Application Note **MSAN-118**

Rate Adaptation, Switching and Transmission of RS-232 Signals

ISSUE 1 January 1986

Contents

1. Introduction

The advent of monolithic devices capable of switching large numbers of PCM encoded voice signals has been instrumental in making the design of digital PBXs practical and cost effective. Traditionally, the role of telecommunication equipment has involved the transfer of voice signals. As a result, the cost of integrated circuit components to handle voice traffic has continued to decline dramatically.

The Mitel MT8950 Data Codec is a device which can be used to capitalize on the low cost of voice band (64 kbps) switching and transmission for data communications. It converts low speed digital signals originating from RS-232 type sources into a format which occupies a single 64 kbps channel on a serial communications stream. The Data Codec can accept digital signals at any speed from 0 to 8 kbps. It can also handle data rates of 9.6 and 19.2kbps. The data can be asynchronous or synchronous. The only restriction for asynchronous data at 19.2 kbps is that it must have two stop bits. The device is completely protocol independent for the other data rates. The incoming low speed data is

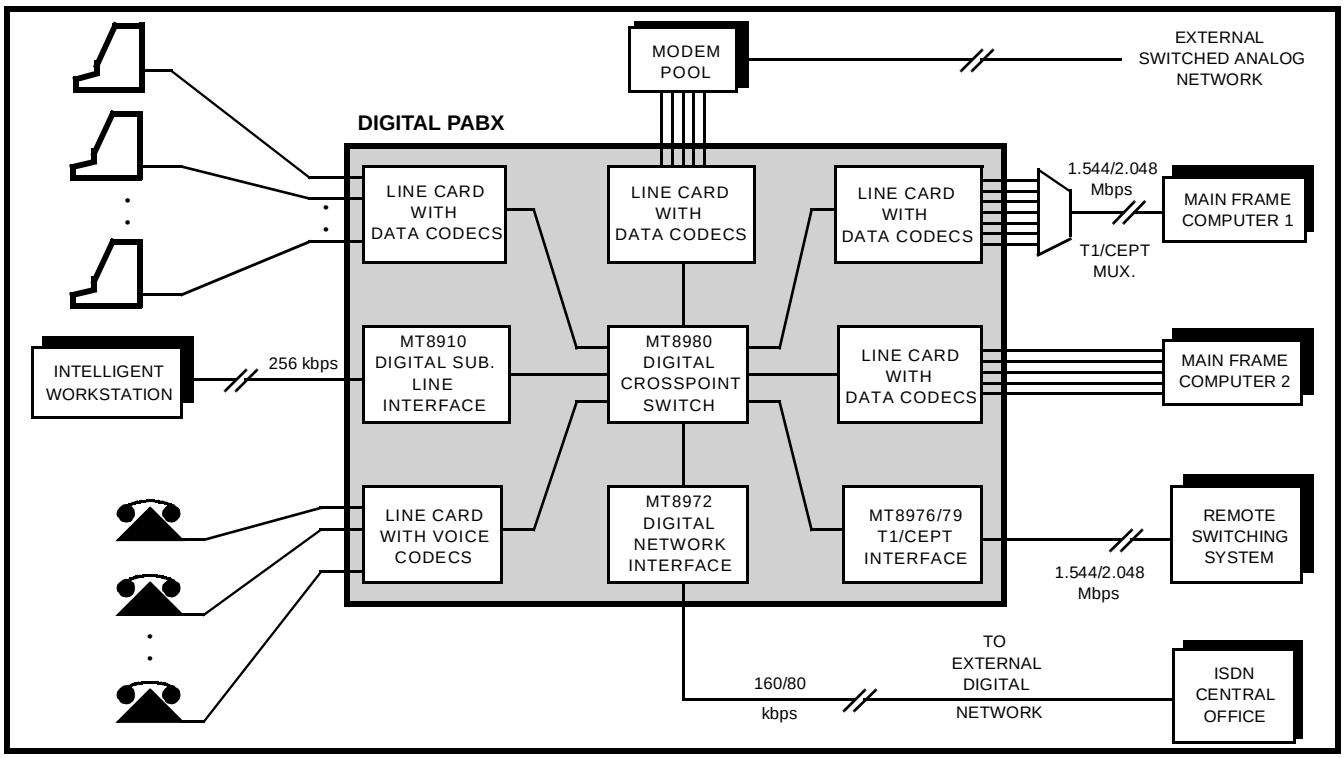

Figure 1 - Voice-Data Integrated PABX Using the Data Codec and Other ST-BUS Components

effectively sampled at 256kHz. The sampled information is encoded on to a single 64 kbps channel. The proprietary encoding/decoding scheme used by the device, significantly reduces the bit period timing error in the regenerated data.

The Data Codec is fully compatible with Mitel's ST-BUS. The ST-BUS is a time division multiplexed serial stream with an aggregate bit rate of 2048 kbps. In a telecommunications environment, the ST-BUS is generally divided into 8 kHz frames with 32 channels per frame. The effective bandwidth of each channel is 64 kbps.

This note will highlight some of the functional features of the MT8950 using practical application circuits. Solutions to specific problems, including the handling of RS-232 control signals and call set up procedures, are also discussed. This document is designed to complement the data sheet. For a more detailed description of the device pinouts and parametric information, the reader should refer to the MT8950 data sheet.

2. General System Application Overview

A typical star configuration of an integrated Voice-Data switching system is illustrated in Figure 1. In this configuration all communication lines terminate in a central unit that contains the switching intelligence required to allow any caller to be connected to any destination. For data communication systems, the callers are usually terminals or personal computers. The destinations are usually mainframe computers, printers or modems. However, any number of different combinations are possible. In all cases, digital equipment operating at low speed is interfaced to the system through the Data Codec.

One of the main attributes of the Data Codec is that the output of the device is in the same format as that of the Voice Codec. Thus, in systems utilizing ST-BUS components, true voice-data integration is possible. The central switching matrix can handle both the encoded voice and data signals. This provides for a flexible system which can be reconfigured to accommodate changing voice and data communications requirements. The encoded information generated by the Data Codec can be transmitted over digital lines to a remote location using one of the ST-BUS transmission components shown in Figure 1.

3. Functional Overview of Data Codec Operation

The Data Codec uses a Mitel proprietary encoding/decoding scheme referred to as Transition Encoded Modulation (TEM). The low speed data is sampled at a frequency of 256 kHz. The position of the first transition, the total number of transitions and the time interval between consecutive transitions within a 125 µs time period is encoded as an eight bit word. The Data Codec at the remote end accepts this eight bit word and regenerates the original signal from it. A summary of the capabilities of the device and the maximum error in the bit period timing introduced due to the encoding/decoding of the signal is presented in Table 1.

The circuit presented in Figure 2 shows a simple RS-232 interface to the ST-BUS. This particular configuration is designed to illustrate the use of the Data Codec in a very basic application where only the rate adaptation features of the device are utilized.

The data originating from the RS-232 TXD pin is level shifted and input at D_X1. Since the data is in the NRZ format, the Data Format (DF) pin on the Codec is tied LOW. This will ensure that the codec receives and transmits data in the NRZ format. The secondary input (D_X2) is not utilized in this particular application and is therefore tied to V_{SS}. The low speed data is encoded by the encoder section to generate an eight bit word. This 8 bit encoded word is transmitted via the DSTo output to the digital switch (the MT8980). The byte can be switched by the MT8980 to any specific remote Data Codec within the PBX. It can also be transmitted using one of the ST-BUS transmission interfaces to a remote location where another Data Codec is used to decode the received information.

Data Rate (bps)	Restrictions		Maximum Bit Period Timing Error*
	Sync. Data	Async. Data	
0-8000	None	None	3.9µs
9600	None	None	
19200	None	Minimum 2 stop Bits	

Table 1. Summary of Data Codec Capabilities

** Refers to the maximum timing error in the bit period of the regenerated signal.*

At the same time as the encoded 8 bit word is being output by the codec, the DSTi input accepts an 8 bit word from the switch. This word is decoded by the codec to regenerate the low speed NRZ signal which is output at D_R1. After level conversion from TTL to RS-232 compatible levels, the regenerated signal is input to the DTE directly.

The Data Codec does not require any programming to indicate to it the bit rate of the data originating from the RS-232 terminal. As mentioned previously, any low speed digital signal transmitted at the specified bit rates will be encoded correctly.

The Framing Type 1 ($\overline{F1i}$) and the Control Address ($\overline{CA}$) inputs are used to enable the ST-BUS interface (DSTi, DSTo and CSTi) on the chip. The relationship between the signals is illustrated in Figure 3. When the $\overline{F1i}$ and the $\overline{CA}$ inputs are both set LOW, the Data Codec will output a serial eight bit word via the DSTo pin. During this same time, serial 8 bit words are also clocked into the device via the CSTi and DSTi inputs. Thus, by enabling the interface at a specific time in the ST-BUS frame, the codec can be assigned a unique channel timeslot during which it will output and accept encoded information.

The eight bit serial word input via the CSTi pin is used to load the internal Control Register. In very simple applications where only the rate conversion feature of the codec is utilized, the CSTi input pin can be tied to V_{SS}. This will load all zeros into the internal Control Register and cause the device to operate in its normal mode. If other modes such as loop-back, local carrier mode, idle, etc. are to be used, an 8 bit word can be loaded into the Control Register via the CSTi serial input. A summary of the various modes of operation affected by the Control Register is presented in Table 2. The internal register is loaded once every ST-BUS frame. The correct 8 bit word can be loaded into the register by using the messaging capabilities of the MT8980. The appropriate channel on the ST-BUS stream connecting the MT8980 digital switch to the CSTi input of the MT8950, is put in message mode (refer to the MT8980 Data Sheet for more details on how this is done). In the MT8980 when a specific output channel is put in message mode, the 8 bit word placed in its connection memory is output during that channel timeslot. For example in Figure 2, output Stream 0 (STo0) from the MT8980 is shown connected to the CSTi pin of the Data Codec. If channel 1 on this stream has been assigned to the Data Codec, then channel 1 on Stream 0 of the MT8980 is put in message mode. The contents of

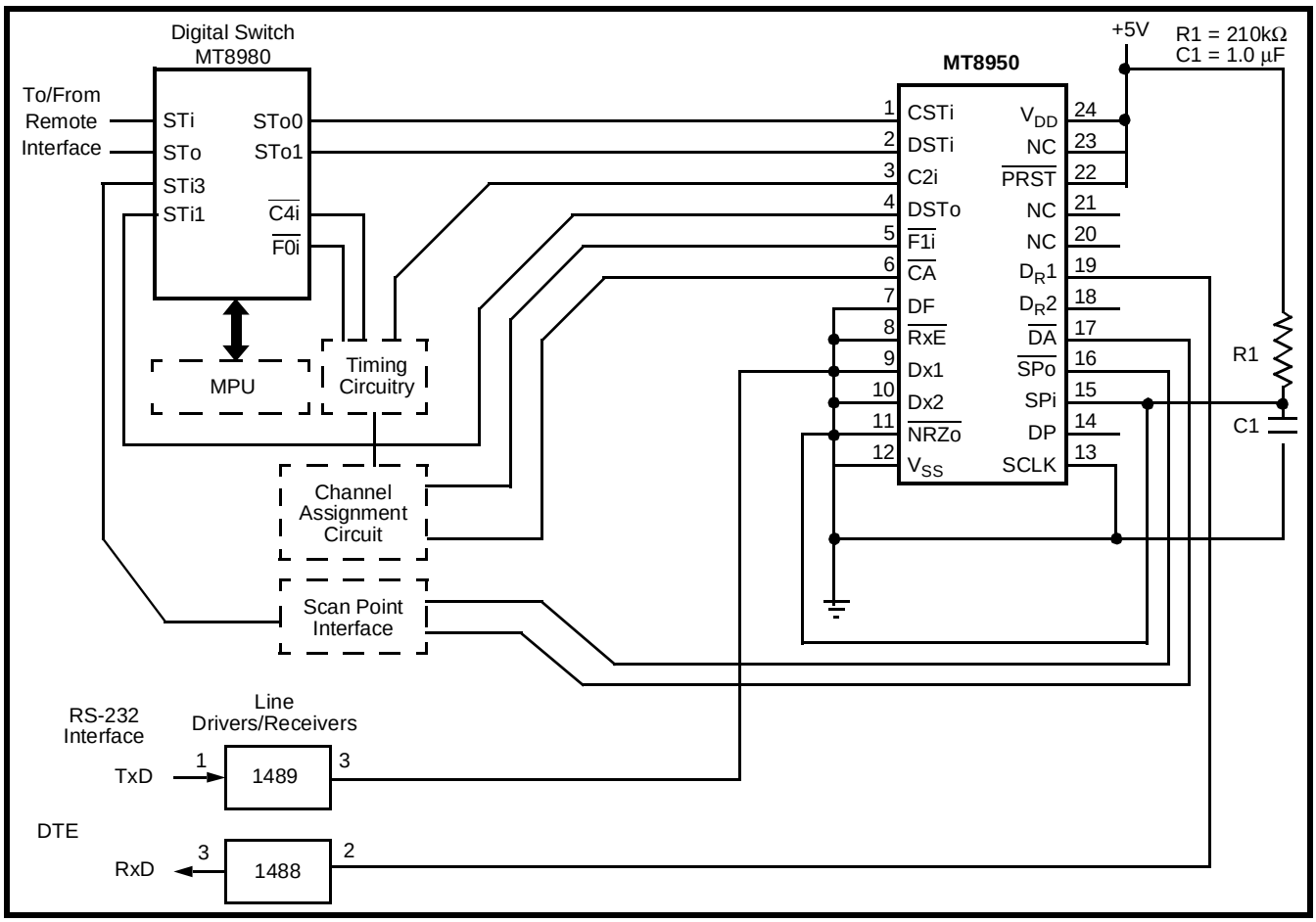

Figure 2 - Simplified RS-232 Interface Using the Data Codec

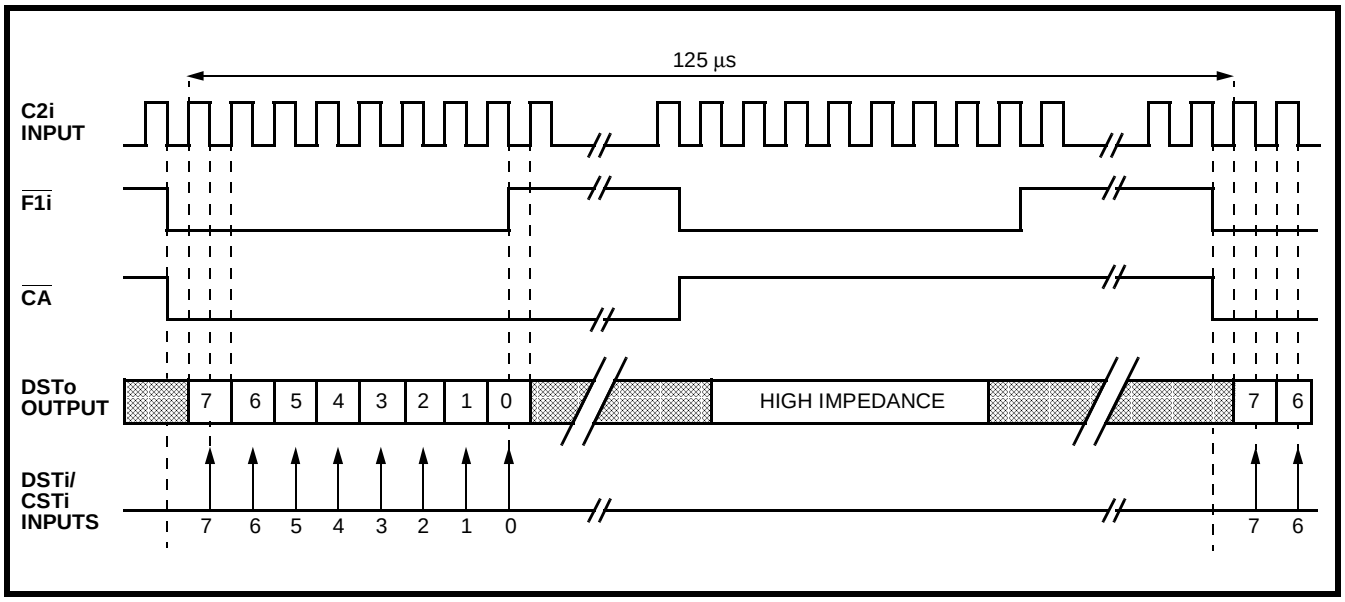

Figure 3 - Timing Diagram - 125µs Frame Period

the connection memory corresponding to this channel and stream, will be output by the MT8980 during the specific channel timeslot and subsequently accepted by the Data Codec at the CSTi pin and loaded into the internal Control Register.

Bit	Function
7,6,5	Device mode control bits. These bits select one of eight modes of operation
4,3,2	These bits control the logic level of bits 1, 4 and 5 in the violation word output in the Local Carrier mode.(See section 4.1)
1	Resets the Data Activity Scan point
0	Resets the Uncommitted Scan point

Table 2. Summary of Control Register Function

The Received Energy ($\overline{RxE}$) pin is used for assigning a specific polarity to the D_X1 and D_X2 pins when the Data Codec is being used in the Return to Zero mode (see MT8950 data sheet for more details). In the Non-Return-to-Zero ($\underline{NRZ}$) format, polarity detection is not required. The $\overline{RxE}$ pin can, therefore, be tied LOW. In both cases, this pin should be asserted LOW for the duration of the call.

There are two output pins on the Data Codec which can be used for monitoring the device. The Data Activity ($\overline{DA}$) pin goes from HIGH to LOW when the first MARK to SPACE transition occurs in the data. The $\overline{DA}$ output can be set HIGH by writing a '1' to bit one of the Control Register. This output pin can be monitored by the system processor to detect the beginning of data activity and used to disconnect a line which has been inactive over a predefined period of time during the initial call set up. It may also be used as a 'request for service' signal by an interface.

The second monitoring feature is the Scan Point Output ($\overline{SPo}$). The level on this output is asserted LOW when the Scan Point Input (SPi) undergoes a HIGH to LOW transition. The $\overline{SPo}$ pin can be set to its original HIGH state by loading a '1' into bit 0 of the Control Register. In conjunction with the $\overline{NRZo}$ pin, the Scan Point output can be used to detect a space signal of varying time durations. The circuitry required to implement this is shown in Figure 4. The values of R1 and C1 in Figure 2 have been chosen to permit detection of a long SPACE exceeding 150ms. In most data terminal equipment, a 150ms long SPACE is generated when the BREAK key on the keyboard is pressed. Thus pressing the BREAK key will cause the Scan Point output to go LOW. The system processor monitoring this output can interpret this as a prompt from the peripheral. The prompt may initiate a call set up process or it may direct the system processor to execute some other predefined routine, in the middle of a communication session. The specific applications depend on the architecture of the system.

The specific applications depend on the architecture of the system.

The Secondary Clock (SCLK) is an external clock input that is used for modulating the signal output on D_R2 in the Local Carrier mode. This clock signal is also used for generating the "synchronization" pulses which are output on D_R2 when the data decoded from the remote end is idling or when the device is operating in mode 2. Since none of these features are required for this particular application, the SCLK pin is tied LOW and the output at D_R2 is not monitored (see Section 4.2 for more information on the uses of this clock).

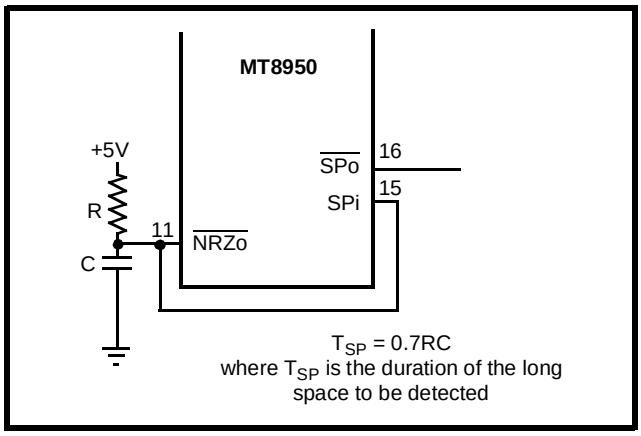

Figure 4 - Long Space Detection Circuit

The Drive Point output (DP) is an uncommitted output pin which is asserted HIGH when the codec is operated in mode 6. It is LOW in all other modes. This output pin can be used to control external circuitry.

4. RS-232 Signalling and Control Through the ST-BUS.

The EIA RS-232 standard defines the interface between Data Terminal Equipment (DTE) and Data Communications Equipment (DCE). In the majority of asychronous applications, only a subset of the extensive list of signal elements defined in the standard is implemented. A summary of the more commonly used signal elements is presented in Table 3. The abbreviations shown in Table 3 are used throughout this document when any reference to these signals is made. Signals such as DTR, RTS, CTS, etc., which are not directly involved in transmitting and receiving data, are collectively referred to as control signals.

In a data switching system such as the one described in Section 2, the handling of the control signals depends on the specific type of connection made through the switch. In DTE-DCE type connections (i.e., a terminal connected through the switch to a modem) the two peripheral devices should effectively believe that they are connected to each other directly. Therefore, RTS turned ON by the terminal is relayed to the modem transparently. Conversely, a CTS turned ON by the modem is relayed back to the terminal. In a switched circuit, this would require transmission of the control information transparently from one interface to the other as shown graphically in Figure 5.

When the two devices connected through the data switch are both terminating type devices, e.g., a terminal connected to a computer, both the RS-232 interfaces on the PBX should appear as DCE to the respective devices. Transparent switching of the control signals from one side to the other would not necessarily serve any useful purpose. Thus some sort of system intervention would be required to interpret the signals asserted by one interface and transmit the appropriate information to the remote interface. In the example protocol illustrated in Figure 6, RTS turned ON at one interface is used to turn ON DCD at the remote end. The remote end echoes back a signal which causes the originating side to turn ON CTS. Since the actual control information exchange uses the same transmission media as the data, the CTS signal indicates that a secure communication channel is available. The local interface has effectively interrogated the remote end to ensure that communication is possible before turning over the channel to the DTEs. Note that if the DTEs connected through the interface were to operate in half duplex, the signal exchange would still be valid.

Pin No.	Signal Description	EIA Ckt.	Common Abbrev.	From DCE	To DCE
1	Protective Ground	AA	GND	X	X
2	Transmitted Data	BA	TXD		X
3	Received Data	BB	RXD	X	
4	Request to Send	CA	RTS		X
5	Clear to Send	CB	CTS	X	
6	Data Set Ready	CC	DSR	X	
7	Signal Ground / Common Return	AB	SG	X	X
8	Received Line Signal Detector	CF	DCD	X	
20	Data Terminal Ready	CD	DTR		X
22	Ring Indicator	CE	RI	X	

Table 3. RS-232 Pin Description

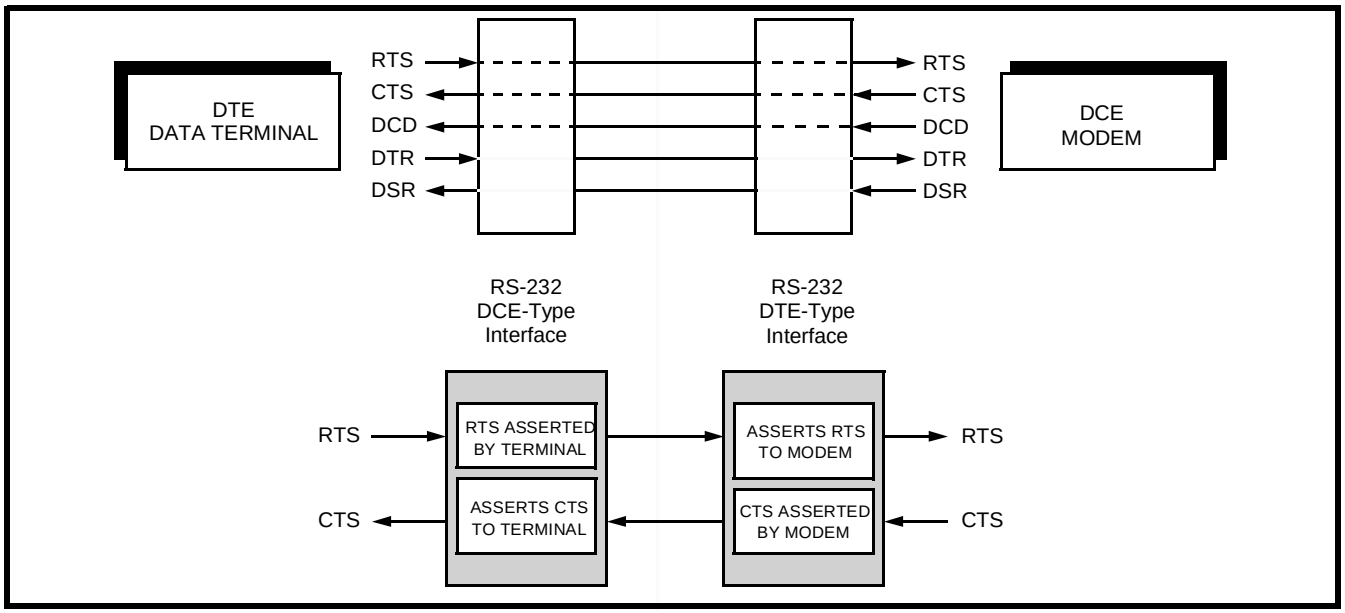

Figure 5 - RS-232 Signal Exchange in a DTE-DCE Type Connection

In this particular example, DSR is turned ON by the interface circuit when it has been connected to the destination interface, or when the main switching controller is ready to receive call set up instructions. The Data Codec interface described above complies with all the RS-232 requirements outlined in the EIA standard specification.

From the preceding discussion it is apparent that two types of interface circuits are required to provide the flexibility of connecting various devices through the data switch. In this document, the circuit configuration which permits direct connection to a DTE is referred to as a 'DCE-type' interface. The signal flow direction for this interface is the same as that for an RS-232 interface on a DCE. Conversely, the circuit which allows direct connection to a DCE is referred to as a 'DTE-type' interface.

A variety of different circuits can be devised around the Data Codec to implement the functionality discussed above. Figure 7 illustrates, in block diagram format, three different configurations which can be implemented with a minimum of extra circuitry.

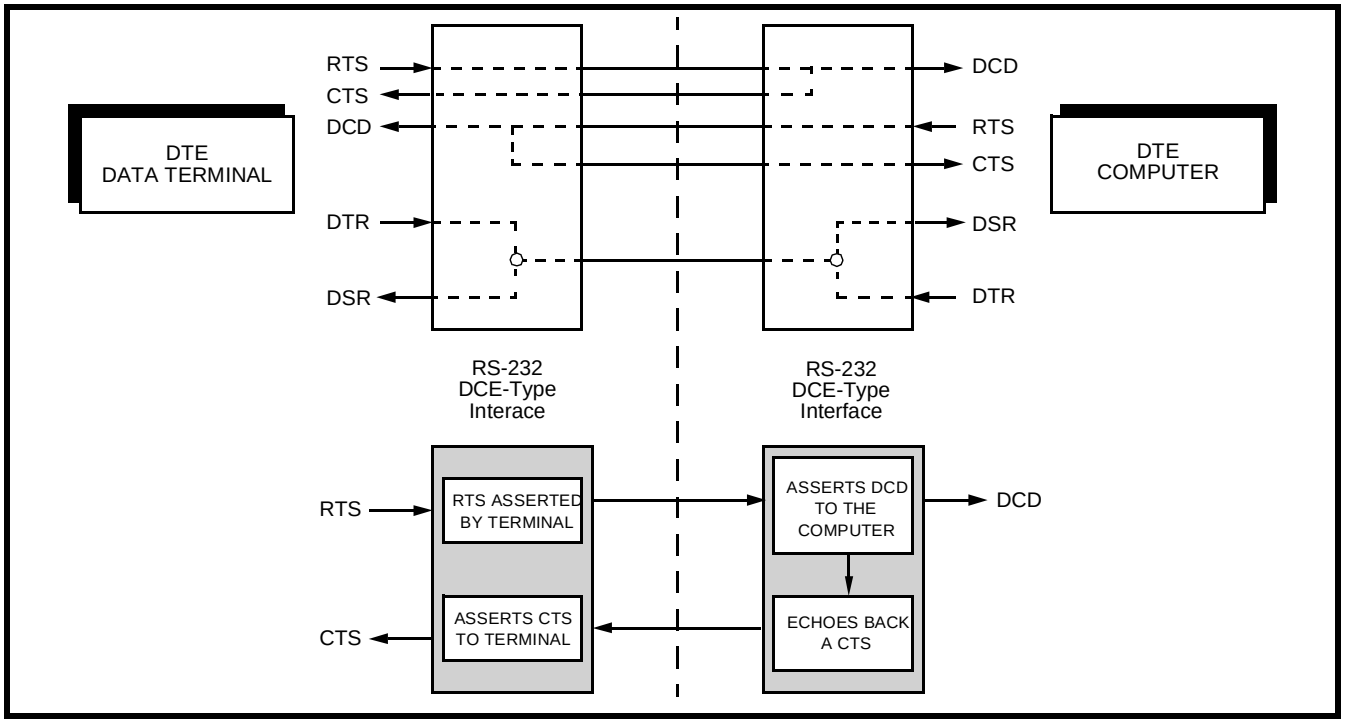

Figure 6 - RS-232 Signal Exchange in a DTE-DTE Type Connection

The block diagram in Figure 7(a), illustrates a technique in which the control information is transmitted and received on the same channel pair as the data. This submultiplexing of the control information with the data can be done most effectively using a single chip microcontroller. In the second configuration, shown in Figure 7(b), the control signals are transmitted and received on one ST-BUS channel pair while the encoded data is transmitted and received on another pair. Both of these configurations are discussed in more detail in subsequent sections.

The third configuration has been presented merely to show how the combination of the techniques described above could be used to submultiplex the control signals on to the data channel and yet still provide the main switching controller with a direct indication of a request for service originating from the peripheral. In this case the PBX directly monitors the level on the DTR line of the interface. The turning ON of DTR by the terminal indicates to the main controller in the PBX that the specific interface has requested service. This is analogous to a phone going off-hook. The Status Point Interface circuit depicted in the block diagram in Figure 7(c) is similiar to the circuit used for the second configuration. In this case, however, the Status Point Interface circuit is shared among a number of interfaces (up to a maximum of eight). Thus an ST-BUS channel originating from this circuit would be made up of individual bits reflecting the status of the DTR lines of eight interfaces.

The choice of configuration depends on the architecture of the switching system and the call processing procedures to be utilized. Generally, if ST-BUS components are used to transmit the information to a remote location, the in-band signalling scheme illustrated in Figure 7(a) would utilize the bandwidth more efficiently. However, if the interface is contained within a single piece of equipment, it may be just as cost effective to use an extra channel to transmit the control information.

In all of the following discussions and illustrations, only the circuit at the local end is described. It is assumed that the remote interface has an identical circuit. The remote interface may be located within the same switching sytem or it may be in some remote location as discussed previously.

4.1 Submultiplexing RS-232 Control Signals with the Data

When the Data Codec is operating in the NRZ format, it accepts data signals at the primary input - D_X1. The secondary input pin, D_X2, can accept a digital signal which is encoded only when there is no activity on the D_X1 pin. The format of this secondary signal is slightly different from that of the primary data signal. Each pulse on the D_X2 input is encoded as a single transition. The minimum time between two consecutive pulses is limited to 125µs. The maximum data rate attainable using the D_X2 input is, therefore, 8 kbps. At the remote end, the regenerated signal appears as a square wave on output pin D_R2. Each transition on this output corresponds to a pulse input at the transmitting end. The general format of the input/output signal is illustrated in Figure 8. This secondary input/output capability can be used to transmit and receive RS-232 control signals when there is no activity on the D_X1 pin (i.e. the data signal is either steady MARK or steady SPACE).

The circuit diagram illustrated in Figure 9 shows an example of how the Data Codec in conjunction with a microcontroller can be used to implement the interface shown in Figure 7(a).

The microcontroller is used to monitor the status of the RS-232 control lines. Based on the logic levels detected, it generates a seven bit word. This word along with the synchronization bit is input at the D_X2 pin of the Codec using a pulse width modulation scheme. The microcontroller also accepts the signal output by the codec at the D_R2 pin and decodes the seven bit word transmitted by the remote end. The information contained within the received word is used to update the status of the control lines. The format of the signal generated and received by the microcontroller is illustrated in Figure 10. In the signal generated by the microcontroller, the time duration between consecutive pulses reflects the logic value of the specific bit in the transmitted word. A logic ‘1’ is represented by a duration of 3.33ms between consecutive pulses while a logic ‘0’ is represented by a duration of 1.67ms. The first bit in each transmitted word is referred to as the synchronization bit. It is represented by a period of 6.7ms between consecutive pulses. Transmission of the signal to the Data Codec is started after a certain minimum time (125µs) has elapsed since the last transition on the TXD data line. Transmission is ceased when any activity is detected on the data line. The microcontroller does not in any way manipulate the data originating from the TXD pin. This signal is transmitted transparently from one end to the other.

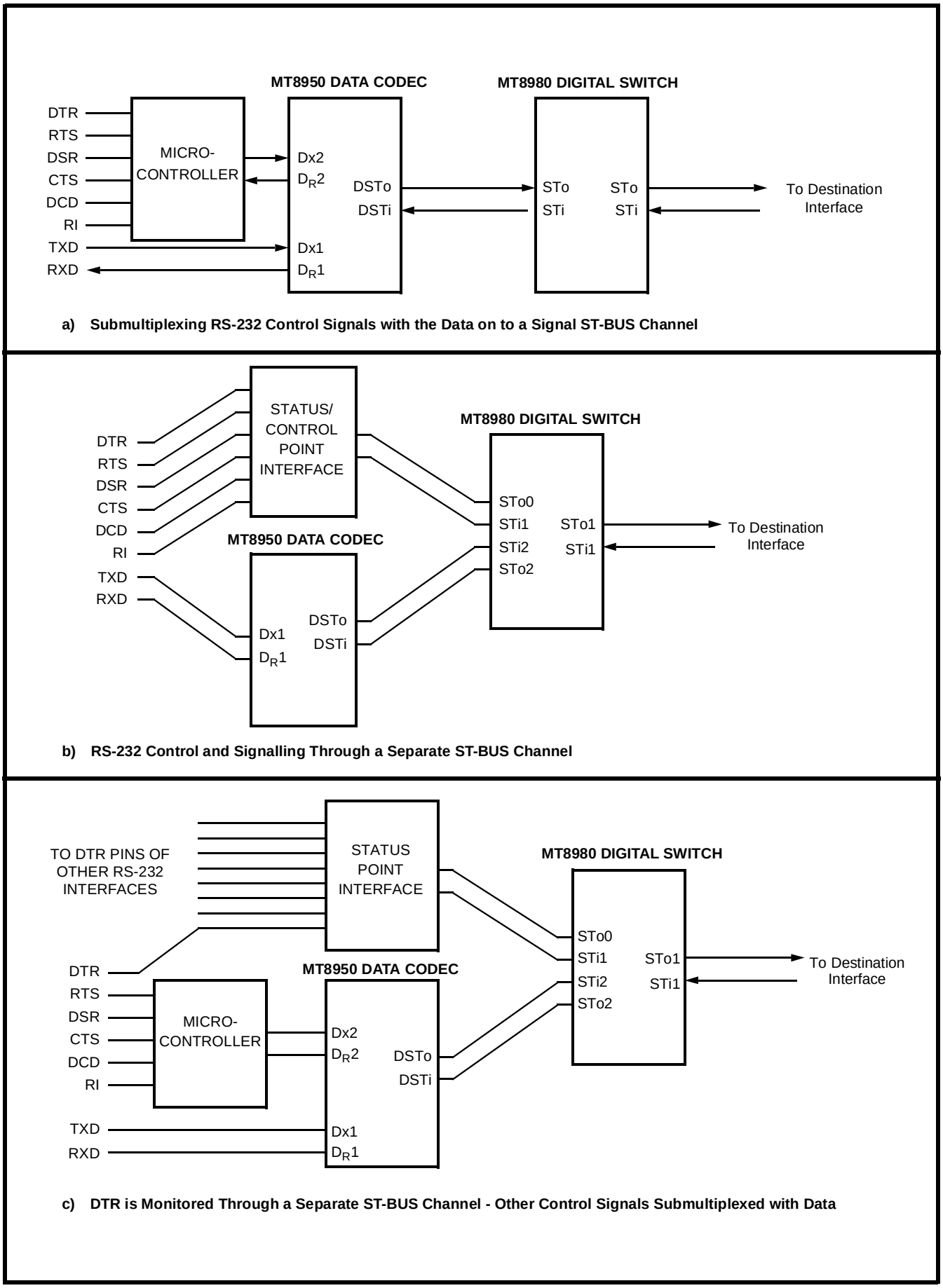

Figure 7 - Implementation of a Complete RS-232 Interface to the ST-BUS Using the Data Codec

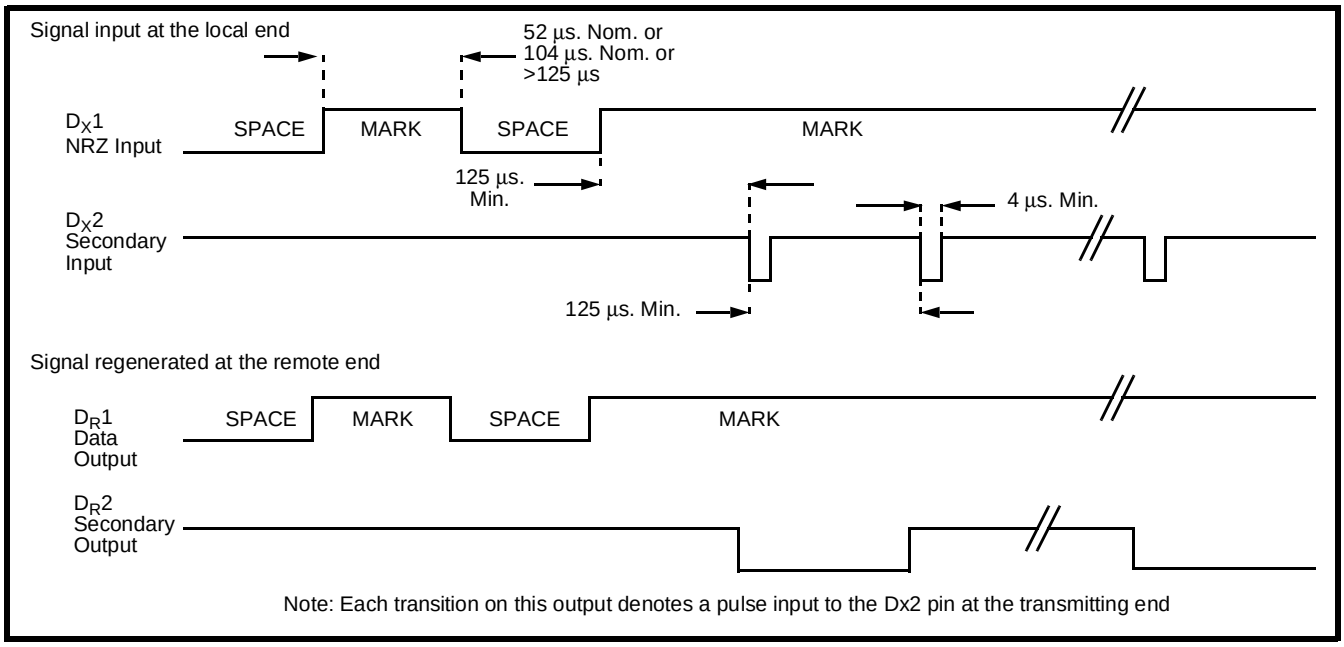

Figure 8 - Example Input/Output Waveform in the NRZ Format (DF = LOW)

This particular circuit configuration can also be used to interpret the 8 bit word generated by the Data Codec in the Local Carrier Mode. The Data Codec can function in any one of eight modes. The required mode of operation is selected through the internal Control Register. When the device is programmed to operate in the Local Carrier mode, it generates an 8 bit word which is output at the D_R2 pin by modulating the secondary clock (SCLK) signal. In the NRZ format, the modulated signal appears as a square wave. The time interval between consecutive transitions in this signal specifies a binary value. A logic '0' is represented by one SCLK time period between the transitions. A logic '1' is represented by two SCLK periods. The first bit in this eight bit word is referred to as the Synchronization bit. It is made up of four clock periods. The logic levels of bits 0, 2, 3 and 6 are fixed as zeros. Bits 1, 4 and 5 can be set to '1' or '0' through the Control Register as illustrated in Figure

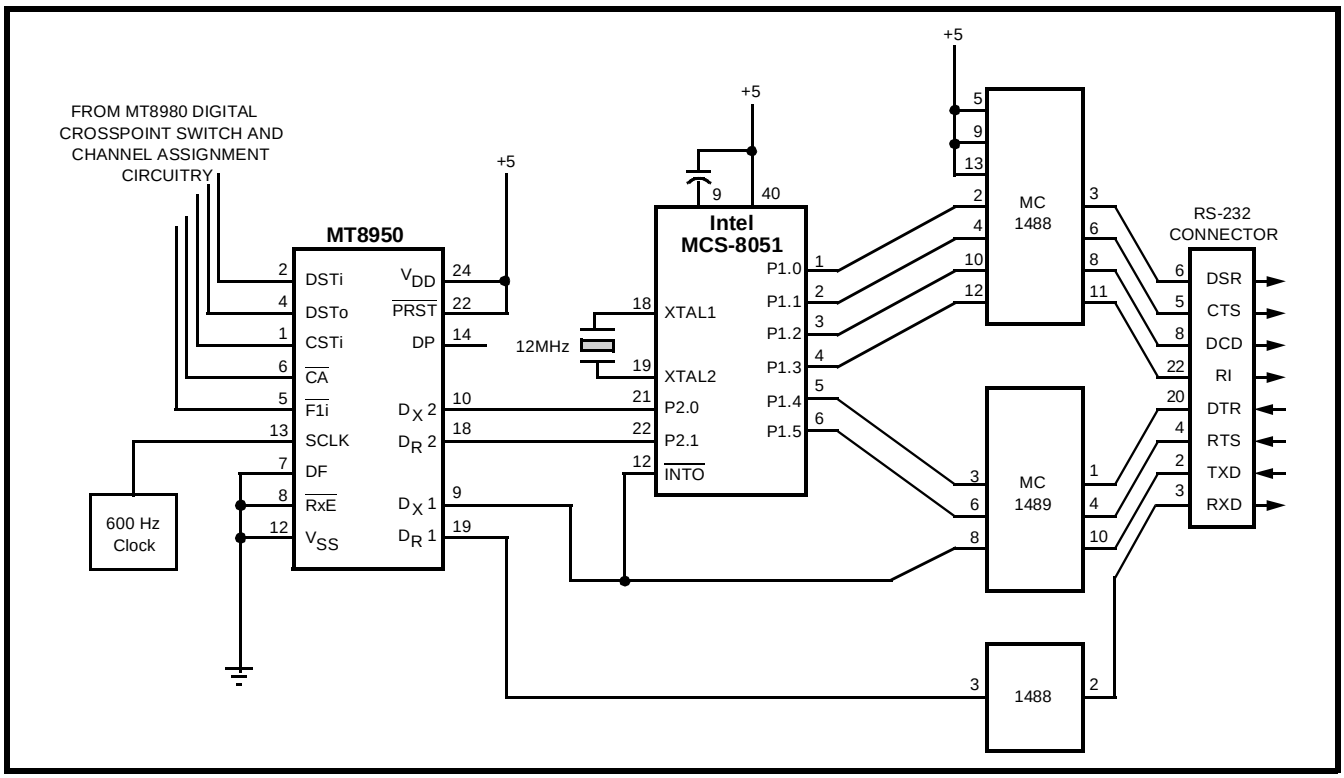

Figure 9 - Submultiplexing RS-232 Signals with Data on to a Single ST-Bus Channel

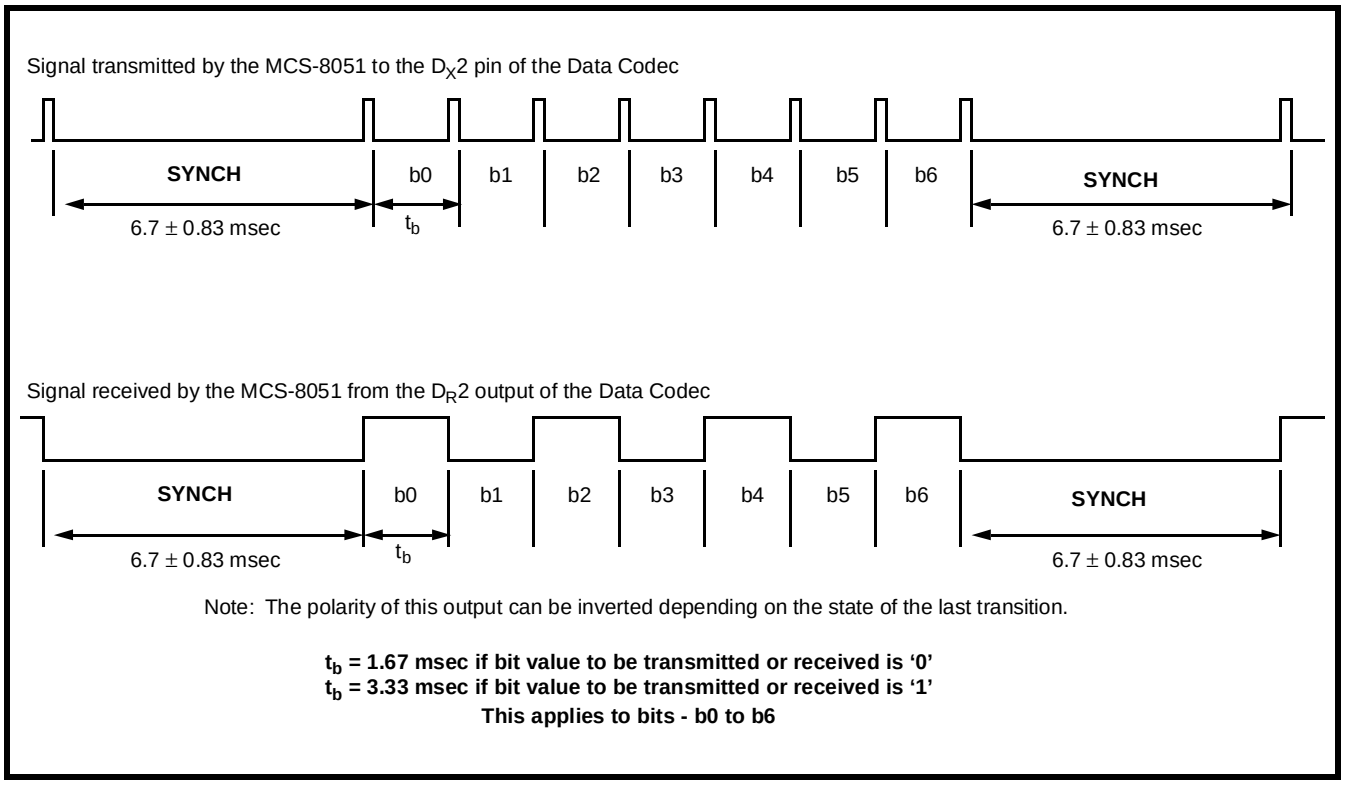

Figure 10 - Signal Received and Generated by the MCS-8051 Microcontroller

11. If a 600 Hz clock signal is used as the SCLK input, the time period between consecutive transitions will correspond to bit logic levels which can be decoded using the decoding scheme described above for the microcontroller circuit. This feature may be used in setting or resetting specific signals (such as DTR, RI, etc.) in the RS-232 interface through the ST-BUS Control Stream (CSTi) originating from the digital crosspoint switch. The central switching controller can therefore exert direct control over some of these signals. This may be used during initial call setup before the connection between two peripheral interfaces is established (see Section 5).

An example of the bit assignment that may be used for the transmitted control word is presented in Table 4. In the seven bit word generated and transmitted by a DCE type interface, bits 0, 2 and 4 are not used. Bit 6 is always set HIGH. This indicates to the remote receiving end that the transmitting interface is a DCE type interface. Bits 1 and 3 reflect the status of the control lines DTR and RTS respectively. Thus bit 1 will be reset to '0' if DTR is turned ON by the terminal connected to the interface. The RTS signal has a similar effect on bit 3. Bit five in the transmitted word is set or reset in response to a RTS from the remote end. It is reset to '0' if an asserted RTS (bit 3) is decoded from the received word. Note that in all cases negative true logic is used in setting or resetting the specific bit in the control word.

For an interface configured as a DTE, the transmitted control word bits 1, 2, 4 and 5 reflect the value of input control lines DSR, DCD, RI and CTS on the interface. Bits 0 and 3 are not used. To indicate to the remote end that the transmitting interface is of a DTE type, bit 6 is reset to '0'.

The control word received by a DCE-type interface can be decoded in one of two ways. If bit 6 in the received word is '0', then it is assumed to have been transmitted by a DTE type interface. If it is '1', then the remote interface is assumed to be a DCE type. In the former case the four bits in the received word corresponding to DSR, DCD, RI and CTS will be used to control the respective local output signals. For example, if the received word bit corresponding to DSR is '0' (indicating that the DSR has been turned ON by the modem at the remote interface) then the receiving end will turn ON DSR to the terminal. The other bits are handled in a similar manner.

In the latter case where bit 6 in the received word is '1', it will be decoded as follows. If the local DTR line is turned ON, the microcontroller will turn ON DSR to the DTE. A logic '0' in bit 3 indicates that the remote end has asserted its RTS. In response to this, the microcontroller resets bit 5 to '0' in the outgoing control word and also asserts DCD to the terminal equipment. A logic '0' in bit 5 in the received word causes the microcontroller to turn ON CTS to the terminal.

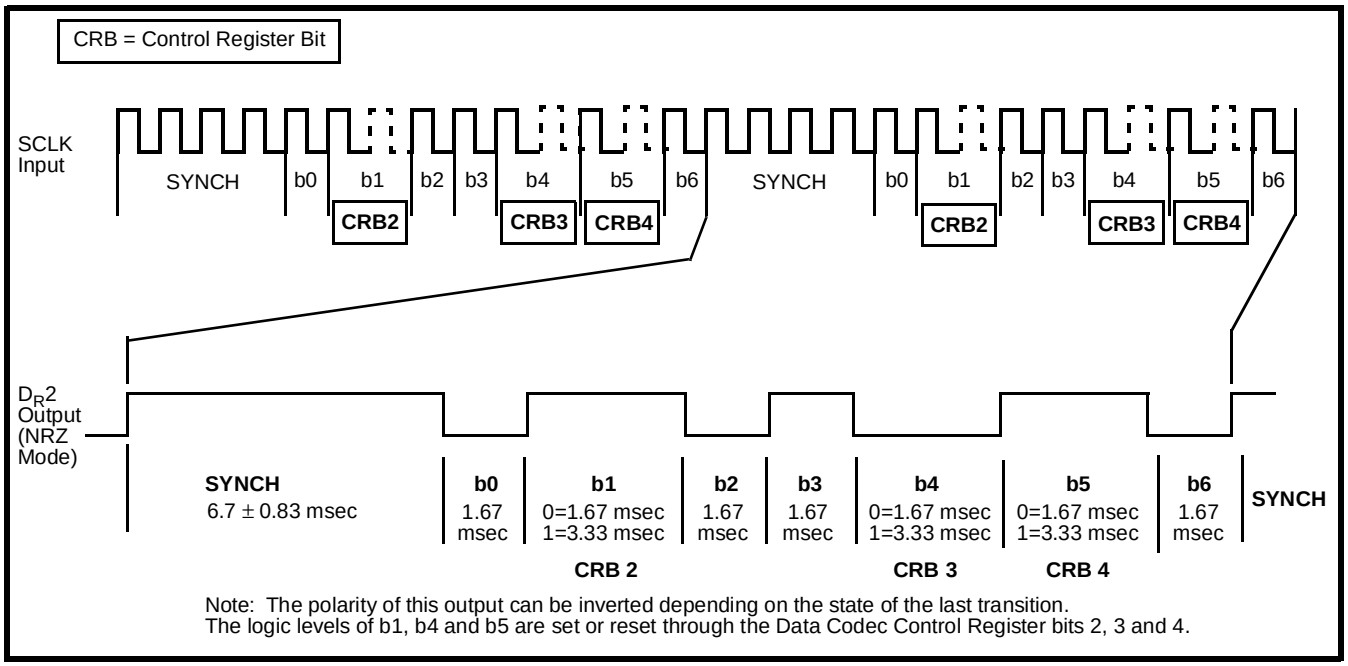

Figure 11 - Signal Output by Data Codec at D_R2 in the Local Carrier Mode for a 600Hz Input to SCLK

The control word received by a DTE-type interface can orginate only from a DCE-type interface. Bit 6 in this word will, therefore, always be '1'. The two bits corresponding to DTR and RTS will be used to control the respective local output signals.

It should be noted that, the signal carrying the control information is transmitted only when there is no activity on the D_X1 input. Thus, there can be a delay between when a change of state occurs in one of the control lines and when that information is made available to the remote end.

As indicated earlier, this particular circuit configuration can also be used to interpret the seven bit word transmitted by the Data Codec in the Local Carrier mode. Depending on the manner in which the received Control Word is decoded by the microcontroller, it is possible to set or reset the level on any of the RS-232 control lines. If, for example, the Data Codec is part of a DCE-type interface, it would be possible to control the DSR, CTS and RI lines by setting or resetting the appropriate bit in the Data Codec Control Register. As mentioned earlier, this register can be written to by the MT8980 via the ST-BUS interface. Note that bit 6 in the Control Word generated by the Codec is fixed as '0'. Thus the microcontroller will decode this word as if it had originated from a DTE type interface. The bit assignment shown in Table 4 could be then used.

The Local Carrier mode could be used during call setup to assert the appropriate control signals for the initial interactive communication between the

Transmitted Control Word			
Bit	**Generated By DCE-TYPE Interface**	**Generated By DTE-TYPE Interface**	**Generated By MT8950 in Local Carrier Mode**
0	X	X	0
1	DATA TERMINAL READY	DATA SET READY	DATA SET READY
2	X	DATA CARRIER DETECT	0
3	REQUEST TO SEND	X	0
4	X	RING INDICATOR	RING INDICATOR
5	ECHO CLEAR TO SEND	CLEAR TO SEND	CLEAR TO SEND
6	1	0	0

Table 4. Control Word Bit Assignment

switching system and the terminal device. For example, if it is necessary to turn ON RI at a called DCE - Type interface, the Data Codec in the interface circuit is put in the Local Carrier mode. Resetting bit 3 in the Data Codec Control Register to '0' will result in a '0' being transmitted to the microcontroller in bit 4 of the control word. The microcontroller will decode this as a signal to assert RI to the interface. Bit 2 in the control word is fixed as '0'. As mentioned earlier, a logic '0' in this bit position will be decoded by the microcontroller as a signal to assert DCD.

4.1.1 Hardware Description

The microcontroller used in the circuit shown in Figure 9 is an Intel MCS-8051. It has two on board timer/counters and three latched input/output ports. The instruction set provides for direct bit addressing and manipulation. Since the Data Codec operates in full duplex, the microcontroller has to generate the output signal and also receive the incoming signal. Thus the microcontroller resources have to be continually shared by the different routines used. It is, therefore, essential to minimize the processing time required for any one subroutine. The boolean processing capabilities of the Intel MCS-8051 greatly aids in this task.

The configuration of the RS-232 control lines depends on whether the interface is to appear as a DCE or DTE to the external world. The circuit shown in Figure 9 is configured as a DCE-type interface. Pins 0 to 3 on the MCS-8051 port 1 are used as outputs. These pins set or reset the levels on the DSR, CTS, DCD and RI lines of the RS-232 connector. Pins 4 and 5 are used as inputs to monitor the status of the DTR and RTS lines. The TXD line is connected to the external interrupt pin. A transition on this line will generate an interrupt. The MC-1489 converts RS-232 levels to TTL compatible levels. The MC-1488 provides the inverse function. Serial communication between the Data Codec and the MCS-8051 is via port 2 pins 0 and 1.

4.1.2 Software Description

The software flowcharts of programs that can be used in this application have been presented in Appendix-1. A more descriptive pseudocode has been included in Appendix-2. Note that the protocol presented has not been fully tested. It has been described here merely as a possible example.

The main routine in the program initializes the microcontroller and configures the input and output pins. The routine reads the status of the input control lines and stores the information in a bit addressable register as the 'Transmit Control Word'. The bits in this control word are assigned different functions depending on whether the interface is a DCE-type or a DTE-type as discussed earlier. One of the two timers is dedicated for timing incoming signals from the Data Codec. The second timer/counter is initially used in conjunction with the external interrupt pin ($\overline{INT0}$) to detect inactivity on the TXD line for a predefined period of time. Upon detecting this inactivity, the program calls a subroutine (OUT_WORD) to generate the outgoing pulse stream. The logic levels of the various bits in the pulse stream are dictated by the contents of the Control Word stored in the bit addressable register. The second timer is reconfigured to time the interval between outgoing consecutive pulses. The timer is loaded with different values depending on the logic level of the specific bit to be transmitted. For example, if the value of the bit is '1', the timer is programmed to expire in 3.33ms. A logic '0' is represented by a time duration of 1.67ms. Every seven bit word is preceded by a SYNC bit which is characterized by 6.7ms between consecutive pulses. The timer is checked every time the routine is executed. Upon expiry of the timer, a 6µs pulse is generated on the output pin. Pulse generation is ceased when activity on the TXD line is detected.

The subroutine IN_WORD is called by the main routine continually, irrespective of whether an outgoing word has to be transmitted or not. This routine measures the time interval between consecutive transitions in the signal received from the Data Codec's D_R2 output . It first looks for the synchronization bit (i.e., a time interval of 6.7ms). Upon detecting this bit, the subsequent transitions are timed to obtain the seven bit received word. Small errors in timing values may be introduced due to the fact that other routines have to be executed before the timer is checked. The magnitude of the error depends on the worst case execution time of the other routines. This has to be taken into account before time decision levels for the bit values are defined. In the algorithm described, a time interval greater than 5.9ms and less than 10ms is assumed to be the SYNC signal. Similarily a time interval greater than 2.9ms but less than 5.9ms is decoded as a logic '1'. A logic '0' is decoded if the time interval between consecutive pulses is less than 2.9ms. All of these decision points reflect worst case execution times not exceeding 400µs. Thus the appropriate timers in each of the routines are accessed at least once in each 400µs time interval.

Two consecutive identical words have to be received for the microcontroller to begin decoding. The DEC_WORD subroutine is called to decode the received word and update the status of the local

control lines. The decoding is done in accordance with the different requirements for DTE and DCE type interfaces, as discussed earlier.

4.2 Using a Separate ST-BUS Channel for RS-232 Control Information

If bandwidth considerations are not too important, a separate distinct channel may be used to convey the control information from one interface to the other. The 'control' channel will then have to be switched along with the 'Data' channel. An example of a circuit which may be used to generate and receive the control signals in the ST-BUS channel format is illustrated in Figure 12. This is the circuit implementation of the Scan and Control Point interface depicted in Figure 7(b). The parallel signals from the RS-232 connector are first serialized using an eight bit parallel in - serial out shift register. This register is clocked with the 2.048 MHz system clock used for the Data Codec and other ST-BUS peripheral devices. The output of the shift register is passed through a tristate gate which is enabled during a specific channel timeslot (in the same way as the ST-BUS interface on the Data Codec is enabled). It is tristated during the remainder of the frame. Thus the serialized control information is contained within one channel timeslot. This channel can be switched using the MT8980 to any remote interface. At the remote interface the serialized eight bit word is converted into parallel format using a serial in - parallel out shift register. The status of the eight lines is latched into an eight bit latch once in each ST-BUS frame when the specified channel has been received. The outputs of the latch are connected, after level conversion, to the appropriate RS-232 lines. Note that in the circuit illustrated in Figure 12, only four of the eight outputs of the latch (74HCT574) and only two of the eight inputs of the parallel to serial converter have been used. The remaining input/ouput lines can be used in a similar manner if more control signals are to be transmittted or received (up to a maximum of eight in each direction) .

The circuit, as described, can be used for transparently switching the control information from one interface to the other. If system intervention is required, the status of the control signals can be monitored by the microprocessor connected to the MT8980 digital switch which is responsible for switching the specific control channel. In order to do this, the microprocessor reads the MT8980's data memory and interprets the eight bits contained within. The required signal status can be transmitted to the remote interface using the messaging mode of the MT8980. The central processor interfaced to the MT8980 could set or reset any particular control line by setting or resetting the appropriate bit in the Connection memory. The contents of the Connection memory would be transmitted to the interface circuit via the ST-BUS as explained earlier.

5. Data Call Set up Techniques

The basic steps involved in setting up a connection between any two peripheral devices through the PBX include the following:

- Detect a request for service and acknowledge
- Receive call destination instructions
- Make the connection
- Break the connection when the service is no longer required

In setting up a normal telephone connection, the PBX senses a request for service by detecting an Off-Hook condition and acknowledges the request with a dial tone. The caller provides the destination address by dialling the appropriate number. In order to interpet the incoming dial pulses or the DTMF tones generated by the telephone set, the controller assigns a special card to the interface originating the call. Once all the digits have been received this card communicates the information to the controller which makes the connection. The connection may be broken once an On-Hook condition is detected.

Essentially the same steps outlined above have to be followed when a connection involving two digital devices has to be made. The signalling methods used, however, can vary depending on the overall archictecture of the switching system. Two different approaches are discussed below. The first approach describes an in-band signalling technique while the second outlines an example of an out-of-band signalling protocol.

5.1 In-Band Signalling

If both the signalling information and the actual data is transmitted over the same path, then the signalization is referred to as being in-band.

In order to detect when a peripheral device connected to an interface wants service, the PBX has to monitor the interface for specific signals. The PBX controller may, for example, be programmed to interpret an asserted DTR as a request for service. It is also possible to use circuitry at the interface which detects energy on the data line. The presence of energy on the line would indicate that the peripheral wants service. The PBX then has to provide some kind of acknowledgement signal to the peripheral and receive call destination instructions. In order to do this, the switch controller may connect

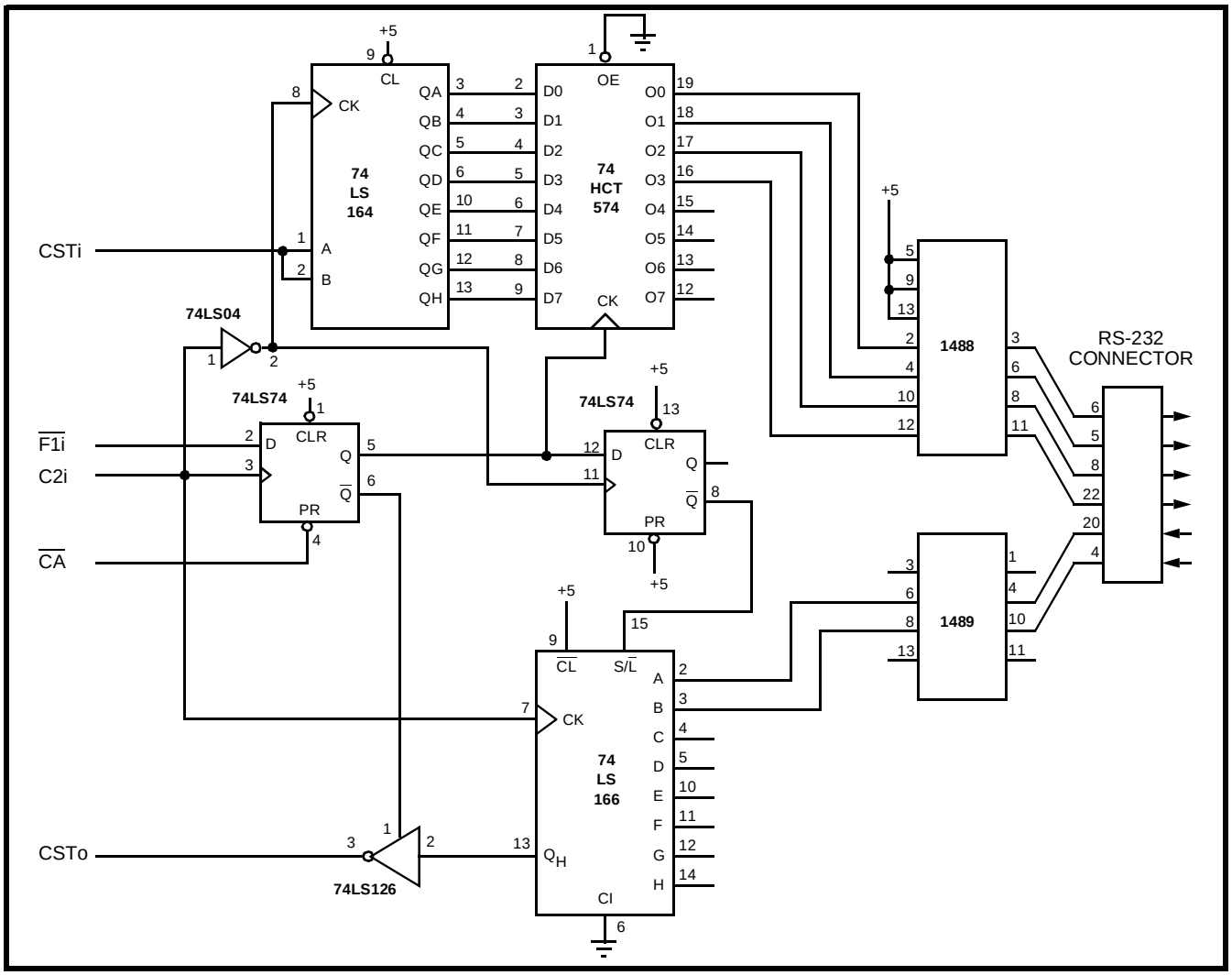

Fig. 12 - Scan and Control Point Interface Circuit for Transmitting and Receiving RS-232 Control Signals. The RS-232 Connector Appears as a DCE to the External Equipment.

a special card, the Data Call Setup (DCS) card, to the calling peripheral interface. A microprocessor on this card communicates with the peripheral device and obtains instructions for making the connection. The complexity of the DCS card effectively determines the capabilities of the overall system.

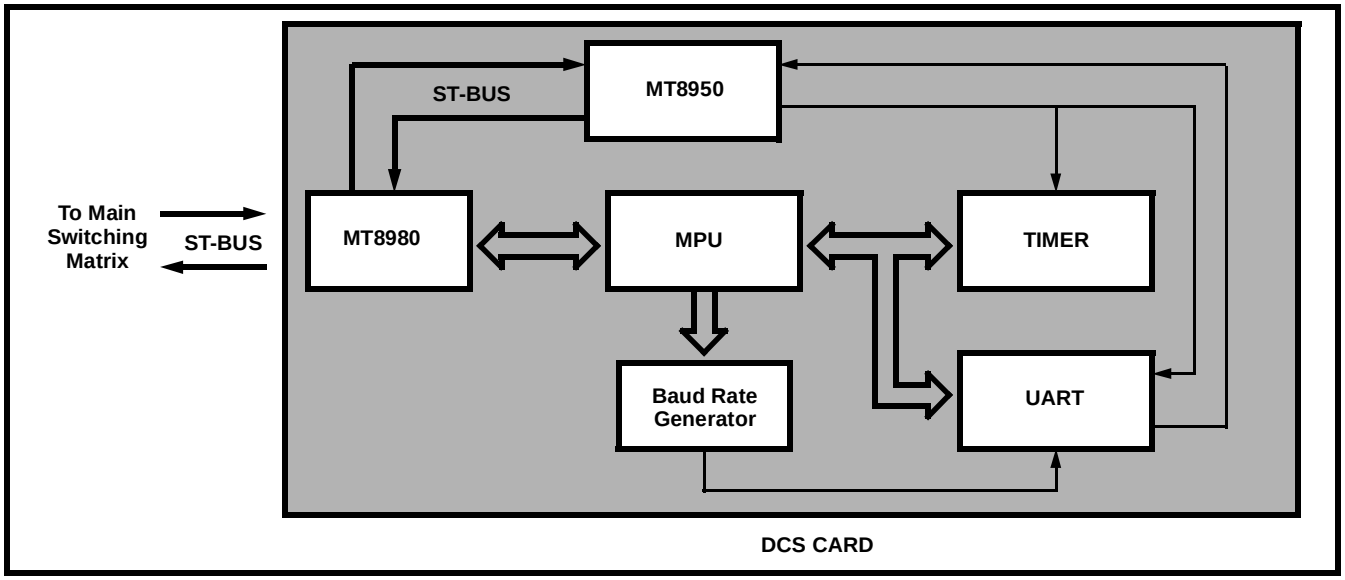

Figure 13 - Data Call Setup Card Block Diagram

Some functional features that would be required in a typical DCS card are illustrated in the block diagram in Figure 13. A Data Codec is used on this card to convert the encoded information transmitted by the codec, on the peripheral card, into a format which the microprocessor can accept. The microprocessor on the card in conjunction with a timer determines the baud rate of the incoming signal. It then sets its own UART baud rate to match that of the terminal so that interactive communication is possible. The MT8980 on the DCS card is used for interprocessor communication. This allows the microprocessor on the DCS card to communicate with the main switching matrix control processor (which makes or breaks the connection) through the ST-BUS. The MT8980 could be shared by a number of other Data Codecs on the card.

In a typical session, the terminal equipment asserts DTR. This is detected by the main processor which assigns a DCS card to the calling data interface. The Data Codec at the calling interface is put in its Local Carrier Mode to turn ON the appropriate RS-232 Control Signals. It is necessary to turn ON DSR, CTS and DCD at the interface to initiate interactive communication. The DCS card waits for a predefined sequence to be transmitted by the DTE to determine the baud rate of the calling party. This predefined protocol may, for example, require that the user press the RETURN key a number of times until the terminal receives a valid prompt from the PBX. The caller is prompted to provide the destination address which could be letters or numbers specifying a certain system or port. This information is made available to the system main processor which then sets up the connection. If it is necessary to turn ON the RI at the called interface, the Data Codec in that circuit is put in the Local Carrier mode and the appropriate bit in its Control Register is reset to '0'. When the peripheral device responds by turning ON its DTR, the two devices are connected. If the port is not available the main switching controller gives the appropriate message to the DCS card indicating that the port is busy. The DCS card in turn passes the message onto the calling terminal.

Note that although it is necessary for the DCS card in the PBX to determine the baud rate of the digital signal for the initial interactive communication, the Data Codec on the line card does not have to be programmed to accept any specific data rate. Thus the line cards can be made relatively dumb and inexpensive. The intelligence and the complexity of the more expensive DCS board is shared among a number of peripheral units and therefore it does not have to be duplicated for every line.

5.2 Out of Band Signalling

The signalling information is said to be out-of-band, if it is passed over a separate and distinct channel or path from the data.

In many modem products currently on the market, turning RTS line ON and OFF by the terminal results in the telephone line circuit to which the device is connected, being pulsed in a manner similar to that used by rotary dial phones. The same principle can be used for call set up with the Data Codec RS-232 interface. In this case the calling equipment has to have software capable of pulsing the RTS line and conveying the information to the PBX using a pulse coding scheme. A typical session would be as follows. The main processor in the PBX detects an Off-Hook condition (DTR asserted) and assigns a pulse decoder card to the particular interface. DSR is turned ON by the interface at this time. This indicates to the DTE that the PBX is ready to receive the dialed pulses. The DTE then pulses the RTS line to indicate to the decoder card the destination address. If the connection can be made, the main processor asserts RI to the called destination. When the remote side answers, the two interfaces are connected to each other. The main processor then asserts CTS and DCD to both sides. This scheme can be more easily implemented in systems where the main switching processor has direct access to the input and output control signals as would be the case if the circuit described in Section 4.2 were to be used.

The principle advantage of using this scheme is that the decoding of the dialed pulses from the caller and the subsequent identification of the destination line, requires circuitry which is similiar to the type used for a normal voice telephone call setup. No baud rate detection is required. Thus, any data rate acceptable to the Data Codec can be used. The disadvantage is that the terminal equipment has to have the appropriate software to provide the pulsing for the dialed digits. This imposes a severe restriction on the types of terminal equipment that can be used. If the complexity is to be contained within a stand alone unit such as a Data Set, then the cost of this unit would correspondingly be high.

NOTES:

Appendix - 1 Software Flowcharts

Software Flow Charts of Programs For Submultiplexing RS-232 Control Signals with the Data

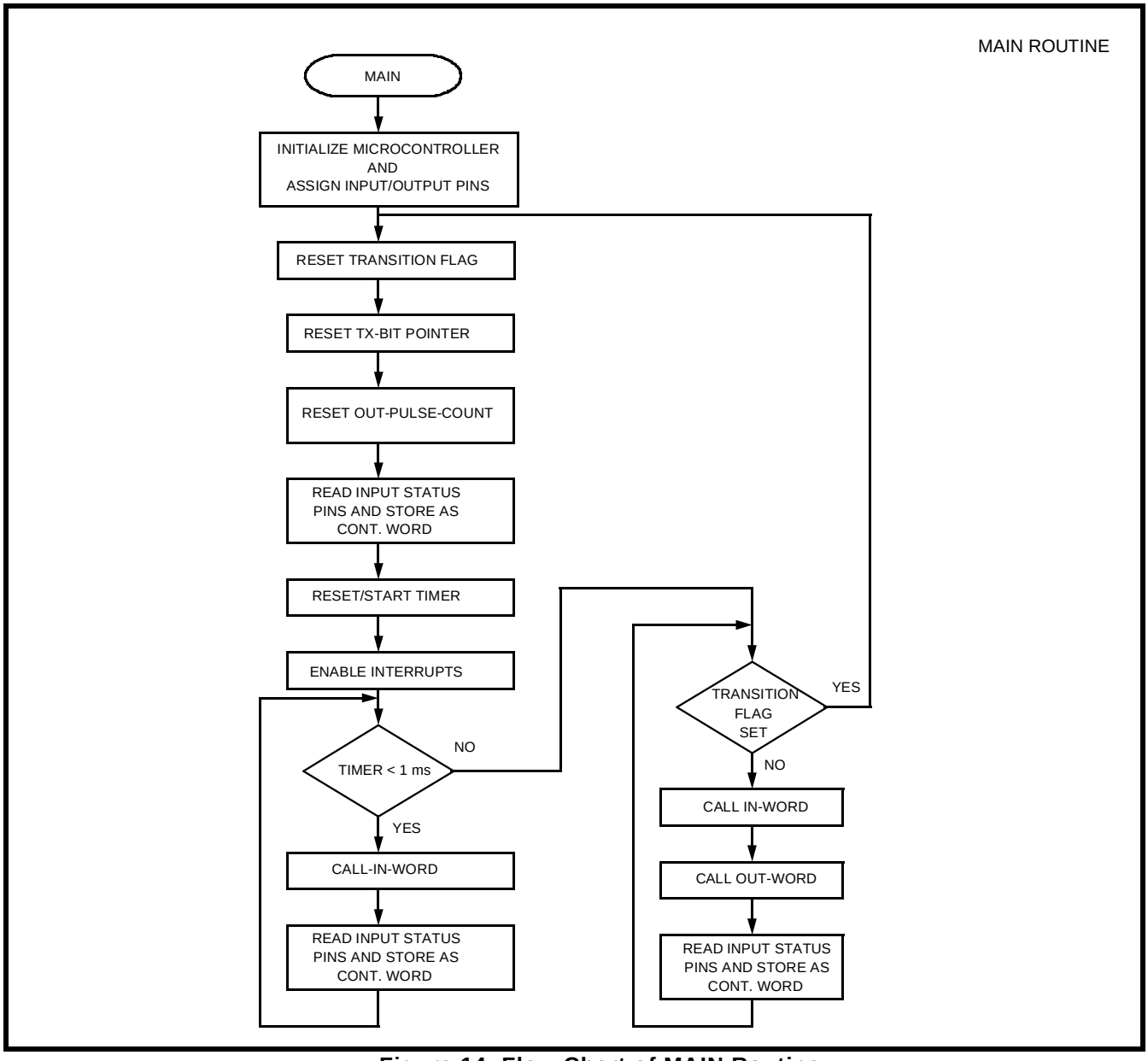

Figure 14- Flow Chart of MAIN Routine

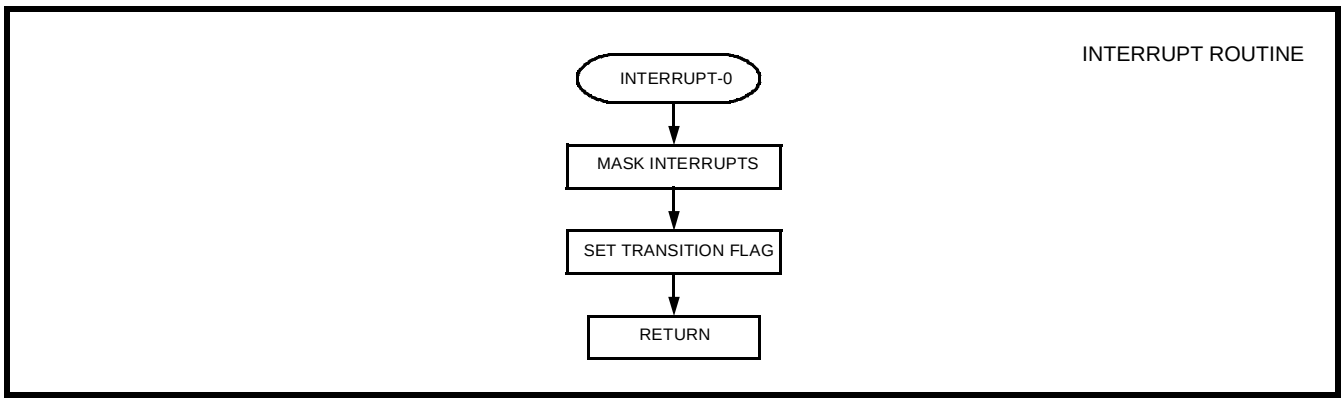

Figure 15 - Flow Chart of INTERRUPT Routine

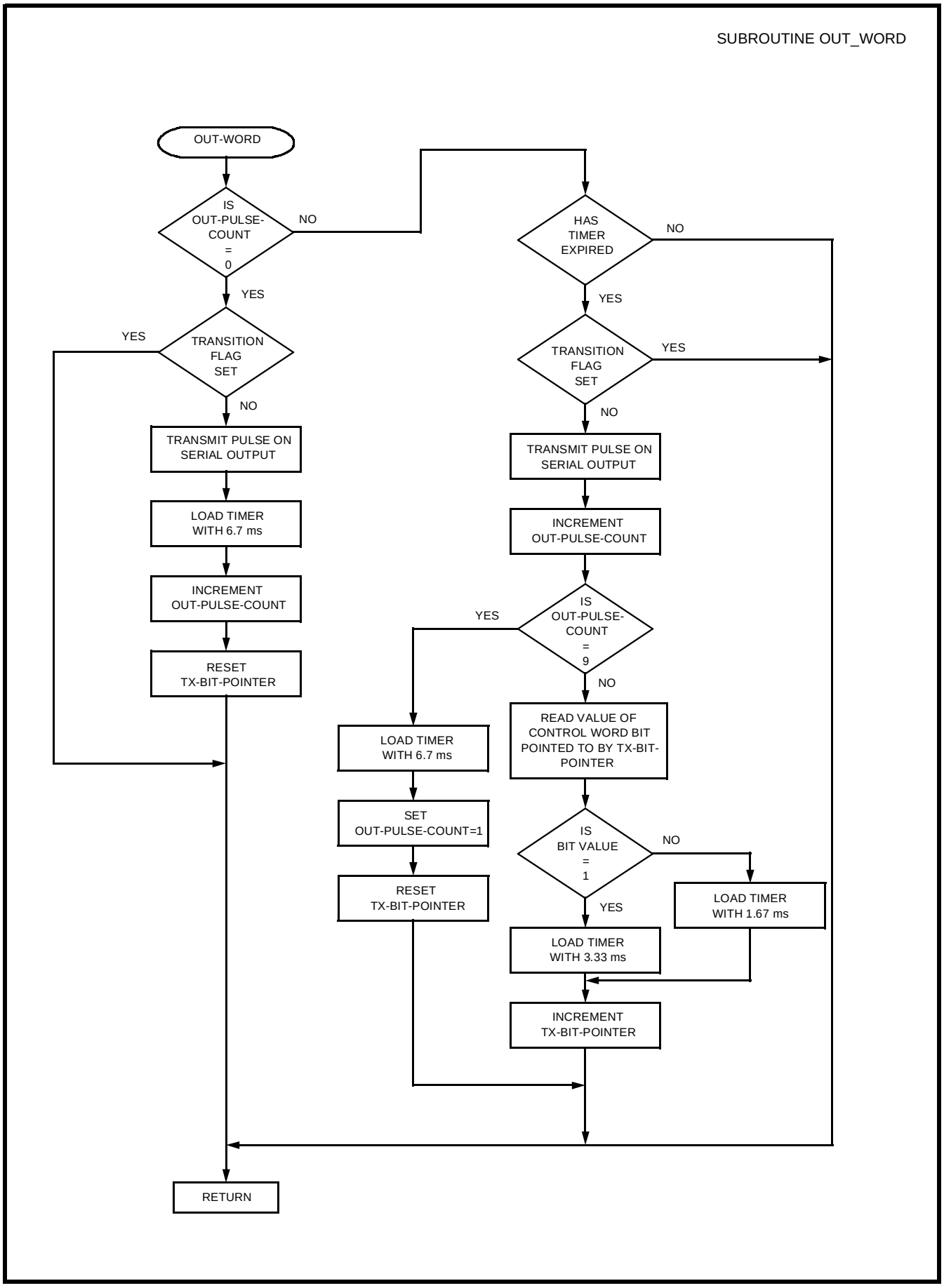

Figure 16 - Flow Chart of Subroutine to Generate Serial Output Word

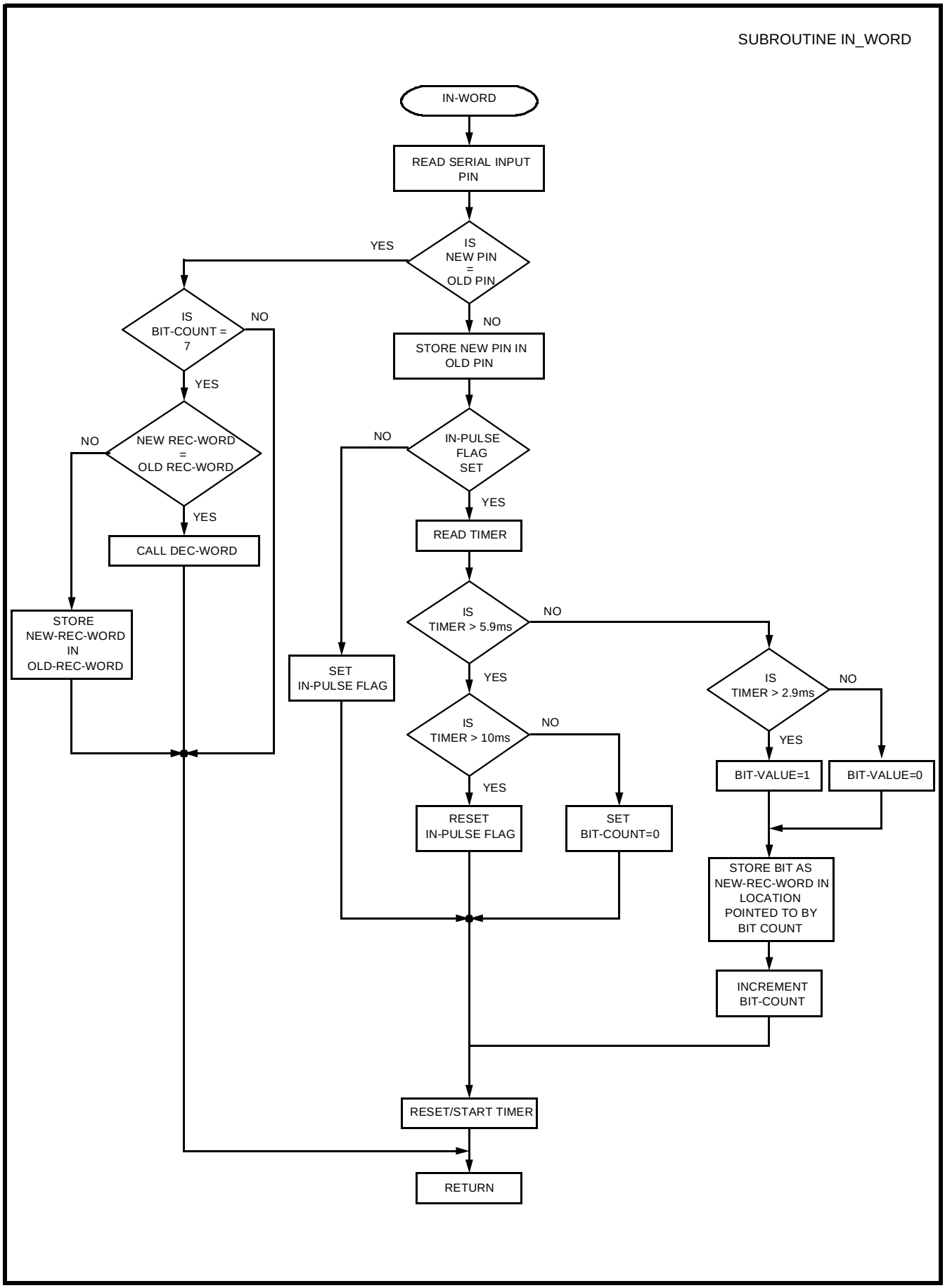

Figure 17 - Flow Chart of Subroutine to Read Incoming Serial Data

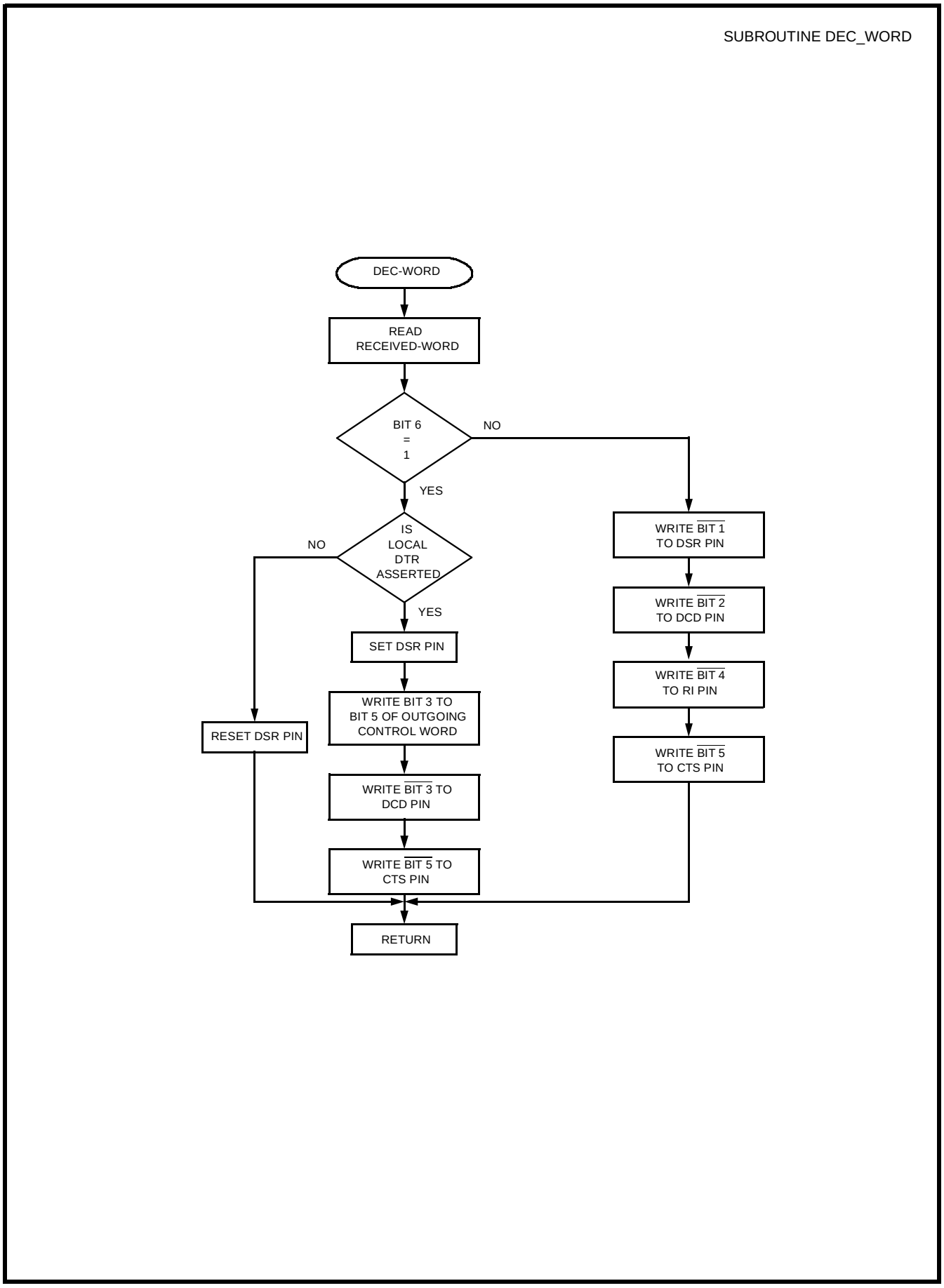

Figure 18 - Flow chart of Subroutine to Decode Received Control Word.

Appendix - 2 Software Documentation

The Software documentation described below has not been tested. It is presented here as an example of the type of protocol that is required to implement the functionality discussed in the application note.

SOFTWARE DOCUMENTATION FOR SUBMULTIPLEXING RS-232 CONTROL SIGNALS WITH DATA ONTO A SINGLE ST-BUS CHANNEL - USING AN MCS-8051 MICROCONTROLLER

MAIN PROGRAM FOR DCE-TYPE INTERFACE

```
MAIN
CONFIGURE INPUT/OUTPUT PINS              /*  INPUTS: 1.4, 1.5, 2.1. OUTPUTS: 1.0, 1.1, 1.2, 1.3, 2.0  */
RESET POINTERS AND FLAGS
EXTERNAL INTERRUPTS-EDGE TRIGGERED
INITIALIZE TIMER MODES - BOTH TIMERS RUN IN MODE - 1

DO WHILE (FOREVER)

   RESET TRANSITION FLAG        /*  WHEN SET, INDICATES THAT TRANSITIONS HAVE BEEN DETECTED ON THE TXD LINE  */
   RESET TX_BIT_POINTER   /*  POINTER INDICATING NUMBER OF BITS IN THE CONTROL WORD THAT HAVE BEEN TRANSMITTED
                              */
   RESET OUT_PULSE_COUNT     /*  REPRESENTS THE NUMBER OF OUTPUT PULSES THAT HAVE BEEN GENERATED ON THE
                                 SERIAL OUTPUT PIN  */
   READ INPUT STATUS PIN AND STORE AS CONTROL WORD              /*  P1.4 AND P1.5  */
   LOAD TIMER-0 TO EXPIRE IN 1 ms -- START TIMER
   ENABLE EXTERNAL INTERRUPTS

   DO WHILE (TIMER-0 < 1 ms)
        CALL IN_WORD
        READ INPUT STATUS PINS AND STORE AS CONTROL WORD
   ENDWHILE

   DO WHILE (TRANSITION FLAG NOT SET)
        CALL IN_WORD
        CALL OUT_WORD
        READ INPUT STATUS PINS AND STORE AS CONTROL WORD
   ENDWHILE

END WHILE.
```

INTERRUPT VECTOR /* THIS ROUTINE IS EXECUTED WHEN AN INTERRUPT IS RECEIVED BY THE MICROCONTROLLER */

```
INTERRUPT - 0                        /* INTERRUPT GENERATED BY INTO PIN */
MASK EXTERNAL INTERRUPT
SET TRANSITION FLAG
RETURN FROM INTERRUPT
```

SUBROUTINE OUT_WORD /* THIS ROUTINE GENERATES THE SERIAL SIGNAL WHICH IS OUTPUT AT PIN 2.0 OF THE MICROCONTROLLER */

```
OUT_WORD
IF (OUT_PULSE_COUNT = 0) THEN
DO:
        IF (TRANSITION FLAG NOT SET) THEN
        DO:
                TRANSMIT 6 MICROSECOND PULSE ON SERIAL OUTPUT
                LOAD TIMER WITH 6.7 ms
                INCREMENT OUT_PULSE_COUNT
                RESET TX_BIT_POINTER
        ENDIF
ELSE
        IF (TIMER-0 HAS EXPIRED) THEN
        DO:
                IF (TRANSITION FLAG NOT SET) THEN
                DO:
                    TRANSMIT 6 MICROSECOND PULSE ON SERIAL OUTPUT
                    INCREMENT OUT_PULSE_COUNT
                    IF (OUTPUT_PULSE_COUNT = 9) THEN
                    DO:
                        LOAD TIMER WITH 6.7 ms
                        SET OUT_PULSE_COUNT = 1
                        RESET TX_BIT_POINTER
                    ELSE
                        READ TRANSMIT CONTROL WORD BIT (ADDRESS POINTED TO BY TX_BIT_POINTER)
                        IF (BIT VALUE = 1) THEN
                        DO:
                                LOAD TIMER-0 TO EXPIRE IN 3.33 ms -- START TIMER
                        ELSE
                                LOAD TIMER-0 TO EXPIRE IN 1.67 ms -- START TIMER
                        ENDIF
                        INCREMENT TX_BIT_POINTER
                    ENDIF
                ENDIF
        ENDIF
ENDIF
RETURN
```

```
SUBROUTINE IN_WORD        /* THIS ROUTINE READS INCOMING DATA FROM THE  INPUT PIN P2.1 OF THE MICROCONTROLLER */

IN_WORD
READ SERIAL PIN              /*P2.1 */
IF (NEW_PIN_VALUE = OLD_PIN_VALUE) THEN                    /*  CHECK FOR TRANSITIONS ON D_R2  */
DO:
    IF (RECEIVED_BIT_COUNT = 7) THEN
    DO:
        IF (NEW RECEIVED WORD = OLD RECEIVED WORD) THEN
        DO:
            CALL DEC_WORD
        ELSE
            STORE NEW_RECEIVED_WORD IN OLD_RECEIVED_WORD
        ENDIF
    ENDIF
ELSE
    STORE NEW_ PIN_VALUE  IN OLD_PIN_VALUE LOCATION
    IF (IN_PULSE FLAG SET) THEN
    DO:
        READ TIMER-1
        IF (TIMER-1 > 5.9 ms) THEN
        DO:
            IF (TIMER-1 > 10 ms) THEN
            DO:
                RESET IN_PULSE FLAG
            ELSE
                SET RECEIVED_BIT_COUNT = 0
            ENDIF
        ELSE
            IF (TIMER-1 > 2.9 ms) THEN
            DO:
                RECEIVED BIT VALUE = 1
            ELSE
                RECEIVED BIT VALUE = 0
            ENDIF
            STORE BIT VALUE AS  NEW RECEIVED_WORD IN BIT POSITION INDICATED  BY RECEIVED_BIT_COUNT
            INCREMENT RECEIVED_BIT_COUNT
        ENDIF
    ELSE
        SET IN_PULSE FLAG
    ENDIF
    RESET TIMER -1 = 0  -- START TIMER
ENDIF

RETURN
```

```
SUBROUTINE DEC_WORD  /* THIS ROUTINE DECODES THE SEVEN BIT RECEIVED CONTROL WORD */


DEC_WORD
READ NEW RECEIVED WORD
IF (BIT 6 IN WORD = 1) THEN
DO:
        IF (LOCAL DTR IS SET) THEN
        DO:
                SET DSR                         /*PIN 1.0 */
                WRITE BIT 3 TO BIT 5 OF OUTGOING CONTROL WORD
                COMPLEMENT BIT 3
                WRITE BIT 3 TO DCD              /* PIN 1.2 */
                COMPLEMENT BIT 5
                WRITE BIT 5 TO CTS              /* PIN 1.1 */
        ELSE
                RESET DSR                       /* PIN 1.0 */
        ENDIF
ELSE
                COMPLEMENT BIT 1
                WRITE BIT 1 TO DSR              /* PIN 1.0 */
                COMPLEMENT BIT 2
                WRITE BIT 2 TO DCD              /* PIN 1.2 */
                COMPLEMENT BIT 4
                WRITE BIT 4 TO RI               /* PIN 1.3 */
                COMPLEMENT BIT 5
                WRITE BIT 5 TO CTS              /* PIN 1.1 */
ENDIF
RETURN.
```