MICROCOMPUTER DATA COMMUNICATION INTERFACING USING THE RS-449 STANDARD

By

NARAYAN MAKARAM

Bachelor of Engineering

Bangalore University

Bangalore, India

1979

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the Degree of
MASTER OF SCIENCE
December, 1984

MICROCOMPUTER DATA COMMUNICATION INTERFACING

USING THE RS-449 STANDARD

Thesis Approved:

M. J. Folk
Thesis Adviser

D. E. Hedrick

Louis G. Johnson

Norman N. Durham
Dean of the Graduate College

ACKNOWLEDGEMENTS

I wish to express my sincere appreciation to my major adviser, Dr. Micheal Folk, for his excellent guidance, encouragement, and incredible patience throughout this study. I would like to thank Dr. L. G. Johnson for the helpful discussions during the circuit design and analysis of results. I would also like to thank Dr. G. E. Hedrick for being helpful and considerate while I was his teaching assistant.

I am deeply indebted to Dr. J. R. Phillips, for his constant inspiration and invaluable guidance throughout my graduate studies and thesis work. I gratefully acknowledge the support of Time Management Software, Inc., for providing me the microcomputer faclities and hardware required for this project. My thanks are due to Mr. Mike Veldman, for being helpful in procuring the I.C.s, and for his assistance during the implentation stage.

I would like to express my gratitude to Dr. D.D. Fisher, Chairman of the department of Computing and Information Sciences, for extending me financial assistance, during my graduate studies.

I greatly cherish the friendship of Omana, Sudha, Ninoo, and Suresh, who have made my stay in Stillwater a memorable one. Finally, I am grateful to my parents and brothers, for their endless love and affection, and their continued encouragement. They have made all my achievements possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

The merging of computers and communications has had a profound influence on the way computers are organized. In early computer systems, users brought their problems to a central facility called a "computer center." This type of a model of a single computer serving all the organizations needs, is rapidly being replaced by one in which a large number of separate but interconnected computers do the job. These systems are called Computer Networks.

Two computers are said to be interconnected if they are capable of exchanging information. The connection may be via a copper wire, optic fibres, radio waves, microwaves, lasers, and even earth satellites. In all these cases digital data is to be communicated between two computers of a network. Hence "Data Communications" has become a vital entity in the organization of present day computer systems.

According to John E. McNamara [13], data communications is defined as - "The interchange of data messages from one point to another over communications channels."

Virtually all computers marketed today include some provision for data communications. For medium and large scale systems, they generally take the form of front end processes with supporting software. For microcomputers, internal provisions are included in the form of handwired adaptors (programmable I/O devices). However, harware is not enough. The logical relationships involved in a data communications system can become complex due to multiple independent data streams, varied line control procedures, varied terminals, varied applications and timing considerations. To manage these various factors, software of appreciable size and sophistication must be developed.

Transmission Links

Nodes of a computer network may be connected through terrestrial, satellite or radio transmission links [1].

Terrestrial Links - These are ground wires that connect network nodes to terminals, hosts or other network nodes. There are two types of terrestrial links.

1. Switched or dialed connection, which uses public telephone lines for data communications.
2. Leased or dedicated link, which employs private lines which are permanently connected.

There is yet another terrestrial communications link called Dataphone Digital Service (DDS). Here one can

transmit signals in digital form, without converting them to analog form.

Satellite Links - Satellites operate in geostationary orbits around the earth, thereby providing a means of communicating signals or messages between earth locations. Compared to terrestrial links, satellite channels may provide transmissions for long distances and large capacities (about 1.5 Mega bits per second, Mbps).

Radio Links - Ground radio links may be used to connect several users together for a large computer network. This network is a collection of microprocessor controlled digital radios. These units are called packet radio units and act as store-and-forward switches. The radio channel operates at 100 or 400 kilobits per second (Kbps).

Optical Fiber - Optical fiber transmission is emerging as a viable technology that is currently being evaluated and implemented. An important advantage of fiber optics is its wide bandwidth. For example, a laser-driven fiber link has the potential of data transmissions at speeds up to 10^{14} bits per second (bps).

Transmission Modes

In this section we characterize and briefly discuss various modes in which data is transferred between microcomputers (network nodes).

Parallel and Serial Data Transfer

Microcomputer systems almost always transfer data in parallel mode, within the processor cabinet. The size of the data bus determines the number of bits transferred in parallel by the microprocessor. External data transfers usually involve peripheral devices that are not housed in the same cabinet or are not in close proximity with the CPU and memory. Such transfers may occur over either parallel or serial interfaces.

Let us compare the serial and parallel data transfer modes relative to the following criteria [13].

Distance - The distance of parallel data transfer is usually less than 100 feet. Serial data transmissions quite commonly traverse distances from a few feet to thousands of miles.

Speed - Parallel data paths associated with typical microcomputer devices support data rates in the range from zero up to 10 Mbps. Serial interface data rates associated with typical microcomputer devices range from zero to about 1 Mbps. For both serial and parallel data transfers, the speed that can be supported adequately is inversely proportional to the distance.

Signal Level - Parallel transfers usually employ Transistor-Transistor-Logic (TTL) levels. Thus, logic 1's and 0's are electrically represented on each parallel wire as +/-5V. Although TTL signalling is used for

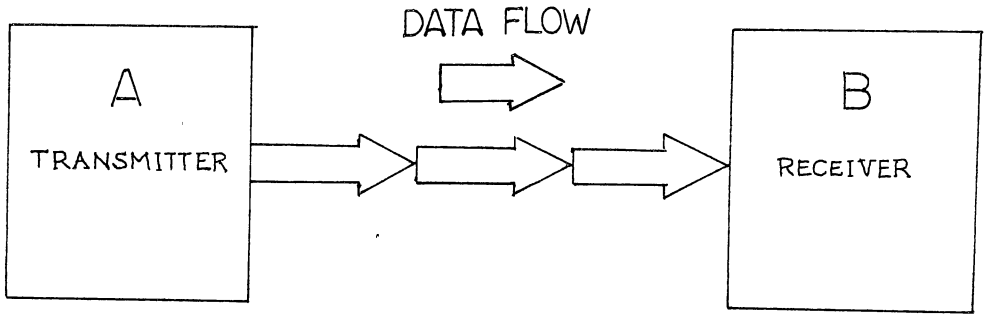
serial interfaces, it is more common to use higher operating voltage levels.

Signal Loss and Amplification - The amount of signal loss is directly proportional to the length of the cable on which it is transmitted. To compensate for signal losses, either high-power transmitters or signal amplifiers can be used to preserve signal character. In addition, another problem encountered in parallel transfers is skewing. Skewing occurs when the differences in individual line propagation delays cause significant discrepancies in the timing of the individual data lines to be sampled by receivers. As the distance increases for parallel transfers, the skewing problem gets worse.

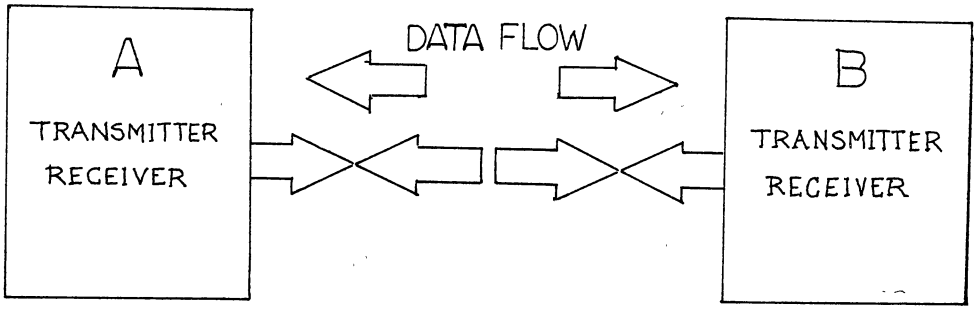
Cost - For distances over 50 feet the cost of running parallel data lines becomes exorbitant. Furthermore, the extra logic required to transform parallel data to serial data for transmission over a single data line, and then to reconstitute the parallel byte at the destination, is less expensive than the hardware required to have an effective parallel communication link over long distances.

Direction of Data Transfer

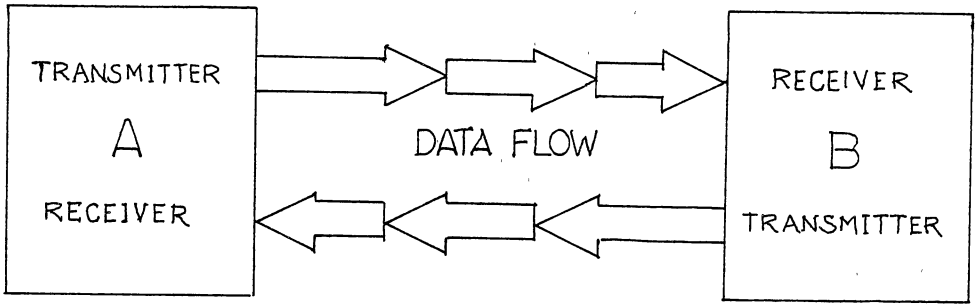
Transmission links may be designed to permit movement of data in either or both directions. Figure 1.1



(a) SIMPLEX - Data flow in only one direction



(b) HALF DUPLEX - Data flow in both directions ALTERNATELY.



(c) FULL DUPLEX - Data flow in both directions SIMULTANEUSLY.

Figure 1.1. Directions of Data Transfer.

illustrates the different types of configurations [15].

SIMPLEX - This data path will support data transfers in only one direction, namely from A to B at all times.

HALF DUPLEX - This will support alternate data transfers between A and B. In particular, device A can transmit to device B, and device B can transmit to device A, but these transmissions must not occur simultaneously.

FULL DUPLEX - This configuration supports simultaneous data transmissions in both directions. In particular, device A can transmit to device B, and at the same time receive data from device B.

Serial Transmission Protocols

Protocols to transmit data on communications links are classified as either asynchronous or synchronous.

Asynchronous Protocols [14] - Here the receiver synchronizes to incoming data on a character-by-character basis. Re-synchronization is coordinated by the use of start bits and stop bits that are collectively referred to as framing bits. The duration of a start bit (pulse) is the same as one information bit. The stop signal is usually 1, 1 1/2, or 2 times the duration of an information bit, as shown in Figure 1.2. Its main advantage is that the character is self-contained with all the information, and the two ends of the link need not be synchronized. Its main disadvantage is the overhead for start and stop signals, and that of false

recognition of start/stop signals due to noise on the channel.

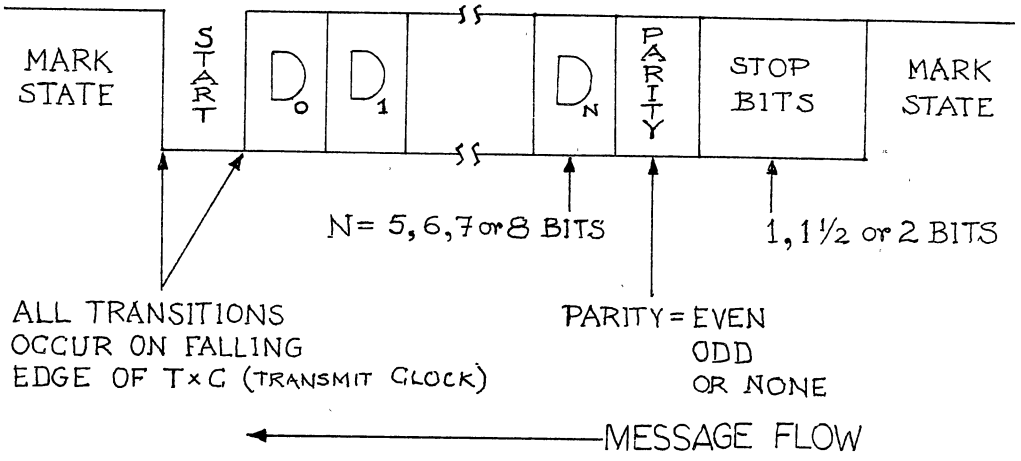


Figure 1.2. Asynchronous Serial Character Transmission

The asynchronous mode of operation is required for devices with keyboards that have no message buffers and operate on one character at a time.

Synchronous Protocol - Synchronous bit-serial transmission uses synchronizing information on groups of characters, such as a message, instead of on each character. Here a separate clock signal is used to provide the necessary co-ordination between source and destination in order to accomplish synchronized data transfer. Data is transmitted serially by providing control characters at the beginning and end of the

message. The advantage here is that the two ends of the link are synchronized, thereby avoiding the overhead for handling synchronizing information for each character. Two types of synchronous transmission protocols are described in Chapter IV, namely: binary synchronous transmission protocol (BSC) and synchronous data link control protocol (SDLC). Figure 1.3 depicts a typical information frame in the SDLC protocol.

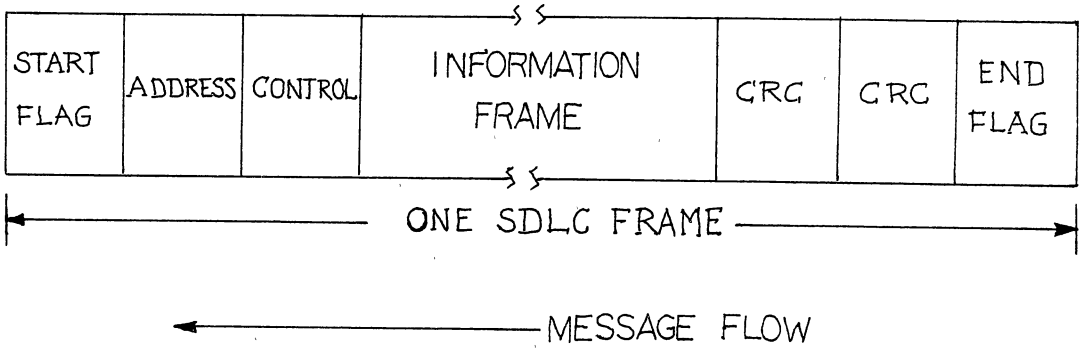


Figure 1.3. Synchronous Serial Communications Protocol.

Project Motivation

The serial communications interface being used in most of the computer systems for the past two decades has been the RS-232-C. But, there are some limitations in a RS-232-C with regards to speed (20 K bits/sec) and distance (50 feet). The interface also uses unbalanced transmitters and receivers, (discussed in chapter III)

which are susceptable to noise. Hence the EIA developed a series of new interface standards - RS-449, RS-422-A and RS-423-A, which meet the advancing state-of-the-art technological changes.

From the standpoint of performance, the new standards are a distinct improvement, which will encourage manufacturers to begin producing equipment with the enhanced interface. However, because of cost and convenience, and because of general resistance of the user community to change, it will be some time before the upgraded standards find widespread acceptance. Hence, there is a need for interfacing new equipment designed for RS-449 standards, with equipment designed for RS-232 standards.

The objective of this project is to do the following.

1. Develop data communications software using the SDLC protocol between two microcomputers connected by a RS-232-C interface.

2. Design a hardware interface circuit (adapter) to convert RS-232-C port (25 pins) to a RS-449 port (37 pins).

3. Implement the same SDLC protocol over the new RS-449 interface channel. Then, compare the performance of the same software (with various transmission speeds) over the new interface.

CHAPTER II

NETWORK ARCHITECTURE

Modern computer networks are designed in a highly structured way. As the number of real time applications of computer networks has grown, the International Standards Organization (ISO) has summarized (on the basis of proprietary networks- ARPANET) both known and anticipated network requirements. This has resulted in a formal specification of a seven layer model [16]. The object of this model is to organize related groups that can be standardized independent of each other. This enables each layer to be defined with minimum reference to any other layer. Thus nodes of a network can be "constructed" with these layers, and communication between nodes is ensured by network protocols.

Layer 'n' on one machine carries on a conversation with layer 'n' on the other machine. The rules and conventions used in this conversation are collectively known as the layer 'n' protocol, as illustrated in Figure 2.1. The entities comprising the corresponding layers on different machines are called Peer Processes. In other words, it is the peer process that communicates using the protocol.

In reality, data is transferred only through the lowest layer called the Physical Layer. Hence at the lowest layer there exists a physical communications link with the other machine, as opposed to a virtual communications used by the higher layers. In Figure 2.1 the virtual communications is shown by dotted lines, and physical communications by a solid line. Between each pair of adjacent layers there is an interface. The interface defines which primitive operations and services the lower layer offers to the upper one.

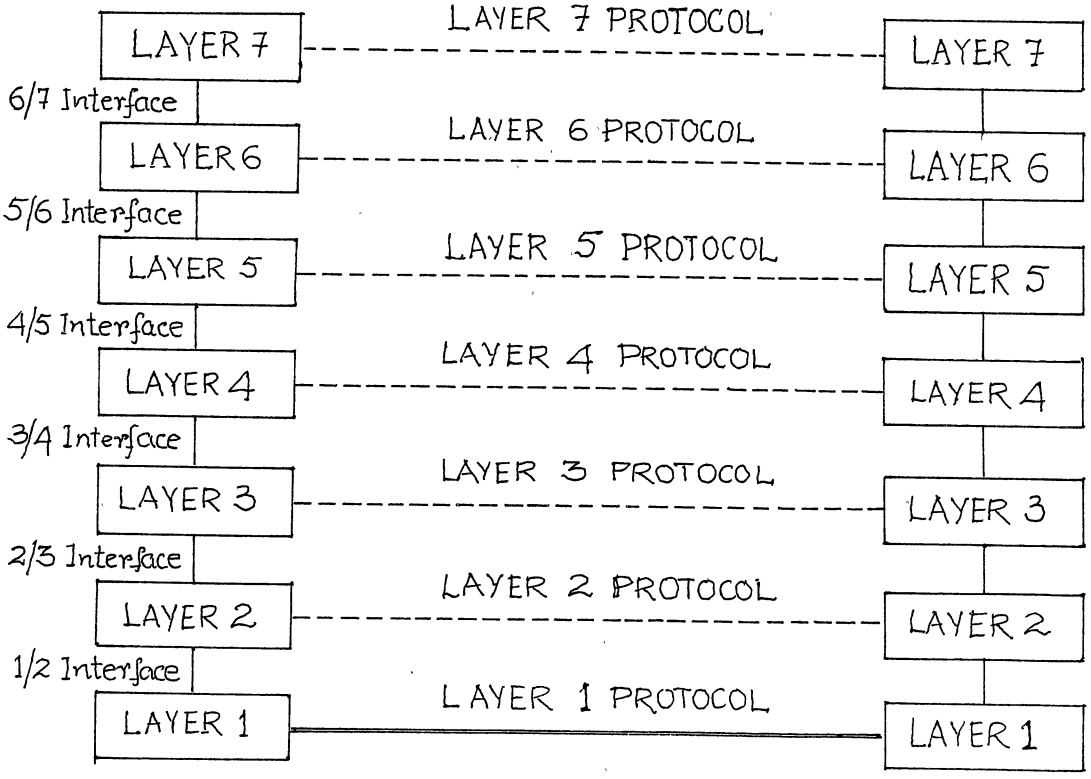


Figure 2.1. Seven-Layered Network Model

ISO Reference Model

An ISO reference model [17] has been accepted as the first step towards standardization of the various layers. This model is primarily applicable to large computer networks, but some of its concepts are useful as background information for our discussion.

There are seven functional layers in the ISO reference model. They are - Physical, Data-Link, Network, Transport, Presentation and Application layers as depicted in Figure 2.2. The layers applicable to general-purpose microcomputer systems are the bottom-most two layers - Physical & Data-Link layers. The top-most five layers are generally not used in simple microcomputer networks. Hence, they will be discussed briefly in order to introduce some basic concepts in Upper-Level network protocols.

Physical and Data-Link Layers

The physical layer represents conventions that are applicable to physical media that interconnect the two correspondents. In particular, this layer must provide the mechanical, electrical, and functional mechanisms for the initial establishment, maintenance and ultimate release of a physical channel linking the source and destination. The typical questions dealt with here are - how many volts should be used to represent a 1 and how many for a 0; how many microseconds a bit occupies;

whether transmission may proceed simultaneously in both directions; how the initial connection is established and how it is torn down when both sides are finished; how many pins the network connector has and what each pin is used for.

We will discuss several physical - layer communication protocols - EIA RS-232-C, EIA RS-449, RS-423-A and RS-422-A. All these protocols are covered in chapter III.

The data link layer takes a raw transmission facility and transforms it into a line that appears free of transmission errors to the network layer. It accomplishes this task by breaking-up the input data into data frames, transmitting the frames sequentially, and processing the acknowledgement frames sent back by the receiver. Since layer 1 merely accepts and transmits a stream of bits without any regard to meaning or structure, it is upto the data link layer to create and recognize frame boundaries. This can be done by attaching special bit patterns to the beginning and end of the frame. There are two types of data link control protocols discussed in Chapter IV. They are - 1) Byte oriented protocols, and 2) Bit oriented protocols.

Upper-Level Network Protocols

The network layer controls the switching and routing of information across the network. Thus the

network layer establishes the physical and logical connections required to transfer the data from its source to its destination(s).

The transport layer is responsible for ensuring high-quality network service. It accepts data from the sessions layer, splits it up into smaller units, if need be, and passes them on the other end. Hence it forms an interface between the sessions and network layer.

The sessions Layer is the user's interface into the network. This layer allows the user to establish a connection with a process on another machine. Once the connection has been established, the sessions layer can manage the dialog in an orderly manner, if the user has requested that service. In short the sessions layer takes the bare bones bit for bit communication service offered by the transport layer and adds application-oriented functions to it.

The presentation Layer provides certain transformation services such as text compression and expansion, encryption and decryption, conversion between character codes (e.g. ASCII to EBCDIC). The presentation layer attempts to alleviate problems associated with incompatible terminals, line and screen lengths, scroll versus page mode, and character sets.

The application Layer is the topmost layer. It communicates with the user's application program or process.

The ISO reference model for a node is illustrated in Figure 2.2 [2], with the connection to network medium at the bottom and the user interface at the top. The two layers used in the implementation of this thesis project are the physical layer and the data link layer.

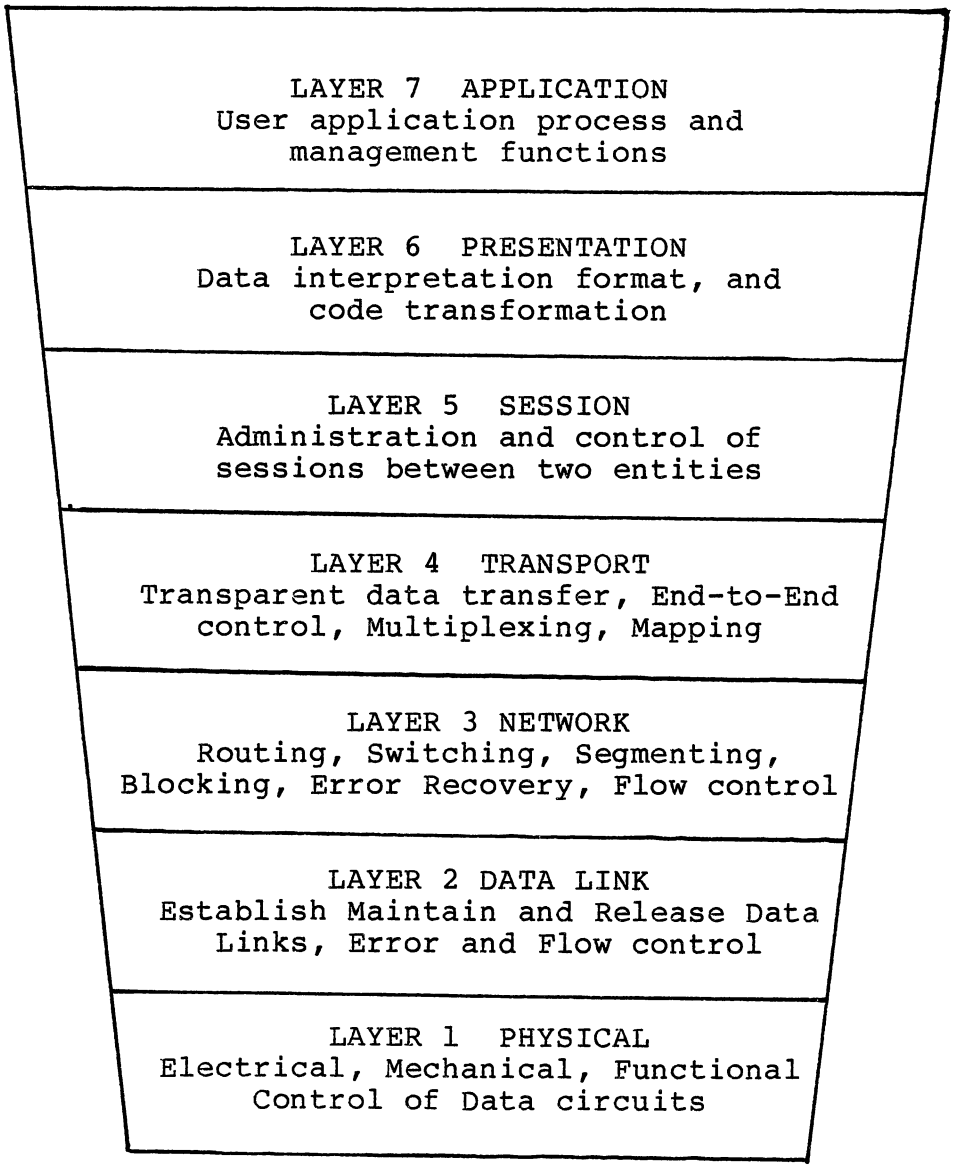


Fig. 2.2. The Seven-Layer ISO Reference Model.

User Interface to Networks

Of all the layers, the application layer is most intimately concerned with the user processes. The term "user" is generally taken to mean any user process or user machine, not necessarily a person using the system directly. The activities associated with the application layer are usually considered to include file transfer control; data base management; remote job entry; electronic mail; and network operating systems.

File Transfer Protocols(FTPs) lay ground rules for transfering files from one host to another. The FTP identifies both the source and destination and locates the files involved. Afetr the initial control phase, the data transfer takes place with associated control between the two servers involved.

Remote Job Entry(RJE) protocols are used to enter jobs or a batch of jobs to a remote computer. ARPANET's NETRJE, developed by UCLA provides this capability.

The computer network we are concerned with, consists of just two microcomputers which are connected by a serial communications channel. A file selected by the user on one microcomputer is transmitted to the other microcomputer using SDLC protocol. Here, we are not using any of the upper level protocols.

CHAPTER III

PHYSICAL LAYER

The physical layer is the most basic protocol level in the heirarchy of data communication protocols. The level covers the physical interface between devices and the rules by which bits are passed from one to another.

The International organization for Standardization (ISO) defines the Physical Layer[3] as follows -

> The physical layer provides mechanical, electrical, functional, and procedural functions in order to establish, maintain and release physical connections between data terminal equipment (DTE), data circuit terminating equipment (DCE) and/or data switching equipment (DSE) (p.433).

Data is transferred between electronic devices via some type of interface that consists of electrical impulses, cable to the devices. The data is commonly represented by changes in current and voltage. To accomplish successful data interchange, devices must follow a physical-layer communications protocol. Such a protocol sets forth standards that unambigously resolve the following issues .

o Mechanical Characteristics - Detailed specifications are given for the connector dimensions,

number of pins and sockets, connector locations, and cable characteristics such as length and numbeer of conductors.

o Electrical Signal Characteristics - The electrical characteristics of data interchange signals and the associated circuitry must be specified. This includes specification of maximum data rates, identification of voltage and current levels that represent signal status conditions, and specification of characteristics of receiver and transmitter circuits.

o Functional Description of Signals - The signals that comprise the interface are usually characterized by function, by whether they are transmitter or receiver, and by their relationship with other signals.

In this chapter we will discuss the physical-layer protocols that are most commonly used in microcomputer based systems. These include several standards that are defined by EIA as RS-232-C and RS-449. The RS-422-A & RS-423-A are subsets of the RS-449 standard.

RS-232-C Interface Standard

The RS-232-C [13] standard addresses communications between data terminal equipment (DTE) and data circuit-terminating equipment (DCE). Essentially, the DTE represents the ultimate source and/or destination of data. Examples of DTEs are printers and data terminals. The DCE facilitates the communication of the data from

its source to its destination. A modem is a device that is a DCE.

The RS-232-C standard was published by EIA in 1969. The letters RS stand for "Recommended Standard." The 232 is an identification number, and the "C" indicates how many revisions the standard has gone through.

Scope

The RS-232-C standard applies to serial binary communications between DCEs and DTEs in which the data rates are in the range from 0 to 20,000 bps. Thus 19.2 K bps is the highest data rate which the RS-232-C supports.

The standard places a limitation of 50 feet upon cable length for error free operation.

Electrical Signal Characteristics

The following is a summary of the RS-232-C electrical signal characteristics.

1. RS-232-C employs negative logic; ON condition is associated with logic 0, while an OFF condition is logic 1.
2. Logic 1 or MARK condition is represented by a voltage between -5 and -15V. The range from +3 to -3V is the transition region for which no signal state is given.

This implies that there is a 2V noise margin in the standard. Thus, a line driver, or signal source, sends a logic 0 by applying a voltage in the range +5V to +15V.

A line receiver, or signal destination, senses a logic 0 by looking for voltages in the range +3 to +15V. The result is that there is allowance for a 2V drop in the signal level between its source and destination as illustrated in Figure 3.1.

NOTE: The departure of RS-232 signals from TTL voltage levels was motivated by the necessity to improve noise immunity and distance capabilities. Noise margin for a TTL interface is 0.4V only.

3. The shunt capacitance of the terminator (destination) side of an RS-232-C circuit must not exceed 2500 pF including the capacitance of the cable.

4. The RS-232-C driver must be able to withstand a short circuit to any other wire in the cable without sustaining damage to itself or associated equipment like the terminal, modem, and I/O port.

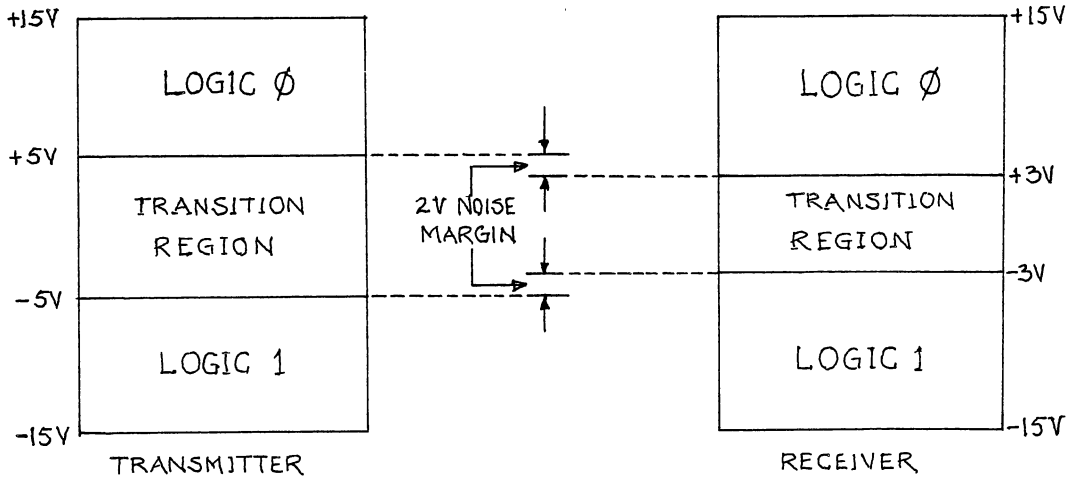


Figure 3.1. RS-232-C Electrical Characteristics.

Interface Mechanical Characteristics

The RS-232-C interface uses a 25 pin connector for connection with DTEs and DCEs. Note that a male connector is associated with DTEs; a female connector is associated with DCEs.

Circuit Functional Characteristics

Interchange circuit functions are typically classified into the following broad categories [15].

Ground or common Return (A)

Data Circuits (B)

Control Circuits (C)

Timing Circuits (D)

Secondary Channel Circuits (S)

Table I groups the RS-232-C circuits by category, showing their two-or-three-letter designation, their direction (to or from DCE), and their cable pin assignment. The function of each of these circuits is described below.

Circuit AA: Protective Ground - This conductor is electrically bonded to the machine or equipment frame. It is optional to connect this circuit.

Circuit AB: Signal Ground or Common Return - This conductor provides the reference point relative to which all other RS-232-C circuits, eccept protective ground, are measured. This is one circuit which is absolutely required, no matter what the cable's application.

TABLE I

RS-232-C INTERCHANGE CIRCUITS BY CATEGORY

Interchange Circuit	Pin Number	Description	From	To
		GROUND		
AA	1	Protective Ground		
AB	7	Signal Ground		
		DATA		
BA	2	Transnitted Data	DTE	DCE
BB	3	Received Data	DCE	DTE
		CONTROL		
CA	4	Request to Send	DTE	DCE
CB	5	Clear to Send	DCE	DTE
CC	6	Data Set Ready	DCE	DTE
CD	20	Data Terminal Ready	DTE	DCE
CE	22	Ring Indicator	DCE	DTE
CF	8	Data Carrier Detect	DCE	DTE
CG	21	Signal Quality Detector	DCE	DTE
CH	23	Data Signal Rate Select	DTE	DCE
CI	23	Data Signal Rate Select	DCE	DTE
		TIMING		
DA	24	Transmitter Clock (DTE)	DTE	DCE
DB	15	External Clock	DCE	DTE
DD	17	Receiver Clock	DCE	DTE
		SECONDARY CIRCUITS		
SBA	14	Secondary Transmit Data	DTE	DCE
SBB	16	Secondary Received Data	DCE	DTE
SCA	19	Sec. Request to Send	DTE	DCE
SCB	13	Sec. Clear to Send	DCE	DTE
SCF	12	Sec. Data Carr. Detect	DCE	DTE

Circuit BA: Transmitted Data - The signals on this circuit are transmitted from DTE to the DCE. The DTE holds Transmitted Data at logic 1 (MARK condition) at all times when no data is being transmitted.

Circuit BB: Received Data - The signals on this circuit are received from the DCE to the DTE. The circuit is held at logic 1 (MARK) during intervals between data streams and at all times when no data is being transmitted.

Circuit CA: Request to Send - This circuit carries a request to transmit data from DTE to DCE. Hence this signal coordinates the action of the local DCE and DTE, and it does not reflect the status of any remote equipment, connected by modems.

Circuit CB: Clear to Send - This is a control signal that is transmitted from the DCE to the DTE to indicate that the DCE is ready to receive data from the DTE on transmit data circuit.

Circuit CC: Data Set Ready - The direction of this control signal is from the DCE to the DTE. It indicates the status of local data set. If the DSR signal is ON, it means the DCE is connected to the communications channel.

Circuit CD: Data Terminal Ready - The direction of this control signal is from the DTE to the DCE. DTR must be ON before the DCE turns on DSR, indicating that it has been connected to communications channel.

Essentially, DTR-DSR implement a more static version of the CTS-RTS protocol. By more static we mean that CTS-RTS protocol addresses channel readiness, and the DTR-DSR protocol addresses equipment readiness.

Circuit CE: Ring Indicator - The direction of this control signal is from DCE to DTE. The major application of this RS-232-C control is in configurations with automatic answer modems.

Circuit CF: Received Line Signal Detector - The DCE sends an ON condition to DTE on this circuit when it is receiving a carrier signal that meets its suitability criteria from the remote DCE. A widely used alternative name for this is Data Carrier Detect (DCD).

Circuit DA: Transmit Signal Element Timing - This signal is the transmit clock generated by the DTE. This circuit is used in synchronous transmission.

Circuit DD: Receiver Signal Element Timing - This signal is the clock received by the DTE. This circuit is also used in synchronous communication.

The remaining circuits are of no significance and are left unconnected in vast majority of microcomputer configurations.

Drawbacks of RS-232-C Interface

- The data transmission rate is limited to 20K bps.
- The distance for error free transmission is limited to 50 feet.

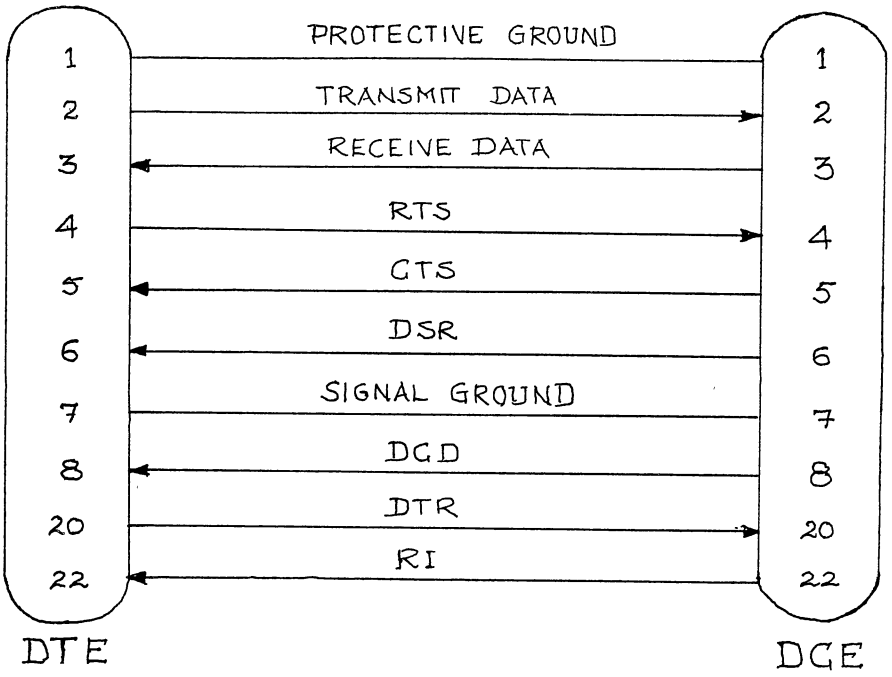
- Only one conductor per circuit is used, with only one signal return (ground) for both directions of transmission. Hence any fluctuations in reference voltage (signal return) can change the logic level of the signal. Hence the RS-232 interface is more susceptible to noise.
- The interface can generate considerable cross-talk among its component signals.

Thus, in November 1977 the EIA issued a new standard called the RS-449, which resolves the disadvantages of RS-232-C.

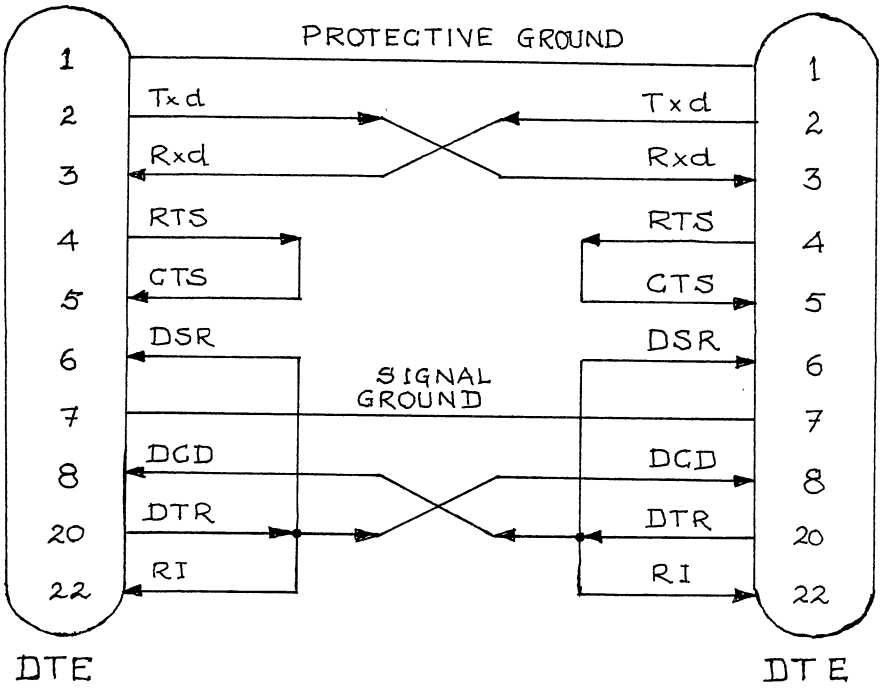
Common Configurations

Full duplex standard cable (STRAIGHT THROUGH): This configuration is used in computer systems which employ modems to communicate with other terminals or computers. Hence, Figure 3.2(a) depicts two connectors labeled as DCE and DTE, and the circuit (cable) is represented by horizontal lines.

NULL MODEM with Luxury Loop-Back: This configuration is used between two microcomputers which are locally connected and behave like two DTE's. The Null Modem constitutes a DCE that sits between two DTEs, and resolves the requirement for a DTE-DCE interface pair. The "cross-over technique" is used in a Null Modem cable to make it appear as a DCE (refer Figure 3.2(b)).



(a) STRAIGHT THROUGH



(b) NULL MODEM

Figure 3.2. Common configurations

RS-449 Interface Standard

Compared to RS-232-C, the RS-449 provides 10 new interchange circuits that permit a variety of new application procedures and diagnostic routines. Serial, binary data speeds up to 2 M bps at distances of at least 200 feet can be accomodated.

EIA defines the RS-449 Standard [12] as -

> "general purpose 37-position and 9-position interface for Data Terminal Equipment and Data Circuit Terminating Equipment employing serial binary interchange"(n.p.).

As a part of the effort to improve on the electrical characteristics of RS-232-C, the EIA produced two additional standards :

EIA Standard RS-422-A [10]," Electrical Characteristics of Balanced Voltage Digital Interface Circuits," December 1978.

EIA Standard RS-423-A [11]," Electrical Characteristics of Unbalanced Voltage Digital Interface Circuits," September 1978.

Balanced and Unbalanced Electric Circuits

Before discussing the EIA standard RS-449, let us investigate what is meant by the terms balanced and unbalanced circuits. Figures 3.3, 3.4 & 3.5 show the schematic diagrams [15] of circuits associated with RS-232-C, RS-423-A, and RS-422-A respectively.

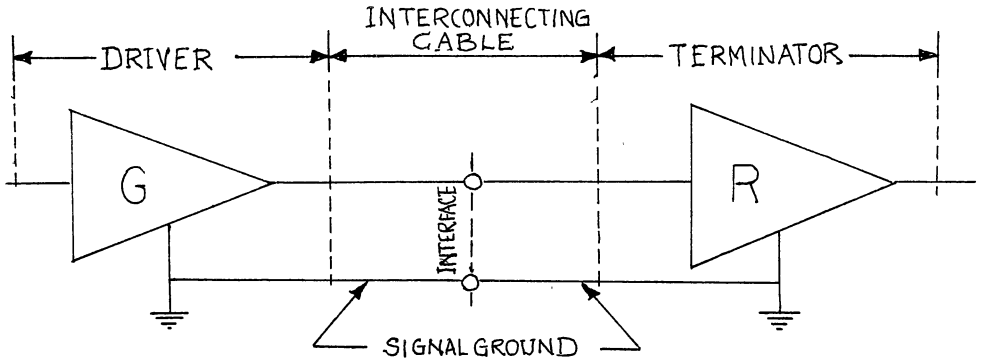


Figure 3.3. RS-232-C Electrical Interface Characteristic.

Designed for discrete component technology.
Unbalanced interface circuit.
Uses one conductor/circuit with one signal return.
Signal rate limited to 20 K bps.
Distance limited to 15 meters(50 feet).
Generates considerable cross talk.

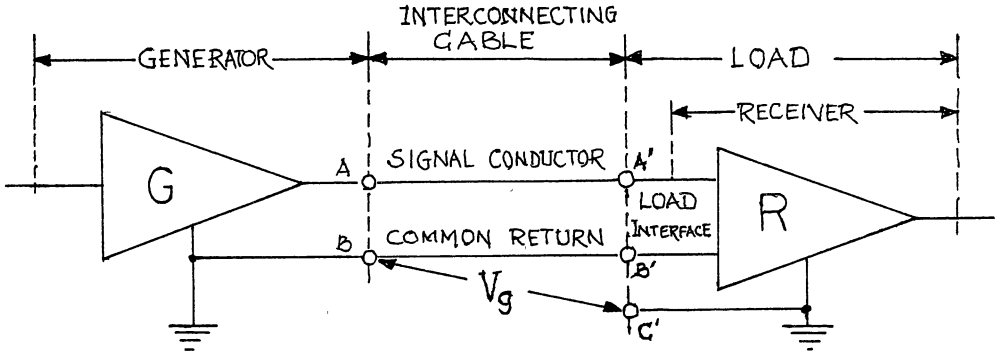


Figure 3.4. RS-423-A Electrical Interface Characteristics.

Designed for IC technology.
Unbalanced generator.
Differential receiver.
One conductor/circuit with independent signal return for each direction.
Signal rate up to 300 Kbps; Distance: 1000m
Reduced cross talk.

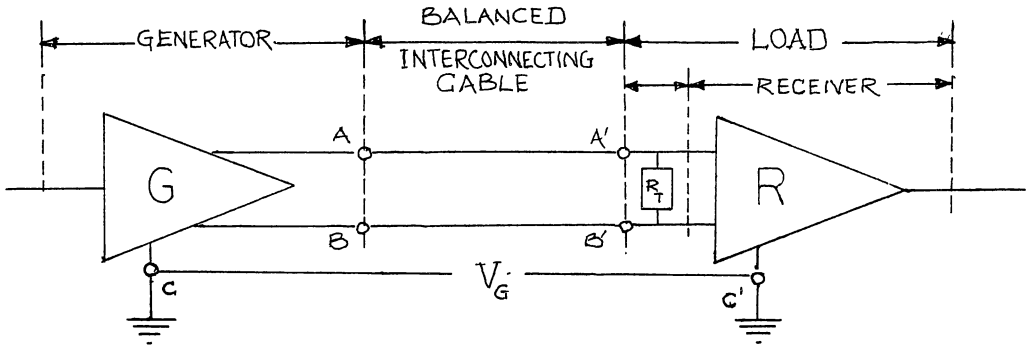


Figure 3.5. RS-422-A Electrical Interface Characteristics.

Designed for IC technology.
Balanced generator.
Differential receiver.
Two conductors per circuit.
Signaling rate up to 10 Mbps.
Distance: 1000m (<100 Kbps) to 10m (at 10 Mbps).
Reduced cross talk.

The RS-423-A [11] uses an unbalanced generator (Fig.3.4) to transmit a signal (data or control). One output of this generator is always the conductor which carries the signal of interest. The other output is the sender's signal ground. The RS-422-A [10] uses a balanced generator (Fig.3.5), and hence the generator transmits a particular signal (voltage level) as the difference of the two outputs of the generator. The signal ground is not used at all. Hence any noise induced in the reference level (ground) does not affect transmission.

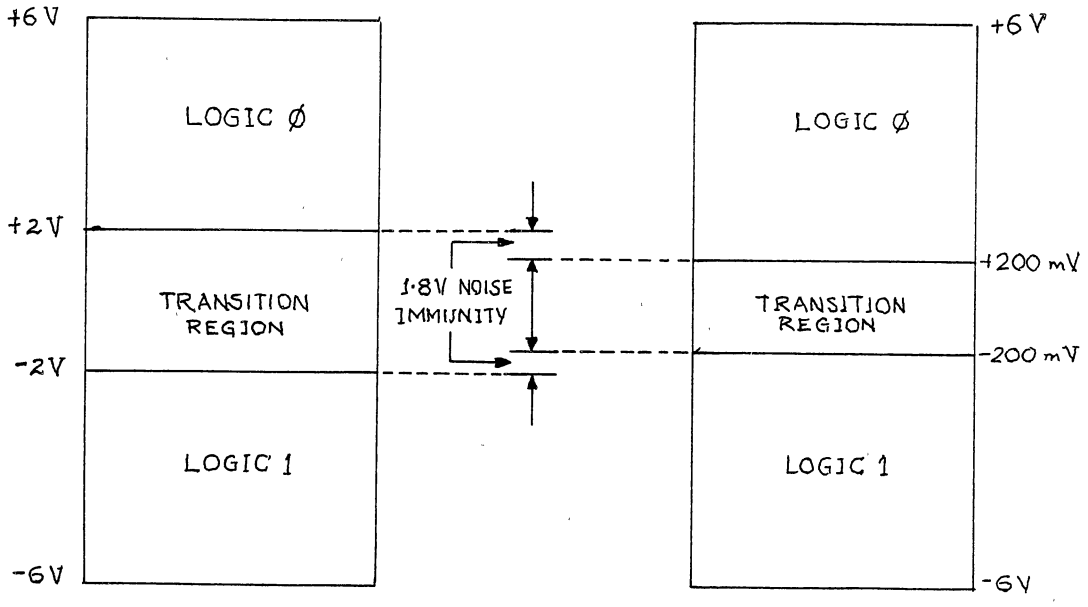
The major advantage of both RS-422-A and RS-423-A over RS-232-C is that they employ differential receivers.

The most important characteristic of this type of receiver is that it measures a difference in voltage between two inputs. Since both inputs pass through the same electrical environment, they will experience the same voltage alteration, say X volts. Hence the "noise-laden" voltages at points A' & B' (Fig. 3.4 & 3.5) are V_A+X and V_B+X assuming the original signals were V_A and V_B . Note that the difference measured by the receiver remains the same as the differential voltage originally sent (RS-422-A), namely V_A-V_B since $(V_A+X) - (V_B+X) = V_A-V_B$.

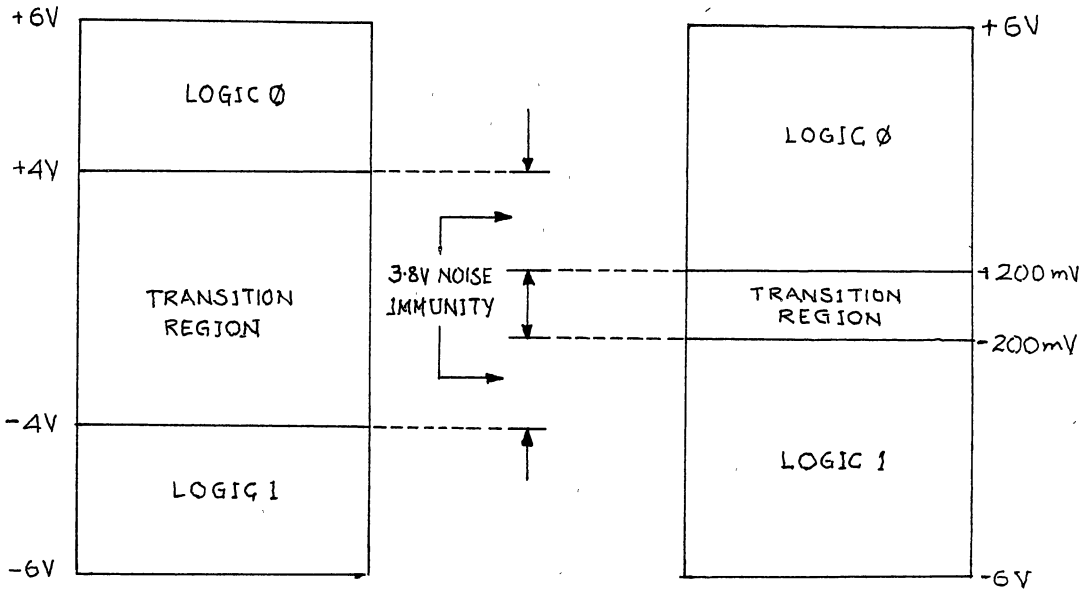
Electrical Characteristics

The RS-449 specification divides the interchange circuits into two categories in defining the implementation of the new electrical characteristics. Catagory I applies to primary data, timing, and five selected control circuits, while catagory II applies to all other circuits.(Table II).

The major objective of the RS-449 standard has been to maintain compatability with RS-232-C interfaces. This has been partially accomplished by allowing the use of unbalanced RS-423-A electrical characteristics in RS-232-C interfaces, provided the data-transmission rate is less than 20 K bps. For data rates over 20 K bps, the RS-232-C signals compatible with RS-449 standard, must use the balanced RS-422-A electrical characteristics.



(a) RS-422-A Interface Electrical Characteristics



(b) RS-423-A Interface Electrical Characteristics

Figure 3.6. RS-449 Interface Electrical Characteristics.

The relationship between voltage and logic states for RS-422-A and RS-423-A are illustrated in Figure 3.6. Note that both standards use a narrower voltage range (-6V to +6V) than RS-232-C (-15V to +15V). In order to compensate for the unbalanced transmitter circuit, the noise margin for RS-423-A is 3.8V, which is significantly greater than 1.8V noise margin for RS-422-A.

Mechanical Characteristics

Since RS-449 incorporates more than 25 signals (used in RS-232-C), the EIA elected to use a 37 pin connector. In order to satisfy the requirements of some foreign standards organizations, the EIA chose another 9 pin connector on which only secondary channels appear.

Functional Characteristics

There are ten signals defined in the RS-449 Standard. Very briefly, their functions are as follows:

o Send Common (SC): This circuit provides a signal common return path for unbalanced circuits used in the direction from DTE to DCE.

o Receive Common (RC): This circuit provides a common return path for unbalanced circuits used in the director from DCE to DTE.

o Terminal in Service (IS): This circuit indicates to the DCE whether or not a DTE is operational. A major application of this circuit is to provide a busy signal

on a telephone line associated with a given DTE in case it is out of order.

o New Signal (NS): This signal is used mostly in multipoint polling applications. The NS circuit is used by the control DTE to signal the control DCE that the message from one remote DTE has ended and a new one is about to begin.

o Select Frequency (SF): This signal is used by the DTE to select the transmit and receive frequencies used by its DCE.

o Local Loop-back (LL): This circuit is used by the DTE to request loop-back testing.

o Remote Loop-back (RL): This circuit is used by the DTE to request the initiation of remote loop-back testing.

o Test Mode (TM): TM is used to indicate to a DTE when a test condition has been established that involves its local DCE.

o Select Standby (SS): This signal is used by the DTE to request a switch to standby equipment to replace prime equipment in case of failure.

o Standby Indicator (SB): SB is used to indicate to the DTE whether regular or standby facilities are in use.

Appendix A gives a complete list of the RS-449 circuits and the equivalent RS-232-C circuit where applicable.

CHAPTER IV

DATA LINK LAYER

The data link layer uses the physical layer to transmit and receive data frames and ensures error free communications. This layer provides data sequencing and error control (detection and correction) techniques using certain protocol standards.

A protocol provides a method for the orderly and efficient exchange of data by establishing rules for the proper interpretations of controls. Use of an appropriate protocol can also reduce transmission inefficiencies. By allowing parameters to be preset before transmission, the extra information required to identify the parameters need not be sent with every data transfer.

The basic tasks required in controlling information exchange over a serial communications facility include:

- Establish and verify a connection.
- Establish and/or verify identities.
- Establish precedence and orderof transmission.
- Handle data sequencing (i.e. blocking).
- Permit interruption or temporary suspension of communications with re-establishment of transmission.

o Permit control of devices and features attached to communicating stations.

There are two basic types of Data Link Control (DLC) protocols:

1. Byte oriented protocols - These protocols are character oriented. They use certain predefined characters from a given code set for supervisory control. (e.g., IBM BSC, DEC DDCMP)

2. Bit oriented protocols - A totally different class of link control protocols which use positionally located control fields rather than code set combinations for supervisory control. (e.g., SDLC, HDLC, ADCCP, BDLC, UDLC etc...)

The two types of transmission protocols described here are IBM's Binary Synchronous Communications (BSC) protocol and synchronous data link control (SDLC) protocol.

IBM Binary Synchronous Communications (BSC)

BSC [8] has been used since 1964 for IBM high speed synchronous data transmission. Synchronization is achieved by achieved by sending specific bit patterns, called SYNC characters, at the start of transmission. The receiving equipment trains on this sequence, adjusts its timing to conform, and operates in step with the transmitter. Data is sent as a serial string of binary digits composed of one or more 1 byte blocks. BSC

permits the use of variable block lengths and is suited for batch transmissions. BSC can be used with devices that transmit/receive EBCDIC, ASCII, or SBT (six bit transcode) code.

BSC defines a set of control characters and sets up a fairly well defined set of data link control procedures using these control characters. The control characters and their respective functions are listed below. The Hexadecimal codes for the control characters are given in Table II.

TABLE II

HEXADECIMAL CODING OF BSC CONTROL CHARACTERS

Control Characters	CODE	
	EBCDIC	ASCII
SYN	32	16
SOH	01	01
STX	02	02
ETX	03	03
ENQ	2D	05
ETB	26	17
NAK	3D	15
EOT	37	04
ACK/0	10/70	10/30
ACK/1	10/61	10/31

o Synchronous Idle (SYN) is used to establish and maintain synchronization and as a time fill in the absence of data. A character-phase sync pattern is two SYNs at the start of each block.

o Start of Heading (SOH) precedes a block of heading characters. A heading is a character consisting of information such as priority, routing etc.

o Start of Text (STX) precedes a block of text characters and terminates a heading.

o End of Text (ETX) terminates a block of characters. ETX requires a reply indicating the status of the receiving station.

o End of Transmission Block (ETB) indicates end of a block of characters that started with SOH or STX.

o End of Transmission (EOT) signals the end of a message, and causes a reset of all stations on the line.

o Enquiry (ENQ) has two uses: to obtain a repeat transmission, or to bid for the line when transmitting point-to-point.

o Affirmative Acknowledgement (ACK 0/ACK 1) consists os two characters, ACK 0 and ACK 1, which are sent alternately as blocks are received and accepted by the receiving station to indicate that each block has been received without error.

o Negative Acknowledgement (NAK) indicates either that the previously received block was in error and the

station is ready for retransmission, or the receiving station is not ready to respond to a station selection.

o Block Check Character (BCC). Each block of data is checked for receive errors. An error checking algorithm is executed for all bits in the data block, and the result is transmitted in the BCC character. The receiver executes the same algorithm and compares its result with received BCC character, thereby determining if there was an error in transmission.

Basic Operation of the Data Link

Block transmission in BSC starts by acquiring the line when it is in an idle (available) status. This is accomplished by sending the enquiry (ENQ) control character and receiving a ready response (ACK 0) from the receiving station. Transmission is ended by transmitting the EOT character, causing the line to be placed in the idle status. An illustration of basic BSC transmission sequences is shown in Table III.

Error Checking

BSC incorporates several techniques to check the accuracy of received data. These checking methods are directly related to the transmission codes used. With EBCDIC and SBT code, Cyclic Redundancy Check (CRC) [13] is employed. CRC is calculated at sending and receiving end by cyclically summing bits of the message as a binary

accumulation and dividing this value by a prime number. Only the remainder is kept as the BCC. With ASCII, Vertical Redundancy Checking (VRC) is used to check each character as it is received, and the entire block is checked by longitudinal Redundancy Checking (LRC). VRC is nothing but a parity check, and LRC character is inserted at the end of the transmission block as the BCC.

TABLE III

BSC CONTROL SEQUENCES FOR POINT-TO-POINT TRANSMISSION

Action	Character Sequences	
	Station A	Station B
Transmission initiated	SYN SYN ENQ FF	SYN SYN ACK0 FF
Transmit data block 1	SYN SYN STX... text..ETB BCC FF	SYN SYN ACK1 FF
Transmit data block 2	SYN SYN STX... text..ETB BCC FF	SYN SYN NAK FF
Retransmit block 2	SYN SYN STX... text..ETB BCC FF	SYN SYN ACK0 FF
End Transmission	SYN SYN EOT FF	--

IBM Synchronous Data Link Control (SDLC)

IBM announced SDLC in 1973 in conjuction with the 3600 teller terminal systems for financial institutions. Today its use has becone widespread, as manufacturers of IBM compatible equipment emulate popular SDLC products eg. 3274/ 3278 terminals. SDLC is a bit oriented protocol and it permits both half and full-duplex operation. It permits up to seven blocks of data to be outstanding before requiring an acknowledgement (buffer storage must be set aside to store the unacknowledged blocks). Also, SDLC is equipment independent.

Stations

A station is the equipment/device located at one of a communications link. One and only one station of a communications line is a primary station; all other stations on the line are secondary stations. The primary station initiates all transmissions from a secondary station by inviting or commanding responses from the addressed secondary station.

Modes of Operation

The communications link can be in one of the three transmission states [7,16] :

o Transient State - Normally, this is the period between the station signalling the modem with a request to send the modem responding with a clear to send.

o Idle State - No transmission of control information or data. It is identified by a succession of 15 or more consecutive binary 1's.

o Active State - Control information or data is being transferred. The secondary station can be in one of three active modes, as established by the primary station-

o Normal Response Mode (NRM) - Responds to poll.

o Normal Disconnect Mode (NDM) - Responds to polls with a request to be put on line or to be initialized; ignores other commands.

o Initialization Mode (INIT) - Procedures for initialization mode are specified by using the system components.

Frame Structure

A common transmission frame layout is used for all transmissions. Three formats within the frame identify the exact interpretation of the contents of the frame. As illustrated in Figure 4.1 [7] the transmission frame consists of the sequence :

FLAG - ADDRESS - CONTROL - INFORMATION - CRC - FLAG

Only the information field is optional. The other fields will always be present in any valid command or information transfer.

Flag. This field is present both at the beginning and ending of a frame, and is the binary number 01111110 (7EH).

Address. This field always contains the identity of the secondary station that is communicating with the primary station. The secondary station may have one of the three types of addresses - station address, group address (common to several stations) or, broadcast address (acceptable to all stations on the link).

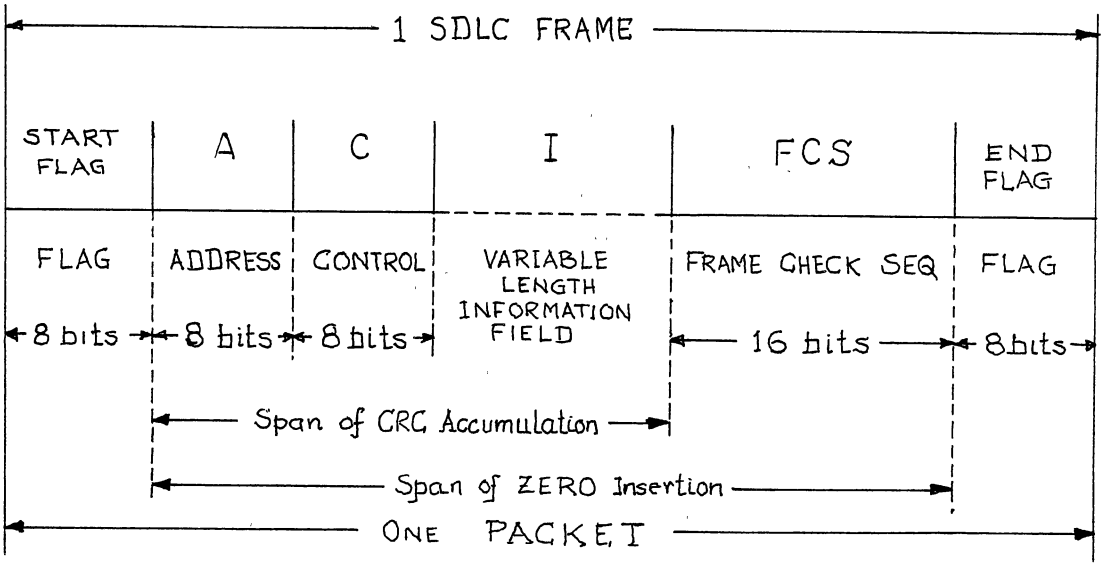


Figure 4.1. SDLC Frame Format

Control. This field identifies the type of transmission. It can assume one of the three formats as illustrated in Figure 4.2 [7].

1. Information Transfer Format - Here the control field contains information that tracks the number and sequence of frames sent and received. Up to 7 unacknowledged numbered information frames may be outstanding (transmitted but not confirmed) at the transmitter. All unacknowledged frames must be retained by the transmitter, because it may be necessary to retransmit some or all of them if transmission errors occur.

2. Supervisory Format - This identifies the three supervisory commands/responses: RR,RNR and REJ.

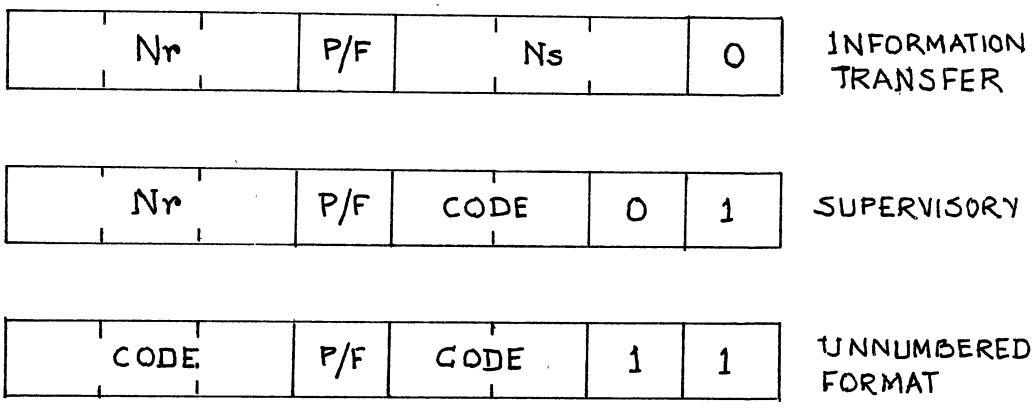
o RR (Receive Ready) is sent by either primary or secondary station. It confirms numbered frames through Nr-1 and indicates that originating station is ready to receive.

o RNR (Receive Not Ready) is sent by either primary or secondary station indicating a temporary busy condition due to buffering or other internal constraints.

o REJ (Reject) is sent by either primary or secondary station to request transmission or retransmission numbered frames.

3. Unnumbered Format - This provides the basic control for setting up information exchanges between primary and secondary stations. There are a total of

fourteen Unnumbered Information (UI) commands and responses. The three most used primary station commands are UA, RIM, DM and FRMR. A summary of command and response control fields [7] is given in Appendix B.



Nr - Number of correct frames received since last ACK. Maximum = 7, (3 bits)
Ns - Number of frames sent since last ACK. Maximum = 7, (3 bits).
P - Poll bit; sent by the primary station.
F - Final frame; sent by secondary station.
CODE - Supervisory and Nonsequenced control code.

Figure 4.2. Control Field Codes.

Zero Insertion[16]

The opening and closing flags must be unique to properly define the transmission frame. Each frame begins and ends with a special bit pattern, namely 01111110.

Whenever the transmitting hardware encounters five consecutive 1's in the Information, Address or Control fields, it automatically stuffs a 0 bit into the outgoing bit stream. Hence, when the receiver sees five consecutive incoming 1 bits, followed by a 0 bit, it automatically destuffs (deletes) the 0 bit.

Table IV illustrates the advantages of a bit oriented protocol (SDLC) over a byte oriented protocol (IBM BSC). One of the problems associated with both BSC and SDLC is the delay involved in waiting for acknowledges. Prolonged delays with BSC are imminent, given the fact that every block of data needs an acknowledgement. This problem is lessened considerably by the use of SDLC.

The ANSI Advanced Data communications Control Procedures (ADCCP) and the ISO High level Data Link Control (HDLC) are largely similar to SDLC. The most important capability these protocols have over SDLC is their ability to support a modulo count of 128 i.e., have up to 127 messages or blocks of data outstanding before an acknowledgement is required.

TABLE IV

ADVANTAGES OF SDLC OVER IBM BSC

Features	SDLC Bit oriented	BSC Byte oriented
1) Mode of Transmission	Half duplex & Full duplex.	Half duplex only
2) Control Codes	Unique codes Not from users data set.	Codes chosen from user's data set.
3) Acknowledges	Permits 7 blocks of data to be outstanding before any ACK is returned.	Every block of data to be acknowledged by receiver.
5) Operation	Device independent.	Device dependent

CHAPTER V

HARDWARE AND SOFTWARE DESIGN AND IMPLEMENTATION

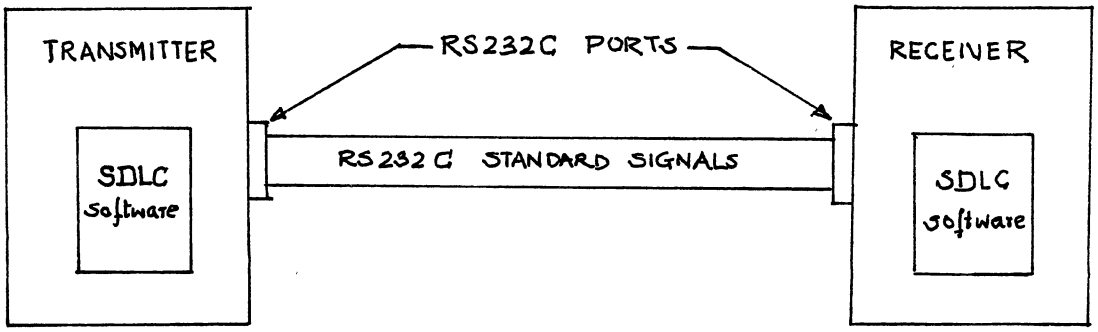
The software and hardware were implemented using a simple computer network consisting of two microcomputers. The microcomputers were Radio Shack TRS-80 model IIs, which are Z80 microprocessor based systems. They behave like Data Terminal Equipment (DTE)[17].

The implementation can be divided into two stages as follows. (refer Figure 5.1 (a) & (b))

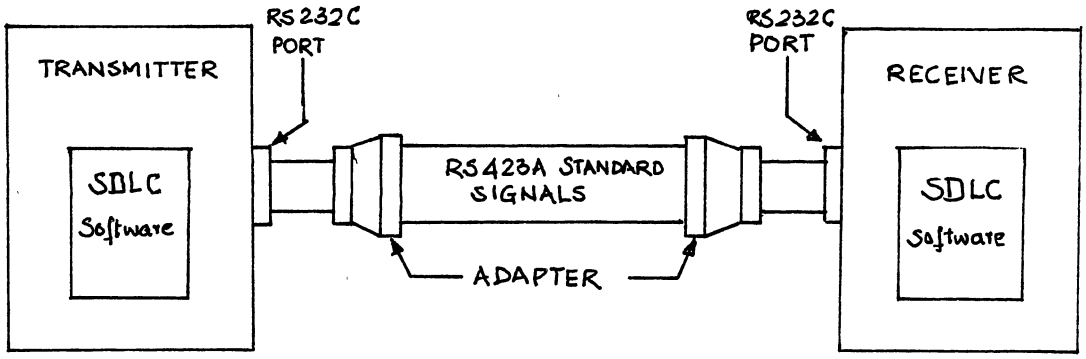
1. Develop data communications software on the TRS-80 model II for inter-microcomputer communications via the RS232-C ports already available. The Null Modem cable is configured as shown in figure 5.2. The purpose of the software is to enable the user to transfer files from one microcomputer to another, using the SDLC protocol.
2. Design the hardware interchange circuit (adapter) to convert the available RS232-C standard signals to the RS449 standard signals. Later, test the hardware with the help of the communications software developed in stage I. As illustrated in figure 5.1 (b), the RS232-C signal available at the transmitter is converted to RS423-A standard signal with the help of the adapter;

and, the same RS423-A signal is converted back to the RS232-C signal at the receiver end.

Finally, The performance of the two types of interface circuits are compared, using the SDLC communications software as a benchmark.



(a) STAGE I - RS232-C standard communications channel.



(b) STAGE II - RS423-A standard communications channel.

Figure 5.1. Block diagram for the two stage implementation.

Software Design

The serial I/O device used in the TRS 80 model II microcomputer is a Z80 SIO. The Z80-SIO [19] is capable of handling asynchronous and synchronous byte oriented protocols such as IBM Bisync, and synchronous bit oriented protocols such as HDLC and SDLC.

Features of Z80 SIO

- Two independent full-duplex channels.
- Data rates in synchronous or isosynchronous modes:
  0 - 500 Kbps with 2 MHz system clock.
  0 - 880 Kbps with 4 MHz system clock.
- Receiver data register quadruply buffered; transmitter doubly buffered.
- IBM SDLC features:
  Abort sequence generation and deletion.
  Automatic zero insertion and deletion
  Automatic flag insertion between messages.
  Address field recognition.
  CRC generation and checking.
- Separate modem control signals for both channels.

Programming Z80 SIO

The SIO offers the choice of polling, interrupts (vectored or non-vectored) and block-transfer modes to

transfer data, status and control information to and from the CPU.

Polling. The Z80-SIO status registers are updated at appropriate times for each function being performed. When the CPU is operating in a polling fashion, it reads the contents of these status registers, before performing an input or output of data bytes. In addition, error and other consideration are also indicated. Polling is adopted in slow asynchronous communications protocols [14].

Interrupts. The SIO has an elaborate interrupt scheme to provide fast interrupt service in real time applications. Transmit interrupts, receive interrupts and external interrupts are the main sources of interrupts. Each interrupt source is enabled under program control, with channel A having higher priority than channel B, and with receive, transmit and external/status interrupts prioritorized in that order within each channel.

When the transmit interrupt is enabled, the CPU is interrupted by the transmit buffer becoming empty. The receiver can interrupt the CPU in one of three ways:

- Interrupt on first received character.
- Interrupt on every received character.
- Interrupt on a special receive condition.

The special receive condition (e.g. End Of Frame interrupt in SDLC) can cause an interrupt, only if the

interrupt on first character received, or interrupt on all received characters mode is selected.

Internal Structure of Z80 SIO: Each channel contains its own set of control and status (write & read) registers, and control and status logic. The registers for each channel are designated as follows :

WR0 - WR7 = Write registers 0 through 7.

RR0 - RR2 = Read registers 0 through 2.

Table V lists the functions assigned to each read and write register. The program first issues a series of commands that initialize the basic modes of operation and then other commands that qualify conditions within the selected mode. The read and write register bit functions are listed in Appendix C & D respectively.

The programming logic for SDLC protocol involves the following steps. 1. Initialization. 2. Operation. 3. Termination. Detailed explanation of these steps are discussed in the Z80 SIO Technical Manual [18].

Initialization

The Z80 SIO is programmed to operate in SDLC mode. The write registers are initialized to the following settings.

o Interrupt logic - Vectored interrupts are enabled. Enable transmit and receive (every character) interrupts. Hence, the special receive condition interrupts (End of Frame, Receive Overrun) are automatically enabled.

TABLE V

Z80 SIO CONTROL AND STATUS REGISTERS

Read Register Functions	
RR0	Transmit/ Receive buffer status, interrupt status and external status.
RR1	Special Receive condition status.
RR2	Modified interrupt vector (Channel B only).
WR0	Register pointers, CRC initialize, Initialization commands for various modes, etc.
WR1	Transmit/Receive interrupt and data transfer mode definition.
WR2	Interrupt vector (Channel B only).
WR3	Receive parameters and control.
WR4	Transmit/Receive misc. parameters and modes.
WR5	Transmit parameters and controls.
WR6	Sync character or SDLC address field.
WR7	Sync character or SDLC Flag.

- Transmit parameters: Data bits per character equals 8. Select SDLC polynomial for CRC calculation. Enable DTR and CTS. Disable transmit logic & CRC.
- Receive parameters: Data bits per characater equals 8. Enable frame hunt and CRC. Disable DCD & CTS. Disable receive logic.
- Address byte is written into WR6 and Flag byte (7EH) is written into WR7.

Operation

The transmitter (primary station) starts transmitting only after CTS is active. CTS becomes active once the receiver (secondary station) is ready to receive. The whole information frame is transmitted using transmit interrupts. When the transmit underrun occurs (at the end of frame), the transmit logic is disabled. The next record is read from the source file. This process is repeated until all records of the file are transmitted.

The receiver is interrupted for the first time on receiving the address byte, and every byte thereafter. An end of frame interrupt is generated at the end of each information frame. The transmitter is enabled by enabling RTS at the beginning of each frame.

Termination

The control field is used to recognize the end of file. The control field equals 0F (Hex) for all records that carry information. The end of file record contains a control field of 00 (Hex). Once the EOF record is received the destination file is closed.

Baud Rate Modification

The Z80 Counter Timer Clock (CTC) [18] supplies the clock input to channels A & B of the Z80 SIO. The Z80

CTC can be programmed to operate in timer or counter mode. In the counter mode, the CTC output clock rate can be varied by changing the time constant loaded into CTC's time costant register.

For a given baud rate the time constant can be calculated as follows.

$p \times T = 1 / f$ where, p = input clock period
T = time constant
f = Baud Rate (frequency)
O = input clock freq.

$p = 1 / O$
$= 1 / 2$ MHz $= 500$ nsec.

For f = 9600 baud,

$T = 1 / f \times p$
$= 1 / 9600 \times 500 \times 10^{-9}$
$= 208$

The time constants for the baud rates ranging from 9600 to 2 Mbps are listed in Appendix E.

Purpose of SDLC communications

The same SDLC communications program resides on both microcomputers. One microcomputer assumes the role of the primary station (transmitter), and the other is the secondary station (receiver). The communications program helps in transferring user selected files from the primary to the secondary station.

The receiver station has to be set up first by going through the following steps.

1. Choose appropriate baud rate (key in the code).

1. Choose appropriate baud rate (key in the code).
2. Key in the destination file name.

Then, the transmitter station is started by the following steps.

1. Choose the same baud rate as the receiver.
2. Key in the source file name that is to be transmitted.
3. Key in a carraige return (CR) if receiver is ready.

An asterisk (*) is displayed for every record transferred on both microcomputers. If the file is transferred properly, a completion message is displayed at the end. However, overrun and CRC errors are detected by the software and appropriate error messages are displayed.

Hardware Design

The hardware design consists of two phases. The first phase is to configure the RS-232-C null modem cable as shown in Figure 5.2. This cable is used as the serial channel connecting the two Radio Shack microcomputers.

The second phase involves the design of the RS-232-C -- RS-449 interface [12]. Interface circuits implemented with RS-422-A will not directly interoperate with RS-232-C receivers, since RS-422 uses balanced circuits and RS-232 uses unbalanced circuits. Therefore, when operating with RS-232-C equipment, RS-423-A generators must be used on category I as well as category II circuits, to facilitate interoperation.

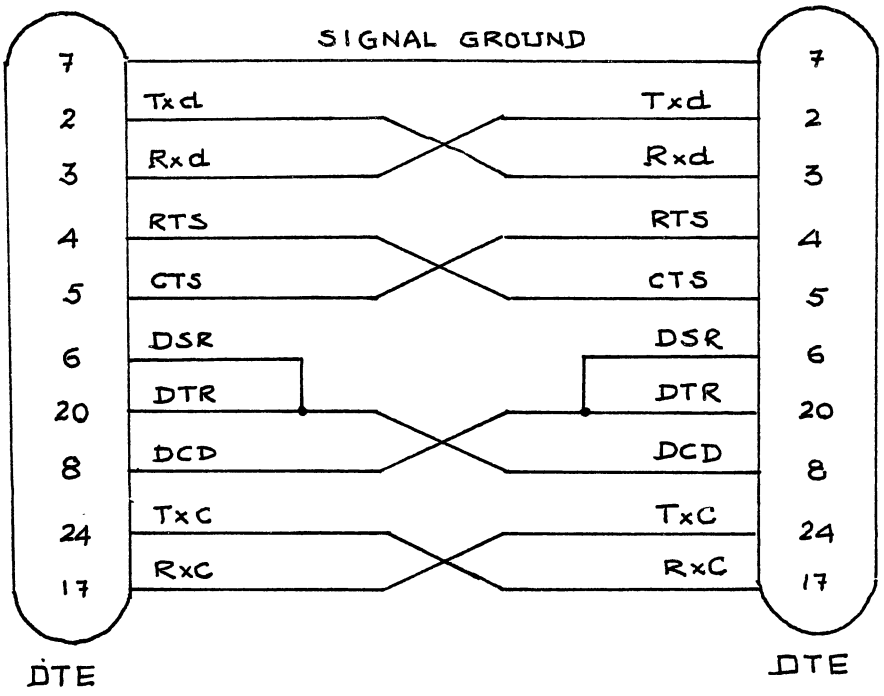


Figure 5.2. RS-232-C Null Modem Cable

The TRS-80 model II microcomputer supports RS-232 standard signals over its serial channels. Hence, to make the signals appear as RS-423-A compatible signals the following hardware modification is implemented at both the primary and secondary stations (externally on the communications channel).

Transmitter: The signal transmitted complies with the RS-232-C standard (without any hardware interface). This signal has to be converted to a RS-423 compatible signal (logic 0 = +6V, logic 1 = -6V). Hence, the transmitted signal is input to a RS 232 receiver (MC1489), which converts the signal to TTL voltage levels. This output is input to an unbalanced RS 423

generator (Am26LS29). The Figure 5.3. depicts this interface circuit.

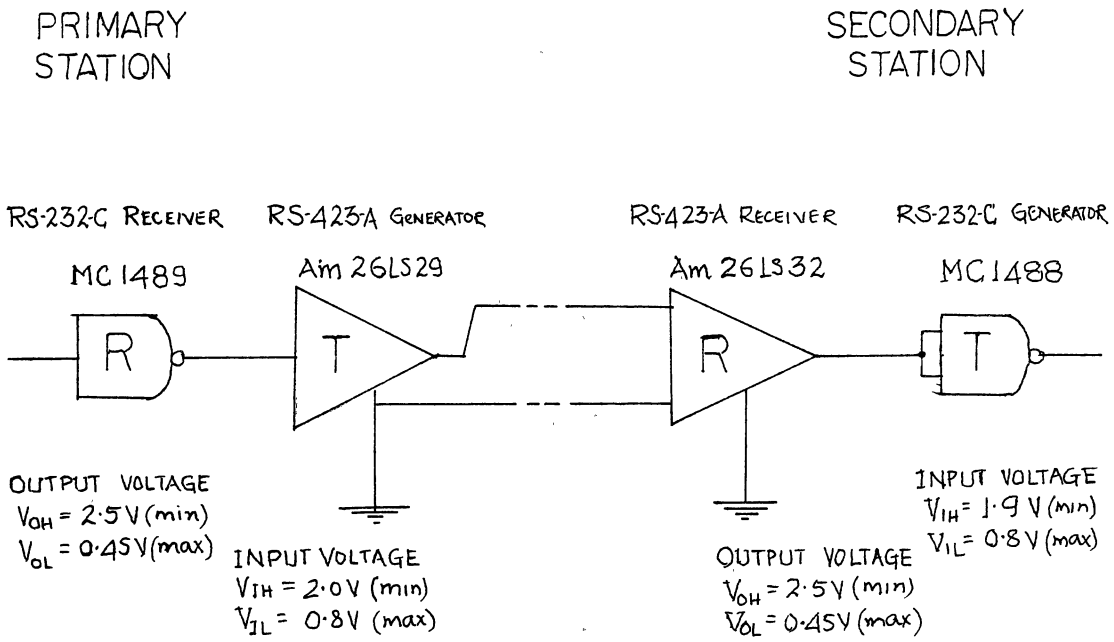


Figure 5.3. Generator and Receiver connections at Interface.

Receiver: The received signal complies with the RS-423 voltage levels (logic 0 = +6V, logic 1 = -6V). This signal has to be converted to RS 232 levels. This is accomplished by a set of RS-423 differential receiver (Am26LS32) and a RS 232 generator (MC1488). This is illustrated in the same circuit diagram (figure 5.3).

The circuit diagram for the entire channel interface is illustrated in Figure 5.4. The cable connecting the RS 423 generator and receiver is a twisted pair - with

the signal circuit and the signal return intertwined. The power supply requirements are +5V, -5V, +12V and -12V. The integrated circuit chips (I.C.'s) used in the circuit are listed in Table VI. The pin connection diagrams for each I.C. used are illustrated in Appendix F.

TABLE VI

Functional Specifications of the I.C.'s Used.

I.C Id. No.	Component	Function
U1, U2	MC1488	RS232-C Quad Line Drivers.
U3, U4	MC1489	RS232-C Quad Line Receivers.
U5, U6	Am26LS29	RS423 Quad Line Drivers.
U7, U8	Am26LS32	RS422/423 Quad Differential Line Receivers.

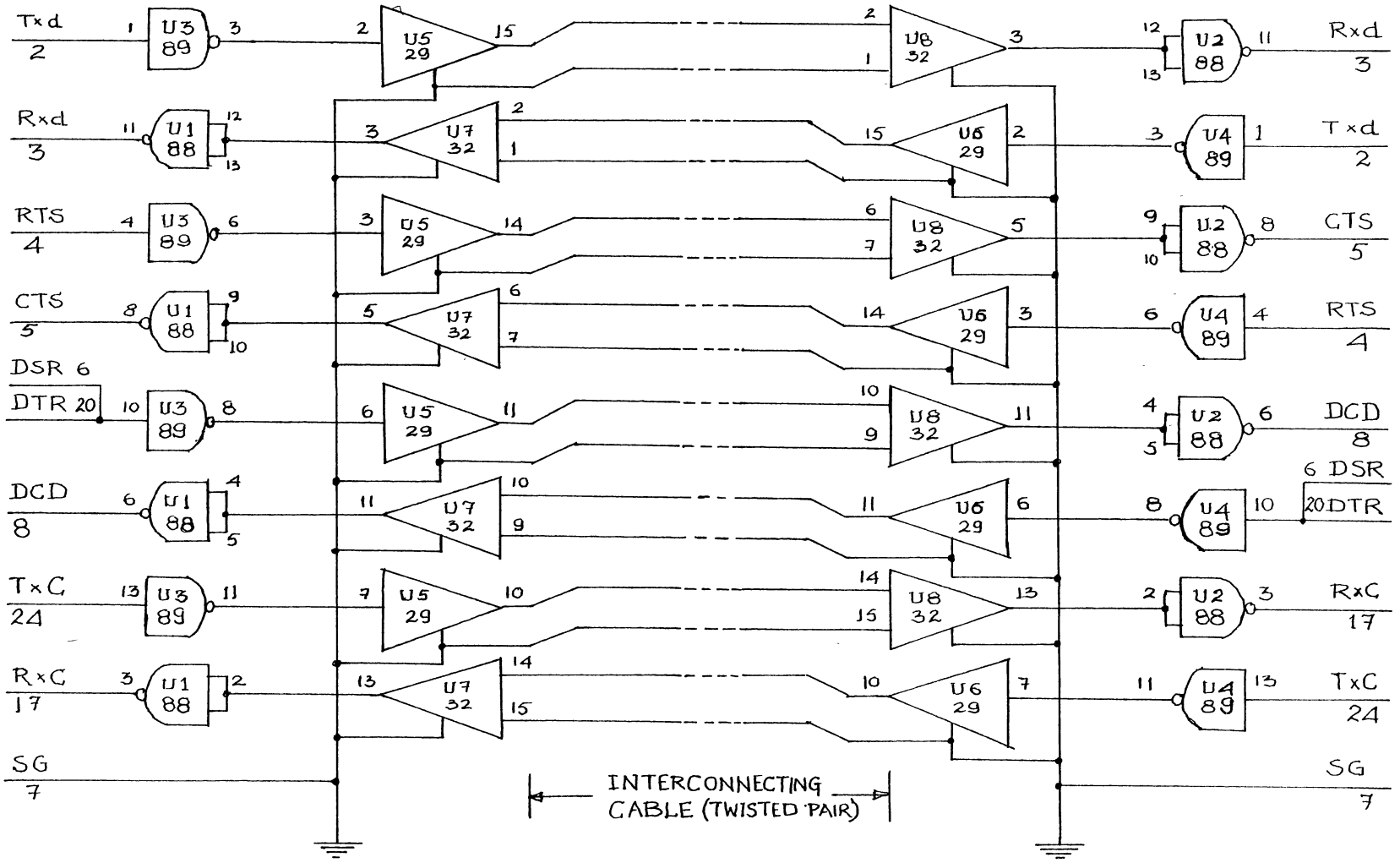


Figure 5.4 RS-232-C/RS-449 Channel Interface Circuit Diagram

CHAPTER VI

SUMMARY, CONCLUSIONS, AND SUGGESTIONS FOR FUTURE DEVELOPMENT

Summary of Results

The main purpose of this project was to evaluate the new interface standard called the RS449 (RS423 in particuler) and compare its performance with the RS232-C interface standard. A communications software, which uses SDLC protocol, was developed to measure the performance of the two types of interface. As discussed in chapter V, a true RS449 standard interface port was not used. Instead, the RS232-C ports available on the microcomputers (TRS-80 model II) were converted to RS423-A standard ports with the help of hardware adapters developed using standard receiver/driver integrated circuit components.

The criteria used for measuring the performance of the two interfaces were -

1. Speed: Rate of data communications in bits per second.
2. Distance: Maximum distance of error free transmission.

Speed: The data rates provided by the software developed, were - 7812 bps, 9600 bps, 19.2 Kbps, 38.4 Kbps, 76.8 Kbps and 153.6 Kbps. The software had a limitation with reference to the maximum rate at which it received characters. The time taken by the Receive Interrupt Service routine determines the minimum (least) duration to receive one character. Hence, the time taken to service a receive interrupt was 55.5 usecs. However, for a data rate of 153.6 Kbps, each byte was arriving at the receiver port, once every 52.1 usecs. This indicates that the software was limited to a maximum data rate of 76.8 Kbps. This was tested out with both types of interfaces (RS232C & RS423A) using short lengths of cables (5 feet long).

Distance: According to the RS423-A specifications, one can transfer data over larger distances without any signal loss as compared to the RS232-C interface. Hence, to evaluate this aspect, a 60 feet long cable (twisted pair) was used to connect the pseudo RS423-A ports (RS232 ports converted to RS423 ports). The software functioned properly (error free) over the RS423 interface channel. Whereas, over a 60 feet long RS232 cable connected between the RS232 ports alone, the software was not functioning consistantly. With the help of an oscilloscope it was determined, that the clock signal which was being transmitted over the RS232 channel, was deteriorating (attenuating) over long

cables. Hence, the receiver end was not synchronized with the transmitter. As a result, the software was not working. The data rates for transmission attempted, were again 7812 bps, 9600 bps, 19.2 Kbps, 38.4 Kbps and 153.6 Kbps. The file transfer did not occur for 153.6 Kbps, because of the software limitation mentioned earlier.

Conclusions

The performance of the communications software on the two types of interfaces - RS232-C and RS423-A, indicate that, with RS423 one can transfer data over larger distances without any signal loss, as compared to a RS232-C interface. According to specifications data rates up to 300 Kbps can be achieved using a true RS423-A interface. However, the data transfer rate is limited to 76.8 Kbps here, because of the software limitation. If the receive interrupt service routine is made shorter, or if a polling scheme is used, then, higher data rates could be achieved. Ultimately, the data transfer rate, using either type of interface, is limited by what the Z80-SIO can support. According to specifications, the maximum data rate must be less than (system clock frequency)/ 4.5 $\backsimeq$ 800 Kbps.

Future Development

In evaluating the RS-232-C -- RS-449 interface, a true RS-449 port was not used at the receiver end.

Instead, a RS-232-C port (available on the microcomputer) was converted to appear as a RS-449 port. For further analysis, one could use a true RS-449 standard port, and measure the performance of the communication link.

The interface developed may be useful in a "gateway", which is a special node that interfaces two or more dissimilar computer networks (..which use different interface standards in the physical layer, for example).

Since VLSI devices and interfaces to support them have appeared, local area networks (LANS) have been multiplying like rabbits. The advantages of connecting a relatively cheap work station to expensive file servers and graphic peripherals are obvious. However, locking users into a particular hardware/software combination could prove counter productive in the long run.

BIBLIOGRAPHY

1. Ahuja, Vijay. Design and Analysis of Computer Communications Networks. McGraw-Hill Book Co., 1982.

2. Berry, David. "Standardizing Upper-level Network Protocols." Computer Design, Vol. 23, 2(Feb 1984), 175-176.

3. Bertine, H. V. "Physical Level Protocols." IEEE Transactions on Communications, Vol. COM-28, 4 (April 1980), 433-444.

4. Carlson, David E. "Bit Oriented Data Link Control Protocols." IEEE Transactions on Comm., Vol. COM-28, 4(April 1980), 455-467.

5. Conrad, James W. "Character Oriented Data Link Control Protocols." IEEE Transactions on Comm., Vol. COM-28, 4(April 1980), 445-454.

6. Cypser, R. J. Communications Architecture for Distributed Systems. Addison-Wesley Publishing Co., Reading, Mass., 1978.

7. Datapro Research Corp. IBM Synchronous Data Link Control (SDLC). Delran, NJ 08075.

8. Datapro Research Corp. IBM Binary Synchronous Communications (BSC). Delran, NJ 08075.

9. Deasington, Richard. A Practical Guide to Computer Communications and Networking. Ellis Horwood Limited, Chichester, England, 1982.

10. Electronic Industries Association. EIA Standard RS-422-A, Electrical Characteristics of Balanced Voltage Digital Interface Circuit. Washington D.C., 20006.

11. Electronic Industries Association. EIA Standard RS-423-A, Electrical Characteristics of Unbalanced Voltage Digital Interface. Washington D.C., 20006.

12. Electronic Industries Association. EIA Applications Notes on Interconnection Between Interface Circuits Using RS-449 and RS-232-C. Industrial Electronics, Bulletin No. 12, Washington D.C., 20006.

13. McNamara, John E. Technical Aspects of Data Communications. Digital Press, Digital Equipment Corp., 1982.

14. Makaram, Narayan. "Design of Asynchronous Communications Module for Microcomputer Systems." (Unpub. Report, Oklahoma State Univ.,1983).

15. Nichols, Elizabeth A., Nichols, Joseph C., and Musson, Keith R. Data Communications for Microcomputers. McGraw-Hill Book Co., 1982.

16. Tenenbaum, Andrew S. Computer Networks. Prentice Hall Inc., Englewood Cliffs, NJ, 1981.

17. TRS-80 Model II Technical Reference Manual. Tandy Corporation/ Radio Shack, Fort Worth, TX 76102.

18. Z80 CTC Counter/Timer Clock Product Specifications. Zilog Inc., Campbell, California, Feb. 1980.

19. Z80 SIO Technical Manual. Zilog Inc., Campbel, California, 1978.

APPENDIX A

RS-449 INTERFACE CIRCUITS AND RS-232-C EQUIVALENTS

TABLE VII

RS449 INTERFACE CIRCUITS AND
RS232-C EQUIVALENTS

EIA RS-449		EIA RS-232-C		Pin Assignment
SG	Signal Ground	AB	Signal Ground	19
SC	Send Common			37
RC	Receive Common			20
IS	Terminal in Service			28
IC	Incoming Call	CE	Ring Indicator	15
TR	Terminal Ready	CD	Data Terminal Ready	12-30
DM	Data Mode	CC	Data Set Ready	11-29
SD	Send Data	BA	Transmitted Data	4-22
RD	Received Data	BB	Received Data	6-24
TT	Terminal Timing	DA	Transmitter Clock (DTE Source)	17-35
ST	Send Timing	DB	Transmitter Clock (DCE Source)	5-23
RT	Receive Timing	DD	Receiver Clock	8-26
RS	Request to Send	CA	Request to Send	7-25
CS	Clear To Send	CB	Clear to Send	9-27
RR	Receiver Ready	CF	Data Carrier Detect	13-31
SQ	Signal Quality	CG	Signal Quality Detecter	33
NS	New Signal			34
SF	Select Frequency			16
SI	Signalling Rate Indicator	CI	Data Signal Rate Selector (DCE source)	2
LL	Local Loop Back			10
RL	Remote Loop Back			14
TM	Test Mode			18
SS	Select Standby			32
SB	Standby Indicator			36

Note: o Category I circuits have two pin assignments to handle differential generators & receivers.
o All Secondary Circuits are omitted in the table.

APPENDIX B

SDLC COMMAND AND RESPONSE

CONTROL FIELDS

TABLE VIII

SUMMARY OF COMMAND AND RESPONSE CONTROL FIELDS

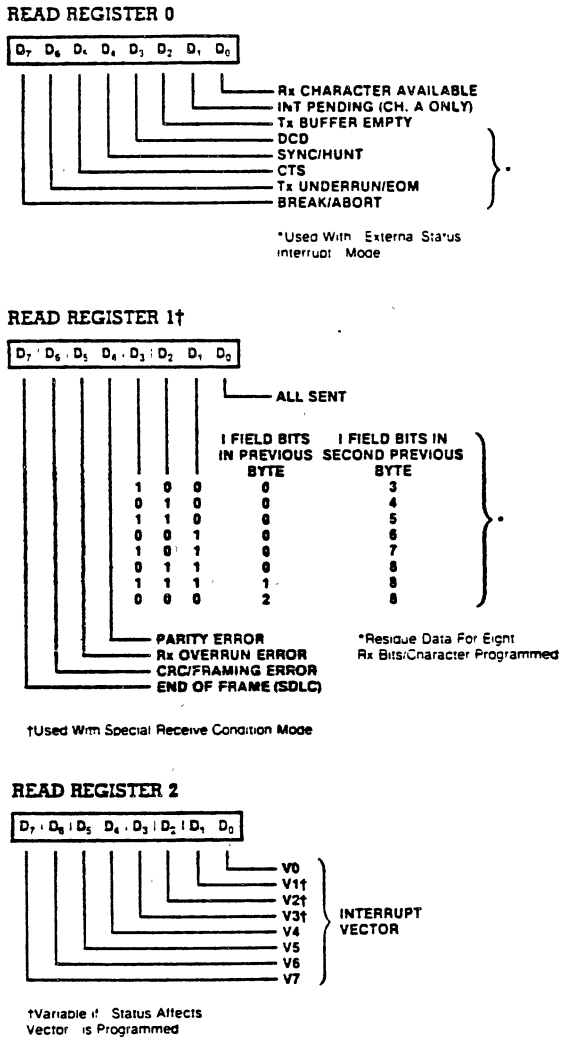
Field Format	Binary Configuration (Sent Last ↓ Sent First ↓)			Acronym	Command	Response	Defining Characteristics
Unnumbered	000	P/F	0011	UI	X	X	Unnumbered command or response that carries information.
	000	F	0111	RIM		X	Request initialization mode; initialization needed; expect SIM.
	000	P	0111	SIM	X		Set initialization mode; the using system prescribes the procedures.
	100	P	0011	SNRM	X		Set normal response mode; transmit on command only.
	000	F	1111	DM		X	This station is in disconnect mode.
	010	P	0011	DISC	X		Do not transmit or receive information (disconnected).
	011	F	0011	UA		X	Acknowledgment for unnumbered commands (SNRM, DISC, SIM).
	100	F	0111	FRMR		X	Frame reject; invalid frame received; must receive SNRM, DISC, or SIM.
	111	F	1111	BCN		X	Beacon; signals loss of input.
	110	P/F	0111	CFGR	X	X	Configure; contains function descriptor in information field.
	010	F	0011	RD		X	Request disconnect; this station wants to disconnect.
	101	P/F	1111	XID	X	X	Exchange station identification; identification in information field.
	001	P	0011	UP	X		Unnumbered poll; response optional if P bit not on
	111	P/F	0011	TEST	X	X	Test; pattern in information field.
Supervisory	Nr	P/F	0001	RR	X	X	Ready to receive.
	Nr	P/F	0101	RNR	X	X	Not ready to receive
	Nr	P/F	1001	REJ	X	X	Reject; transmit or retransmit, starting with frame Nr
Information	Nr	P/F	Ns 0	I	X	X	Sequenced I-frame

Source: Datapro Research Corp. IBM Synchronous Data Link Control (SDLC). Delran, NJ 08075.

APPENDIX C

Z80-SIO READ AND WRITE REGISTER SPECIFICATIONS

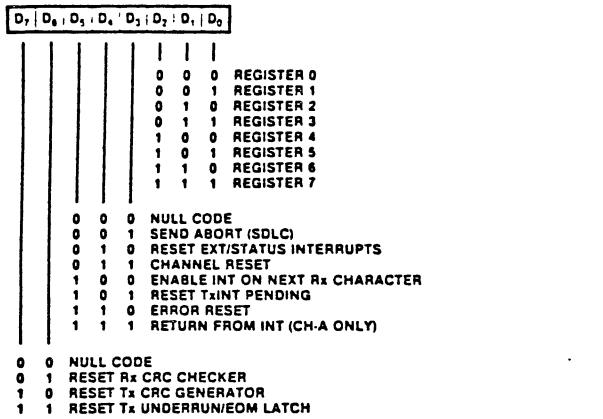
Read Register Bit Functions



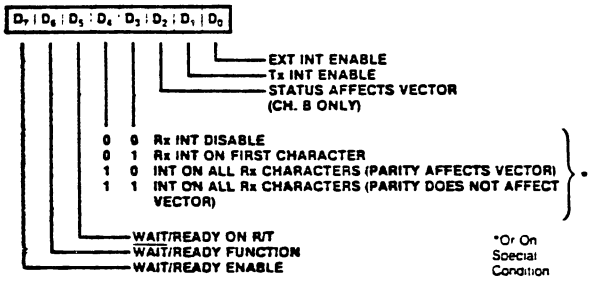
Source: Z80 SIO Technical Manual. Zilog Inc., Campbell, California, Feb 1978.

Write Register Bit Functions

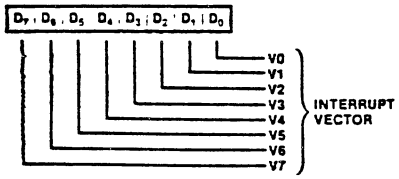
WRITE REGISTER 0



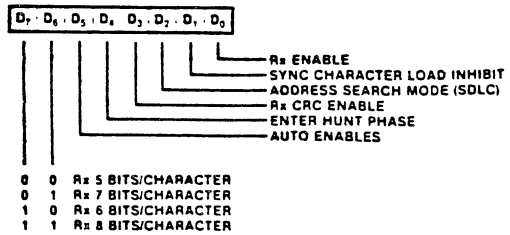
WRITE REGISTER 1



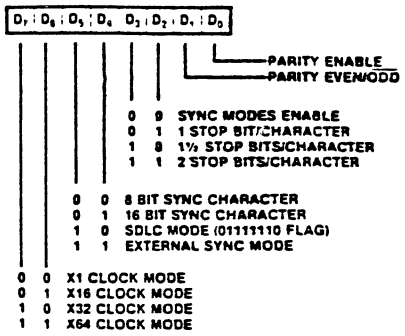
WRITE REGISTER 2 (CHANNEL B ONLY)



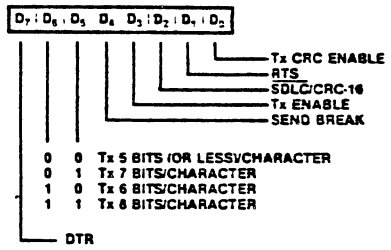
WRITE REGISTER 3



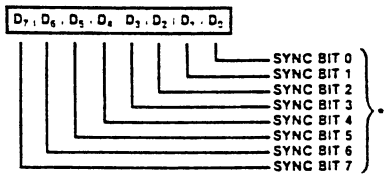
WRITE REGISTER 4



WRITE REGISTER 5

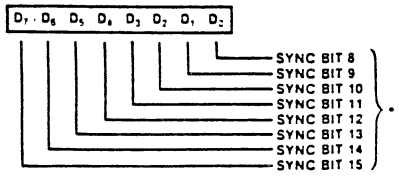


WRITE REGISTER 6



*Also SDLC Address Field

WRITE REGISTER 7



*For SDLC it Must Be Programmed to 01111110 For Flag Recognition

Source: Z80 SIO Technical Manual. Zilog, Inc., Campbell, California, 1978.

APPENDIX D

Z80-CTC TIME CONSTANTS FOR VARIOUS BAUD RATES

TABLE IX

Z80 TIME CONSTANTS FOR SELECTED

BAUD RATES

Baud Rates (bps)	CTC Time Constant
7812	256
9600	208
19,200	104
38,400	52
76,800	26
153,600	13
307,200	7
614,400	3
1,228,800	2
2,000,000	1

NOTE:
- Time Constant = 1 / (Baud rate x Clock period)
- System Clock Period = 500 x 10 secs.
- All Time Constants are rounded off to the nearest integer.

APPENDIX E

PIN CONNECTION DIAGRAMS

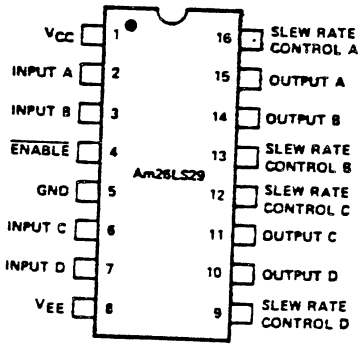
OF I.C.'s USED

Am26LS29 : Quad Three-State Single Ended RS-423 Line Driver

DISTINCTIVE CHARACTERISTICS

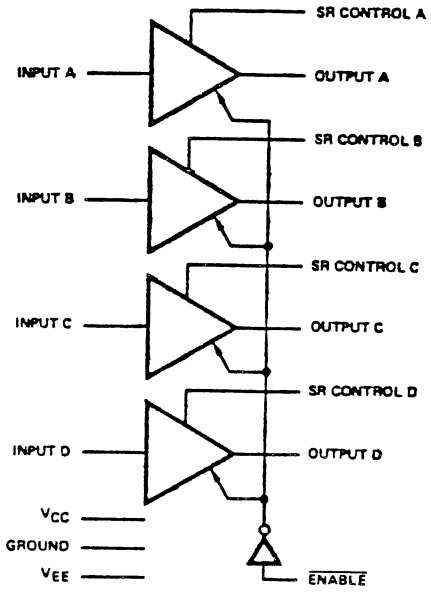
- Four single ended line drivers in one package for maximum package density
- Output short-circuit protection
- Individual rise time control for each output
- High capacitive load drive capability
- Low I_{CC} and I_{EE} power consumption (26mW/driver typ.)
- Meets all requirements of RS-423
- Three-state outputs for bus oriented systems
- Outputs do not clamp line with power off or in hi-impedance state over entire transmission line voltage range of RS-423
- Low current PNP inputs compatible with TTL, MOS and CMOS
- Available in military and commercial temperature range
- Advanced low power Schottky processing

CONNECTION DIAGRAM
Top View



Note: Pin 1 is marked for orientation.

LOGIC DIAGRAM

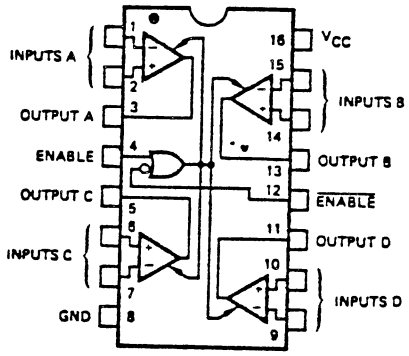


Am26LS32 : Quad Differential Line Receiver.

DISTINCTIVE CHARACTERISTICS

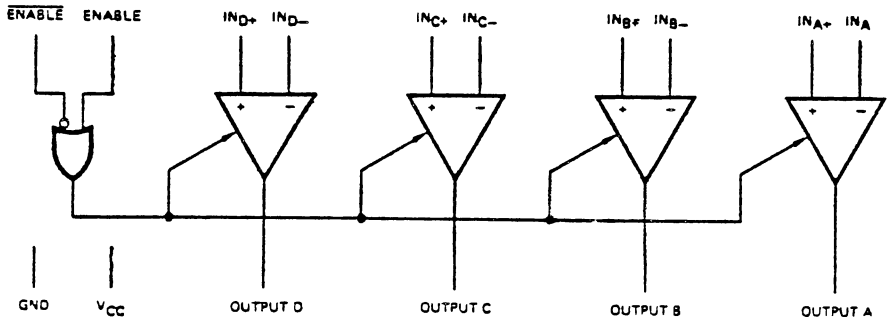
- Input voltage range of 15V (differential or common mode) on Am26LS33; 7V (differential or common mode) on Am26LS32
- ±0.2V sensitivity over the input voltage range on Am26LS32; ±0.5V sensitivity on Am26LS33
- The Am26LS32 meets all the requirements of RS-422 and RS-423
- 6k minimum input impedance
- 30mV input hysteresis
- Operation from single +5V supply
- 16-pin hermetic and molded DIP package
- Fail safe input-output relationship. Output always high when inputs are open.
- Three-state drive, with choice of complementary output enables, for receiving directly onto a data bus.
- Propagation delay 17ns typical
- Available in military and commercial temperature range
- Advanced low-power Schottky processing

CONNECTION DIAGRAM
Top View



Note: Pin 1 is marked for orientation.

LOGIC DIAGRAM



MC1488 : Quad MDTL Line Driver RS-232-C.

QUAD LINE DRIVER

The MC1488 is a monolithic quad line driver designed to interface data terminal equipment with data communications equipment in conformance with the specifications of EIA Standard No. RS 232C

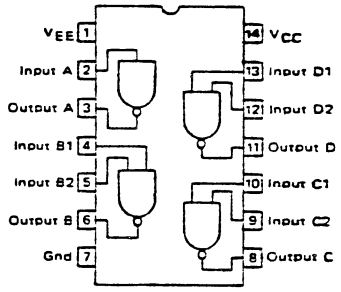
Features

- Current Limited Output
  ±10 mA typ
- Power-Off Source Impedance
  300 Ohms min
- Simple Slew Rate Control with External Capacitor
- Flexible Operating Supply Range
- Compatible with All Motorola MDTL and MTTL Logic Families

L SUFFIX
CERAMIC PACKAGE
CASE 632-02
MO-001AA

P SUFFIX
PLASTIC PACKAGE
CASE 646-05

PIN CONNECTIONS

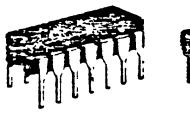


MC1489 : Quad MDTL Line Receiver RS-232-C.

QUAD LINE RECEIVERS

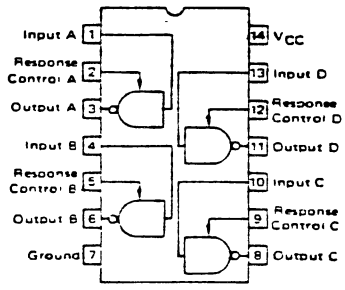
The MC1489 monolithic quad line receivers are designed to interface data terminal equipment with data communications equipment in conformance with the specifications of EIA Standard No. RS 232C

- Input Resistance – 3.0 k to 7.0 kilohms
- Input Signal Range – ± 30 Volts
- Input Threshold Hysteresis Built In
- Response Control
  a) Logic Threshold Shifting
  b) Input Noise Filtering



L SUFFIX
CERAMIC PACKAGE
CASE 632-02
MO-001AA

P SUFFIX
PLASTIC PACKAGE
CASE 646-05



APPENDIX F

SOURCE LISTING OF THE SDLC COMMUNICATIONS PROGRAM

```
;*************************************************
;*      SOFTWARE INTERFACE FOR MICROCOMPUTER      *
;*            COMMUNICATIONS USING                *
;*               SDLC  PROTOCOL                   *
;*_______________________________________________*
;*                                                *
;*             M . S .  THESIS  PROJECT           *
;*                                                *
;*  Author :  NARAYAN MAKARAM                     *
;*  Program : COMMF.MAC          April 1984       *
;*_______________________________________________*
;*                                                *
;*   This software interface package transfers a  *
;*  file from a primary station (microcomputer) to *
;*  a secondary station ( another microcomputer    *
;*  with an address = AA hex ).  The protocol used *
;*  for communications is Synchronous Data Link    *
;*  Control (SDLC).  The transfer rates for comm-  *
;*  unications available to the user are - 7812 bps,*
;*  9600 bps, 19.2 Kbps, 38.4 Kbps & 76.8 Kbps.    *
;*  The software uses interrupt processes to       *
;*  transmit and receive every character.          *
;*                                                 *
;*  USER PROMPTS AND RESPONSES -                   *
;*  ( User prompts are underlined & comments are   *
;*    enclosed in "/*" -- comments -- "*/". )      *
;*                                                 *
;*  1) A> COMMF                                    *
;*  2>    Baud Rates => 7812 9600 19.2K 38.4K 76.8K *
;*        CODES ======> 0    1     2     3     4   *
;*        Key in code =  1 /* 9600 baud selected */ *
;*        INITIALIZATION COMPLETED                 *
;*  3)    Transmit(T), Receive(R); T or R => T     *
;*      /* This example shows transmit responses */ *
;*      /* The next prompt is not displayed for  */ *
;*      /* the Receiver.                         */ *
;*  4)  TYPE <CR> IF RECEIVER STATION IS READY:<CR> *
;*      *****.........*****                        *
;* /* "*" displayed for every record transmitted */ *
;*  5)  >> COMPLETED TRANSMISSION >>               *
;*_______________________________________________*
;*                                                *
;*  Note: The receiver (secondary station) should  *
;*   initialized first ( steps 1-3 ), before the   *
;*   transmitter (primary station) is initialized. *
;*                                                 *
;*************************************************
;
;*************************************************
;*                                               *
;* Routine            Description                *
;* =======            ===========                *
;* INITVECT :  Initializes the interrupt vector  *
;*             table with service routine addresses.*
;* BAUDSEL  :  Prompts user to select baud rate. *
;* INITSIO  :  Initializes Z80 SIO to operate in *
;*             SDLC mode, and the Z80 CTC is set *
;*             to provide the clock signal for the *
;*             desired baud rate.                *
;* TRANSMIT :  Transmits an user selected file until*
```

```
;*            EOF is reached.                          *
;* TRNSFRM    Transmits one SDLC frame, which is       *
;*            equal to 1 record of 128 bytes.          *
;* TRNSINT  : Transmitter interrupt service routine*
;*            Transmits one byte at a time.            *
;* RECEIVE  : Receives a file and writes it on the     *
;*            the diskette. (user selects file name)
;* REC$FRM  : Receives one SDLC frame .                *
;* REC$INT  : Receiver interrupt service routine.      *
;*            Interrupts generated on every byte.      *
;* SPL$INT  : Special condition interrupt service      *
;*            routine. Detects End of Frame, CRC &     *
;*            Overrun Errors.                          *
;* BUFIN    : Inputs file name from console buffer     *
;*            and moves it into the File Control       *
;*            block (FCB).                             *
;*                                                     *
;*                                                     *
;*******************************************************

          .Z80

RBOOT     EQU     0000H       ; system reboot address.
BDOS      EQU     0005H       ; bdos entry point.
CONIN     EQU     1           ; console input function.
CONOUT    EQU     2           ; console output function.
PRINTF    EQU     9           ; print string until '$'.
RDBUF     EQU     10          ; read console buffer function.
OPENF     EQU     15          ; file open function.
CLOSEF    EQU     16          ; file close function.
READF     EQU     20          ; Sequential read.
WRITEF    EQU     21          ; Sequential write.
MAKEF     EQU     22          ; Make file function.
SETDMA    EQU     26          ; Set up DMA address.

SFCB      EQU     005CH       ; Source file control block.
SFCBCR    EQU     SFCB+32     ; Current record number.
TBUFF     EQU     0080H       ; Transmit buffer.
RBUFF     EQU     0080H       ; Receive buffer.
CR        EQU     0DH         ; Carraige return.
LF        EQU     0AH         ; Line feed.
;-------- SIO PORT ADDRESSES ------------------------------
DATA$A    EQU     0F4H        ; Channel A Data port.
DATA$B    EQU     0F5H        ; Channel B Data port.
STAT$A    EQU     0F6H        ; Channel A Status port.
STAT$B    EQU     0F7H        ; Channel B Status port.
COMD$A    EQU     0F6H        ; Channel A Command port.
COMD$B    EQU     0F7H        ; Channel B Command port.
;-------- CTC PORT ADDRESSES ------------------------------
CTC0      EQU     0F0H        ; CTC Channel 0.
CTC1      EQU     0F1H        ; CTC Channel 1.
CTCW      EQU     55H         ; CTC Control word.
;------- CONTROL WORDS FOR SIO----------------------------
WR3       EQU     03H         ; *** WRITE REGISTER 3 ****
CW3       EQU     0E9H        ; Enable receive & CRC ,
                              ; Enter Flag hunt mode.
ERR$RS0   EQU     33H         ; Reset overrun & CRC bits.
REC$DS3   EQU     0E8H        ; Disable receive logic.
;
WR5       EQU     15H         ; *** WRITE REGISTER 5 *****
CW5       EQU     0E0H        ; Disable Xmit logic & CRC.
TRN$RS5   EQU     2DH         ; Reset Xmit Int. Pending & select WR5.
TRN$EN5   EQU     0EBH        ; Enable Xmit logic & CRC.
RS$CRC0   EQU     80H         ; RESET TX CRC TO OFFH FOR SDLC PROTOCOL.
UDR$RS0   EQU     0C0H        ; Reset Xmit Underrun & EOM interrupts.
TRN$DI0   EQU     28H         ; Reset Xmit Interrupt Pending.
ABT$WR0   EQU     08H         , Transmit ABORT (SDLC).
```

```
;-------------------------------------------------------
SECSAD      EQU     0AAH        ; Secondary station address.
REC$SZ      EQU     130         ; Information Frame size = 130 bytes.
;------- Interrupt Vector Addresses -------------------------
RECV        EQU     015CH       ; Receive interrupt vector addr.
TRAN        EQU     0158H       ; Transmit interrupt vector addr.
EXTN        EQU     015AH       ; External interrupt vector addr.
SPCL        EQU     015EH       ; Special rec. condition vector addr.
            ORG     100H        ; Beginning of TPA.
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
USER:       LD      SP,STACK    ; initialize stack pointer.
            CALL    INITVECT    ; initialize Interrupt Vector Table.
            CALL    BAUDSEL     ; select desired baud rate ( time const.)
            CALL    INITSIO     ; initialize Z80 SIO Channel A.
;
RESTART.    LD      DE,TRN$REC  ; prompt user to receive/transmit.
            CALL    DISPLAY
            CALL    CI          ; get keyboard input.
;
            CP      'T'         ; does user want to transmit ?
            JP      Z,TRANS     ; YES,then jump.
            CP      'R'         ; does user want to receive ?
            JP      Z,RECVE     ; YES, then jump.
            JP      REBOOT      ; return to system.
;
TRANS:      CALL    TRANSMIT    ; Call TRANSMIT function.
            JP      REBOOT
RECVE:      CALL    RECEIVE     ; Call RECEIVE function.
            JP      REBOOT
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$     I N I T V E C T   R O U T I N E          $
;$  This module initializes the INTERRUPT       $
;$  VECTOR TABLE to the corresponding addresses  $
;$  of the following interrupt service routines  $
;$                                               $
;$  INTERRUPT SERVICE ROUTINE   VECTOR ADDRESS   $
;$     1. Receive               005CH            $
;$     2. Transmit              0058H            $
;$     3. Special rec. condition 005EH           $
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
INITVECT:   PUSH    HL
            LD      A,01H       ; INTERRUPT REG.= High address byte of
            LD      I,A         ;                 interrupt VECTOR TABLE.
;
            LD      HL,TRN$INT  ; HL = Addr. of TRANSMIT interrupt routine.
            LD      (TRAN),HL   ; Store it in VECTOR TABLE.
            LD      HL,REC$INT  ; HL = Addr. of RECEIVE interrupt routine.
            LD      (RECV),HL   ; Store it in VECTOR TABLE.
            LD      HL,SPL$INT  ; HL = Addr. of SPECIAL RECEIVE CONDITION
                                ;      interrupt routine.
            LD      (SPCL),HL   ; Store it in VECTOR TABLE.
;
            IM      2           ; Set Z80 to MODE 2 interrupts.
            POP     HL
            RET
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$     I N I T S I O   R O U T I N E          $
;$ This module initializes the Z80 SIO to operate$
;$ in SDLC mode. The desired baud rate is set    $
;$ by programming the CTC.                       $
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
INITSIO:    DI
            PUSH    AF
            PUSH    BC
```

```
        PUSH    DE
        PUSH    HL
;** Initialize SIO channel A ****
        LD      B,13            ; Number of control words in the table.
        LD      C,COMD$A        ; Output to COMMAND REG.( Chan.A )
        LD      HL,INITAB       ; HL -> table of control words.
        OTIR                    ; Block output to SIO Write Regs.
;*** Initialize SIO channel B (INTERRUPT VECTOR REG>) ***
        LD      B,5             ; Number of control words.
        LD      C,COMD$B        ; Output to COMMAND REG.( Chan.B )
        LD      HL,INTAB$       ; HL -> table of control words.
        OTIR                    ; Block output to SIO write regs.
;*** Program CTC for proper BAUD RATE ***
        LD      A,CTCW          ; CTC control word.
        OUT     (CTC1),A        ; Output to CTC Chan.1.
        LD      A,(TCONST)      ; CTC Time Constant.
        OUT     (CTC1),A        ; Output to CTC Chan.1.
;*** Display initialization completed message ***
        LD      DE,INITMES
        CALL    DISPLAY
        POP     HL
        POP     DE
        POP     BC
        POP     AF
        EI
        RET
;----------**** SIO Initialization table ****----------
INITAB: DEFB    18H             ; Reset SIO Channel A.
        DEFB    14H             ; Select WR4.
        DEFB    20H             , SDLC mode, x1 clock,
                                ; disable parity.
        DEFB    13H             , Select WR3.
        DEFB    0E8H            ; 8 bits/char, Receive SDLC frame,
                                ; Disable receive logic .
                                ; Enable Frame Hunt mode & CRC.
                                ; Enable DCD & CTS ( Autoenable )
        DEFB    15H             ; Select WR5.
        DEFB    0E0H            ; 8 bits/char, select SDLC polynomial,
                                ; Disable Xmit logic & CRC.
                                ; Enable DTR & RTS.
        DEFB    06H             ; Select WR6.
        DEFB    0AAH            ; Secondary station address = AA.
        DEFB    07H             ; Select WR7.
        DEFB    7EH             ; FLAG = 7EH.
        DEFB    11H             ; Select WR1
        DEFB    16H             ; Enable vectored interrupts
                                ; Enable Xmit  & receive interrupts.
INTAB$: DEFB    18H             ; Reset SIO channel B.
        DEFB    02H             ; Select WR2.
        DEFB    50H             ; Low address byte.
        DEFB    11H             ; Select WR1.
        DEFB    16H             ; Enable vectored interrupts.
;-----------------------------------------------------------------
;***********************************************
;*      T R A N S M I T   R O U T I N E        *
;*  Transmits a file from this primary station  *
;*  (microcomputer) to a secondary station (an- *
;*  other microcomputer) using SDLC protocol.   *
;*  At a time the Infomation Frame consists of  *
;*  only 1 record ( 128 bytes ).                *
;***********************************************
TRANSMIT: PUSH  BC
        PUSH    DE
        PUSH    HL
```

```
;-------------------------------------------------------
;** Set up DMA address for READ ( for transmit ) **
          LD        C,SETDMA                ; Set DMA address.
          LD        DE,TBUFF                ; DE -> transmit buffer.
          CALL      BDOS
;-------------------------------------------------
;** Open the file to be transmitted ***
          LD        DE,TRNFILE              ; display message to accept
          CALL      DISPLAY                 ; Source file name.
          CALL      BUFIN                   ; Input string from buffer.
;
          CALL      OPEN                    ; Open source file.
          INC       A                       ; A = status of open.
          JP        NZ,RDY$XT               ; A <> 0, then OK.
          LD        DE,NOFILE               ;         else ERROR.
          CALL      DISPLAY                 ; display error message.
          JP        REBOOT                  ; return to system.
;-------------------------------------------------
;** Verify whether the RECEIVING STATION is ready **
RDY$XT:   LD        DE,VER$MSG              ; display verify message.
          CALL      DISPLAY
          CALL      CI                      ; Accept any character.
          CALL      CRLF                    ; Carraige Return, Line Feed.
;-----------------------------------------------------------------
;**** Loop to transmit a 128 byte record ***
NXT$REC:  LD        DE,SFCB                 ; DE-> File Control Block.
          CALL      READ                    ; Read 1 record from file.
          OR        A                       ; A = status of read.
          JP        NZ,LAST                 ; If A <> 0, then EOF reached.
;--- Initialize buffer pointer and control bytes --
          LD        A,RECSSZ                ; Record Size = 130.
          LD        C,0FH                   , C = control byte.
          JP        ENSTRN
;
LAST:     LD        A,2                     ; Record size of last rec.= 2.
          LD        C,00H                   ; C = control byte = 00.
;--- Transmit 1 SDLC frame to SECONDARY STATION ---
ENSTRN:   CALL      TRN$FRM                 ; Call transmit Frame.
;--- Check if last frame sent was EOF frame -------
          LD        A,(TBUFF-1)             ; Is control byte = 00h ?
          CP        0
          JP        Z,TRN$EOF               ; YES, then exit loop.
          JP        NXT$REC
;-----------------------------------------------------------------------
;*** End of File reached ***
TRN$EOF:  LD        DE,SFCB                 ; Close source file.
          CALL      CLOSE
          INC       A                       ; A = status of file.
          JP        Z,F$ERR                 ; If A= 0, ERROR during Close.
;**** transmission completed *****
          LD        DE,CMP$TRN              ; Display completed Xmit msg.
          CALL      DISPLAY
          POP       HL
          POP       DE
          POP       BC
          RET
F$ERR:    LD        DE,ERR$MSG              ; display Error message.
          CALL      DISPLAY
          JP        REBOOT
;*********************************************
;*       TRANSMIT FRAME ROUTINE              *
;*********************************************
TRN$FRM:  PUSH      BC
          PUSH      DE
```

```
          PUSH      HL
;---- Initialize COUNT & buffer pointer ------
          LD        (COUNT),A
          LD        HL,TBUFF-1          ; HL -> transmit buffer addr.
          LD        (TRN$PTR),HL
          LD        (HL),C              ; Store CONTROL BYTE.
          LD        A,0
          LD        (TRN$FLG),A         ; TRANSMIT FLAG = 0.
;------ Wait for CTS to be active ------------
WAIT$TR:  LD        A,10H               ; Wait for Receiver to get ready.
          OUT       (COMD$A),A
          IN        A,(STAT$A)          ; Check if CTS is active.
          BIT       5,A
          JP        Z,WAIT$TR
;---- Enable transmitter ----------------------
          LD        A,WR5               ; Select WR5.
          OUT       (COMD$A),A
          LD        A,TRNSEN5           ; Enable Xmit logic.
          OUT       (COMD$A),A
          LD        A,RS$CRC0           ; Reset Xmit CRC generator.
          OUT       (COMD$A),A
;---- Delay loop ------------------------------
          LD        C,0FFH
WAIT$LP:  DEC       C
          JP        NZ,WAIT$LP
;---- Send address byte -----------------------
          LD        A,SEC$AD            ; output Addr. of sec.station.
          OUT       (DATA$A),A
          LD        A,UDR$RS0           ; Reset Xmit Underrun/EOM latch.
          OUT       (COMD$A),A
;---- Wait for End Of Frame -------------------
TRN$WT:   IN        A,(STAT$A)          ; Input Chan.A status.
          BIT       6,A                 ; End of Message interrupt ?
          JP        NZ,D$XMIT           ; Yes, then jump.
          BIT       7,A                 ; Abort received ?
          JP        NZ,AB$REC           ; Yes, then jump.
          JP        TRN$WT
;----- Disable transmitter logic --------------
D$XMIT:   LD        A,1                 ; Transmit flag = 1.
          LD        (TRN$FLG),A
          LD        A,'*'               ; Display * for each Rec. Xmitted
          CALL      CO
          JP        OV$FRM
;----- ABORT RECEIVED --------------------------
AB$REC:   LD        DE,ABTMES           ; Display ABORT message.
          CALL      DISPLAY
          JP        REBOOT
;
OV$FRM:   LD        BC,0000H            ; Delay loop.
WAITF:    DEC       BC
          LD        A,C
          OR        B
          JP        NZ,WAITF
          POP       HL
          POP       DE
          POP       BC
          RET
;***********************************************
;*     TRANSMIT INTERRUPT SERVICE ROUTINE      *
;***********************************************
TRN$INT:  PUSH      AF
          PUSH      BC
          PUSH      DE
          PUSH      IX
```

```
;----- Check if End of I-Frame ? ----------------
          LD        A,(TRN$FLG)             ; Is TRN$FLG = 1 ?
          CP        1
          JP        Z,DS$TRN                ; YES, then disable Xmitter.
;--- Update BYTE COUNT -------------------------
          LD        A,(COUNT)               ; decrement byte count.
          DEC       A
          LD        (COUNT),A               ; Restore count.
          JP        Z,RS$TRN                ; Count = 0, then jump.
;-------------------------------------------
;** Transmit next byte from Buffer **
          LD        IX,(TRN$PTR)            ; IX -> Transmit buffer.
          LD        A,(IX)                  ; get next byte.
          OUT       (DATA$A),A              ; Output next character.
          INC       IX                      ; Increment buffer pointer.
          LD        (TRN$PTR),IX
          JP        OV$TRN
;-------------------------------------------
;** Disable transmitter on End of Message **
DS$TRN:   LD        A,TRN$RS5               ; Disable Transmitter.
          OUT       (COMD$A),A
          LD        A,CW5
          OUT       (COMD$A),A
          JP        OVSTRN
;-------------------------------------------
;** Reset Transmit interrupt pending ****
RS$TRN:   LD        A,TRN$DIO               ; disable further interrupts.
          OUT       (COMD$A),A
;
OVSTRN:   POP       IX
          POP       DE
          POP       BC
          POP       AF
          EI
          RETI
;*********************************************
;*      R E C E I V E   R O U T I N E        *
;* This routine receives a file from a primary *
;* station and creates a destination file in   *
;* the secondary station (on floppy disk).     *
;*********************************************
RECEIVE:  PUSH      DE
          PUSH      HL
;
;-------------------------------------------
;*** Open file to Receive records ***
          LD        DE,RECFILE              ; display user prompt message.
          CALL      DISPLAY
          CALL      BUFIN                   ; input file name.
          CALL      MAKE                    ; Create file on disk.
          INC       A                       ; check status.
          JP        NZ,REC$RDY              ; A <> 0, Good Status.
          LD        DE,DSK$FUL              ; A = 0, Disk is FULL.
          CALL      DISPLAY
          JP        REBOOT
;----- Set up DMA address to WRITE ONTO DISK ( for Receive) ---
REC$RDY:  CALL      CRLF
          LD        C,SETDMA
          LD        DE,RBUFF
          CALL      BDOS
;-----------------------------------------------------------
;**** Loop to receive a 128 byte record ***
REC$NXT:  CALL      REC$FRM                 ; Receive another Frame.
;
```

```
;---- Check if frame received is EOF frame -----
          LD        A,(RBUFF-1)             ; check if Info frame recvd is
                                            ; EOF record ?
          CP        0
          JP        Z,RECSEOF               ; YES, then Close file.
;---- Write the received record on Diskette ----
          LD        DE,SFCB                 ; Write record on disk.
          CALL      WRITE
          OR        A                       ; Check status.
          JP        NZ,WRSERR
          JP        RECSNXT                 ; receive next Rec.
;-----------------------------------------------------------------
RECSEOF:  LD        DE,SFCB                 ; Close receive file.
          CALL      CLOSE
          INC       A
          JP        Z,FLSERR                ; jump on error.
          LD        DE,CMPSREC              ; display completion message.
          CALL      DISPLAY
          POP       HL
          POP       DE
          RET
;
WRSERR:   LD        DE,DSKSFUL              ; Disk is full.
          CALL      DISPLAY
          JP        REBOOT
;** Error during CLOSE FILE ***
FLSERR:   LD        DE,ERRSMSG              ; display error message.
          CALL      DISPLAY
          JP        REBOOT
;************************************************
;*     RECEIVE FRAME FUNCTION                   *
;************************************************
REC$FRM:  PUSH      AF
          PUSH      BC
          PUSH      DE
          PUSH      HL
;-- Initialize BUFFER POINTER ------------------
          LD        HL,RBUFF-2              ; HL -> Receive buffer
          LD        (REC$PTR),HL            ; Buffer Pointer = HL.
          LD        A,0                     ; RECEIVE FLAG = 0.
          LD        (REC$FLG),A
          LD        (STATUS),A              ; STATUS = 0.
;-----------------------------------------------
;** Enable Transmitter Station **
          LD        A,WR5                   ; Select WR5.
          OUT       (COMD$A),A
          LD        A,0E2H                  ; Enable DTR & RTS.
          OUT       (COMD$A),A
;-----------------------------------------------
;** Enable FRAME HUNT mode for receive **
          LD        A,WR3                   ; address Write Reg.3
          OUT       (COMD$A),A
          LD        A,CW3                   ; disable Frame Hunt mode.
          OUT       (COMD$A),A
;-----------------------------------------------
;** Wait for End of FRAME interrupt ***
EOM$WT:   LD        A,(STATUS)              ; check STATUS of RECEIVE.
          CP        0                       ; GOOD status ?
          JP        NZ,ABRTR                ; NO, then abort receive.
;
          LD        A,(REC$FLG)             ; Check if Receive Flag= 1?
          CP        1
          JP        NZ,EOM$WT               ; NO, then wait.
;-----------------------------------------------
```

```
;** Disable Transmitter Station ***
          LD        A,WR5                   ; Select WR5.
          OUT       (COMD$A),A
          LD        A,CW5                   ; Disable RTS.
          OUT       (COMD$A),A
;
          POP       HL
          POP       DE
          POP       BC
          POP       AF
          RET
;---- Display ABORT message --------------------
ABRTR:    LD        DE,ABTMES
          CALL      DISPLAY
          JP        REBOOT
;************************************************
;*     RECEIVE INTERRUPT SERVICE ROUTINE        *
;************************************************
REC$INT:  PUSH      AF
          PUSH      BC
          PUSH      DE
          PUSH      IX
;
;         LD        A,'R'
;         CALL      CO
;-----------------------------------------------
;** Check if a character has been received **
CHK$REC:  IN        A,(STAT$A)              ; Input RR0 channel A.
          BIT       0,A                     ; Bit 0 = 1 ?
          JP        NZ,REC$CHR              ; YES, then Char. received.
          BIT       7,A                     ; Bit 7 = 1 ?
          JP        NZ,ABORT                ; Yes, then ABORT received.
          LD        DE,UNSERR
          CALL      DISPLAY
          JP        OV$REC
;** Input character **-------------------------
REC$CHR:  LD        IX,(REC$PTR)            ; IX -> receive buffer ptr.
          IN        A,(DATA$A)              ; Input DATA BYTE.
          LD        (IX),A                  ; store it in recv. buffer.
          INC       IX                      ; Increment buffer ptr.
          LD        (REC$PTR),IX            ; Restore buffer ptr.
;
OV$REC:   POP       IX
          POP       DE
          POP       BC
          POP       AF
          EI
          RETI
;
ABORT:    LD        A,'@'
          CALL      CO
          JP        REBOOT
;************************************************
;*   SPECIAL RECEIVE CONDITION INTERRUPT        *
;************************************************
SPL$INT:  PUSH      AF
          PUSH      BC
          PUSH      DE
          PUSH      IX
;         LD        A,'S'
;         CALL      CO
;** Check Channel A status ( READ REG.1 ) ***
          LD        A,1                     ; address READ  Reg.1.
          OUT       (COMD$A),A
```

```
        IN      A,(STAT$A)              ; Input Chan.A status.
        BIT     7,A                     ; Is RR1 bit 7 = 1 ?
        JP      NZ,EOF$RC               ; Yes,then End of Frame.
        BIT     5,A                     ; Is RR1 bit 5 = 1 ?
        JP      NZ,OVR$ERR              ; Yes, then OVERRUN error.
        JP      OV$SPL
;-----------------------------------------------
;** End of Frame received ***
EOF$RC: BIT     6,A                     ; Is RR1 bit 6 = 1 ?
        JP      NZ,CRC$ERR              ; YES, Then CRC error.
        IN      A,(DATA$A)              ; Input CRC2 byte.
        LD      A,ERR$RS0               ; Reset underrun interrupts.
        OUT     (COMD$A),A
        LD      A,REC$DS3               ; disable receiver logic.
        OUT     (COMD$A),A
        LD      A,1                     ; set receive flag.
        LD      (REC$FLG),A
        LD      A,'*'
        CALL    CO
        JP      OV$SPL
;-----------------------------------------------
;** CRC Error detected ***
CRC$ERR: LD     A,40H                   ; Reset Rx CRC checker.
        OUT     (COMD$A),A
        LD      DE,CRCMES               ; display CRC error message.
        CALL    DISPLAY
        LD      A,31H                   ; set STATUS = '1'
        LD      (STATUS),A
        JP      OV$SPL
;-----------------------------------------------
,** Overrun Error ***
OVR$ERR: LD     A,ERR$RS0               ; Reset overrun error.
        OUT     (COMD$A),A
        LD      A,REC$DS3               ; Disable eceive logic.
        OUT     (COMD$A),A
        LD      DE,OVRMES
        CALL    DISPLAY
        LD      A,32H
        LD      (STATUS),A
;
OV$SPL: POP     IX
        POP     DE
        POP     BC
        POP     AF
        EI
        RETI
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$     B A U D   R A T E   S E L E C T           $
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
BAUDSEL: PUSH   DE
        PUSH    HL
;
        LD      HL,CONST                ; Display available baud rates.
        LD      DE,BD$MES
        CALL    DISPLAY
;
        CALL    CI                      ;accept CODE for Baud selected.
        AND     0FH                     ; Convert ASCII to BCD.
        LD      DE,0
        LD      E,A                     ;DE -> Index into T const.table.
        ADD     HL,DE                   ; HL -> Time constant entry.
        LD      A,(HL)                  ; A= Time constant for Baud rate
        LD      (TCONST),A              ; Save the Time constant.
;
```

```
          POP       HL
          POP       DE
          RET
,$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$;
;$     D I S P L A Y  S U B R O U T I N E     $;
;$  DISPLAYS CONTENTS OF BUFFER ADDRESSED BY   $;
;$  D,E REGISTER PAIR UPTO '$' SIGN.            $;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$;
DISPLAY:  PUSH      AF
          PUSH      BC
          PUSH      HL
          LD        C,PRINTF
          CALL      BDOS
          POP       HL
          POP       BC
          POP       AF
          RET

;**********  CONSOLE INPUT    *************;
CI:       PUSH      HL
          PUSH      DE
          PUSH      BC
          LD        C,CONIN
          CALL      BDOS
          POP       BC
          POP       DE
          POP       HL
          RET
;**********  CONSOLE OUTPUT   *************;
CO        PUSH      BC
          PUSH      DE
          LD        C,CONOUT
          LD        E,A
          CALL      BDOS
          POP       DE
          POP       BC
          RET
;**********  CRLF            **************;
CRLF:     LD        A,CR
          CALL      CO
          LD        A,LF
          CALL      CO
          RET
;**********  READ RECORD SEQUENTIALLY  ****;
READ:     LD        C,READF
          JP        BDOS
;**********  WRITE RECORO SEQUENTIALLY ****;
WRITE:    LD        C,WRITEF
          JP        BDOS
;**********  OPEN  FILE          ***********;
OPEN:     LD        DE,SFCB
          LD        C,OPENF
          JP        BDOS
;**********  CLOSE  FILE         ***********;
CLOSE:    LD        C,CLOSEF
          JP        BDOS
;**********  MAKE  FILE          ***********;
MAKE:     XOR       A
          LD        (SFCBCR),A
          LD        DE,SFCB
          LD        C,MAKEF
          JP        BDOS
;********** INIT FCB WITH FILE NAME **********;
BUFIN.    PUSH      BC
```

```
            PUSH    DE
            PUSH    HL
;
RETRY:      LD      DE,CNBUF
            LD      C,RDBUF
            CALL    BDOS
;--- ZERO OUT DRIVE CODE IN FCB -----
            LD      A,0
            LD      (SFCB),A
;* Move file name from console buffer to FCB **;
            LD      HL,CNLIN
            LD      DE,SFCB+1
            LD      BC,8
MOVBLK:     LDI
            LD      A,(HL)
            CP      '.'
            JP      Z,OUTLP
            LD      A,0
            CP      C
            JP      NZ,MOVBLK
            LD      DE,LONG
            CALL    DISPLAY
            JP      RETRY
,* Pad rest of the file name chars with blanks **
OUTLP:      LD      A,20H
            LD      (DE),A
            INC     DE
            DEC     C
            JP      NZ,OUTLP
;
            INC     HL
            LD      BC,3
            LDIR
;* zero out the current extent number    ******;
            LD      A,0
            LD      (DE),A
            POP     HL
            POP     DE
            POP     BC
            RET
;***** MESSAGES TO THE CONSOLE *****
TRN$REC:    DEFB    0AH,0DH,'TRANSMIT (T), RECEIVE (R); TYPE T or R=> $'
TRNFILE:    DEFB    0AH,0DH,'NAME OF THE SOURCE FILE => $'
RECFILE:    DEFB    0AH,0DH,'NAME OF THE DESTINATION FILE => $'
LONG:       DEFB    0AH,0DH,'FILE NAME > 8 CHARS;**  ENTER AGAIN => $'
NOFILE:     DEFB    0AH,0DH,'SORRY, FILE NOT FOUND!$'
DSK$FUL:    DEFB    0AH,0DH,'SORRY, DISK IS FULL !$'
ERR$MSG:    DEFB    0AH,0DH,'** ERROR ** DURING CLOSE FILE.$'
INITMES:    DEFB    0AH,0DH,'INITIALIZATION COMPLETED$'
VER$MSG:    DEFB    0AH,0DH,'TYPE <CR> IF RECEIVER STATION IS READY :$'
ST$TRN:     DEFB    0AH,0DH,'START OF TRANSMISSION.$'
ST$REC:     DEFB    0AH,0DH,'START OF RECEIVE.$'
CMP$TRN:    DEFB    0AH,0DH,'>>COMPLETED TRANSMITTING FILE.$'
CMP$REC:    DEFB    0AH,0DH,'>>COMPLETED RECEIVING FILE.$'
TRN$ERR:    DEFB    0AH,0DH,'** ERROR ** DURING TRANSMIT $'
REC$ERR:    DEFB    0AH,0DH,'** ERROR ** DURING RECEIVE $'
EOMMES:     DEFB    0AH,0DH,'END OF INFORMATION FRAME.$';
OVRMES:     DEFB    0AH,0DH,'OVERRUN ERROR,ABORT !$'
CRCMES:     DEFB    0AH,0DH,'CRC ERROR,ABORT !$'
ABTMES:     DEFB    0AH,0DH,'ABORT CHARACTER DETECTED$'
EXT$ERR:    DEFB    0AH,0DH,'** ERROR ** External Interrupt $'
UNSERR:     DEFB    0AH,0DH,'Unknown ERROR $'
CONST:      DEFB    0,208,104,52,26,13
BDSMES:     DEFB    0AH,0DH,'BAUD RATES => 7812  9600 19.2K 38.4K 76.8K 153.6K'
```

```
          DEFB    0AH,0DH,'CODES ------>  0    1    2    3    4    5'
          DEFB    0AH,0DH,'Key in code => $'
;
TCONST:   DEFS    1
TRN$FLG:  DEFS    1
REC$FLG:  DEFS    1
STATUS:   DEFS    1
TRN$PTR:  DEFS    2
REC$PTR:  DEFS    2
COUNT:    DEFS    1
;** CONSOLE INPUT BUFFER ***
CNBUF:    DEFB    CNLEN          ; max. length of console buffer.
CNSIZ:    DEFS    1              ; resulting size after read.
CNLIN:    DEFS    32             ; length of busser= 32 bytes.
CNLEN     EQU     $-CNSIZ        ;
          DEFS    50
STACK:
          END
```

APPENDIX G

GLOSSARY

TERM GLOSSARY

Term	Description

ACK
Acknowledge character. A transmission control character used as a response in Byte-oriented protocols like Async and BSC.

ADCCP
Advanced Data Communications Control Procedure developed by ANSI. It is a bit-oriented protocol.

Address
The location of a terminal, a peripheral device, a node, or any other unit or component in a network.

ANSI
American National Standards Institute. Accepted and proposed standards include transmission code and protocol- ASCII, ADCCP, and high level languages such as FORTRAN & COBOL.

ARPANET
A computer communications network developed by the Department of Defense Advanced Research Projects Agency in the early 1970's. It used the public switching network to interconnect many different computers at locations across the United States.

ASCII
American Standard Code for Information Interchange. The standard code consists of 7-bit coded characters (8 bits including parity check), used for information transfer among data processing systems, and data communication systems.

Asynchronous Transmission
Transmission in which each information character is individually synchronized, usually by means of start and stop bits.

Attenuation
The difference (loss) between transmitted and received power due to transmission loss through equipment, lines or other communications devices.

Bandwidth
The range of frequencies that can pass over a given circuit. Generally, the greater the bandwidth, the more information that can be sent through the circuit in a given amount of time.

Baud
Roughly equivalent to bits per second. A unit of measuring the signalling speed (data rate).

BDLC
Burroughs Data Link Control, a Bit-oriented data link control protocol.

Binary Synchronous (BSC/ BISYNC)
A half duplex, character-oriented synchronous data communications protocol developed by IBM in 1964.

bps (Mbps, Kbps)
Bits per second- defines the data communications speed. Mbps - Mega bits per second = 10 bps.
Kbps - Kilo bits per second = 10 bps.

Broadcast
Transmission to multiple receiving locations simultaneously.

CCITT
Consultive Committee International for Telephony and Telegraphy. An advisory committee established under the United Nations to recommend world wide standards.

Channel
1) A path for electrical transmission. Also called a circuit, line, link, or path.
2) A specific and discrete bandwidth allocation in the radio frequency spectrum utilized to transmit information signals.

Circuit Switching
A switching technique in which an information path (ie. circuit) between calling and called stations is established on demand, for exclusive use by the connected parties until the connection is released. (e.g. telephone).

CRC (Cyclic Redundancy Check)
An error-checking algorithm which is included in a packet, during transmission. The receiver checks the CRC on each packet it receives and strips it off before giving the packet to the destination station.

DCE (Data Circuit Terminating Equipment)
Equipment installed at the users premises which provides all the functions required to establish, maintain and terminate a connection, and provides a signal conversion and coding between the data terminal equipment and common carrier's line. e.g. modem.

DTE (Data Terminal Equipment)
User equipment. The end-user machine (terminal, computer, controller, etc.) which plugs into a unit which is the the termination point of the communications equipment (DCE).

EIA (Electronics Industries Association)
The US national organization of electronic manufacturers. It is responsible for the development and maintanence of industry standards for the interface between data processing machines and data communications equipment.

Ethernet
A local area network specification developed jointly by Xerox Corporation, and Digital Equipment Corporation to interconnect computer equipments using a coaxial cable and "transceivers".

Full Duplex
Simultaneous two way communications.

Half Duplex
Alternate two way communications.

HDLC
High Level Data Link Control - a bit-oriented protocol developed by ISO (International Standards Organization).

ISO Reference Model
A standard approach to network architecture which introduces modularity by dividing the complex set of functions into more manageable, self-contained, functional slices.

Interface
The hardware and (sometimes) software link between two separate devices.

Leased Line
A line rented excusively to one customer.

Local Area Network
A network that is located in a localized geographical area (e.g. an office, building, or campus), and whose communications technology provides a high-bandwidth, low-cost medium to which many nodes can be connected.

Mark
The signal (communications channel state) corresponding to a binary one. The marking condition exists when voltage is between -3V to -15V (RS232 standard).

Modem

A contraction of modulate and demodulate; a conversion device installed in pairs at each end of an analog communications line. The modem at the transmitter end, modulates digital signals received locally from the computer or terminal; the modem at the receiver end, demodulates the incoming analog signals, converting it back to its original (digital) format, and passes it to the destination device.

Packet

A collection of bits that contain both control information and data. A SDLC frame. A packet can be of fixed or variable length, but generally has a specified maximum length.

Polling

A method of controlling the sequence of transmission by devices on a multipoint line, by requiring each device to wait until the controlling processor requests it to transmit.

Port

The entrance or physical access point to a computer, device, or a network where signals may be supplied, extracted or observed.

Primary Station

Transmitter, master node in the network which controls data flow to secondary stations (receivers).

Protocol

A set of rules and conventions that govern the orderly and meaningful exchange of information between or among communications parties. Both hardware protocols and software protocols can be defined.

Remote Station

Data terminal equipment for communication with a data processing system in a distant location.

RJE

Remote Job Entry.

RS-232-C

A technical specification published by the EIA that specifies the mechanical and electrical characteristics of the interface for connecting DTE and DCE. The physical connection is made through a 25 pin connector. Data transmission speeds up to 20 Kbps can be achieved in full or half-duplex modes.

RS-422
An EIA standard that specifies electrical characteristics for balanced circuits (circuits with their own ground leads).

RS-423
A standard that specifies the electrical characteristics for unbalanced circuits (circuits using common or shared grounding techniques).

RS-449
General purpose 37-position and 9-position interface for Data Terminal Equipment employing serial binary interchange. RS-422 & RS-423 are subsets of RS-449.

SDLC (Synchronous Data Link Control)
An IBM communications line discipline or protocol associated with SNA. In contrast with Bisync, SDLC provides full-duplex transmission and is more efficient.

Serial Interface
An interface which requires serial transmission, or the transfer of information in which the bits composing a character are sent sequentially. Implies only a single transmission channel.

SNA (Systems Network Architecture)
A network architecture developed by IBM.

Start Bit
In asynchronous transmission, the beginning of a character.

Stop Bit
In asynchronous transmission, at the end of a character; usually required to be 1, 1.5, or 2 bit times long.

Switched Line
Used to refer to the public telephone network.

Synchronous transmission
Transmission in which there a constant time between successive bits, characters or events. The timing is achieved by sharing of clocking.

Topology
1) Physical topology - The configuration of network nodes and links. Description of the physical geometric arrangement of the links and nodes that make up the network. 2) Logical topology - Description of the possible logical connections between network nodes, indicating which pair of nodes are able to communicate, whether or not they have a direct physical connection. Examples of network topologies are :

- Bus
- Ring
- Star
- Tree

X.25

An interface developed for packet switching procedures by CCITT, the standard-writing organization for international telephone carriers.

VITA[2]

Narayan Makaram

Candidate for the Degree of

Master of Science

Thesis: MICROCOMPUTER DATA COMMUNICATION INTERFACING USING THE RS-449 STANDARD

Major Field: Computing and Information Sciences

Biographical:

Personal Data: Born in Bangalore, India, December 20, 1956, the son of Lt.Col. M.S.Perumal.

Education: Received Bachelor of Engineering degree from Bangalore University, Bangalore, India, in December 1979; completed requirements for the Master of Science degree at Oklahoma State University, Stillwater, Oklahoma, in December, 1984.

Professional Experience: Project Assistant at School of Automation, Indian Institute of Science; Bangalore, India, August 1979 to July 1981. Graduate teaching assistant, Department of Computing and Information Sciences, Oklahoma State University, Stillwater, Oklahoma, 1981 to 1983.