknowledgement. This status bit must be acknowledged.

RDC Remote DTE Status Change

Indicates that at least one bit in the RDS register has changed since the last acknowledgement. This status bit must be acknowledged.

SYC Synchronization Status Change

Indicates that at least one bit in the SYS register has changed since the last acknowledgement. This status bit must be acknowledged.

EXI Extended Interrupt

Indicates that at least one bit is set in the EXIR register.

RF Receiver and RFIFO Status

Indicates that at least one bit in the RFS register has changed since the read operation.

XF **Transmitter and XFIFO Status**

Indicates that at least one bit in the XFS register has changed since the read operation.

SC **Special Condition Status**

Indicates that at least one bit in the SCS register has changed since the read operation.

EXtended Interrupt Register (EXIR) Address: 22, 23H Read

Value after Reset: 07H

7							0
SXC A	ESC A	MOSC	STRC	CTUR A	1	1	1

SXC **S-and X-Bit Status Change**

The SXC bit is set if a change has occured on the received S- or X-bits. This status bit must be acknowledged.

ESC **E-Bit Status Change**

The ESC bit is set if a change has occured on the received E-bits. The status bit must be acknowledged.

MOSC **MONITOR Channel Status Change**

The MOSC bit is set if a change has occured in the MONITOR status register. The status bit is cleared after the changes in the MOSR register are acknowledged.

STRC **Status Register Change**

The STRC bit is set, if a change in the STR register has occured and the corresponding interrupt enable bit is set. This bit is cleared after the changes in the STR register are acknowledged.

CTUR **Counter Underrun**

The CTUR status bit is set with a delay of one clock cycle after the counter underruns (00 → FF). When the counter underruns, it will automatically reload the value of the counter registers. The CTUR status bit must be acknowledged.

Local DTE Status Register (LDS) Address: 30, 31H Read

Value after Reset: --H

Non-X.21 (GCR:V24 = 1)

7							0
DTR	RTS	MI1	MI2	MI3	1	1	1

DTR **Data Terminal Ready interchange circuit state.**
1: DTR = 'OFF' (1)
0: DTR = 'ON' (0)

RTS **Request To Send interchange circuit state**
1: RTS = 'OFF' (1)
0: RTS = 'ON' (0)

MI1-3 Multifunctional Input circuit states
1: MIx = 'OFF' (1)
0: MIx = 'ON' (0)

X.21 (GCR:V24 = 0)

7							0
LON0	LOF01	LL3	0	0	1	1	1

LON0 Local (0, ON) state
LOF01 Local (0101..., OFF) state
LL3 Local Loop 3, Local (00001111..., OFF) state

Extended Local DTE Status (ELDS) Address: 32, 33H Read

Value after Reset: 03H

7							0
LONX	LOFX	LON1	LOF1	LOF0	LL2	1	1

*significant only in X.21 mode (GCR:V24 = 0)

LONX Local (X,ON) state
LOFX Local (X,OFF) state
LON1 Local (1,ON) state
LOF1 Local (1,OFF) state
LOF0 Local (0,OFF) state
LL2 Local Loop 2, Local (0011...,OFF) state

Remote DTE Status (RDS) Address: 34, 35H Read

Value after Reset: 03H

7							0
RONX	ROFX	RON1	ROF1	ROF0	RL2	1	1

RONX Remote (X,ON) state
ROFX Remote (X,OFF) state
RON1 Remote (1,ON) state
ROF1 Remote (1,OFF) state
ROF0 Remote (0,OFF) state
RL2 Remote Loop2, Remote (0011...,OFF) state

Synchronization Status (SYS) Address: 36, 37H Read

Value after Reset: 8DH

7							0
FSL	RSI	RSS	0	1	VN2	VN1	VN0

FSL **Frame Sync Loss**

The intermediate rate receiver has detected at least one or three consecutive frames with erroneous frame synchronization pattern. After frame sync loss was detected, the value programmed in NRF frames is counted down, before a RSI status is generated.

RSI **ReSynchronization Impossible**

The intermediate rate converter has not achieved synchronization within the number of frames which has been programmed in the NRF register. The IRC receiver continuous to search for the frame synchronization pattern.

RSS **ReSynchronization Successful**

The intermediate rate receiver has achieved synchronization after a FSL or RSI status.

VN2-VN0 Version Number

VN2	VN1	VN0	Version
1	1	1	A1
1	1	0	B1
1	0	1	V2.2

S- and X-Bit Status (SXS) Address: 38, 39H Read

Value after Reset:1FH

7							0
RS/RSA	RSB	RX	1	1	1	1	1

RS/RSA Received S- or SA-Bit
If (GCR:V110 = 0): S
If (GCR:V110 = 1): SA

1: received S/SA-bit was 1

0: received S/SA-bit was 0

RSB Received SB-Bit
*significant only if GCR:V110 =1

1: received SB-bit was 1

0: received SB-bit was 0

RX Received X-Bit

1: received X-bit was 1

0: received X-bit was 0

E-bit Status (ES) Address: 3A, 3BH Read

Value after Reset: 01H

7							0
RE1	RE2	RE3	RE4	RE5	RE6	RE7	1

RE1-7 Received E-Bits

RE7 is always 1 for NR = 600 bit/s

Receiver and RFIFO Status (RFS) Address: 3C, 3DH Read/Write

Value after Reset: 00H

7							0
HRD	URD	HRFF	URFF	RMER A	RME A	FER A	PER A

HRD HDLC Receive Data available

The HRFIFO contains at least one byte.

URD USART Receive Data available

The URFIFO contains at least one character/byte.

HRFF HDLC Receive FIFO Full

All 9 bytes of the HRFIFO are occupied. At least one byte must be read before the next one is completed in the shift register. Otherwise a SCS:HRFO status is generated.

URFF USART Receive FIFO Full

All 9 bytes of the URFIFO are occupied. At least one byte must be read before the next one is completed in the shift register. Otherwise a SCS:URFO status bit is generated.

RMER Receive Message End Reception

This bit is set after the receiver status byte was written into the HRFIFO. It indicates that the end of a frame is stored in the HRFIFO. This status bit must be acknowledged.

RME **Receive Message End**

This bit indicates, that the byte previously read from the HRFIFO was the last byte of a frame. This status bit must be acknowledged.

FER **Framing Error**
*active only if UMR:ASYC = 1 and HMR:HRLC = 0.

The USART receiver detected a framing error (missing stop bit) on the byte previously read from the URFIFO. This status bit must be acknowledged.

PER **Parity Error**
*active only if UMR:PTY = 1 while receiving the character.

The USART receiver detected a parity error on the byte previously read from the URFIFO. This status bit must be acknowledged.

Transmitter and XFIFO Status (XFS) Address: 3E, 3FH Read/Write

Value after Reset: 07H

7							0
HXW	UXW	HXFE	UXFE	HXDUA	UXSE	1	1

HXW **HDLC Transmit FIFO Write enable**

At least one more byte can be written into the HXFIFO.

UXW **USART Transmit FIFO Write enable**

At least one more character can be written into the UXFIFO.

HXFE **HDLC Transmit FIFO empty**

The HXFIFO is empty. Up to nine bytes can be written into the HXFIFO.

UXFE **USART Transmit FIFO empty**

The UXFIFO is empty. Up to nine bytes can be written into the UXFIFO.

HXDU HDLC Transmit Data Underrun

Non-DMA:
All data from the HXFIFO was transmitted and no HCC:XME (transmit message end) command was issued. The frame will be closed with an abort sequence. The HXDU status bit is set during the transmission of the last bit from the last byte.

DMA:
The HXDU bit indicates that the RME bit has been generated internally and the frame will be closed by the CRC field and the closing flag. A new frame may be started by writing the HDMS command after the HXDU bit has been acknowledged.

This status bit must be acknowledged.

UXSE USART Transmit Shifter Empty

The USXE bit indicates that the USART shifter is empty.

If asynchronous operation of the USART is selected (UMR:ASY = 1 and UMR:BTM = 0), the USXE bit will be set to '1' after the last stop bit has been transmitted and no further data is available in the UXFIFO.

If asynchronous bit transparent mode is selected (UMR:ASYC = 1, UMR:BTM = 1), the UXSE bit will be set after the last data bit has been transmitted and no further data is available in the UXFIFO.

If synchronous operation is selected (UMR:ASYC = 0), the USXE bit will set after the last bit of the last byte has been transmitted. During idle-state it will stay '1' although the last byte will be transmitted continuously.

The USXE bit will be cleared after a new data byte has been transfered from the UXFIFO to the shift register. The UXSE bit is active both in DMA and in non-DMA operation. The UXSE bit is not acknowledgeable. It will generate an interrupt, if XFSE:EXSE = 1 and ISEN:XF = 1.

Special Condition Status (SCS) Address: 40, 41H Read/Write

Value after Reset: 00H

7							0
HRFO A	URFO A	BRB A	BRE A	LC1 A	LC2 A	RC1 A	RC2 A

HRFO HDLC Receive FIFO Overflow

The start of a frame could not be stored in the HRFIFO due to a filled HRFIFO. Thus at least one entire frame was lost. This status bit must be acknowledged.

URFO USART Receive FIFO Overflow

A character which was completed in the receiver shift register could not be stored in the URFIFO due to a filled URFIFO. This status bit must be acknowledged.

BRB Break Signal Begin

This bit is set if the USART receiver has detected 2M-2 bits of start polarity if it is connected to the network side (IRC receiver) or M bits of start polarity if it is connected to the DCE interface. This status bit must be acknowledged.

BRE Break Signal End

This bit is set when the first stop bit is received by the USART receiver after a BRB status indication. This status bit must be acknowledged.

LC1 Local Character 1 recognized

The character programmed in the LCAR1 register has been recognized from the local DTE. This status bit must be acknowledged.

LC2 Local Character 2 recognized

The character programmed in the LCAR2 register has been recognized from the local DTE. This status bit must be acknowledged.

RC1 Remote Character 1 recognized

The character programmed in the RCAR1 register has been recognized from the received D-bits. This status bit must be acknowledged.

RC2 Remote Character 2 recognized

The character programmed in the RCAR2 register has been recognized from the received D-bits. This status bit must be acknowledged.

MONITOR Channel Status Register (MOSR) Address: 42, 43H Read/Write

Value after Reset: 00H

7							0
1	1	1	1	MDRA	MERA	MDAA	MABA

MDR MONITOR Channel Data Received

A new data byte is available in the MOR1 register. This status bit must be acknowledged.

MER MONITOR Channel End of Reception

The end of reception condition has been recognized by the MONITOR channel handler. This status bit must be acknowledged.

MDA MONITOR Channel Data acknowledge

The opposite side has acknowledged the transmitted data byte written into MOX1. This status bit must be acknowledged.

MAB MONITOR Channel Data abort

The opposite side has aborted the transmitted data written into MOX1. This status bit must be acknowledged.

Status Register (STR) Address: 44, 45H Read/Write

Value after Reset: --H

7							0
CAC	CIS	RLA	IDLE	RCHR A	RDB A	XDB A	OVS A

CAC Command Accepted

The bit must be polled before a HRR, HXR, HDMS, URR, UDMS, SBK, HNT and TRA command is issued. After a command was accepted, this bit is set. Writing any of the previous commands will set the CAC bit to '0' until the command is accepted. During this period no further command may be written into the HCC or UCC register.

CIS Character Insertion Successful

1: The character written to INSC has been inserted in the received character stream (to the DCE interface). Another character may be inserted using INSC.

0: The character written into INSC has not been inserted yet. A write operation to INSC may overwrite the insertion character.

RLA Receive Line active

1: HDLC flags or messages are being received by the HDLC receiver.

0: Neither flags nor messages are being received.

IDLE Idle State on the HDLC received data

1: At least 12 consecutive '1' have been detected by the HDLC receiver.

0: No idle state.

RCHR Receive Character detected

1: At least on start bit was detected from the DCE interface since the last acknowledgement.

0: Only stop bits have been received from the DCE interface since the last acknowledgement.
This status bit must be acknowledged.

RDB **Receive Data Byte (from SNI)**

This bit is set after every eighth bit in the received X.30/V.110 frame (P8, Q8, R8 or D8, D16, D24 ...) when SYS:RSS = 1.
This status bit must be acknowledged.

XDB **Transmit Data Byte (to the SNI)**

This bit is set after every eighth bit in the transmitted X.30,V.110 frame (P8, Q8, R8 or D8, D16, D24 ...) when ENFR = 1.
This status bit must be acknowledged.

OVS **Overspeed**

This bit is set if the ASC receives a data stream where the data rate exceeds the selected tolerance range. At least one bit in the output data stream has been lost.
This status bit must be acknowledged.

Receive HDLC Frame Status Byte (RSTA) Address: --H

Value after Reset: --H

7							0
RDO	CRC	RAB	0	0	VB2	VB1	VB0

The Receive HDLC Frame Status Byte is not accessable via a register address. It's value is added to the data of the received frame as the last byte.

RDO **Receive Data Overflow**

1: At least one byte of the message could not be stored due to an occupied HRFIFO.

CRC **CRC Check correct**

1: No CRC error has been detected in this frame.

RAB **Receive Abort**

1: An abort sequence was received.

VB2-0 Valid bit count

VB2-0 shows the valid bit count of the last received byte.

VB 2 1 0
0 0 0 = 1 Bit
0 0 1 = 2 Bit
0 1 0 = 3 Bit
0 1 1 = 4 Bit
1 0 0 = 5 Bit
1 0 1 = 6 Bit
1 1 0 = 7 Bit
1 1 1 = 8 Bit

3.3 Interrupt Enable Registers

Interrupt Status Enable Register (ISEN) Address: 48, 49H Read/Write

Value after Reset: 00H

7							0
Enable LDC	Enable ELDC	Enable RDC	Enable SYC	Enable EXI	Enable RF	Enable XF	Enable SC

A'1' in a bit of the ISEN enables the interrupt from the corresponding bit position of the IST register. If an interrupt is disabled, the bit in IST will still indicate the status and thus disabled interrupts may be polled.

Receiver and RFIFO Interrupt Enable Register (RFIE) Address: 4A, 4BH Read/Write

Value after Reset: 00H

7							0
Enable HRD	Enable URD	Enable HRFF	Enable URFF	Enable RMER	Enable RME	Enable FER	Enable PER

A '1' in a bit of the RFIE enables the status in IST (IST:RF) being activated by the corresponding bit in the RFS. If an interrupt is disabled, the bit in RFS will still indicate the status and thus disabled interrupts may be polled.

Transmitter and XFIFO Interrupt Enable (XFIE) Address: 4C, 4DH Read/Write

Value after Reset: 00H

7							0
Enable HXW	Enable UXW	Enable HXFE	Enable UXFE	Enable HXDU	Enable UXSE		

A '1' in a bit of the XFSE enables the status in IST (IST:XF) being activated by the corresponding bit in the XFS. If an interrupt is disabled, the bit in XFS will still indicate the status and thus disabled interrupts may be polled.

Special Condition Interrupt Enable Register (SCIE) Address: 4E, 4FH Read/Write

Value after Reset: 00H

7							0
Enable HRFO	Enable URFO	Enable BRB	Enable BRE	Enable LC1	Enable LC2	Enable RC1	Enable RC2

A '1' in a bit of the SCIE enables the status in IST (IST:SC) being activated by the corresponding bit in the SCS. If an interrupt is disabled, the bit in SCS will still indicate the status and thus disabled interrupts may be polled.

Status Register Interrupt Enable (STIE) Address: 46, 47H Read/Write

Value after Reset: 00H

7							0
Enable CAC	Enable CIS	Enable RLA	Enable IDLE	Enable RCHR	Enable RDB	Enable XDB	Enable OVS

A '1' in a bit of the STIE enables the status in EXIR (EXIR:STRC) being activated by the corresponding bit in the STR. If an interrupt is disabled, the bit in STR will still indicate the status and thus disabled interrupts may be polled.

3.4 Configuration Register

General Configuration Register (GCR) Address: 50, 51H Read/Write

Value after Reset: 00H

7							0
PU	DOE	V24	V110	ASY	ENFR	DMA	DMH

PU Power-Up

1: ITAC is in power-up state.

0: ITAC is in power-down mode.

DOE DCE Interface Output Enable

0: all output pins of the DCE interface are tri-state.

1: all output pins are operating.

V24 V.24 or X.21 Control lines

Selects the meaning of the DCE interface control lines and the meaning of the LDS status register.

0: X.21 interchange circuits

1: V.24 interchange circuits

V110 Mapping of S bits

0: S bits are mapped according to X.30

1: S bits are mapped according to V.110

ASY Selects asynchronous or synchronous DCE interface operation

0: Synchronous DCE interface.
The data transfer over the DCE interface is synchronized by the clock signal supplied by the S clock output. The ASC is inactive.

1: Asynchronous DCE Interface
The ASC is active and no clock signal is output on S. The synchronization is performed on each received start bit.

ENFR **Enable frame output to the Synchronous Network Interface**

0: No frames are transmitted to the SNI. The receiver is inactive. The status bit in the SYS-register are set to FSL = 1 and RSI = 0.

1: Frames are generated and output to the SNI. The receiver searches for the frame pattern.

ENFR = 1 will become active after the next FSC signal.
ENFR = 0 will tri-state the SNI-interface regardless of the current position in the frame.

Caution:
ENFR = 1 should be programmed after the DPLL is locked and the network rate, the frame format and the active time slot has been programmed. Changes of the network rate, frame format or active time slot needs the ENFR bit to be cleared for one clock cycle of the previous intermediate rate.

DMA **Controls the DMA block**

0: DMA interface is inactive. The DMA request outputs are tri-state.

1: DMA interface is active as selected by the DMA bit. The DMA request outputs are push-pull outputs.

DMH **Selects DMA mode for USART or HDLC controller**

0: DMA interface is connected to the USART.

1: DMA interface is connected to the HDLC controller.

Special Configuration Register (SCR) Address: 52, 53H Read/Write

Value after Reset: 00H

7							0
TS5	DLL	LCS	RCS	F56	DCL	OD	SFS

TS5 **Time-Slot Select MSB**

TS5 holds the value of the most significant bit of the time-slot register.

0: The value of TSR accesses time-slots 0 - 31.

1: The value of TSR accesses time-slots 32 - 63.

DLL **Selects Double Last Lock for S-bit mapping.**
*RSC must be '1'

0: Receive status bit changes are mapped to the DCE interface directly.

1: Receive status bit changes are mapped to the DCE interface after two identical values (double last lock). The contents of the SXS register is changed immediately.

LCS **Selects Local Character Stop**

0: The characters programmed in the LCAR1 and LCAR2 registers are not removed from the data stream.

1: The characters programmed in the LCAR1 and LCAR2 registers are deleted. Stop bits are transmitted instead of the character bits.

RCS **Selects Remote Character Stop**

0: The characters programmed in the RCAR1 and RCAR2 registers are not removed from the data stream.

1: The characters programmed in the RCAR1 and RCAR2 registers are deleted. Stop bits are transmitted instead of the character bits.

F56 **Selects the frame format for a user rate of 56 kbit/s (according to V.110).**

0: Bit 8 of the 64 kbit/s channel is filled with '1'.

1: Bit 8 of the 64 kbit/s channel is filled with the pattern '0 X SA SB 1 1 1 1'.

DCL **Double Bit Clock**

The DCL bit selects the type of clock signal used on the serial network interface SNI.

0: CLK is a single bit clock signal (one clock period per data bit).

1: CLK is a double bit clock signal (two clock periods per data bit). This will switch to IOM-2 timing.

OD **Open Drain Output**

The OD bit specifies the operation of the SDX/DU output.

0: The SDX/DU driver is push-pull during the selected time-slot bits.

1: The SDX/DU driver is open-drain during the selected time-slot bits. An external pull-up resistor is required.

The SDX/DU output is always tri-state during the inactive time-slot bits.

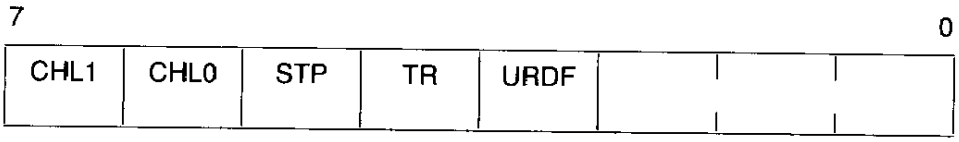
SFS **Single Frame Sync loss**

0: The Frame Sync Loss (FSL) status is set after three incorrect frame sync patterns have been detected.

1: The Frame Sync Loss (FSL) status is set after one incorrect frame sync pattern has been detected.

Async Interface Configuration Register (AICR) Address: 54, 55H Read/Write

Value after Reset: 00H



CHL1-0 Character Length

CHL1	CHL0	Character Length
0	0	eight bits
0	1	seven bits
1	0	six bits
1	1	five bits

The character length does not include a possible parity bit.

STP Stop Bits

0: One stop bit per character.

1: Two stop bits per character.

This selection affects only the transmit direction. In receive direction only the first stop bit is checked.

TR Tolerance Range of the ASC

0: Normal tolerance range (12,5 %). Every 8th stop bit may be removed from the data stream. In transmit direction the length of the stop bit is reduced.

1: Extended tolerance range (25 %). Every 4th stop bit may be removed from the data stream. In transmit direction the length of the stop bit is reduced.

URDF URFIFO Data Format

0: The received parity bit is stored in the URFIFO.

1: The received parity bit is not stored in the URFIFO.

Data Path Configuration Register (DPCR) Address: 56, 57H Read/Write

Value after Reset: 00H

7							0
RDC1	RDC0	XDC1	XDC0	RSC	XSC	TL2	TL3

Any change in the DPCR is active after the data bits D8, D16, D24, D32, D40, D48 in transmit direction when ENFR = 1, in receive direction when RSS = 1.
If ENFR = 0 the change is active after the next intermediate rate period.

RDC0-1 Receive Data Connect
Selects the path for the RxD output pin of the DCE interface.

RDC1	RDC0	
0	0	RxD is connected to the register bit RD
0	1	RxD is connected to the Serial Communication Logic
1	0	RxD is connected to the D-bits
1	1	reserved

XDC0-1 Transmit Data Connect
Selects the path for the input of the IRC transmitter.

XDC1	XDC0	
0	0	D-bits are equal to the register bit XD
0	1	D-bits originate from the Serial Communications Logic
1	0	D-bits originate from TxD
1	1	reserved

RSC Receive S-Bit Connect

0: Received S/SA, SB bits (from the SNI) are written to the register.

1: Received S/SA, SB bits (from the SNI) are additionally mapped onto the DCE interface interchange circuits.

XSC Transmit S-Bit Connect

0: Transmitted S/SA,SB bits (to the SNI) originate from the RDR register.

1: Transmitted S/SA,SB bits (to the SNI) are mapped from the DCE interchange circuits.

TL2 Test Loop 2 activation

0: Normal operation

1: Test loop 2 is active. The received data from the network interface is looped back at the output of the IRC (sync DCE-Interface) or ASC (async DCE-Interface).

TL3 Test Loop 3 activation

0: normal operation

1: Test loop 3 active. The loop indside the BRC is closed. The output of the BRC is looped back.

HDLC Mode Register (HMR) Address: 58, 59H Read/Write

Value after Reset: 00H

7							0
HRLC	HXLC	HREN	HXEN	HINV	ITF		

HRLC HDLC Receiver to Local DCE Connection

0: The USART receiver is connected to the DCE interface. The HDLC receiver is connected to the IRC.

1: The USART receiver is connected to the IRC. The HDLC receiver is connected to the DCE interface.

HXLC HDLC Transmitter to Local DCE Connection

0: The USART transmitter is connected to the DCE interface. The HDLC transmitter is connected to the IRC.

1: The USART transmitter is connected to the IRC. The HDLC transmitter is connected to the DCE interface.

HREN HDLC Receiver Enable

0: The HDLC receiver is inactive.

1: The HDLC receiver is active.

HXEN HDLC Transmitter Enable

0: The HDLC transmitter is inactive.

1: The HDLC transmitter is active. Any data entered into the HXFIFO will be transmitted.

HINV HDLC Inverted

0: The data stream of the HDLC transmitter and receiver is not inverted.

1:The data stream of the HDLC transmitter and receiver is inverted.

ITF Interframe Time Fill

0: Continuous ones are transmitted as interframe time fill.

1: Non-shared flags are transmitted as interframe time fill.

USART Mode Register (UMR) Address: 5A, 5BH Read/Write

Value after Reset: 00H

7							0
ASYC	PTY	UREN	UXEN	SCM	PY1	PY0	BTM

ASYC Asynchronous mode

0: The USART operates in synchronous mode.

1: The USART operates in asynchronous mode.

PTY Parity

0: parity check / generation disabled.

1: parity check / generation enabled. The parity type is specified by bits PY0-1.

UREN USART Receiver Enable

0: The USART receiver is disabled.

1: The USART receiver is enabled.

UXEN **USART Transmitter Enable**

0: The USART transmitter is disabled.

1: The USART transmitter is enabled.

SCM **Synchronous Communication Mode**
*significant only in synchronous operation (ASYC = 0)

0: Bisync operation of the USART receiver.

1: Monosync operation of the USART receiver.

PY0-1 **Parity Type**
*PTY has to be set '1' to enable parity check / generation

PY1	PY0	Parity type
0	0	0
0	1	odd
1	0	even
1	1	1

BTM **Bit Transparent Mode**
*ASYC must be set to '1'

0: Normal operation of the USART.

1: Bit transparent mode of the USART.

MONITOR Channel Configuration Register (MOCR) Address: 5C, 5DH Read/Write

Value after Reset: 00H

7							0
IOM2	CIX6	CIX5	SAW	MRE	MRC	MIE	MXC

IOM2 IOM2 Mode Selection

0: MONITOR channel handler is disabled, CIX5, CIX6 and SAW have no affect. Time-slot 7 and 8 are not influenced.

1: MONITOR channel handler is enabled. CIX5, CIX6 and SAW are active.

CIX6 Command/Indicate Transmit Bit 6
*significant only if IOM2 = 1

0: C/I 6 of IOM channel 1 is set to '1'.

1: C/I 6 of IOM channel 1 is set to '0'.

CIX5 Command/Indicate Transmit Bit 5
*significant only if IOM2 = 1

0: C/I 5 of IOM channel 1 is set to '1'.

1: C/I 5 of IOM channel 1 is set to '0'.

SAW Software Awake
*significant only if IOM2 = 1

0: Normal operation of the SDX/DU output.

1: Software awake function of the IOM-2 interface. The SDX/DU output will be forced to low while the following condition is true: FSC*IOM2*SAW = 1.

MRE MONITOR Channel Receiver Enable
*significant only if IOM2 = 1

0: The MDR interrupt is masked.

1: The MDR interrupt is enabled. The MCR bit controls the generation of the MDR interrupt.

MRC **MONITOR Channel Receiver Control**
*significant only if IOM2 = 1

0: The transmitted MR-bit is always '1'. The MDR interrupt is generated only for the first byte of a message.

1: The MR-bit is controlled by the MONITOR channel handler. The MDR interrupt is generated when a new MONITOR data byte has been received. The MR-bit performs the acknowledgement after the MOR1 register has been read.

MIE **MONITOR Channel Interrupt Enable**
*significant only if IOM2 = 1

0: The interrupts of the status bits MER, MDA and MAB are masked.

1: The interrupts of the status bits MER, MDA and MAB are enabled.

MXC **MONITOR Channel Transmitter Control**
*significant only if IOM2 = 1

0: The MX-bit is always '1'. The MONITOR channel transmitter is inactive.

1: The MX-bit is controlled by the MONITOR channel handler. The transmitter outputs the MONITOR channel data byte from the MOX1 register and performs the proper handshake. The MER, MDA and MBA status bits report status changes.

Bit Rate Select (BRS) Address: 5E, 5FH Read/Write

Value after Reset: 00H

7							0
UR3	UR2	UR1	UR0	NR3	NR2	NR1	NR0

UR0-3 User Rate
Specifies the user rate.

UR3	UR2	UR1	UR0	User rate	ASYNC DCE-IF	SYNC DCE-IF
0	0	0	0	300bit/s	X	–
0	0	0	1	600bit/s	X	X
0	0	1	0	1200bit/s	X	X
0	0	1	1	2400bit/s	X	X
0	1	0	0	4800bit/s	X	X
0	1	0	1	9600bit/s	X	X
0	1	1	0	19200bit/s	X	X
0	1	1	1	38400bit/s	X	X
1	0	0	0	48000bit/s	–	X
1	0	0	1	56000bit/s	–	X
1	0	1	0	64000bit/s	–	X
1	0	1	1	reserved		
1	1	x	x	reserved		

NR0-3 Network Rate

Specifies the network rate. A change of these bits becomes active after the ENFR bit has been cleared for one clock cycle of the previous intermediate rate.

NR3	NR2	NR1	NR0	Network rate	Intermediate rate
0	0	0	0	–	–
0	0	0	1	600bit/s	8000 bit/s
0	0	1	0	1200bit/s	8000 bit/s
0	0	1	1	2400bit/s	8000 bit/s
0	1	0	0	4800bit/s	8000 bit/s
0	1	0	1	9600bit/s	16000 bit/s
0	1	1	0	19200bit/s	32000 bit/s
0	1	1	1	38400bit/s	64000 bit/s
1	0	0	0	48000bit/s	64000 bit/s
1	0	0	1	56000bit/s	64000 or 56000 bit/s
1	0	1	0	64000bit/s	64000 bit/s
1	0	1	1	reserved	–
1	1	x	x	reserved	–

Time-Slot Register (TSR) Address: 60, 61H Read/Write

Value after Reset: 00H

7							0
TS4	TS3	TS2	TS1	TS0	ICS2	ICS1	ICS0

TS0-4 Time-slot Select

Selects the active time-slot on the SNI. If TS5 is '0', TS0-4 specifies a time-slot between time-slot 0 to 31. If TS5 is '1', TS0-4 specifies a time-slot between time slot 32 to 63.

ICS0-2 Intermediate Rate Channel Select

Selects the position of the first bit used by bearer rate converter within the specified time-slot.

Number of Retry Frames (NRF) Address: 62, 63H Read/Write

Value after Reset: 00H

7							0
N7	N6	N5	N4	N3	N2	N1	N0

Specifies the number of frames between a FSL status change and the generation of the RSI status.

3.5 Control Registers

Local DCE Control Register (LDR) Address: 64, 65H Read/Write

Value after Reset: 00H

7							0
DCD/I	DSR	CTS	RD	MO1	MO2		

DCD/I **State of the DCD/I (109) interchange circuit**
*significant only if DPCR:RSC = 0

0: DCD/I is 'ON' (0).

1: DCD/I is 'OFF' (1).

DSR **State of the DSR (107) interchange circuit**
*significant only if DPCR:RSC = 0

0: DSR is 'ON' (0).

1: DSR is 'OFF' (1).

CTS **State of CTS (106) interchange circuit**

0: CTS is 'ON' (0).

1: CTS is 'OFF' (1).

RD **State of the RxD (104) data circuit**

0: RxD is '0'.

1: RxD is '1'.

MO1-2 **State of the multifunctional outputs MO1, MO2**

0: MOx is 'ON' (0).

1: MOx is 'OFF' (1).

Remote DCE Control Register (RDR) Address: 66, 67H Read/Write

Value after Reset: 00H

7							0
XS/XSA	XSB	XX	XD				

XS/XSA **Value of the transmitted S/SA bit**
*significant only if DPCR:XCS = 0.

0: Transmitted S/SA-bit is 'ON' (0).

1: Transmitted S/SA-bit is 'OFF' (1).

XSB **Value of the transmitted SB bit**
*significant only if DPCR:XSC = 0.

0: Transmitted SA-bit is 'ON' (0).

1: Transmitted SA-bit is 'OFF' (1).

XX **Value of transmitted X bit**

0: Transmitted X-bit is 'ON' (0).

1: Transmitted X-bit is 'OFF' (1).

XD **Value of the transmitted D bits**
*significant only if DPCR:XDC1-0 = 00.

0: Transmitted D-bits is '0'.

1: Transmitted D-bits is '1'.

Transmitted E-Bit Register (XER) Address: 68, 69H Read/Write

Value after Reset 00H

7							0
XE1	XE2	XE3	XE4	XE5	XE6	XE7	

Value of the transmitted E bits in the intermediate rate frame. The value of XE7 has no meaning if the network rate is set to 600 bit/s. In this case E7 is controlled by the ITAC internally.

3.6 Command Registers

The synchronization status of both the HCC and UCC commands are reported by the STR:CAC status bit. It must be polled before a new value is written into the HCC or UCC register.

HDLC Controller Command Register (HCC) Address: 6C, 6DH Write

Value after Reset: 00H

7							0
HRR	HXR	XME	HDMS				

HRR HDLC Receiver Reset

The HDLC receiver is reseted. The HRFIFO is cleared and the DMOR-line becomes inactive if the DMA controller is connected to the HDLC controller and DMA is enabled.

HXR HDLC Transmitter Reset

The HDLC transmitter is reseted. The HXFIFO is cleared and the DMIR-line becomes inactive if the DMA controller is connected to the HDLC controller and DMA is enabled.

XME Transmit Message End

XME acts as a frame delimiter. XME marks the last byte in the HXFIFO as the last byte of a HDLC frame. After the HDLC controller has transmitted this byte, it will add the CRC field and the closing flag.

HDMS DMA start for the HDLC transmitter
*significant only if GCR:HMH = 1 and GCR:DMA = 1.

HDMS will start the generation of DMIR requests. After entering the first byte in the HXFIFO, the HDLC transmitter will start to transmit a frame. The HDLC frame will be closed by the CRC field and the closing flag if the HDLC transmitter becomes empty.

USART Controller Command Register (UCC) Address: 6E, 6FH Write

Value after Reset: 00H

7							0
URR	UXR	0	UDMS	SBX	HNT	TRA	

URR **USART Receiver Reset**

The USART receiver is reseted. The UXFIFO is cleared and the DMOR-line becomes inactive if the DMA controller is connected to the HDLC controller and DMA is enabled.

UXR **USART Transmitter Reset**

The USART transmitter is reseted. The UXFIFO is cleared and the DMIR-line becomes inactive if the DMA controller is connected to the HDLC controller and DMA is enabled.

UDMS **DMA start for the USART transmitter**
*significant only if GCR:HMH = 0 and GCR:DMA = 1.

UDMS will start the generation of DMIR requests.

SBK **Send Break**
*significant only if ASYC = 1.

If the send break command (SBK → UCC) is issued, the output of the USART transmitter is set to '0' regardless of any character currently transmitted. The output will stay '0' until a new command is entered with the SBK bit set to '0'.

HNT **Set Hunt Mode**
*significant only if ASYC = 0.

Forces the USART to search for sync character(s) before starting to store data regardless of any data being received at this time. The hunt mode is default after reset or UCC:URR = 1.

TRA **Set Transparent Mode**
*significant only if ASYC = 0.

Enables the USART to store received data without searching for character synchronization.

Counter Value 0 (CTV0) Address: 30, 31H Write

Value after Reset: 00H

7							0
CT7	CT6	CT5	CT4	CT3	CT2	CT1	CT0

Counter Value 1 (CTV1) Address: 32, 33H Write

Value after Reset: 00H

7							0
CT15	CT14	CT13	CT12	CT11	CT10	CT9	CT8

Counter Value 2 (CTV2) Address: 34, 35H Write

Value after Reset: 00H

7							0
		CTRN	CONT	CT19	CT18	CT17	CT16

CT0-19 Counter Value 0-19

Specify the number of bits which are counted down before the CTUR status bit is set.

CTRN Counter Run

0: Counter is stopped.

1: Counter is active.

CONT Continuous Operation

0: The counter is reloaded with every start bit.

1: The counter counts down regardless of the received data bits.

3.7 Constants and Special Character Registers

Synchronization Character Register (SYN) Address: 76, 77H Read/Write

Value after Reset: 00H

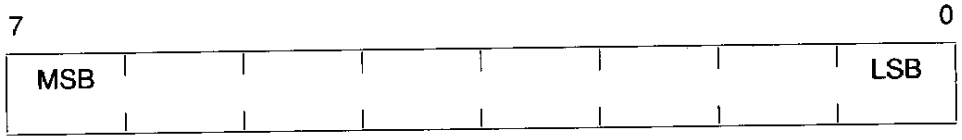
*significant only if UMR:ASYC = 0
In the hunt phase of the USART receiver it searches for one or two characters of the value stored in the SYN register before it stores characters in the URFIFO.

Local Character 1 (LCAR1) Address: 78, 79H Read/Write

Value after Reset: 00H

Local Character 2 (LCAR2) Address 7A, 7BH Read/Write

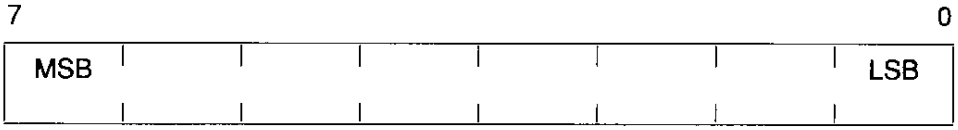
Value after Reset: 00H



Characters received by the DCE interface are compared with the programmed value of the LCAR1 and LCAR2 register. Upon a match, the SCS:LC1 or SCS:LC2 status bit is set. If SCR:LCS is set to '1', the character bits are replaced by stop bits.
In case of a character length less than eight bits, the unused bits of the LCAR1 and LCAR2 register must be set to '1'.

Remote Character 1 (RCAR1) Address: 7C, 7DH Read/Write

Value after Reset: 00H



Remote Character 2 (RCAR2) Address: 7E, 7FH Read/Write

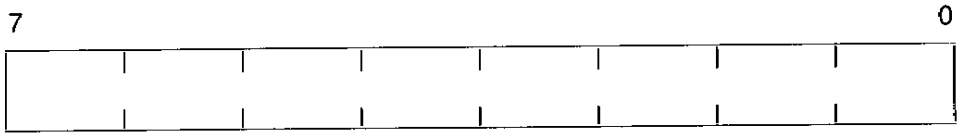
Value after Reset: 00H

Characters received by from the IRC receiver are compared with the programmed value of the RCAR1 and the RCAR2 register. Upon a match, the SCS:RC1 or SCS:RC2 status bit is set. If SCR:RCS is set to '1', the character bits are replaced by stop bits.
In case of an character length less than eight bits, the unused bits of the RCAR1 and RCAR2 register must be set to '1'.

3.8 Other Registers

Test Register 1 (TEST1) Address: 2E, 2FH Write

Value after Reset: 00H



This register is reserved for factory testing.

Bit 0 controls a test loop located between the IRC and the BRC. It loops the output of the IRC to its input and the output of the BRC to its input.
Note that there is no octet alignment provided by the BRC loop. Thus it is unlikely that the data received by the IRC will be transmitted with the same alignment on the B-channel. It is recommended to use a serial bit-oriented protocol like HDLC for testing the BRC loop.

Test Register 2 (TEST2) Address: 2C, 2DH Write

Value after Reset: 00H

This register is reserved for factory testing. The user should not write to this register.

4 Electrical Characteristics

Absolute Maximum Ratings

Parameter	Symbol	Limit Values		Unit
		min.	max.	
Ambient temperature under bias	T_A	0	70	°C
Storage temperature	T_{stg}	– 65	125	°C
Voltage on any pin with respect to ground	V	– 0.4	V_{DD} +0.4	V
Maximum voltage on V_{DD}	V		6	V

DC Characteristics

T_A = 0 to 70 °C; V_{DD} = 5 V ± 5 %, V_{SS} = 0 V

Parameter		Symbol	Limit Values		Unit	Test Condition	
			min.	max.			
L-input voltage		V_{IL}	– 0.4	0.8	V		All pins
H-input voltage		V_{IH}	3.0	V_{DD} +0.4	V		
L-output voltage		V_{OL}		0.45	V	I_{OL} = 2 mA	All pins except DU/SDX
L-output voltage		V_{OL}		0.45	V	I_{OL} = 7 mA	DU/SDX
H-output voltage		V_{OH}	2.4		V	I_{OH} = – 400 µA	All pins
H-output voltage		V_{OH}	V_{DD} – 0.5		V	I_{OH} = – 100 µA	
Power supply voltage	power-down	I_{CC}		0.7	mA	V_{DD} = 5 V; inputs at 0 V or V_{DD}; no output loads; CLK = 0 MHz	V_{DD}
	operational			6.0	mA	V_{DD} = 5 V; inputs at 0 V or V_{DD}; no output loads; CLK = 4 MHz	
Input leakage current		I_{IL}		10	µA	0 V < V_{IN} < V_{DD}	All pins
Output leakage current		I_{LO}		10	µA	0 V < V_{IN} < V_{DD}	

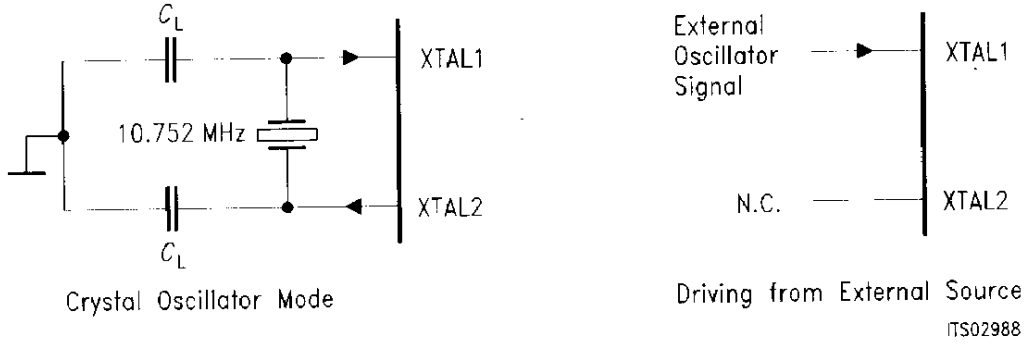
Capacitances

T_A = 25 °C; V_{DD} = 5 V ± 5 %, V_{SS} = 0 V; f_C = 1 MHz, unmeasured pins grounded

Parameter	Symbol	Limit Values		Unit	
		min.	max.		
Input capacitance	C_{IN}		7	pF	All pins except XTAL1, 2
I/O capacitance	$C_{I/O}$		7	pF	
Load capacitance	C_L		50	pF	XTAL1, 2

Recommended Oscillator Circuits

Figure 48
Oscillator Circuits



Crystal Specification

Parameter	Symbol	Limit Values	Unit
Frequency	f	10.752	MHz
Frequency calibration tolerance		max. 100	ppm
Load capacitance	C_L	max. 50	pF
Oscillator mode		fundamental	

Note: The load capacitance C_L depends on the recommandation of the crystal specification. Typical values for C_L are 18 .. 22 pF.

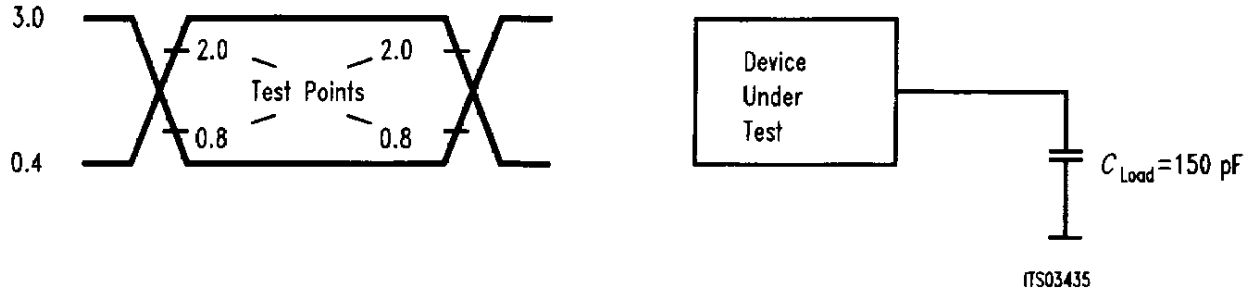
AC Characteristics

T_A = 0 to 70 °C; V_{DD} = 5 V ± 5 %

Inputs are driven to 3.0 V for a logical '1' and to 0.4 V for a logical '0'. Timing measurements are made at 2.0 V for a logical '1' and at 0.8 V for a logical '0'.

The AC testing input/output waveforms are shown below.

Figure 49
Input/Output Waveform for AC Tests



Microprocessor Interface Timing

Figure 50
µP Read Cycle

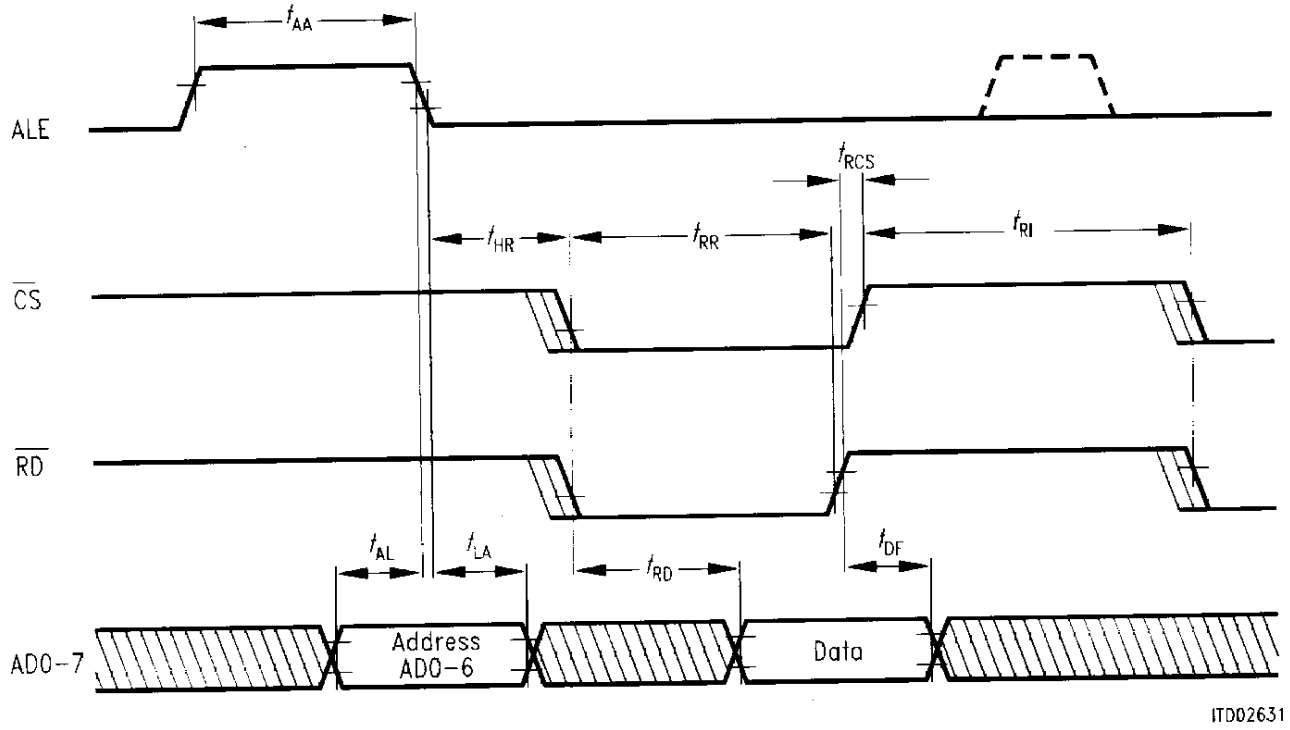


Figure 51
µP Write Cycle

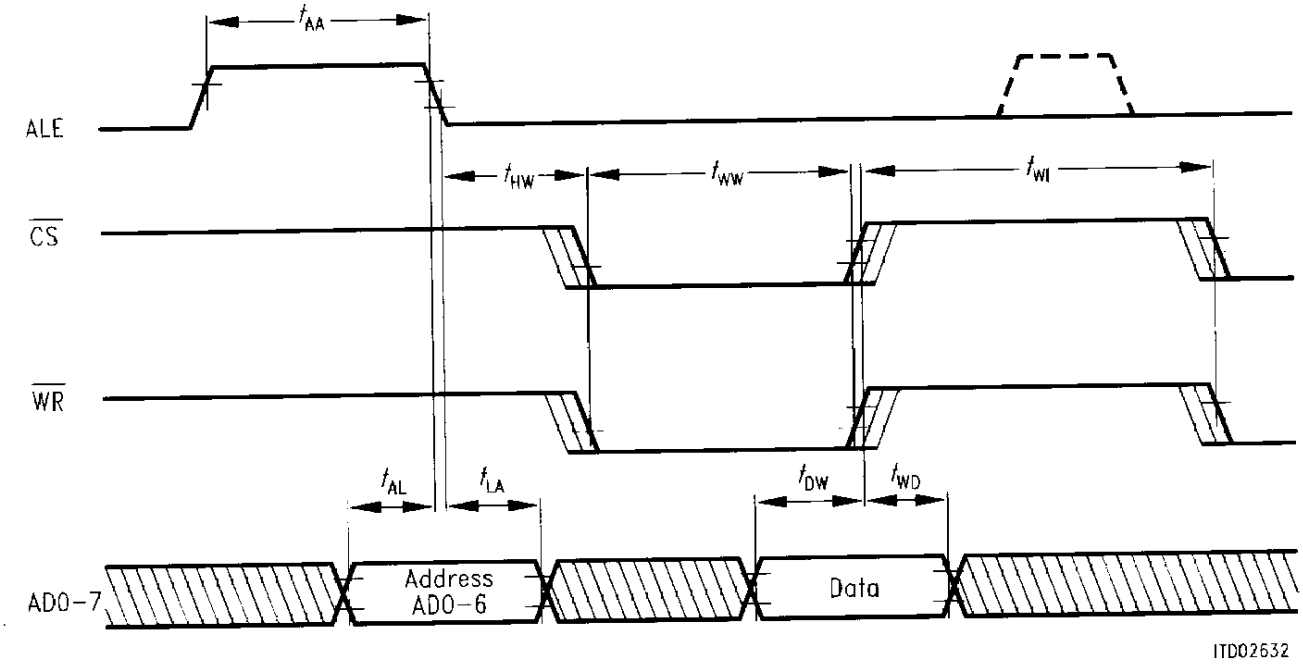


Figure 52
DMA Read Cycle

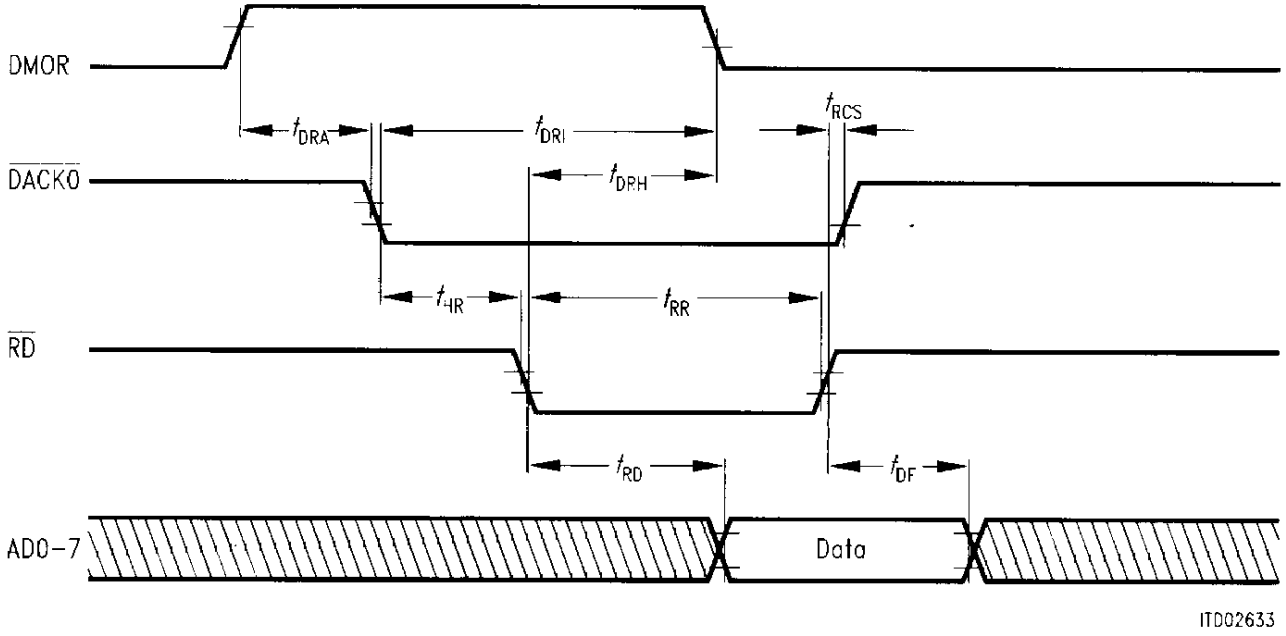


Figure 53
DMA Write Cycle

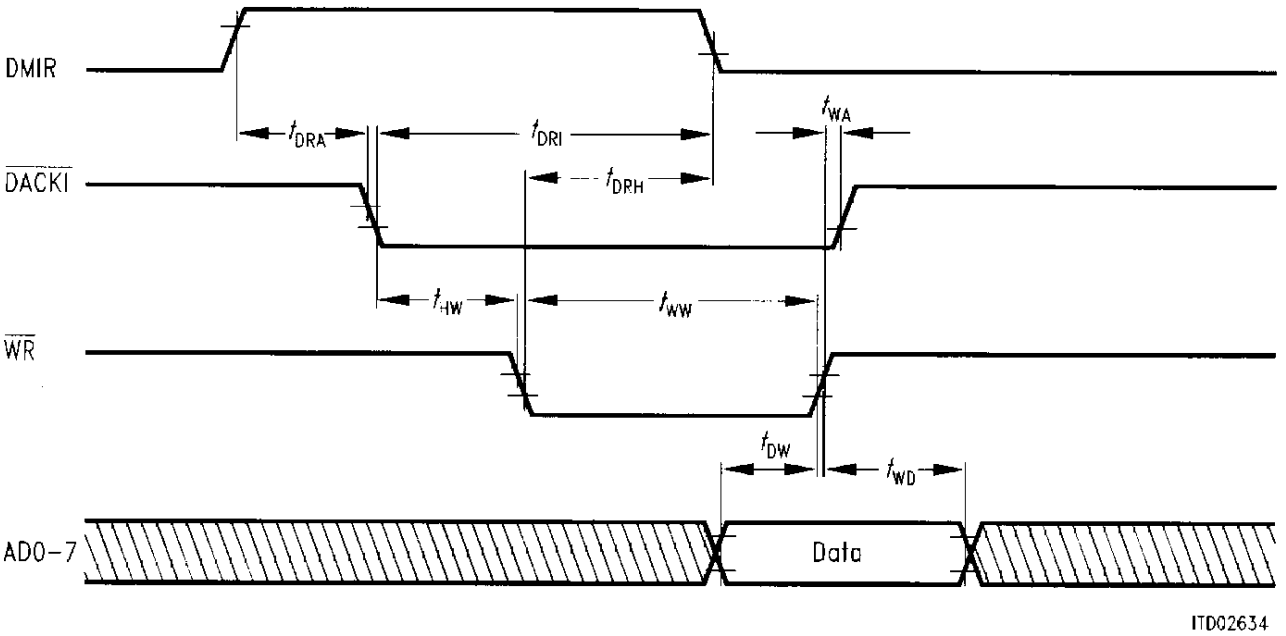
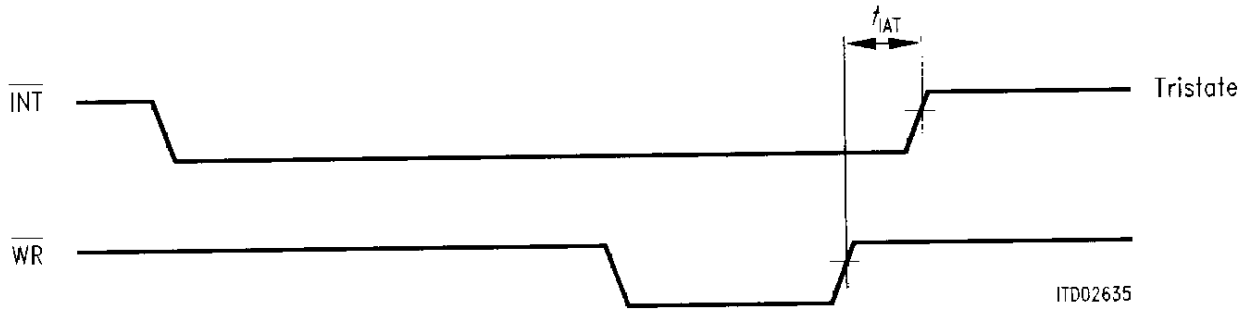


Figure 54
Interrupt Release Timing

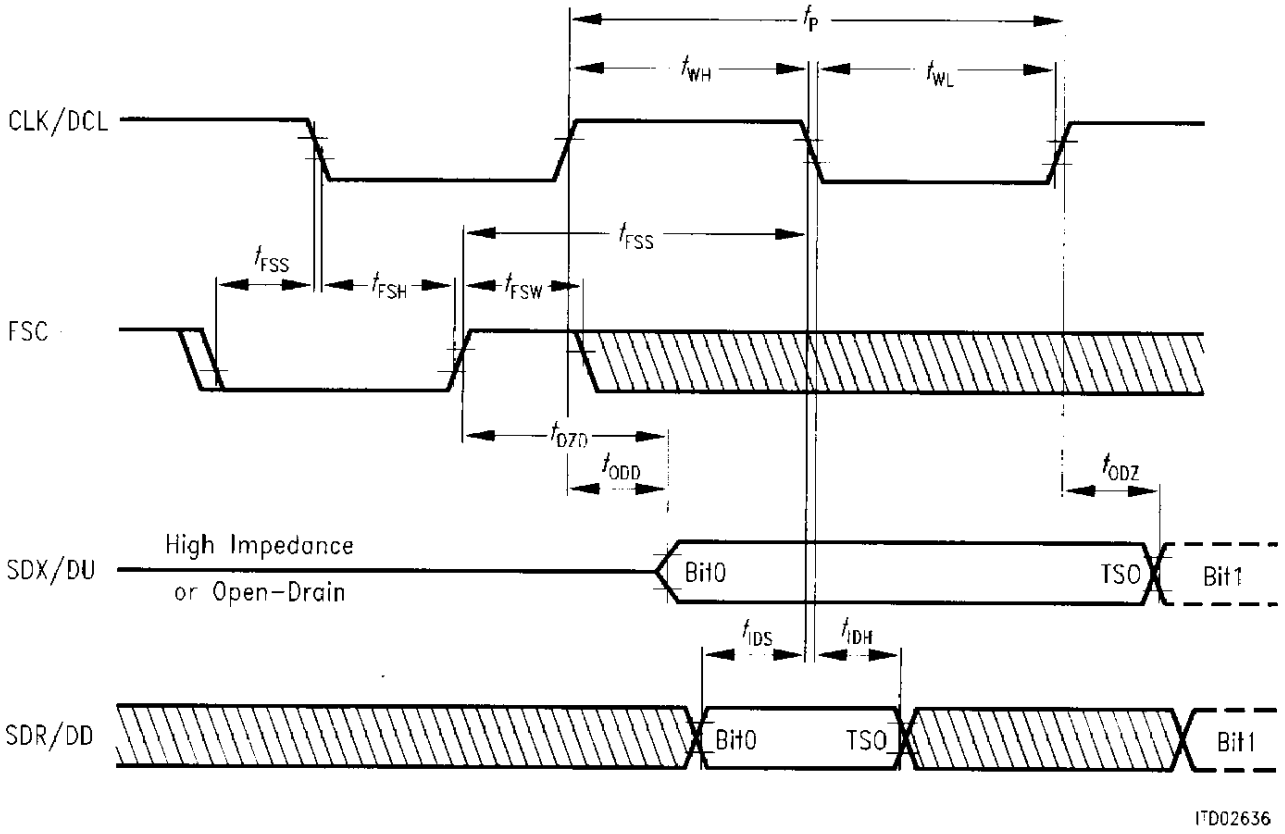


Microprocessor Interface Timing

Parameter	Symbol	Limit Values		Unit
		min.	max.	
ALE pulse width	t_{AA}	55		ns
Address setup time to ALE	t_{AL}	20		ns
Address hold time from ALE	t_{LA}	10		ns
Address hold time to read	t_{HR}	0		ns
Address hold time to write	t_{HW}	0		ns
Data output delay from $\overline{RD}$	t_{RD}		100	ns
Data float from $\overline{RD}$	t_{DF}		25	ns
$\overline{RD}$ pulse width	t_{RR}	110		ns
$\overline{RD}$ to $\overline{CS}$ inactive	t_{RCS}	5		ns
$\overline{RD}$ control interval	t_{RI}	85		ns
$\overline{WR}$ pulse width	t_{WW}	110		ns
Data setup time to $\overline{WR}$ x $\overline{CS}$	t_{DW}	30		ns
Data hold time from $\overline{WR}$ x $\overline{CS}$	t_{WD}	10		ns
$\overline{WR}$ control interval	t_{WI}	85		ns
DMA request to acknowledge	t_{DRA}	0		ns
DMA request inactive delay	t_{DRI}		80	ns
DMA request hold delay	t_{DRH}		85	ns
DACK hold after write	t_{WA}	5		ns
Interrupt acknowledge to tri-state	t_{IAT}		100	ns

Synchronous Network Interface

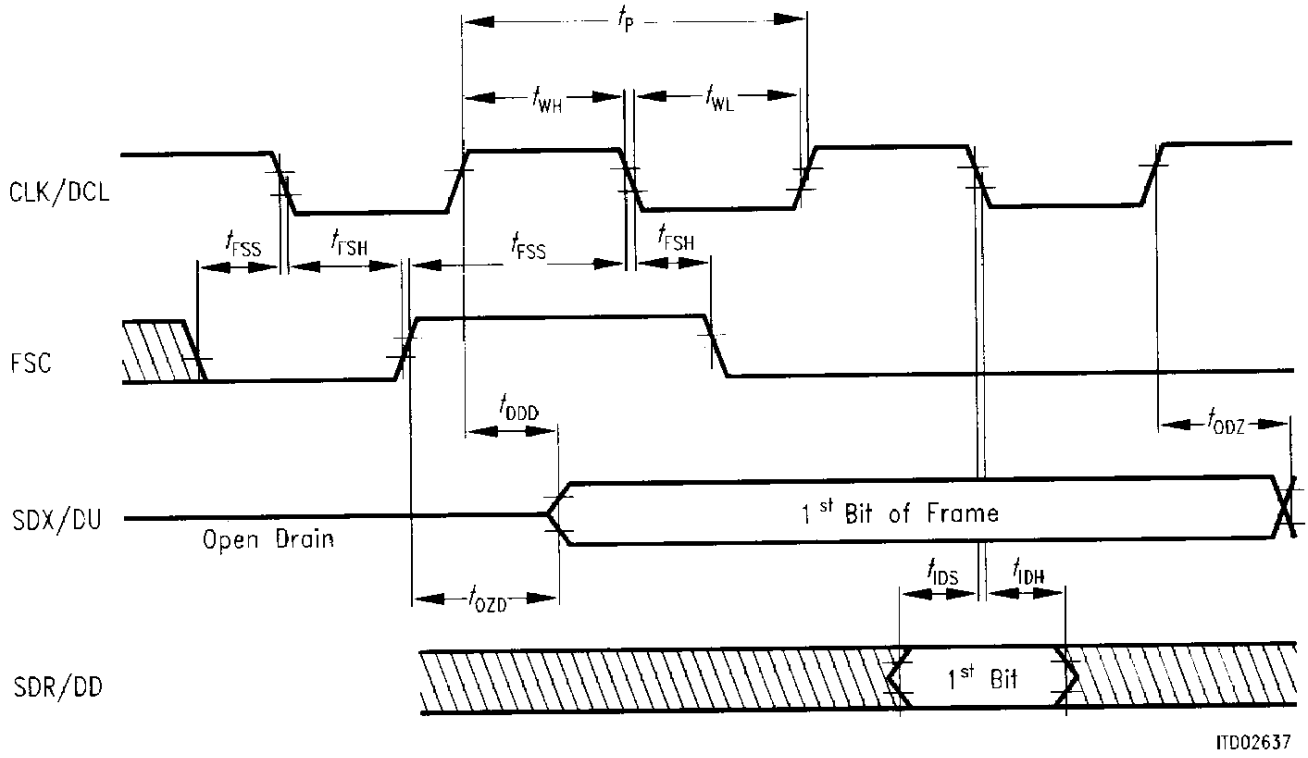
Figure 55
PCM Interface Timing



Parameter	Symbol	Limit Values		Unit
		min.	max.	
CLK/DCL period	t_P	244	$t_{IR}/2$	ns
H-CLK period	t_{WH}	100		ns
L-CLK period	t_{WL}	100		ns
Frame sync setup	t_{FSS}	40		ns
Frame sync hold	t_{FSH}	40		ns
Frame sync width	t_{FSW}	40		ns
Output data from high impedance to active	t_{OZD}		100	ns
Output data delay from CLK	t_{ODD}		100	ns
Output data from active to high impedance	t_{ODZ}		80	ns
Input data setup	t_{IDS}	20		ns
Input data hold	t_{IDH}	40		ns

Note: t_{IR} = Intermediate Rate Period

Figure 56
IOM®-2 Interface Timing

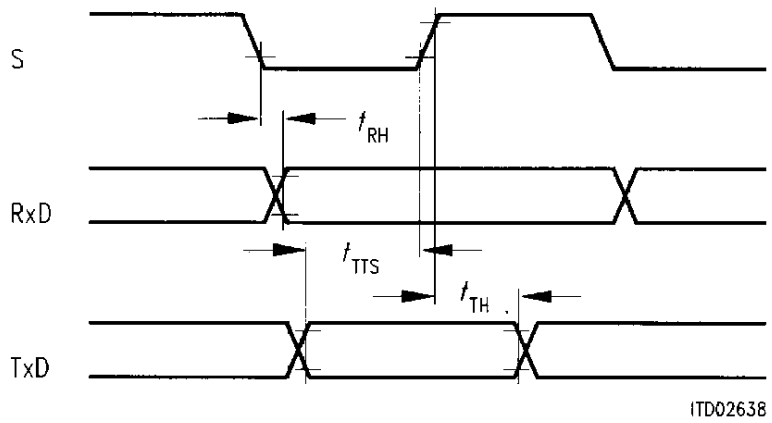


Parameter	Symbol	Limit Values		Unit
		min.	max.	
CLK/DCL period	t_P	244	$t_{IR}/4$	ns
H-CLK period	t_{WH}	100		ns
L-CLK period	t_{WL}	100		ns
Frame sync setup	t_{FSS}	40		ns
Frame sync hold	t_{FSH}	40		ns
Output data from high impedance to active	t_{OZD}		100	ns
Output data delay from CLK	t_{ODD}		100	ns
Output data from active to high impedance	t_{ODZ}		80	ns
Input data setup	t_{IDS}	20		ns
Input data hold	t_{IDH}	40		ns

Note: t_{IR} = Intermediate Rate Period

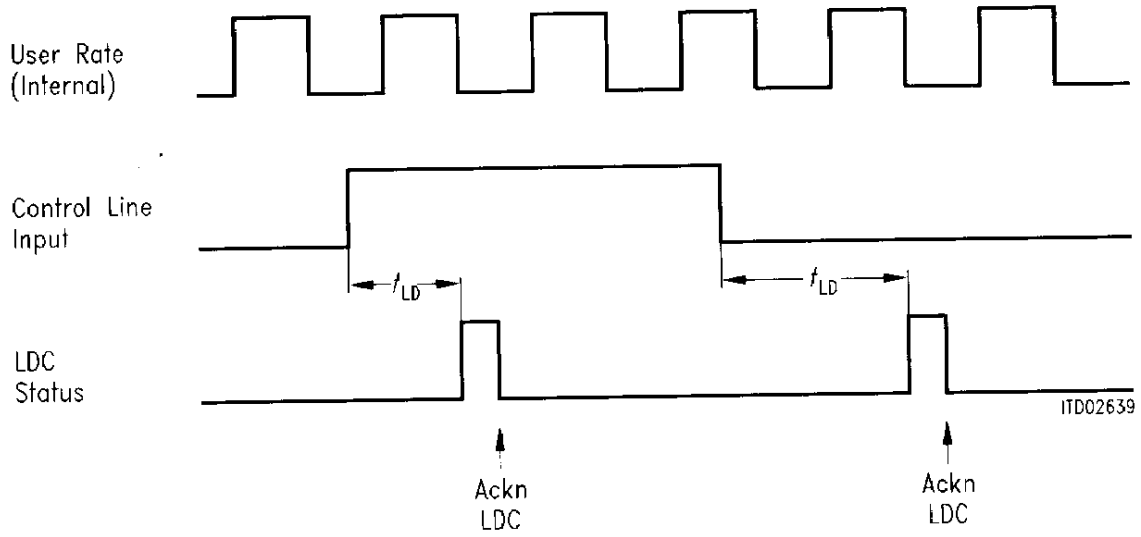
DCE Interface

Figure 57
Synchronous DCE Interface Timing



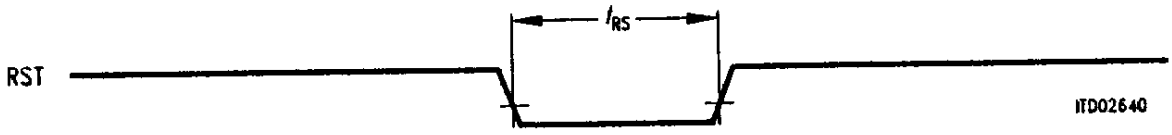
Parameter	Symbol	Limit Values		Unit	Test Condition
		min.	max.		
R x D hold time	t_{RH}		200	ns	
T x D setup time	t_{TTS}	200		ns	User rate <> 56 kbit/s
T x D setup time	t_{TTS}	7000		ns	User rate = 56 kbit/s
T x D hold time	t_{TTH}	200		ns	

Figure 58
Asynchronous DCE Interface



Parameter	Symbol	Limit Values		Unit
		min.	max.	
Local control delay	t_{LD}	0.5	1.5	User rate periods

Figure 59
Reset Timing



Parameter	Symbol	Limit Values		Unit
		min.	max.	
Reset pulse width	t_{RS}	2		µs

5 Application Note Summary

The following paragraphs describe application information for the ITAC. They cover the topics

- connecting the ITAC to the ISAC-S via SSI
- support of the V.110 by the ITAC
- autobauding
- alternative solution for autobauding
- octet aligned data transfer at 64 kbit/s

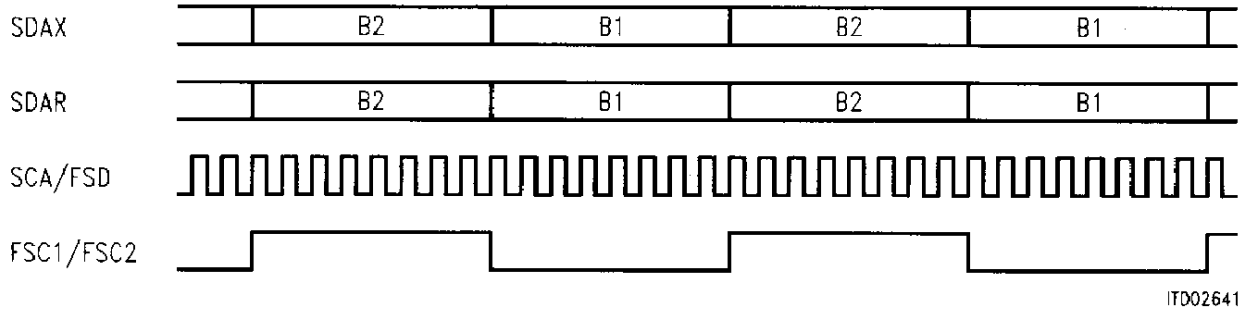
5.1 Connecting the ITAC® (PSB 2110) to the ISAC®-S (PEB 2085) via SSI

The SSI Interface of the ISAC®-S

The SSI interface serves as a full duplex connection to B-channel sources/destinations in terminal equipment at a data rate of 128 kbit/s.

It consists of one data line for each direction (transmit and receive), a 8-kHz frame synchronization signal and a 128-kHz clock signal. The default polarity of the frame synchronization signal is such that it is high during the transmission of channel B2 and low during the transmission of channel B1.

Figure 60
Signals of the SSI Interface



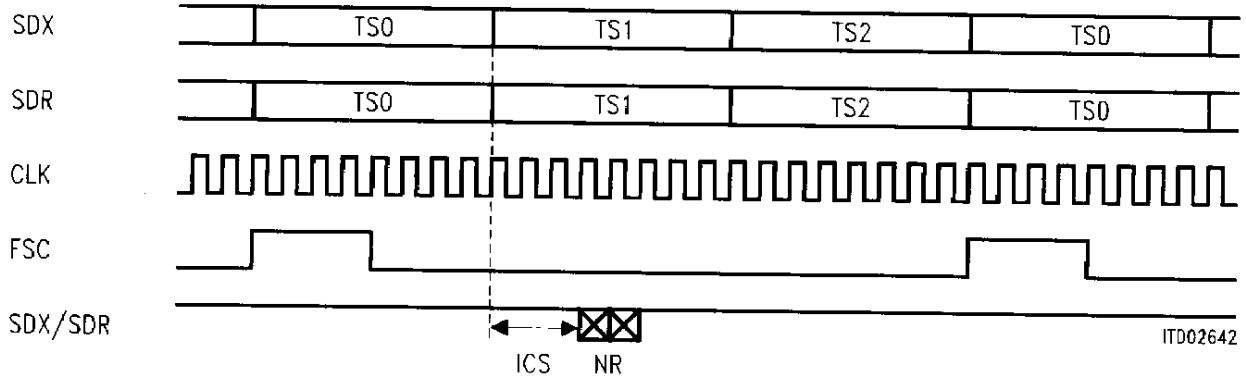
The Serial Network Interface of the ITAC®

The serial network interface of the ITAC also consists of four lines. One data line in each direction (transmit and receive), a frame start signal and a data clock signal. The network interface is defined as a time-slot oriented interface. Each time-slot has the fixed width of eight bits. Six bits, five in the TSR and one in the SCR register, specify the time-slot which is used for transmission and receiption.

The number of used bits within the time-slot is determined by the programmed network rate (BRS:NR0-NR3). If the network rate is less or equal to 4800 bit/s, the bearer rate converter uses one bit of the time-slot. If the network rate is 9600 bit/s, the bearer rate converter uses two bits of the time-slot. If the network rate is 19200 bit/s, the bearer rate converter uses four bits of the time-slot. If the network rate is greater or equal to 38400 bit/s, the bearer rate converter uses all eight bits of the time-slot.
The start position within the time-slot is programmed by bits ICS0 - ICS2 in the TSR register.

ICS2	ICS1	ICS0	Start position
0	0	0	bit 0
0	0	1	bit 1
0	1	0	bit 2
0	1	1	bit 3
1	0	0	bit 4
1	0	1	bit 5
1	1	0	bit 6
1	1	1	bit 7

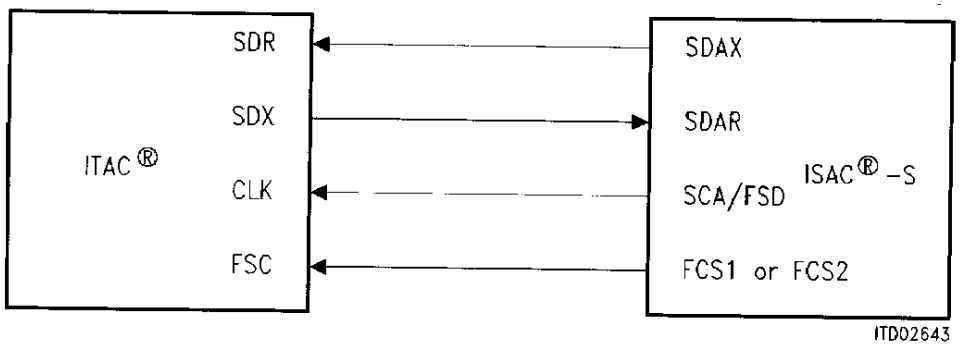
Figure 61
Signals of the Serial Network Interface (time-slot 1, ICS = 3, NR = 9600 bit/s)



Connecting ITAC® and ISAC®-S

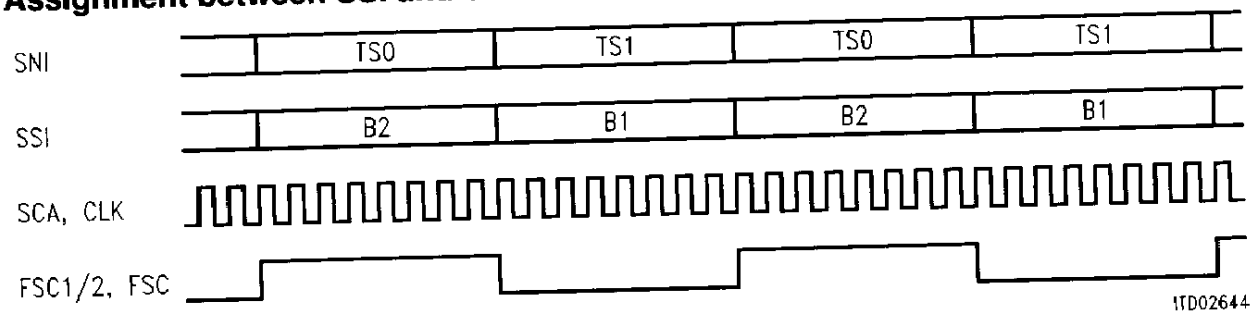
The ITAC and the ISAC-S are connected as illustrated in **figure 62**.

Figure 62
Connecting the ITAC® and the ISAC®-S via SSI



The ITAC treats the SSI interface as a time-slot oriented interface with two time-slots. If default polarity of the FSC signal is used, time-slot 0 transmits channel B2 and time-slot 1 transmits channel B1.

Figure 63
Assignment between SSI and SNI



Programming the ISAC®-S

In the ISAC-S, the ADFR register is programmed to default polarity of the FCS signal. This is done by setting bit 1 to '0' if FSC1 is used or setting bit 2 to '0' if FSC2 is used.

ADFR = x x x x x x 0 x	for FSC1
ADFR = x x x x x 0 x x	for FSC2

The SPCR register must be set to connect the used B channel to the SSI interface. This is done with the B1C1, B1C0 bits for channel B1 and with the B2C1, B2C0 bits for channel B2.

SPCR = x x x x 1 0 x x	for B1 to SSI
SPCR = x x x x x x 1 0	for B2 to SSI

Programming the ITAC®

In the ITAC, the TSR, the SCR, the BRS and the GCR register must be programmed.

The TSR register specifies the used B-channel and the start position of the used bits within the time-slot.

TSR = 0 0 0 0 1 ICS2 ICS1 ICS0 for B1

TSR = 0 0 0 0 0 ICS2 ICS1 ICS0 for B2

The TS5 bit in the SCR register must be set to zero.

SCR = 0 x x x x 0 0 x

The network rate is selected by the NR0 - NR3 bits in the BRS register.

BRS = x x x x NR3 NR2 NR1 NR0

The number of used bits depends on the selected network rate as described before.

The ENFR bit in the GCR register controls the output of the network interface.

If ENFR = '0', the SDX output is tri-state for all positions in the selected time-slot. The receiver is inactive.

If ENFR = '1', the SDX output sends data at the programmed bit positions within the selected time-slot. The receiver receives data at the programmed bit positions within the selected time-slot.

Important Note

The ITAC needs a maximum time of 84 ms to synchronize the internal clock to the FSC signal after the ITAC is powered up by the PU bit in the GCR register or after the position of the frame start signal has changed. Therefore it is recommended to select B1 or B2 in the ITAC by changing the TSR register and not by changing the FSC polarity in the ISAC-S.

5.2 Support of V.110 by the ITAC®

This chapter describes the register settings for the ITAC to implement chapter four of the V.110 recommendation (blue book version).

The chapter numbers are according to the ones used in the recommendation.

4 Operation sequence.

4.1 TA-A duplex operation.

4.1.1 Idle state.

4.1.1.1 During the idle (or ready) state the TA (DCE) will be receiving the following from the DTE:

Circuit 103 (TxD) = '1'
Circuit 105 (RTS) = ON
Circuit 108/1 = OFF; Circuit 108/2 (DTR) = ON

This is inputed to the ITAC DCE interface and may be checked using the LDS register.

4.1.1.2 During the idle state the TA will transmit continuous binary 1s into the B- and D-channels.

Therefore, the output of the SNI must be disabled. This is done by setting

GCR:ENFR = 0

4.1.1.3 During the idle state the TA (DCE) will transmit the following toward the DTE:

Circuit 104 (RxD) = '1'
Circuit 107 (DSR) = OFF
Circuit 106 (CTS) = OFF
Circuit 109 (DCD) = OFF

This is done by setting

LDR = '1111xx00' b,'F0' hex
DPCR = '00xx0x00' b,'00' hex

Setting all bits to zero in the DPCR registers means to connect the control and data output lines of the DCE interface to the register bits of the LDR register. Since they are set to one, the output at the DCE pins is also high and thus these lines are 'OFF'.

Note: The LDR register must be programmed before the GCR:DOE (DCE Output Enable) bit is set to one. If the DOE bit is set to one before the LDR register is programmed, the output pins will go low for a short moment which will be interpreted as a start bit. Thus an unknown character may be received by the terminal.

4.1.2 Connect TA to line state.

4.1.2.1 Switching to the data mode causes the TA to transmit the following toward the ISDN:

a) Frame synchronization pattern, as follows:
- octet 0 = all binary 0
- bit number one of octets 1-9 = binary 1

b) Data bits = binary 1

c) Status bits S = OFF and X = OFF

This is done by setting:

BRS:NR3-NR0 to the corresponding network rate
RDR = '11110000' b,'F0' hex
DPCR = 'xx00x000' b,'00' hex
GCR:ENFR = 1

The DPCR register is set so that the value of the transmitted D-, S- and X-bits originate from the bits of the RDR register. The value is independent of the status on the DCE interface.

4.1.2.2 The receiver will start to search for the frame synchronization pattern in the received bit stream. At the same time, a timer T1 shall be started with a time out value of at least 10 seconds.

After setting GCR:ENFR to '1' , the receiver will automatically search for the sync pattern. The FSL (frame sync loss) bit in the SYS register will be set to '1' and the programmed value of the NRF register (number of retry frames) will be counted down. If the counter reaches zero, a RSI = 1 status (resynchronization impossible) is generated.

If at any time, the receiver recognizes the frame pattern, it will set FSL to '0' and generate the SYC status change in the IST register. This can generate an interrupt if enabled in the ISEN register.

The timer can be realized in two ways.

In the first case it is realized with the microprocessor. The software must check the FSL bit, to see if it becomes 0 within the timer period.

In the second case, the NRF register is programmed and the RSI status is used to reload the NRF register and to increment the software retry counter. The value for the NRF register and the value for the software counter is determined by the intermediate rate. If the intermediate rate is equal to 8 kHz, a total of 1000 frames (1000 * 80 bits) must be searched before the 10 s time out occurs. To count 1000 frames, the NRF register may be programmed to 250 dec. = 'FA' hex and the software counter must count four RSI status changes. If the intermediate rate is equal to 16 kbit/s a total of 2000 frames must be searched. Therefore, the NRF register is programmed with 250 dec and the software counter must count eight RSI status changes. For an intermediate rate of 32 kbit/s, the software counter must count sixteen RSI status changes.

4.1.2.3 When the receiver recognizes the frame synchronization pattern, it causes the S- and X-bits in the transmitted frames to be turned ON.

Therefore, the S- and X-bits are set to '0' in the RDR register.

RDR = '00010000' b, '10' hex

4.1.2.4 When the receiver recognizes that the status bits S and X are in the ON condition it will perform the following functions:

a) Turn ON circuit 107 (DSR) toward the DTE and stop timer T1

LDR = '1011xx00' b, 'B0' hex

b) Then, circuit 103 (TxD) may be connected to the data bits in the frame; however the DTE must maintain a binary 1 condition until circuit 106 is turned ON in the next portion of the sequence.

DPCR = '00100000' b, '20' hex

The transmitted D-bits originate from the DCE interface.

c) Turn ON circuit 109 (DCD) and connect the data bits to circuit 104 (RxD).

LDR = '0011xx00' b, '30' hex
DPCR = '10100000' b, 'A0' hex

d) After a N (N = 24) bit interval it will turn ON circuit 106 (CTS).

Wait for three status changes of bit STR:RDB (3*8 = 24 = N)
and set:

LDR = '0001xx00', '10' hex
DPCR = '10101100' b, 'AC' hex (connect S- and X-bit to DTE)

4.1.3 Data transfer state.

DPCR = '10101100' b,'AC' hex

S- and X-bits are connected to DTE.

4.1.4 Disconnect mode.

4.1.4.1 At the completion of the data transfer phase, the local DTE will indicate a disconnect request by turning OFF circuit 108 (DTR). This will cause the following to occur:

a) The status bit S in the frame toward the ISDN will turn OFF, status bits X are kept ON;

RDR = '110x0000' b, 'C0' hex
DPCR = '10101000' b, 'A8' hex

Set the S-bits in the RDR register to one and connect the transmitted S-bits to the register bits.

b) Circuit 106 will be turned OFF.

LDR = '0011xx00' b, '30' hex
DPCR = '10101000' b, 'A8' hex

c) The data bits in the frame will be set to binary 0.

RDR = '11000000' b, 'C0'hex
DPCR = '10001000' b, '80' hex

Set XD to zero and connect the transmitted D-bits to the register bit.

4.1.4.2 If circuit 108 (DTR) is still ON at the remote TA, this TA will recognize the transition of the status bits from ON to OFF and the data bits from data to binary 0 as a disconnected signal and it will turn OFF circuits 107 (DSR) and 109 (DCD).

A change in the received S-bits is indicated by the SXC bit in the EXIR register. An interrupt is generated, if the 'ENABLE EXI' bit is set in the ISEN register.

The received D-bits can be monitored by the USART receiver. Therefore the USART receiver is connected to the received D-bits.

HMR = '1xxxxxxx' b, '80' hex

The USART is set to synchronous operation.

UMR = '00000000' b, '00' hex

Look for a change in the received S-bits (IST:EXI, EXIR:SC). If both SXS:RSA and SXS:RSB are '0', turn on the USART (UMR:UREN = 1), start reception (UCC:TRA = 1) and read the next characters. If they are '00' disconnect RxD, set S-bits to OFF and/or disable frame.

```
Wait for IST:EXI
Check for EXIR:SC
If SXS:RSA = 1 and SXS:RSB = 1:
HMR   = '10000000' b, '80' hex     connect USART receiver to frame
UMR   = '00100000' b, '20' hex     USAR sync, ON
UCC   = '00000010' b, '02' hex     TRA=1

Wait for RFS:URD
Read URFIFO
If data = '00' then
LDR   = '1111xx00' b, 'F0' hex     set all DCE outputs to OFF
DPCR  = '00000000' b, '00' hex     connect DCE interface to register bits
GCR:ENFR = 0                       disable V.110 frame

Return to disconnect mode
```

4.1.4.3 The TA at the station that originated the disconnect request will recognize reception of S = OFF or the loss of framing signals as disconnect acknowledgement and turn OFF circuits 107 (DSR) and 109 (DCD) and transfer to disconnect mode.

Wait for IST:EXI, EXIR:SC or IST:SYC
If SXS:RSA = 1 and SXS:RSB = 1 then return to disconnect mode
If SYS:RSL then return to disconnect mode

4.1.5 Loss of frame synchronization.

4.1.5.1 In the event of loss of frame synchronization, the TA should attempt to resynchronize as follows:

a) Place circuit 104 (RxD) in binary 1 condition (passes from the data mode).

LDR = 'xxx1xx00' b, '10' hex
DPCR = '00101100' b, '2C' hex

b) Turn OFF status bit X in the transmitted frame

RDR = 'xx1x0000' b, '20' hex

c) The remote TA upon recognition of status bit X OFF will turn OFF circuit 106 (CTS) which will cause the remote DTE to place circuit 103 in a binary 1 condition.

LDR = 'xx1xxx00' b, '20' hex

d) The local TA should attempt to resynchronize on the incomming signal.

e) If after an interval of three seconds the local TA cannot attain synchronization, it should send a disconnect request by turning OFF all of the status bits for several (at least three) frames with data bits set to binary 0 and then disconnect by turning OFF circuit 107 (DSR) and transferring to the disconnect mode.

The timer of three seconds can be realized in two ways like timer T1 during switching to data mode (4.1.2.2). A microprocessor timer can be used and the SYS:RSS bit is polled or register NRF is programmed.

f) If resynchronization is achieved, the TA should turn ON status bit X toward the distant station.

RDR = 'xx0x0000' b, '00' hex

g) If resynchronization is achieved, the TA (which has turned OFF circuit 106) should, after a N bit interval, turn ON circuit 106. This will cause circuit 103 to change from binary 1 to the data mode.

LDR = 'xx0xxx00' b, '00' hex

5.3 Autobauding

Introduction

One of the major areas of application of the ISDN Terminal Adapter Circuit (ITAC PSB 2110) is the connection of terminals and PC's supporting RS232C/V.24 to the Integrated Services Digital Network. This is implemented according to the V.110 specification. If one thinks in terms of today's technologies the ITAC could be compared, in this application, to a highly sophisticated modem since it interfaces a terminal/PC to the telephone network. Of course there is no modulation/demodulation going on and the additional features offered by the ITAC constitute a major advance on current equipment. Whereas most modern modem's work at dedicated rates with limited configuration possibilities the ITAC can be programmed to any combination of the following;

bit rate	38400 bit/s 19200 bit/s 9600 bit/s 4800 bit/s 2400 bit/s 1200 bit/s 600 bit/s 300 bit/s
parity	None, Odd, Even, Space, Mark
character bit	8, 7, 6, 5 bits
stop bit	1, 2 stop bits

What is in itself very flexible is not without some programming overhead under certain circumstances. Consider the case where the ITAC is at the front end of a 'ISDN Modem' sitting in some office somewhere. Now in this office there are several V.24 terminals of various calibres all wanting access to the ISDN on an occasional basis. What may seem like a minor configuration headache can be eased considerably if we consider the possibility of Autobauding. By this procedure the 'ISDN Modem' would be able to recognize automatically the configuration settings of the terminal that's plugged into it. Hence any of the terminals lying around the office can be connected into the ISDN with the minimum of fuss.

This is just one fanciful example where autobauding may prove indispensable. This chapter will propose a possible implementation. The example discussed will limit itself to character lengths of 7 and 8, speeds up to and including 19200 bit/s and all parity types except Mark. The characters sent will be in ASCII format. This does cover most of the normal applications encountered in the real world. In most cases these limits are self-inflicted and the principles explained below can equally be extended to accommodate a wider range of configurations.

The Hardware etc.

Before considering any code it is appropriate to mention the hardware on which this algorithm was realized. Those familiar with the Siemens ISDN PC User Board (SIPB) will know that it is based around an 80188 microprocessor. An ITAC add-on module was plugged into the mainboard. The algorithm was built into the mainboard firmware. It can be initiated by sending a command to the firmware via a command/response mailbox which lies at the core of the software architecture of the system. The code was written in 'C' and compiled with Microsoft C Compiler version 5.1. The PC was used as a V.24 terminal and connected to a V.24 driver which in turn was connected to the ITAC.

The Procedure

In order for the 'ISDN Modem' to automatically recognize the set-up characteristics of the terminal that has been plugged into it, some input from that terminal is required. In other words the terminal has to tell the 'ISDN Modem' a little bit about itself! In keeping with the syntax of the ubiquitous Hayes Smartmodem the string;

AT

is used for this purpose. Also in keeping with Hayes the string

at

is also supported. Entering either of these character sequences at the terminal is sufficient for the 'ISDN Modem' to calculate its characteristics. All the information required is there; it only remains for the code to extract it.

First let us see how these characters would look on the TxD line from the terminal. The corresponding character strings are shown below;

A	(41 Hex)	s	1000001(8 P)S
T	(54 Hex)	s	0010101(8 P)S
a	(61 Hex)	s	1000011(8 P)S
t	(74 Hex)	s	0010111(8 P)S

where
- s - start bit
- 8 - possible eight character bit
- P - possible parity bit
- S - stop bit

The first thing to note is that the spaces in the character strings are only there to make it more legible. The leftmost bits are only the first transmitted by the terminal and the first received by the ITAC. In the middle of the string is the ASCII code of the character. Note that the LSB of the character is transmitted first. The character string begins with a bit of start polarity (0) and ends one of stop polarity (1). An eight character bit (in these cases 0) and a parity bit (could be 0 or 1) may also be present.

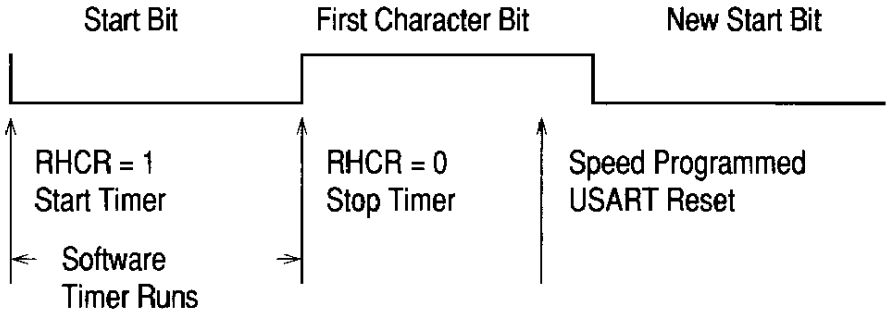
The problem devides naturally in two. Firstly the code has ascertain the terminal's speed. The first two bits of the string are used for this purpose. Having performed this the ITAC is programmed accordingly and as much of the remainder of the string as possible is stored in the USART which ist switched to eavesdrop on the V.24 TxD line. This residual string in the FIFO is used to figure out parity and character bit length settings.

The problem is complicated somewhat by asynchronous nature of the system. Any number of stop bits can be sent between characters. These superfluous stop bits can effect what gets stored in the FIFO. In some cases the presence of a third character directly after the t/T can also have some effect. The algorithm attempts to take all these factors into account and to recognize the terminal characteristics despite their confusing effects.

Problem 1: Recognizing the Speed

This is by far the more demanding of the two problems because it is time critical. In effect we time the length of the start bit. Considering the character strings shown above it should be noted that the first two bits in the at/AT sequence are always 0 followed by 1. This is independent of the number of character bits or the parity settings. It is important to realize that the line contains all stop bits (i.e. ones) when no character is sent. Hence we have a situation as depicted in **figure 64**. We have a isolated 0 character in the line which we know is only one bit long. If we can measure the length of this bit we can work out the speed at which the character is being sent. In fact we don't even have to be very accurate in our measurement since the possible speeds increase in multiples of two. For example if a bit at a given speed is x long we know that it would be 2x long for the next lower speed and x/2 for the next higher.

Figure 64
Speed Measurement



In the ITAC if the status/interrupt bit RCHR (receive character detected) is set in means that at least one start bit has been detected on the DCE/DTE interface since the last acknowledgement of RCHR. In fact it is set as soon as a start polarity is detected on the TxD line and not after the complete start bit has been detected. Hence it can be used to detect the beginning of the start bit and to start a software timer. Furthermore by repeatedly acknowledging the bit and testing it again, the continuing presence of the start polarity can be ascertained. As soon as the RCHR ceases to be set we know that the stop polarity is present on the line. In this way the end of the start bit can be determined and the timer stopped.

This is the relevant section of the code;

```
Start_count = 0;
                        /* wait for character to arrive*/
while (!(*it->pt_r_str & STR_RCHR))
    ;
do
    {
    Start_count++;                          /* Increment */
    *it->pt_r_stra |= STR_RCHR;             /* Acknowledge*/
    }                                       /* Test RCHR */
while (*it->pt_r_str & STR_RCHR);
```

As soon as the first 1 is received the do loop is exited. In order to store as much of the remainder of the string as possible we must program the ITAC for the detected speed and clear the USART before the current bit (1) is finished (see **figure 64**). Because we have the shortest time for the fastest bit rate the 19200 bit/s case is tested first. This accounts for the rather cumbersome testing of the value of Start_count. To save time the USART is already connected to the TxD line. It had been programmed to a speed of 38400 bit/s. It only requires that it be reprogrammed and reset.

```
If (Start_count <   4)                    /*19200 */
    *it->pt_r_brs  = BRS_U_19K  | BRS_N_19K;
else if (Start_count <   7)               /*9600 */
    *it->pt_r_brs  = BRS_U_9K6  | BRS_N_9K6;
else if (Start_count <  15)               /*4800 */
    *it->pt_r_brs  = BRS_U_4K8  | BRS_N_4K8;
else if (Start_count <  30)               /*2400 */
    *it->pt_r_brs  = BRS_U_2K4  | BRS_N_2K4;
else if (Start_count <  60)               /*1200 */
    *it->pt_r_brs  = BRS_U_1K2  | BRS_N_1K2;
else if (Start_count < 120)               /*600 */
    *it->pt_r_brs  = BRS_U_600  | BRS_N_600;
else                                      /*300 */
    *it->pt_r_brs  = BRS_U_300  | BRS_N_600;
                          /* Reset Rx USART but don't bother
*/
                          /*   waiting   for   command   ac-
cepted      */
    *it->pt_r_ucc = UCC_URR;
```

The boundaries for Start_count were determined empirically. These will vary for different microprocessors, C compilers etc. The progression of values reflect the fact that lengths increase as a factor of two as speeds decrease.

Once the reset has been given the time critical stage is passed. It only remains to acknowledge any break signal which may be recognized. Because the original speed had been set to 38400 bit/s it is possible that the start bit was longer than the break signal for this high speed.

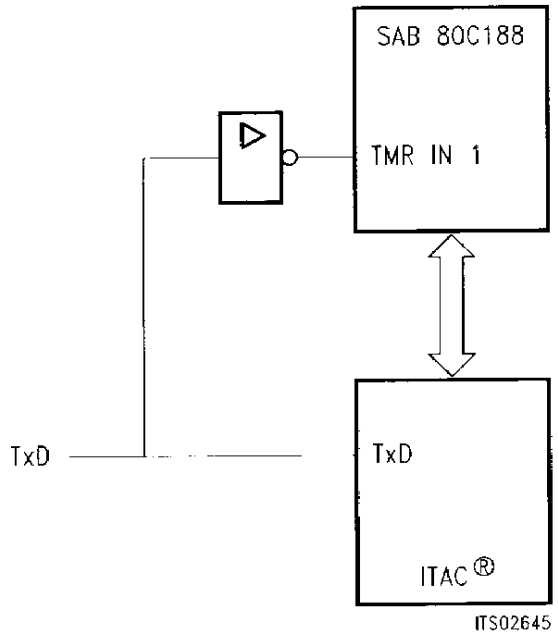
Software Failings

- In the above case the beginning of the start bit is detected in a while loop. Bearing in mind that the 'ISDN Modem' to terminal connection is useless before the speed of the terminal has been recognized, it is not so important that the firmware is in a blocked state while waiting for the start bit. Although RCHR is an interrupt it was found that on our system the use of an interrupt service routine took too much time. In the worst cases the start bit was already over by the time the routine was entered.
- The software timer is not very accurate. However, as was said above, the length of the start bit for different speeds increases/decreases by a factor of two. The timer doesn't need to be more accurate.

Because of these failings the timing method described here is admittedly neither very elegant nor very portable. However considering the small amount of time available to the program it does the job efficiently. It has the advantages that it can be written completely in 'C' and requires no special hardware modifications.

A more elegant method is outlined in **figure 65**. In this case the TxD line is input directly into the timer unit of the microprocessor. The timer is set-up and enabled to begin counting clock pulses on the falling edge of the first start bit. The rising edge will inhibit the clocking and signal a timer interrupt. The timer count is an indication of the data speed which is used to program the ITAC during the interrupt service routine. This method is more accurate and more efficient and such a method is to recommended when hardware is to be specially developed for an autobauding application.

Figure 65
Start Bit Timing in µP



Problem 2: Recognizing Bit Length and Parity

The Basic Procedure

Having recognized the speed, by whatever method, it is now possible to store the remaining bits in the character string A/a and all the bits of the T/t character string in the FIFO. From the stored values we can figure out the set-up characteristics of the terminal.

The first 3 bits received in all cases are 010. The first zero is the start bit which is used to measure the bit length. The time during the receiption of the first one is used to program the ITAC and reset the USART. If the USART reset is completed before the end of the receiption of the 1 then the second zero will be interpreted as a new start bit and the subsequent characters will be stored in the USART. To store the remaining bits of the A character string the USART is set to 7 bits wide, no parity and one stop bit. So, for example, for A with 8 character bits and a parity bit the following would occur;

A s 1000001 8 P S

s is used to measure the bit length.

1 is lost because the programming of the ITAC takes place at this time.

0 is interpreted as the new start bit.

000018 P is stored in the USART.

S The ITAC expects that the character stored has no parity bit and so the bit immediately following the character should be of Stop bit polarity.

Note: The blank spaces are meant to aid legibility.

Look at the value stored in the USART. It includes the parity bit and the eight character bit. In principle, the information we're looking for is in the FIFO.

The presence of fhe Stop bit is also important. As will be seen there will cases where a zero will be in the position where a stop polarity is expected. It is important to understand how this is interpreted by the ITAC in order to understand the algorithm itself. Take the example where there are seven bits, no parity, one stop bit and the subsequent T follows the A immediately.

s 1000001 S s 0010101 S SSSSSS

s is used to measure the bit length

1 is lost because the programming of the ITAC takes place at this time.

0 is interpreted as the new start bit.

00001Ss is stored in the USART.

0 In this case the expected stop bit is a zero. The ITAC now releases a framing error (FER) interrupt and interprets this 0 as a new start bit.

010101 S is stored in the USART FIFO as the second character.

S Finally a Stop bit is present at the end of the character as expected.

The Framing Error will be used extensively in the algorithm. Its main purpose is to overcome the difficulties caused by the fact that in an asynchronous system it is not known in advance how many stop bits will separate the two characters input.

Having dealt with the general principles we move on to the specific. **Figure 66** lists the possible sequences which can be received when using the sequence AT as input. In the algorithm used the usual sequence is to use a 7 bit wide USART to store the remaining bits of the A character string and an 8 bit wide USART to store the T character string. The stored character is available in the FIFO.

First we'll look at the simple case where there are plenty of stop bits between the characters of the character string. Consider the first example in **figure 66** illustrating what is received when AT is input from a terminal configured for 7 character bits, no parity, one stop bit (7,N,1). In this case two extra stop bits were sent between the characters. The significance of this will become apparent later. The first character read out of the FIFO is 70 and no framing error is reported since the bit immediately following is 1. Once again the LSB is received first (leftmost bit) and hence the value stored appears to be the reverse of the value in the bit sequence. The USART is then set to be 8 bits wide, again without parity, and the USART is reset. The next value read is D4. Because it is not know if a subsequent character is going be received immediately after, there could be a framing error reported. In this case however this is unimportant. We have stored all the remaining bits of the character string using this 7 bits wide USART/8 bits wide USART technique and all other information is superfluous. Note that the 7,N,1 setting can produce a number of special cases which will be dealt with in section The Decision. For the present these special cases will be ignored.

Now consider one of the cases marked with a * (e.g. 8,E,1). A framing error occurs at the end of the T character. In this case this information is not unimportant as this point coincides with the arrival of the parity bit of the T character. The same situation arises for all cases marked with a *. In these cases this information is vital to decide what type of parity is being used. Therefore the occurrence of the framing error at the end of the T character is always checked. If it occurs it is recorded for further evaluation. It is indicated by the presence of a '~' character in the tables.

So to recap the usual procedure would be;

a. determine the speed

b. use a 7 bit wide USART to store the remains of the A/a character

c. use an 8 bit wide USART to store the t/T character

d. use the framing error at the end of the second stored character to capture information which could not be squeezed into the FIFO.

There is one case, however, when this algorithm is inadequate. This occurs when a framing error occurs at the end of the first character (a/A). When this happens a slightly different approach must be used (described in the next section). Unfortunately the liberal widespread use of the framing error may prove a little confusing. If so the flow charts in Appendix A should help differentiate the different cases.

Figure 66
The AT Sequence Possibilities

	Character Strings								FIFO Contents	
	s	ASCII	8	P	S(S)	s	ASCII	8 P S		
7,N,1	0	1000001			11110		0010101	1X	70	D4 (˜)
7,N,1	0	1000001			11110		0010101	1X	70	D4 (˜)
7,E,1	0	1000001		0	11110		0010101	1 1X	50	D4
7,O,1	0	1000001		1	11110		0010101	0 1X	70	54
7,M,1	0	1000001		1	11110		0010101	1 1X	70	D4
7,S,1	0	1000001		0	11110		0010101	0 1X	50	54
8,N,1	0	10000010			11110		00101010	1X	50	54
8,E,1 *	0	10000010		0	11110		00101010	1 1X	10	54
8,O,1 *	0	10000010		1	11110		00101010	0 1X	50	54 ˜
8,M,1 *	0	10000010		1	11110		00101010	1 1X	50	54
8,S,1 *	0	10000010		0	11110		00101010	0 1X	10	54 ˜

without sufficient stop bits

	s	ASCII	8	P	S(S)	s	ASCII	8 P S		
7,N,1	0	1000001			1	0	0010101 !	1X	30 ˜	6A (˜) -> 54 -> D4
7,N,1	0	1000001			1 1	0 !	0010101	1X	70 ˜	54 -> D4
7,E,1	0	1000001		0	1	0	0010101	1 1X	50 ˜	54 -> D4
7,O,1	0	1000001		1	1	0	0010101	0 1X	70 ˜	54 ˜ -> 54
7,M,1	0	1000001		1	1	0	0010101	1 1X	70 ˜	54 -> D4
7,S,1	0	1000001		0	1		0010101	0 1X	50 ˜	54 ˜ -> 54
8,N,1	0	10000010			1		00101010	1X	50 ˜	54 ˜ ->54

where
0/1 bits stored in the FIFO.
! framing error in first character.
˜ framing error detected in second character.
(S) denotes superfluous stop bits which may or may not be present.

Table 49
The AT Decode Table

Parity	7 Bits		8 Bits	
N	7	D (¯)	5	5
E	5	D	1	5
O	7	5	5	5 ¯
S	5	5	1	5 ¯

Framing Errors after a/A and How to Deal With Them

A framing error can occur for certain character settings when there aren't sufficient stop bits between the characters. The procedure outlined above must be slightly modified when this occurs. Consider the case 7,E,1 when there are no superfluous stop bits between the characters of the sequence (**figure 66**; without sufficient stop bits section). A framing error is generated since a 0 follows the value stored in the FIFO and the ITAC was expecting a 1 in this position. Furthermore the FIFO will interpret this zero as a start bit and begin storing what comes afterwards as the second charater. This presents us with a problem. If, after storing the first 7 bit wide character in the FIFO, we proceed to change the FIFO width to 8 we will loose this character. The FIFO will discard the start bit already recognized and go looking for the next falling edge. This would mess up the second value stored in the FIFO. So in this case the FIFO is not switched to 8 bits wide but left at 7 bits wide. But as we have already seen we need to store 8 bits in the second character. When we look at what's stored in the FIFO in the 7,E,1 case we find 54. More importantly we see that the parity bit isn't stored in the FIFO . So we appear to have lost an important bit of information. In order to retrieve it we use the framing error again. If this bit is zero a framing error occurs and we can put a zero in the MSB position of the value read from the FIFO. Otherwise we put a 1 in this position. In this way we have patched together the value which would have been stored had we been able to use an eight bit wide USART in the first place. This correction is indicated in **figure 66** by the '→' transitions.

So in this case the framing error is used to identify the problem (when it occurs after a/A) and to correct for the shortfall in the FIFO data contents. In section Basic Procedure we saw the framing error being used for different settings and to a different end. Therefore there is no conflict in the use of this status bit. It is important to realize that although in both cases the parity bit is recognized using the framing error the subsequent procedures are completely different. In this case the framing error is used to recognize bits which would, under normal circumstances, be stored in the FIFO. In section Basic Procedure the framing error records the polarity of bits which would never under any circumstances be stored in the FIFO.

Note also that the case 7,N,1 with no extra stop bits between the characters is a special case which be handled in a special way.

No doubt the above appears quite complicated. Studying **figure 66** in detail is highly recommended to understand the cases and the flow chart diagrams to understand how to deal with them.

Evaluating What We Get

Once we have stored two characters in the FIFO we are in a position to figure out the terminal characteristics. It is important to note that the lower byte is the same in all cases (either 0 or 4 in the AT case). These bits are only used to double check that the expected characters were typed at the terminal. This prevents the algorithm from jumping to the false conclusions if 'Wq' were to be typed at the terminal.

The information pertaining to the parity and character length is in the upper byte. **Table 49** shows how these are evaluated. For example if the upper byte of the first FIFO character is 7 and of the second is D then the terminal must be set at 7 bits, no parity. With one exception each of the characteristic combinations produce a unique combination making it easy to identify the characteristics.

The exception is not such a great problem. In **table 49** we see that the cases 8,N,1 and 7,S,1 are indistinguishable. Compare the character A using these settings;

8,N,1	s	1000001	8	S
7,S,1	s	1000001	P	S

The character strings are exactly identical excepting the bits 8 and P. But, because of Space Parity, bit P is always zero. Furthermore for the seven bit character set bit 8 is always zero. The 7,S,1 case is really a subset of the 8,N,1 case. Hence the case 7,S,1 can be programmed as 8,N,1 without causing any errors.

In one case however the procedure fails to identify the settings. The mark parity bit looks the same as a superfluous stop bit. Both arrive after the ASCII character and both are 1. Since this is an asynchronous system any number of stop bits are allowed between characters. Hence it is impossible to tell the mark parity bit from the stop bits using this method. Similarly stop bits can be sent as part of the character string or as inter-character time fills. It is therefore also impossible to identify the number of stop bits which are programmed at the terminal. To overcome this limitation it would be necessary to ensure that the AT/at sequence is sent without any superfluous stop bits in between. This makes life more difficult at the terminal. Besides these settings do not occur all that frequently.

Therefore this program ignores this difficulty and interprets all these cases as No parity, One stop bit. This doesn't cause problems with the receiption of the characters. It simply looks as though there are always plenty of stop bits between characters. If a V.110 link is made no problems will be encountered as the link will be transparent and the extra 'stop bits' will be forwarded on to the remote. What will be problematical, however, will be those cases when the local FIFO is used to **write** to the local DTE. In this case the FIFO oper-

ation will not append the required number of ones at the end of the character in the FIFO. The terminal will see errors in the data stream. For example if it is expecting two stop bits per character it may only receive one. How it reacts to these errors is terminal dependant.

The AT String and What to do with it

The final case to be considered is the string at. **Figure 67** and **table 50** detail this case. The principles are the same. The ASCII character itself is different since it has a extra '1' bit in the lower case characters (see section above). This bit is referred to as the lower case bit. This bit effects the contents of the FIFO not only by simply being there but also because the even and odd parity values are affected. **Table 50** details the results received for the at case.

Figure 67
The AT Sequence Possibilities

	Character Strings								FIFO Contents	
	s	ASCII	8	P	S(S)	s	ASCII	8 P S		
7,N,1	0	1000011			11	1110	0010111	1X	78	F4 (˜)
7,N,1	0	1000011			11	1110	0010111	1X	78	F4 (˜)
7,E,1 *	0	1000011		1	11	1110	0010111	0 1X	78	74
7,O,1 *	0	1000011		0	11	1110	0010111	1 1X	58	F4
7,M,1 *	0	1000011		1	11	1110	0010111	1 1X	78	F4
7,S,1 *	0	1000011		0	11	1110	0010111	0 1X	58	74
8,N,1 *	0	10000110				11110	00101110	1X	58	74
8,E,1 *	0	10000110		1		11110	00101110	0 1X	58	74˜
8,O,1 *	0	10000110		0		11110	00101110	1 1X	18	74
8,M,1 *	0	10000110		1		11110	00101110	1 1X	58	74
8,S,1 *	0	10000110		0		11110	00101110	0 1X	18	74˜

without sufficient stop bits

	s	ASCII	8	P	S(S)	s	ASCII	8 P S		
7,N,1 *	0	1000011			1	0	0010111 !	1X	38˜	7A(˜) -> 74 -> F4
7,N,1 *	0	1000011			1 1	0	0010111	1X	78˜	74 -> F4
7,E,1 *	0	1000011		1	1 1	0	0010111	0 1X	78˜	74˜ -> 74
7,O,1 *	0	1000011		0	1	0	0010111	1 1X	58˜	74 -> F4
7,M,1 *	0	1000011		1	1	0	0010111	1 1X	78˜	74 -> F4
7,S,1 *	0	1000011		0	1		0010111	0 1X	58˜	74˜ -> 74
8,N,1 *	0	10000110				1	00101110	1X	50˜	54˜ -> 54

where 0/1 bits stored in the FIFO.
! framing error in first character.
˜ framing error detected in second character.
(S) denotes superfluous stop bits which may or may not be present.

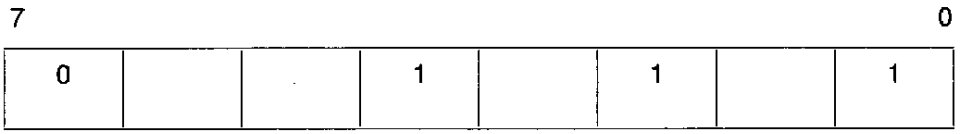
Table 50
The AT Decode Table

Parity	7 Bits		8 Bits	
N	7	F (˜)	5	7
E	7	7	5	7˜
O	5	F	1	7
S	5	7	1	7 ˜

The Decision

We now have all the information we can possibly get and its about time that we make a decision. However the information is bit scattered. We have two tables for each of the two character strings and we also have some additional information which is not contained in the upper bytes we have extracted from the FIFO characters.

First of all the lower case bit of the A/a character appears in the lower byte of the first FIFO word. Secondly there is still some framing error information in the tables (described using the symbol (˜) which isn't included in the decision word. We want to incorporate this information into the decision word to simplify the decision making process (simple switch/ case decision).
Now if we consider the words at present in tables below we find the following bits are always constant.



I used two of these erstwhile un-used bits to store the 'missing' information. The MSB was used the store the lower case bit of the A/a character. The LSB was reset when a framing error occurs (e.g the value 57˜ becomes 56). The two tables were combined into one and the corresponding modifications have been made. The result is the decision table in **table 51**.

Table 51
The Decision Table

Parity	7 Bits		8 Bits	
N N	7 F	D F	5 D	5 7
E E	5 F	D 7	1 D	5 6
O O	7 D	5 F	5 9	4 7
S S	5 D	5 7	1 9	4 6

The Special Cases of 7,N,1

Finally there are special cases which can occur for the 7,N,1 setting. These have been ignored up to now to avoid unnecessary complication, although they were detailed in **figures 66** and **67**. **Table 51** states that upper bit combinations which can be obtained with 7,N,1 are 7D and FF. However we saw in **figures 66** and **67** that a framing error can occur randomly at the end of the second FIFO character depending on whether another character follows the at/AT sequence or not. Because of the way the algorithm treats a framing error the cases 7C and FE are also valid when the setting is 7,N,1.

Another special case occurs when there are no superfluous stop bits between the characters. Here are the relevant lines from **figures 66** and **67**.

	s	ASCII	8	P	S(S)	s	ASCII	8	P	S		
7,N,1	0	1000001			1	0	0010101			1X	30˜	6A (˜) -> 54 -> D4
							!					
7,N,1	0	1000011			1	0	0010111			1X	38˜	7A (˜) -> 74 -> F4
							!					

To simplify matters a bit in the second byte is changed into 54 or 74 (the equivalent t/T character) in the code;

```
if (ichr == 0x6A)

ichr = 0x54;

if (ichr == 0x7A)

   ichr = 0x74;
```

This allows us to continue dealing with this case without having to change any of the procedures used for the non-special cases. It is now only a question of working out all the possible result combinations which could occur. Once again a framing error can occur randomly since the bit following what is stored in the FIFO is unknown (X). In this case, however, this causes the MSB of the upper byte to be set or reset randomly (see section Framing Errors). Hence the cases which can occur are;

35, B7, 3D, BF;

When dealing with these special cases I have simply let them filter through the algorithm, making as little modifications as possible to the FIFO data and simply reacting to every possible result combination which could occur. Although this is a bit complicated to follow, it generated as little extra code as possible. The switch case statement ended up with a few extra cases, but the basic handling of the received characters remained the same. A easier to follow (but less economical) way of dealing with these special cases would be to treat them separately. For example as soon 30/38 is read from the FIFO the program would jump to a separate part of the code which waited for 6A/7A. If some other value was found we would simply start again normally, looking for A/a. Framing errors etc. would be ignored. This is easier to visualise but more difficult to realize.

Appendices

Appendix A contains a flow chart description of the procedure described above.

Appendix B contains the corresponding code. For the most part the code should be self-explanatory. However a little overview wouldn't go amiss. The structure table

```
register ITACTAB    *it = itac;
```

contains all the pointer to the registers of ITAC. For example to reset the USART the code lines reads;

```
*it->pt_r_ucc = UCC_URR;
```

*it->pt_r_ucc	points to the ucc register of the ITAC
UCC_URR	refers to the URR bit of the UCC register

Other important definitions include;

BRS_U_9K6	The byte necessary to program the brs register to yield a User rate of 9600 bit/s
BRS_N_600	The byte necessary to program the brs register to yield a User rate of 600 bit/s
UMR_P_ODD	The byte necessary to program odd parity

Armed with this knowledge the code itself should be easy to follow.

For the sake of clarity the program is implemented in a continuous routine. In a many applications it will prove more sensible to use interrupt routines to detect the presence of the characters in the FIFO rather than waiting for URD in a while routine as is the case in the code presented here.

Appendix A

Figure 68
Speed Recognition Flow Diagram

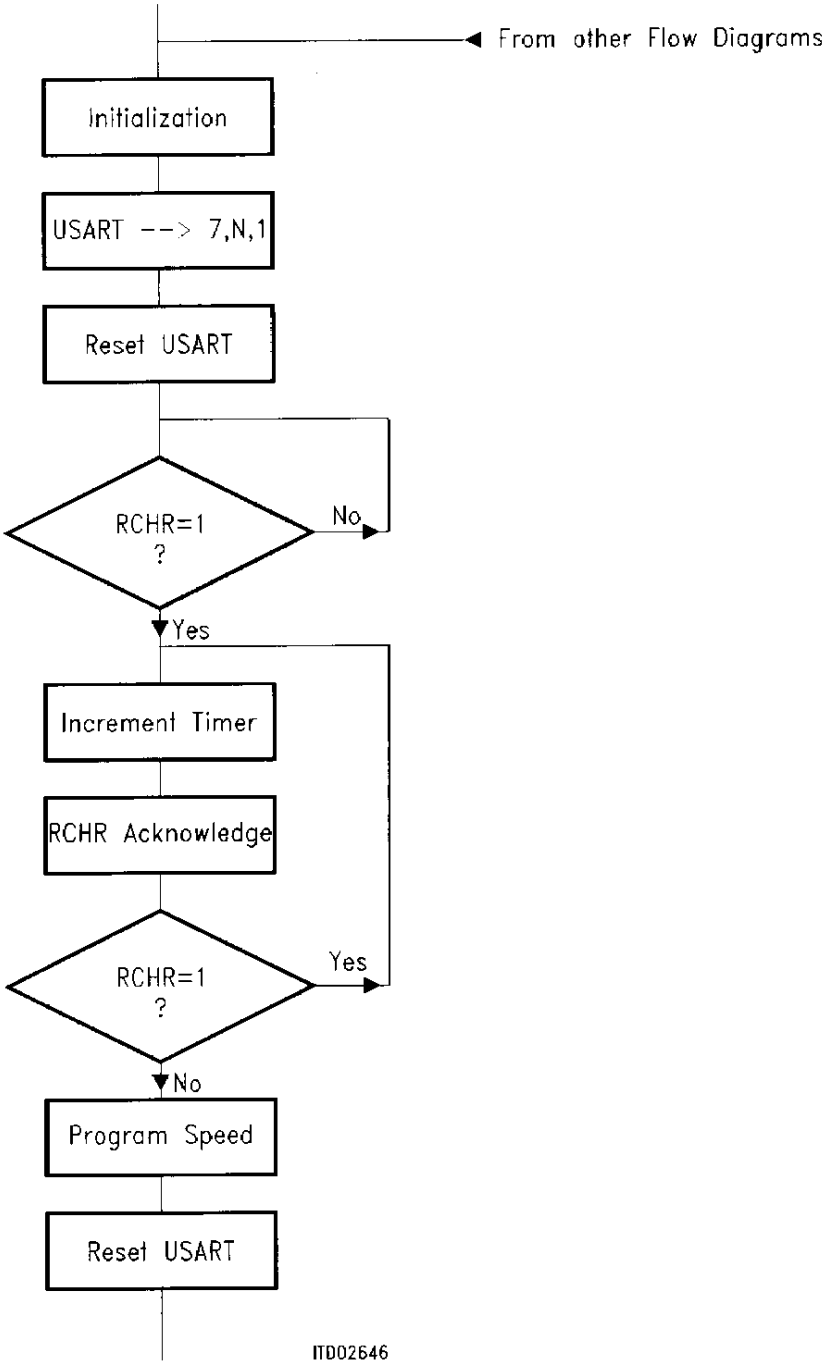


Figure 69
Reception of Remaining Bits of A/a (C1)

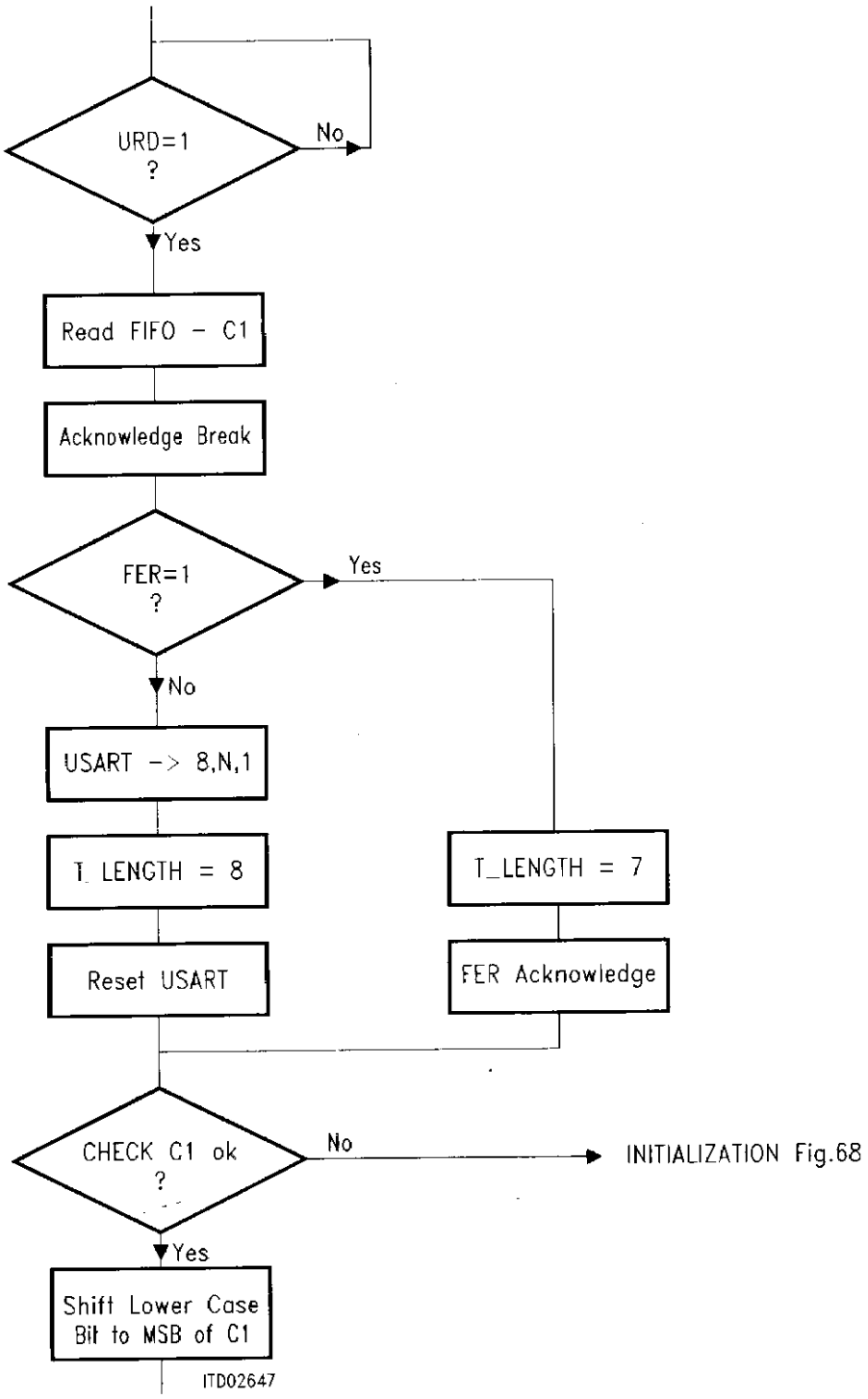
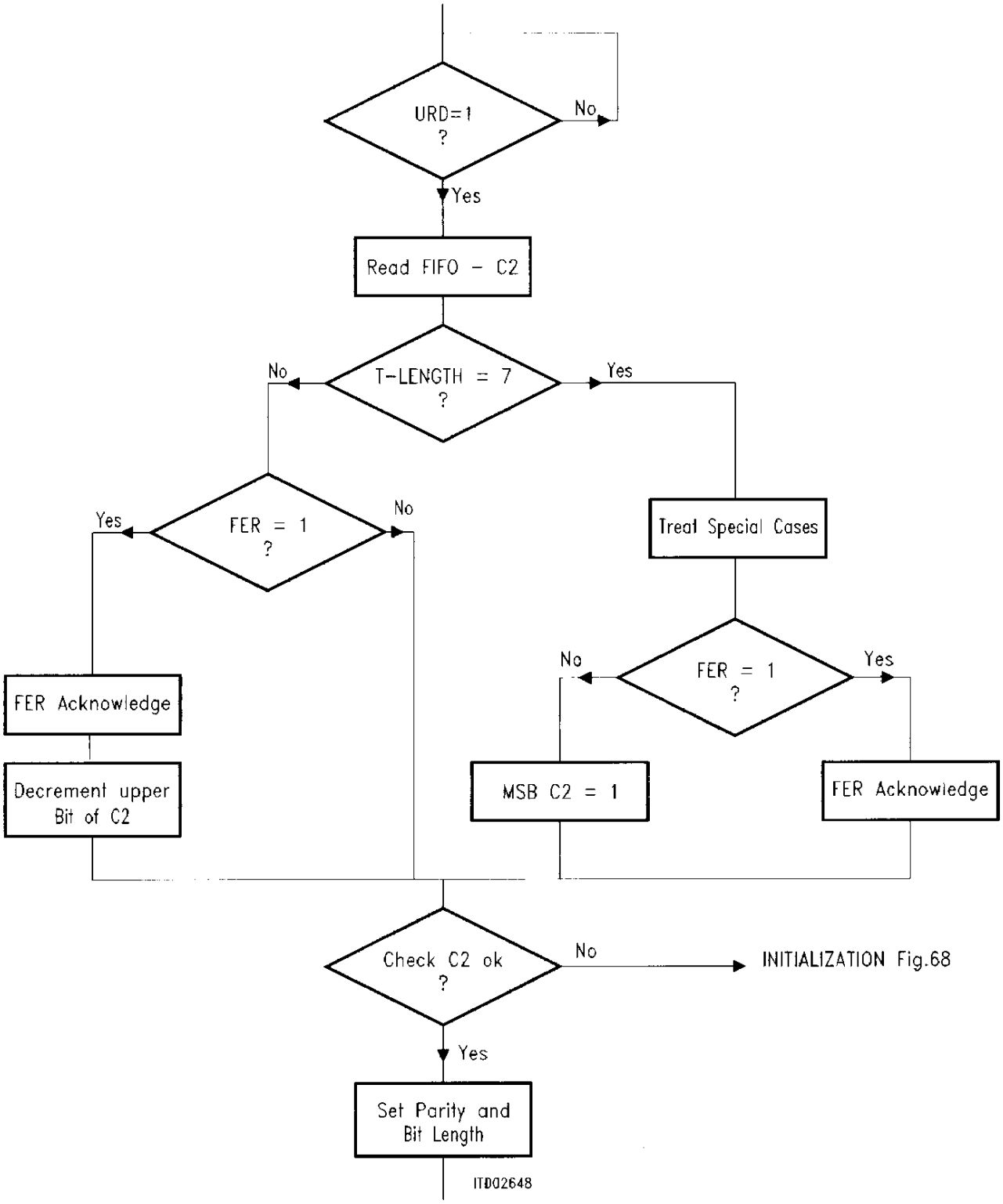


Figure 70
Reception of Character of T/t (C2)



Appendix B

C Code

```
*****************************************************************
*                  Function: Auto_speed_ITAC ()                 *
*                  Parms:                                       *
*                  purpose:                                     *
*****************************************************************/

EXPORT int
Auto_speed_ITAC ()
        {
        unsigned int   Start_count, T_length, i;
        unsigned char  ichr, Test_Byte;
        register ITACTAB   *it = itac;
                           /* USART connected to the TxD line    */
                           /* of the V.24                        */
        *it->pt_r_hmr &= ~HMR_HRLC;
                           /* USART is set to async and enabled  */
        *it->pt_r_umr  = UMR_ASYC | UMR_UXEN | UMR_UREN;
                           /* Don't want anything interrupting   */
                           /* our time-critical routines         */
        *it->pt_r_mask = MASK_ALL;

/*---------------AUTOBAUDING SPEED CALCULATION ------------------*/
TEST:
                           /* Program USART to be 7,N,1 at       */
                           /* 38400 bps                          */
        *it->pt_r_brs  = BRS_U_38K | BRS_N_38K;
        it->pt_r_aicr = AICR_B7 | AICR_SP_1;
                           /* Reset USART                        */
        *it->pt_r_ucc  = UCC_URR;
                           /* acknowledge everything used to     */
                           /* prevent residual effects           */
        *it->pt_r_stra |= STR_RCHR;
        *it->pt_r_rfsa |= (RFS_FER | RFS_PER);

        A_found = 0x00;
        T_found = 0x00;

        Start_count = 0;
                           /* wait for character to arrive       */
        while (!(*it->pt_r_str & STR_RCHR))
             ;
        do
             {
             Start_count++;
             *it->pt_r_stra |= STR_RCHR;
        }
```

```
while (*it->pt_r_str & STR_RCHR);
                    /* Work out the speed and program     */
                    /* BRS accordingly                    */
                    /* NOTE the boundaries have been      */
                    /*          chosen from empirical     */
                    /*          measurements              */
                    /* NOTE the most time critical        */
                    /*      speeds must be programmed     */
                    /*      first                         */

if (Start_count < 4)/* 19200                              */
     *it->pt_r_brs  = BRS_U_19K | BRS_N_19K;
else if (Start_count <   7)/*  9600                       */
     *it->pt_r_brs  = BRS_U_9K6 | BRS_N_9K6;
else if (Start_count <  15)/*  4800                       */
     *it->pt_r_brs  = BRS_U_4K8 | BRS_N_4K8;
else if (Start_count <  30)/*  2400                       */
     *it->pt_r_brs  = BRS_U_2K4 | BRS_N_2K4;
else if (Start_count <  60)/*  1200                       */
     *it->pt_r_brs  = BRS_U_1K2 | BRS_N_1K2;
else if (Start_count < 120)/*   600                       */
     *it->pt_r_brs  = BRS_U_600 | BRS_N_600;
else                /*   300                             */

     *it->pt_r_brs  = BRS_U_300 | BRS_N_600;
                    /* Reset Rx USART but dont bother     */
                    /* waiting for command accepted       */
*it->pt_r_ucc  = UCC_URR;
                    /* Clear any erroneous break          */
                    /* detection                          */
if (*it->pt_r_scs & (SCS_BRB | SCS_BRE))
     *it->pt_r_scsa = SCS_BRB | SCS_BRE;

Start_count = 0;
                    /* wait till all a/A character is     */
                    /* there                              */
                    /* NOTE; the timeout prevents the     */
                    /* program hanging because of         */
                    /* incorrect speed conclusions due    */
                    /* to erroneous characters being      */
i = 0;              /* input from the terminal            */

while (!*it->pt_r_rfs & RFS_URD))
     {
     if (i++ > 2000)
          goto TEST;
     }
                    /* Read in the byte from FIFO         */
```

```
ichr = *it->pt_r_ufifo;

if (!(*it->pt_r_rfs & RFS_FER))
     {                  /* Reset width of FIFO only if no      */
                        /* other byte is in FIFO               */
     *it->pt_r_aicr = AICR_B8;
     *it->pt_r_ucc  = UCC_URR;
                        /* To differentiate FIFO length       */
     T_length = 8;
     }
else
     {
     *it->pt_r_rfsa |= RFS_FER;
     T_length = 7;
     }
                        /* if the character is wrong start    */
                        /* autobauding speed process again    */
if ((ichr & 0x17) != 0x10).
     goto TEST;
A_found = (ichr & 0x70);

                        /* small letters being used           */
if (ichr & 0x08)
     A_found |= 0x80;

                        /* wait for next character 't/T'      */
while (!*it->pt_r_rfs & RFS_URD))
     ;

                        /* read in the byte from FIFO         */
ichr = *it->pt_r_ufifo;
                        /* if the FIFO was not changed        */
                        /* to eight bits wide this section    */
                        /* manipulates the T_found            */
                        /* character to be the same as it     */
                        /* would have been had the FIFO       */
                        /* been set to 8 bits wide            */
if (T_length == 7)
     {
                        /* treat special rare cases           */
     if (ichr == 0x6A)
          ichr = 0x54;
     if (ichr == 0x7A)
          ichr = 0x74;
     T_found = ichr;

     if (!(*it->pt_r_rfs & RFS_FER))
                        /* put one in upper bit               */
```

```
            T_found |= 0x80;
        else            /* acknowledge it                          */
            *it->pt_r_rfsa |= RFS_FER;
        }
    else
        {
        T_found = ichr;
        if (*it->pt_r_rfsa & RFS_FER)
            {           /* reset bit in value to indicate          */
                        /* FER has occurred                        */
            T_found &=~0x10;
                        /* Note that the FER interrupt has         */
                        /* a different meaning for the             */
                        /* T_length = 8 case                       */
            *it->pt_r_rfsa |= RFS_FER;
            }
        }
    if ((T_found & 0x4F) != 0x44)
                        /* if the character is wrong start         */
                        /* autobauding process again               */
        goto TEST;
                        /* prepare registers for changes           */
    *it->pt_r_aicr &= ~(AICR_B5 | AICR_SP_2);
    *it->pt_r_umr &= ~(UMR_PTY | UMR_P_EVEN);
                        /* Strip off the all- importanttart        */
                        /* upper bytes                             */
    TEST_Byte = (A_found | (T_found >> 4));
    switch (Test_Byte)  /* DECISION TABLE                          */
        {               /* 7 bits Even Parity One stop bit         */
        case 0x5D:
        case 0xF7:
            *it->pt_r_aicr |= AICR_B7 | AICR_SP_1;
            *it->pt_r_umr  |= (UMR_PTY | UMR_P_EVEN);
            break;
        case 0x75:      /* 7 bits Odd Parity                       */
        case 0xDF:
            *it->pt_r_aicr |= AICR_B7 | AICR_SP_1;
            *it->pt_r_umr  |= (UMR_PTY | UMR_P_ODD);
            break;
                        /* 7 bits No Parity One stop bit           */
        case 0x7D:
        case 0xFF:
        case 0x7C:
        case 0xFE:
        case 0x35:      /* These cases arises rarely;              */
        case 0xB7:      /* if 't' follows 'a' directly;            */
        case 0x3D:      /* without sufficient stop bits            */
        case 0xBF:
```

```
            *it->pt_r_aicr |= AICR_B7 | AICR_SP_1;
            *it->pt_r_umr  &= ~UMR_PTY;
            break;
                        /* 8 bits No Parity One stop bit      */
        case 0x55:
        case 0xD7:
            *it->pt_r_aicr |= AICR_B8 | AICR_SP_1;
            *it->pt_r_umr  &= ~UMR_PTY;
            break;
                        /* 8 bits Even Parity One stop bit    */
        case 0x15:
        case 0xD6:
            *it->pt_r_aicr |= AICR_B8 | AICR_SP_1;
            *it->pt_r_umr  |= (UMR_PTY | UMR_P_EVEN);
            break;
                        /* 8 bits Odd Parity One stop bit     */
        case 0x54:
        case 0x97:
            *it->pt_r_aicr |= AICR_B8 | AICR_SP_1;
            *it->pt_r_umr  |= (UMR_PTY | UMR_P_ODD);
            break;
                        /* 8 bits Set Parity One stop bit     */
        case 0x14:
        case 0x96:
            *it->pt_r_aicr |= AICR_B8 | AICR_SP_1;
            *it->pt_r_umr  |= (UMR_PTY | UMR_P_O);
            break;
                        /* get any other possibilities:       */
                        /* start again                        */
        default:
            goto TEST;
            break;
        }
    return (ACK_DONE);
    }
```

5.4 Alternative Solution for Autobauding

Introduction

Autobauding is used in terminal adapters to detect the data rate at which the terminal transfers its data. Two characters at the begin of each command string are evaluated and indicate the data rate and the character format.

Problems with the Current Implementation of an Autobauding Procedure

A solution for implementing autobauding with the ITAC has been issued in several implementations from various customers showed two weaknesses. First it is not possible to detect a character format of 8 bit plus parity bit in the case where the 'A' and 'T' character follow with only one stop bit. Second, the period to reset the USART receiver after reprogramming its character format is less than half a bit period. For high data rates, the interrupt handling takes longer and thus the character immediately following the 'T' got lost.

These problems can be solved in two ways. One way is to specify a minimum number of two stop bits between the 'A', the 'T' and the next character. The second way is by using an alternative autobauding solution which is limitied to data formats of 7 bit plus parity or 8 bit without parity.

Alternative Solution to Detect the Character Format

In most cases, the terminal adapter needs to detect only five character formats. These are 7 data bit plus one parity bit or 8 bit without parity bit. They have in common a total character length of 8 bit plus start bit and stop bit.

In the chapter 5.3 it is described that the USART has to be reset if the character format changes. This is only true if the total number of bits change, e.g. from 7 bit no parity to 8 bit with parity. If the overall character length is kept the same, the character format need to be changed before the parity bit is received. This gives at least 7 bit periods time to reprogram the USART.

In contrary to the chapter 5.3, the character format of the USART is now set to 8 bit no parity (8N1) before the speed is detected. Once the speed is detected, the USART receiver is enabled and receives both characters. The software will evaluate the pattern of the 'A' and the 'T' character. The first byte from the USART represents the 'A' character and it is compared against the following pattern:

MSB LSB

x 1 x 1 x 0 0 0

A framing error on the first character indicates that the 'T'- character follows immediately or with one stop bit. The MSB of the first character is tested to distinguish between two cases of handling the second character. If the MSB is zero than the second USART-FIFO character is compared against:

MSB LSB

1 x 1 x 1 0 1 0

A framing error on the second character indicates again that another character is following immediately. The software has to reprogram the USART format within the next 7 character bits. The character following the 'T' will be received correctly.

If no framing error is indicated or the MSB of the first character is one, the gap between the first and the second character is at least one additional stop bit. In this case, the second USART-FIFO byte is compared against:

MSB LSB

x 1 x 1 0 1 0 0

The software has more than 7 character bits time to reprogram the USART character format.

Evaluation of the Character Format

The character format is determined by testing three bits from both FIFO-values. The significant bits of the first FIFO character are bit 3 to 5. Bit 5 to 7 from the second FIFO character are the significant bits if no framing error has been detected. If a framing error has been detected, the significant bits are bit 4 to 6.

The bit values and the corresponding character format are shown in the following table.

Table 52

First Char	Second Char	Char Format
543	765	if no framing error on first char.
543	654	if framing error on first char.
010	010	8N1 or 701
011	011	8N1 or 701
110	010	7O1
011	110	7O1
010	110	7E1
111	011	7E1
110	110	711
111	111	711

8N1	8 bit, no parity, 1 stop bit	AICR=00H, UMR=B0H
701	7 bit. 0 parity, 1 stop bit	AICR=40H, UMR=F0H
7O1	7 bit, odd parity, 1 stop bit	AICR=40H, UMR=F2H
7E1	7 bit, even parity, 1 stop bit	AICR=40H, UMR=F4H
711	7 bit, 1 parity, 1 stop bit	AICR=40H, UMR=F6H

Figure 71 shows the timing diagram for receiving the AT-characters with a different number of stop bits. **Figure 72** shows the flow diagram for the character evaluation.

Figure 71
Receiving AT-Characters

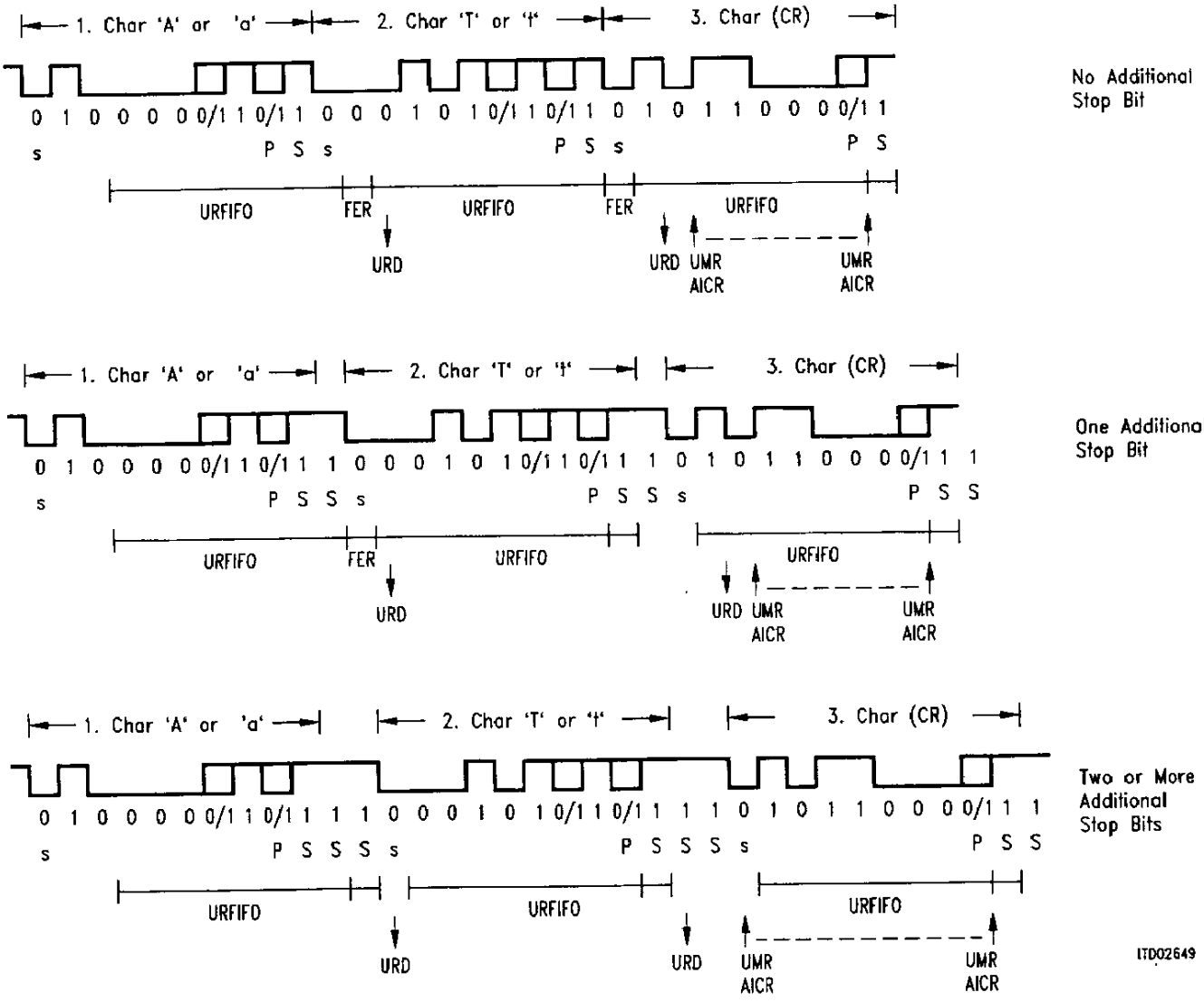
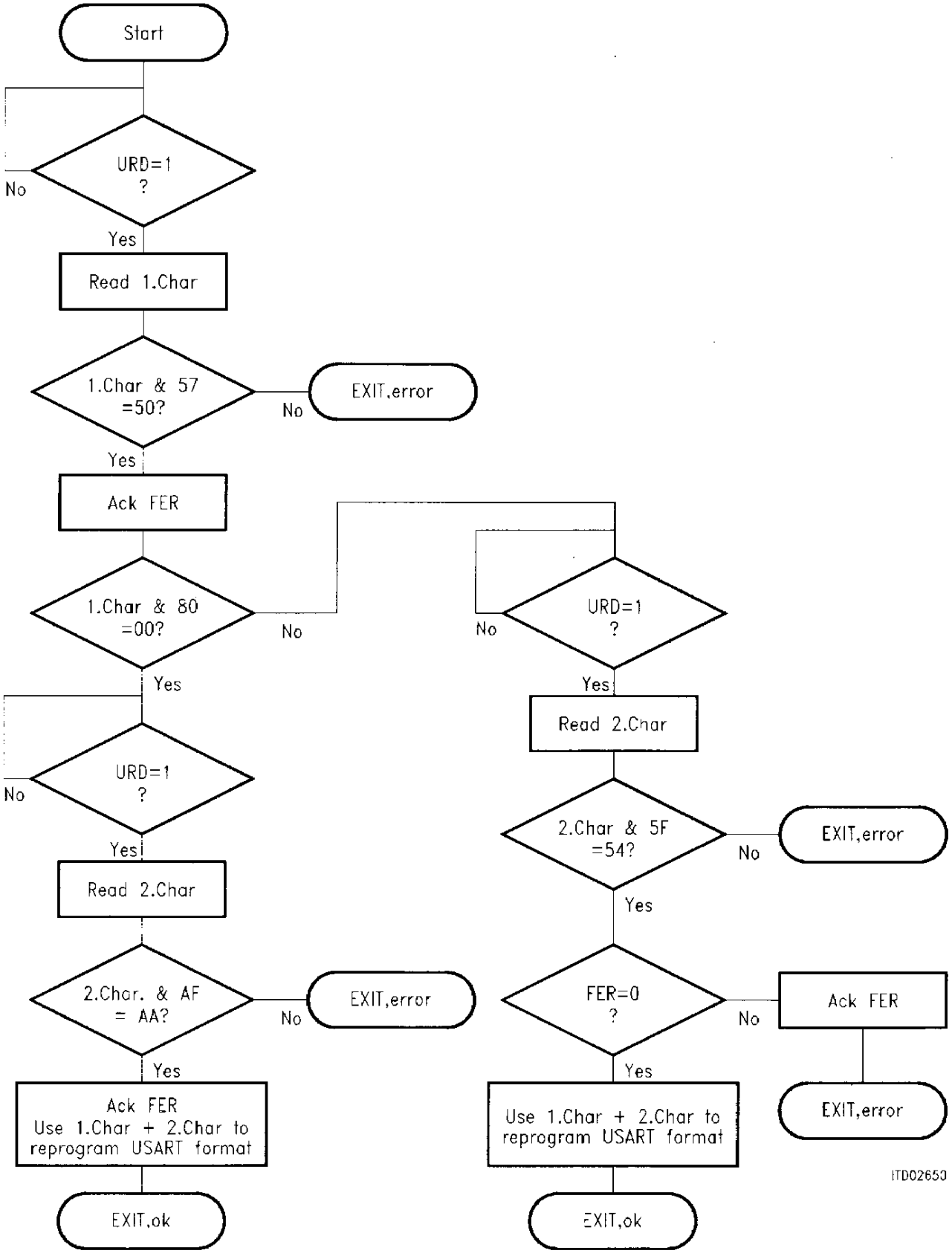


Figure 72
Flow Diagram



5.5 Octet Alignment for Data Transfer at 64 kbit/s by the ITAC®

Introduction

In many applications which transfer data at 64 kbit/s the octet alignment provided by the B-channel is used for synchronization. Examples for these applications are In-band Parameter Exchange (IPE), voice processing or handling of proprietary protocols. One may have assumed that aligned octet data transfer would occur by programming the ITAC into a data rate of 64 kbit/s and the USART to transparent mode. However the USART does not just transfer the data with the octet alignment provided by the time-slot.

How the SNI and USART Operate

In the receive direction, the synchronous network interface receives the time-slot information and forwards it to the USART as a continuous data stream at 64 kbit/s. The USART will start to receive octets after the TRA command has been entered or the USART receiver has been enabled. The relation between the time-slot information and the USART FIFO data depends on the timing when the command is executed.
In the transmit direction, the USART sends a continuous data stream at 64 kbits/s to the SNI. The SNI cuts the continuous data stream into octets and transmits them in the selected time-slot. The relation between the USART FIFO data and the data on the time-slot depends on when the USART transmitter is enabled.

How to Achieve the Octet Alignment

With the following procedure it is possible to transfer data octets aligned to the time-slot. The procedure is time critical and therefore it is recommended to turn off all interrupts. The synchronization procedure roughly needs 500 microseconds.

The IRC transmitter generates an XDB interrupt after eight data bits have been sent. Since the network rate is set to 64 kbit/s, the XDB interrupt is generated at the end of a time-slot. For the transmitter the alignment is achieved by enabling it after a delay of between 83 and 97 microseconds from the XDB interrupt.
In the receive direction the principle is for the receiver to synchronize on a known character being sent from the transmitter and looped round to the receiver by 'test loop 3'. Therefore a known character is written into the UXFIFO and into the SYNC register. Since the transmitter continues to repeat the last character entered in the UXFIFO, only one value must be entered. The receiver is set to mono-sync which will generate an URD interrupt after two know characters have been received. Once the correct value is received, 'FF' is written to the UXFIFO to transmit idle bits and the test loop 3 is deactivated. The USART transfers now all data aligned to the time-slot.

IT SHOULD BE NOTED THAT A USART TRANSMITTER OR RECEIVER RESET COMMAND WILL DESTROY THE ALIGNMENT.

Also note that there will be another one or two synchronization characters sent over the SNI before the 'FF' value is sent. If this causes problems in the application, the deactivation of test loop 3 may be delayed until the 'FF' character is read from the URFIFO.

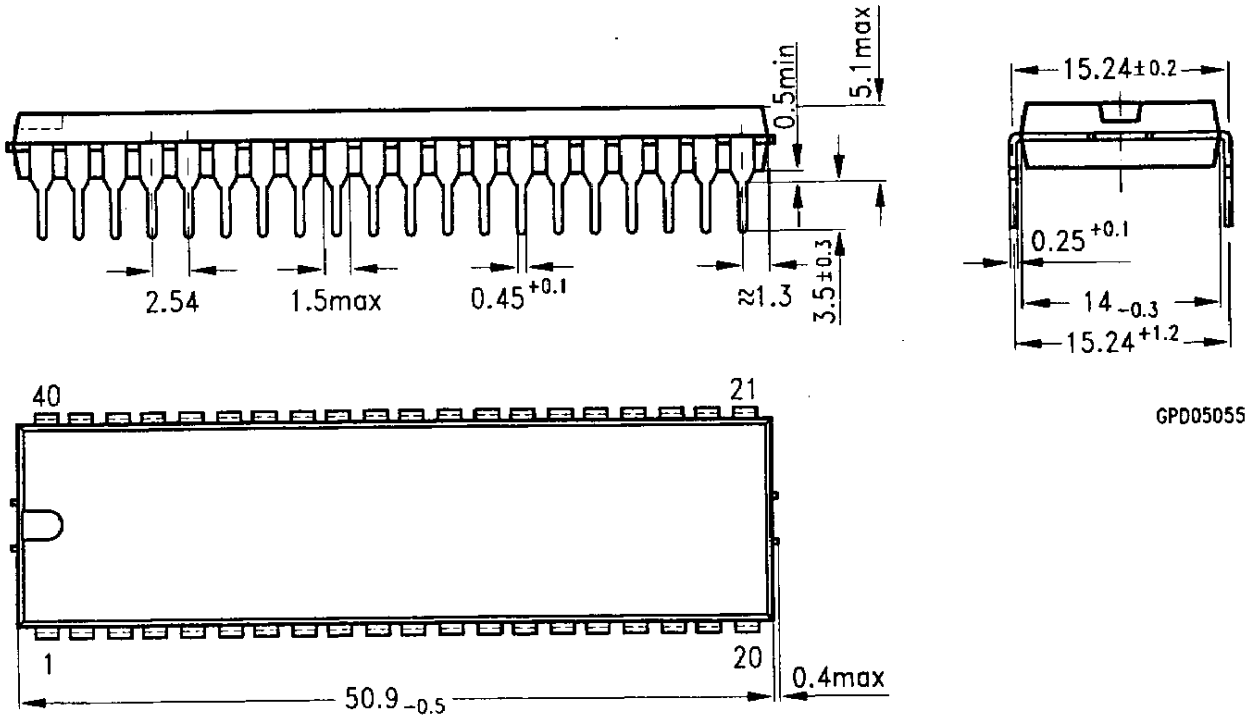
Pseudo Code

This listing shows the necessary register setting of the ITAC. It has been tested on a system using a 80C31 CPU, an ITAC and a HSCX to varify the alignment.

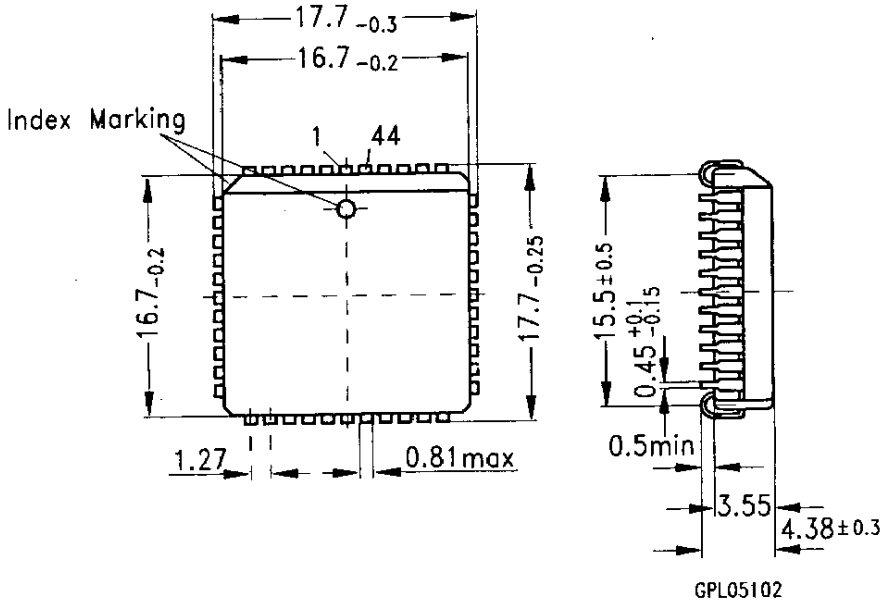
```
write to ITAC_GCR reg C0H          ; this powers up the chip, enables DCE as sync DCE
wait 100ms                         ; waiting to ensure that the DPLL is locked before the
                                   ; IRC is enabled
write to ITAC_BRS reg AAH          ; ur = nr = 64000
write to ITAC_DPCR reg 51H         ; connect SCL to D bits, activate test loop 3
write to ITAC_GCR reg C4H          ; Enable IRC by setting ENFR = 1
write to ITAC_IST reg FFH          ; acknowledge all interrupts outstanding
write to ITAC_HMR reg C0H          ; connect usart to IRC
write to ITAC_UMR reg 00H          ; disable usart
write to ITAC_UCC reg C0H          ; reset usart transmitter and receiver
write to ITAC_ISEN reg 00H         ; disable all interrupts
disable all system interrupts
write to ITAC_SYN reg 41H          ; prepare sync. character
write to ITAC_UFIFO reg 41H        ; prepare usart fifo with character
write to ITAC_STR reg 02H          ; acknowledge XBD
wait till ITAC_STR bit XDB = 1
wait 83 microseconds               ; Delay of 83 to 97 us before enable TX
write to ITAC_UMR reg 38H          ; enable TX. and RX. TX is now octet aligned. RX is
                                   ; mono-sync
wait till ITAC_RFS bit URD =1      ; the receiver is than sync
read ITAC_UFIFO                    ; should be 41H (sync character)
write to ITAC_UFIFO reg FFH        ; set USART to idle (transmit '1')
write to ITAC_DPTR reg 50H         ; turn off test loop as we are now synchronized
activate all system interrupts
```

6 Package Outlines

Plastic Dual-in-Line Package, P-DIP-40



Plastic-Leaded Chip Carrier, P-LCC-44 (SMD)



SMD = Surface Mounted Device