Asynchronous Communications User Guide
(for Model 2236MXE Terminal Processor and Option W Terminal Processor)

REVIEW COPY

FOR INTERNAL USE ONLY

April 11, 1983

PREFACE

Information is provided in this manual regarding usage of the asynchronous communication capabilities of the Model 2236MXE and Option W Terminal Processors. The 2236MXE terminal processor can be used with the 2200MVP and 2200LVP central processors, whereas the Option W terminal processor can be used with the 2200SVP central processor.

Chapter 1 describes the 2236MXE/Option W asynchronous communication capabilities and some important modem considerations for communication applications. Chapter 2 describes programming techniques related to the communication capabilities of the 2236MXE/Option W terminal processors.

Readers of this manual should be familiar with the Wang BASIC-2 language capabilities and the general programming techniques of the Wang 2200 system that is to conduct asynchronous communications using the 2236MXE or Option W terminal processor.

CONTENTS

TABLES

FIGURES

CHAPTER 1

COMMUNICATION CAPABILITIES

1.1 GENERAL INFORMATION

The Wang Model 2236MXE or Option W Terminal Processor is a terminal interface device. The 2236MXE version provides 4 terminal ports; the Option W version provides 3 terminal ports. The 2236MXE/Option W terminal processor is generally used for communications between any combination of Wang 2236D, 2236DE, 2236DW, 2336DW, and 2336DE terminals and a 2200 Series central processor. This device is also capable of serving as a multi-port, RS-232-C compatible communications controller. With the specialized telecommunications (TC) software supporting the terminal processor when it functions as a controller, the user can alternatively choose any port to act as an asynchronous communications controller. Via the port, the user can attach compatible transmission equipment locally or remotely to the terminal processor.

For program control of data transmission and reception via the 2236MXE/Option W port, the $GIO statement in BASIC-2 is used. Release 2.5 or greater of BASIC-2 is required.

A 25-pin EIA (Electronic Industries Association) RS-232-C, CCITT (Consultative Committee on International Telephony and Telegraphy) V.24 compatible cable is used for the connection of a 2236MXE/Option W port to a modem. (RS-232-C/V.24-compatible cables are available from Wang Laboratories, Inc.) A modem is needed for communication applications since data signals from a computer must be converted (modulated) into a range of frequencies suitable for transmission over telephone lines; similarly, data signals received via telephone lines must be demodulated before transfer to a computer. Modem considerations related to the asynchronous communication capabilities of the 2236MXE/Option W terminal processors are presented in Section 1.3.

Though primarily designed for communication applications, the ports on a 2236MXE/Option W terminal processor are well-suited for direct connection of RS-232-C compatible asynchronous transmission equipment such as the following: local CRT terminals, graphic display terminals, analytical equipment, and laboratory instrumentation in general. For hookup of such equipment, a null modem (available from Wang Laboratories, Inc.) may be required.

The 2236MXE/Option W terminal processor has an integrated microprocessor, and each port has multicharacter input/output buffers to simplify telecommunications control procedures and reduce CPU processing requirements. The TC support software is downloaded into the 2236MXE/Option W microprocessor at system configuration time, in a step that is transparent to the user. The downloaded software supports the following:

- data buffering
- code translation
- substitution for characters received in error
- communications control, e.g., monitoring CPU ready/busy conditions, monitoring modem signals, and implementing line turnaround procedures
- break signal detection and transmission
- detection of received timeouts
- detection of end-of-record characters
- automatic insertion and removal of shift characters
- automatic transmission delays following several specified characters.

The standard and special features of the 2236MXE/Option W enable a Wang system to be programmed to transmit and receive data using the line discipline of a variety of asynchronous CRT and mechanical printer terminals.

In addition to containing the TC support code, the 2236MXE/Option W's random access memory is used for storage of initialization information for each port (including code translation tables and a communications control vector), storage of current status information, and input/output data buffering. The desired transmission rate, communication mode, character format options, and special operations for a particular application are selected under program control by specified values in a communications control vector. The vector is loaded into the 2236MXE/Option W by a $GIO statement in the user's application program operating in the CPU.

A selectable-speed clock on the 2236MXE/Option W supports serial asynchronous transmission and reception at line speeds from 50 to 9600 bits per second (bps).

The 2236MXE/Option W supports the following transmission modes:

- full-duplex mode (independent transmission two-ways simultaneously)
- full-duplex mode with automatic deletion of received null characters.
- half-duplex mode (independent transmission one-way-at-a-time alternately)
- half-duplex mode with automatic deletion of received null characters.

The desired mode is set via a particular byte-position in the communications control vector.

Via other byte-positions in the communications control vector, the character format can be selected from the following options:

- parity — odd, even, or no parity
- number of data bits per character — 5, 6, 7 or 8
- number of stop bits per character — 1, 1.5 or 2.

The communication features of the 2236MXE/Option W terminal processor are numerous; descriptions of particular features are presented in this chapter. Programming techniques are presented in Chapter 2, where the format of the communications control vector is shown byte-by-byte in Figure 2-1 and valid values representing the asynchronous transmission options are presented in Table 2-1; also, the format of the communications status vector is given, and $GIO microcommand sequences needed to operate the 2236MXE/Option W are supplied. An example is included to illustrate some of the programming techniques.

1.2 INSTALLATION

Installation of the 2236MXE/Option W terminal processor is the responsibility of a Wang Customer Engineer. If the device arrives as an addition to existing equipment, call the Wang Customer Engineer; do not attempt to install the 2236MXE/Option W terminal processor—any such attempt may void the warranty.

If the 2236MXE/Option W is to be used for point-to-point, dial-up asynchronous telecommunications applications, an RS-232-C/V.24-compatible cable should be plugged into a suitable modem and into the selected port on the 2236MXE/Option W terminal processor. (RS-232-C/V.24-compatible cables are available from Wang Laboratories, Inc.) Installation of a modem is not the responsibility of a Wang Customer Engineer.

Alternatively, if the 2236MXE/Option W is to be used to interface RS-232-C compatible asynchronous transmission equipment such as a non-Wang CRT terminal or a laboratory instrument, the other end of the cable may need to be plugged into a null modem (available from Wang Laboratories, Inc.) or may be suitable for direct connection to the non-Wang equipment. Installation or interfacing of non-Wang equipment is not the responsibility of a Wang Customer Engineer. Information regarding the interface connector pin assignments and voltage levels is given in Section 1.4.

1.3 MODEM CONSIDERATIONS

The modem used with the 2236MXE/Option W may be rented from the telephone company serving the locality where a Wang system is installed or may be purchased from any of several modem vendors, including Wang Laboratories, Inc. In either case, arrangements should be made with the telephone company for installation of a modem since modems purchased from a vendor must be connected to the telephone network via telephone company equipment.

Before a telephone company representative arrives to install a modem, the location of the Wang system must be planned to ensure its proximity to the telephone equipment. Normally, modems are wired permanently to a wall; in such cases, subsequent relocation of the Wang system any great distance would necessitate having the telephone company relocate the modem. The RS-232-C standard recommends use of short cables (less than 50 feet or 15 meters) between data terminal equipment and communications equipment. Longer cable distances are possible for operations at a lower range of transmission rates in an environment relatively free of electromagnetic interference.

The microprocessor on the 2236MXE/Option W can sense the value of the following modem signals:

- Received Data on Pin 3
- Clear to Send on Pin 5
- Data Set Ready on Pin 6

The microprocessor can set the level of the following modem signals:

- Data Terminal Ready on Pin 20
- Request to Send on Pin 4
- Transmitted Data on Pin 2.

A modem with appropriate full-duplex, asynchronous capabilities, such as the Wang WA3451 Asynchronous/Synchronous Modem, can be used with the 2236MXE/Option W. Very likely, other RS-232-C-compatible modems commonly used with asynchronous terminals having transmission rates in the range covered by the 2236MXE/Option W may prove suitable.

For asynchronous operations, the WA3451 modem has three functional modes: 103J-compatible mode, 212A-compatible mode, and 3400 mode. The 103J-compatible mode supports asynchronous communications with a Bell 103J modem at line speeds up to 300 bps. The 212A-compatible mode supports asynchronous communications with a Bell 212A at a line speed of 1200 bps. The 3400 mode supports asynchronous communications at either a line speed of 1200 bps, or any standard commercial line speed of 300 bps and below, with another WA3451 modem, a Racal-Vadic 3400 series modem, or an Anderson-Jacobson 1200 modem.

The WA3451 modem can be used for three types of communication operations over a switched telephone line: manually originated calls, manually answered calls, and automatically answered calls.

NOTES:

1. Modems used at both ends of a communications link must be of similar type. For example, if a Wang WA3451 modem operating in the 103J-compatible mode is used at one end, a Wang WA3451, Bell 103J, or equivalent modem must be used at the other end.

2. If acoustic couplers are used at both ends of a communications link, one must have the "originate" feature and the other must have the "answer" feature -- ideally each should have both features.

1.4 CONNECTOR PIN ASSIGNMENTS

Information in this section is provided for readers responsible for interfacing non-Wang equipment to a Wang system via the 2236MXE/Option W terminal processor. Other readers may wish to ignore the section.

The 2236MXE/Option W conforms to the nationally recognized EIA RS-232-C and the internationally recognized CCITT V.24 standards for voltage levels and pin connections. The signal polarity and the voltage of driven and detected signals are as follows:

Logic Level	Applied Voltage	Detected Voltage
0 or ON (Spacing)	+8 vdc	+5 to +15 vdc
1 or OFF (Marking)	-8 vdc	-5 to -15 vdc

The interface connector pin assignments are listed in Table 1-1, with both the EIA and the CCITT designations given for the circuit associated with each pin. Also, the signal descriptions and sources are included in the table.

Table 1-1. Connector Pin Assignments

Pin	EIA	CCITT	Signal Description	Source
1	AA	101	Protective Ground	
2	BA	103	Transmitted Data	Controller
3	BB	104	Received Data	Modem
4	CA	105	Request to Send	Controller
5	CB	106	Clear to Send	Modem
6	CC	107	Data Set Ready	Modem
7	AB	102	Signal Ground	
8	CF	109	Received Line Signal Detector*	Modem
9			Unconnected	
10			Unconnected	
11			Unconnected	
12			Unconnected	
13			Unconnected	
14			Unconnected	
15			Unconnected	
16			Unconnected	
17			Unconnected	
18			Remote Analog Loopback*	Controller
19			Unconnected	
20	CD	108.2	Data Terminal Ready*	Controller
21			Remote Digital Loopback*	Controller
22	CE	125	Ring Indicator*	Modem
23	CH/CI	111/112	Data Signalling Rate Selector*	Controller/Modem
24			Unconnected	
25			Unconnected	

* This pin is not connected on port 1 of an SVP Option W; ports 2 and 3 use these signals.

1.5 ASYNCHRONOUS TRANSMISSION AND RECEPTION

Generally speaking, in asynchronous transmission (often called start-stop transmission) each character is framed by start and stop elements as shown in Figure 1-1. The start element is represented by a transition from a logic "1" voltage level to a logic "0" voltage level. The nominal interval during which the logic "0" level is maintained for a start element is the same length as the interval used for each data bit.

Depending upon the design of the transmitting equipment, the data-bit interval is a fixed value or one of several possible values if the transmission rate for the equipment is selectable. Immediately following transmission of a start element, the voltage level is changed or not changed depending upon whether the first data bit is 1 or 0. (See Figure 1-1.) Similarly, after the first data-bit interval, the voltage level is changed or not, as required, to represent the second data-bit -- and so on successively for each data bit. The number of data bits transmitted is a fixed value or one of several possible values, depending upon the equipment being used.

After the last data bit is transmitted, a parity bit may be transmitted if provisions for parity information are included in the equipment design. The parity bit interval is the same length as the data bit and start bit intervals. The voltage level may be a logic "0" or "1" depending upon the type of parity (odd or even) and also upon the number of 1's occurring in the preceding data bits.

Finally, the stop element is transmitted using a logic "1" voltage level which is maintained until the next character is transmitted. Usually, there is no upper limit to the length of a stop element; however, there is a lower limit, a fixed value or one of several possible values, depending upon the design of the transmitting equipment.

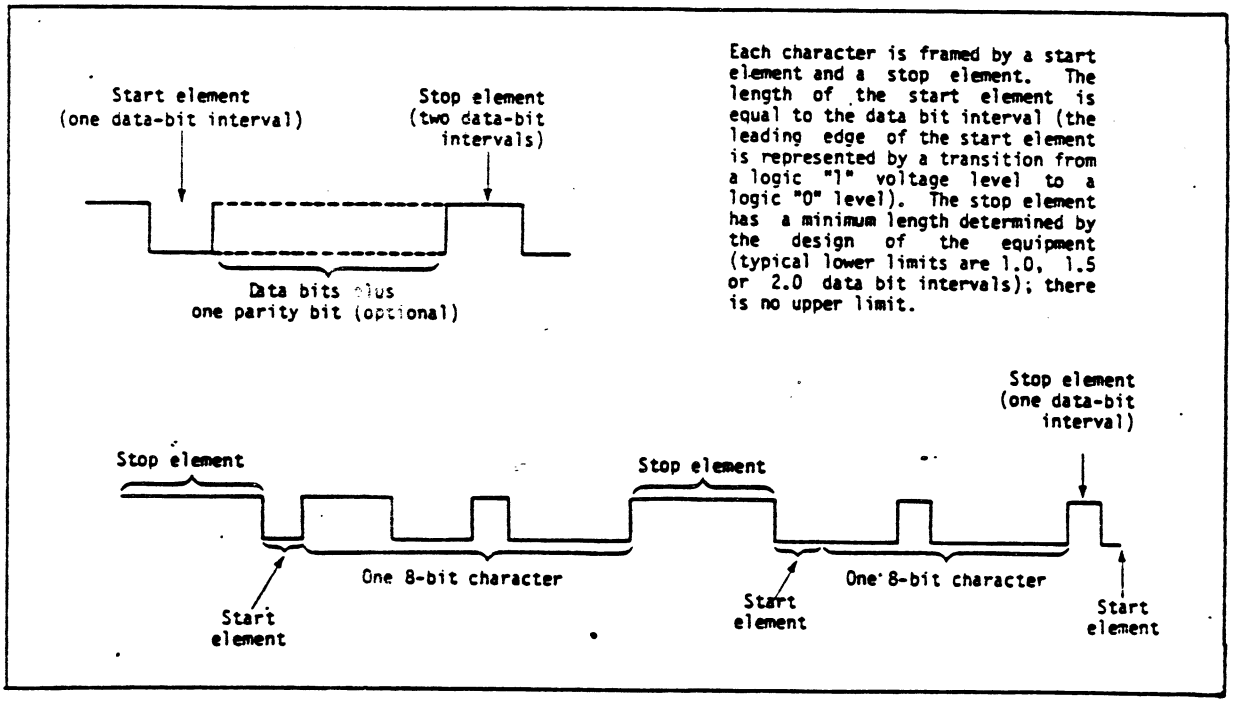

Figure 1-1. Asynchronous Data Transmission

Character Transmission

Wang's 2236MXE/Option W terminal processor transmits each character over a port selected for TC by modulating the Transmitted Data signal on Pin 2 in the connector as follows:

1. The Transmitted Data signal is set to "0" for one bit-time, representing the start bit.

2. Successively, low-order bit first, the signal is set for one bit-time to the value of each data bit until the number of transmitted data bits equals the number specified in the communications control vector; therefore, only the low-order 5, 6, 7 or 8 bits of a character are transmitted.

3. If odd or even parity is specified in the communications control vector, the signal is set for one bit-time to the appropriate value for the type of parity specified. In particular, if odd parity is specified, the parity bit is equal to 1 when the preceding data bits contain an even number of 1-bits; thus, for odd parity, the total number of 1's in the data bits plus the parity bit is an odd number. If even parity is specified, the parity bit is equal to 1 when the preceding data bits contain an odd number of 1-bits; thus, for even parity, the total number of 1's in the data bits plus the parity bit is an even number. If no parity is specified, Step 3 is omitted.

4. The Transmitted Data signal is set to "1" for a minimum interval equal to 1, 1.5 or 2 bit-times depending upon the number of stop bits specified in the communications control vector.

When no character is being transmitted, the Transmitted Data signal on Pin 2 is held at the value "1".

Character Reception

A 2236MXE/Option W port receives a character by detecting changes in the Received Data signal on Pin 3 in the connector as follows:

1. A transition from the voltage level representing logic "1" to the level representing logic "0" for at least one-half a bit-time is interpreted as the leading edge of the start bit for an incoming character.

2. The Received Data signal is sampled successively at times corresponding to the nominal center of each data bit. In particular, the nominal center of the first data bit is 1.5 bit-times after the leading edge of the start bit. The center of each subsequent bit occurs one bit-time after the center of its predecessor. Successively, low-order bit first, the bits in the character being received are set to correspond to the sampled values. The number of data bit samples taken by the 2236MXE/Option W equals the number of data bits specified in the communications control vector. If the number of samples is less than 8, the remaining high-order bits in the received character are automatically set to 0 (unless the shift character option is in effect).

3. A parity bit, if specified, is read by sampling the Received Data signal again — one bit-time after the last data-bit is sampled. The sampled parity value is compared with a calculated value based on the received data bits and the type of parity specified. If the received and calculated parity values are unequal, a designated bit in the communications status vector is set to 1 to indicate a parity error has occurred.

4. One bit-time after the Received Data signal is sampled for a parity value (or for the last data bit if a no-parity option is in effect), the signal is sampled again. Now, if the signal is "1", a valid stop bit is recognized. On the other hand, if the signal is "0", a framing error has occurred and a designated bit in the status vector is set to 1.

NOTE:

For each application, the transmission rate, number of data bits, type of parity, and number of stop bits specified for a 2236MXE/Option W port must match the specifications for equipment in use at the other end of a communications link.

1.6 DATA BUFFERING

Each port of the 2236MXE/Option W terminal processor has two multicharacter data buffers, a 256-byte transmit buffer and a 256-byte receive buffer. With these buffers, data transmission/reception operations performed by the 2236MXE/Option W with respect to the modem attached to a port can overlap data input/output operations performed by the 2200 CPU with respect to the I/O peripherals designated for a communications application.

For example, after the 2200 CPU sends a data string to the 2236MXE/Option W, the CPU is free to perform an independent task such as fetching the next string of data to be transmitted from the input device — while the 2236MXE/Option W is performing such tasks as code translation, character formatting, and transmission to the modem.

NOTE:

If the transmit buffer of a port becomes full while the 2200 CPU is sending data to the buffer, the data transfer rate from the 2200 CPU to the 2236MXE/Option W automatically slows to the rate at which characters are being transmitted from the buffer. No characters are lost.

On the other hand, the 2236MXE/Option W is free to receive a data string, perform such operations as code translation, and store the data in the receive buffer of a port — while the 2200 CPU is performing an independent task such as outputting data to a designated peripheral.

NOTE:

If characters are received when the receive buffer of a port is full, a buffer overrun condition occurs and the appropriate error bit is set in the communications status vector. No other action is taken by the 2236MXE/Option W.

1.7 SUBSTITUTION FOR CHARACTERS RECEIVED IN ERROR

When a character is received with either a parity or a framing error, a substitute character (defined by byte 4 in the communications control vector) is automatically supplied by the 2236MXE/Option W, and the appropriate error bit is set in the port's communications status vector. For example, a special character such as @, or any other character not likely to occur in the incoming data, can be specified as the character to automatically replace any characters received in error. Replacement occurs before code translation, if any, is performed.

1.8 TRANSMISSION DELAYS FOLLOWING SPECIFIED CHARACTERS

When sending data to a mechanical printer terminal such as an IBM 2741, time delays are needed after transmission of TAB and CR (carriage return) characters to allow the printing element sufficient time to reach the proper position without loss of subsequent characters.

A special feature of the 2236MXE/Option W removes the necessity for the user's application program operating in the 2200 CPU to introduce time delays following transmission of special characters. Bytes 11 through 18 of the communications control vector can be used, in four two-byte groups, to define up to four special characters and the transmission delay associated with each special character. During data transmission, the 2236MXE/Option W automatically delays for the specified time following transmission of any character matching one of the specified characters.

1.9 CODE TRANSLATION

The code translation feature of the 2236MXE/Option W allows data interchange between the 2200 CPU and the 2236MXE/Option W in the ASCII code (American Standard Code for Information Interchange) native to the 2200 CPU — regardless of the transmission/reception code for a particular application.

Each port has space reserved in the 2236MXE/Option W for two 256-byte code translation tables, a transmit-code translation table and a receive-code translation table. Specification of such tables is optional. Translation tables, supplied in the user's application program operating in the 2200 CPU, must be loaded into the 2236MXE/Option W by appropriate $GIO statements in the program. (See Section 2.4.)

The automatic code translation operation is enabled by loading a transmit or a receive code translation table (or both) after loading the communications control vector. If no tables are loaded, the code translation feature is effectively disabled.

During transmission, a character sent from the 2200 CPU to the 2236MXE/Option W becomes an 8-bit index for a table lookup in the transmit-code translation table. An 8-bit character obtained from the table is placed in the transmit buffer; however, if byte 3 of the communications control vector specifies less than 8 data bits per character, only the relevant low-order bits of each character are actually transmitted.

During reception, a character received by the 2236MXE/Option W is used as an 8-bit index for a table lookup in the receive-code translation table, and an 8-bit character is obtained from the table. If the translated character is a null character, i.e., HEX(00), and the high-order hexdigit in byte-position-2 in the communications control vector has the value 3, the character is discarded; otherwise, the translated character is placed in the receive buffer.

NOTE:

Superfluous characters used for timing or fill can be removed automatically by translating them to null characters. This feature is applicable to the full-duplex and half-duplex operations supported by the 2236MXE/Option W terminal processor. See Table 2-1.

1.10 INSERTION AND REMOVAL OF SHIFT CHARACTERS

For applications involving data transmission and reception in a code set which utilizes shift characters, e.g., a Baudot code set or an IBM 2741 code set, a special feature of the 2236MXE/Option W removes the necessity for the user's program operating in the 2200 CPU to handle insertion and removal of shift characters. Instead, the upshift and the downshift characters are defined in bytes 7 and 8 of the communications control vector. The number of data bits per character is set to 5 or 6 (depending upon the application) by choosing an appropriate value for the high-order hexdigit in byte 3 of the communications control vector. Then, to activate automatic insertion and removal of shift characters, code translation tables are suitably defined and loaded into the 2236MXE/Option W.

During data transmission, the 2236MXE/Option W examines and interprets the two high-order bits of each translated character in the transmit buffer as "shift status" bits as follows:

Two High-order Bits	Meaning
00	Downshifted character.
01	Upshifted character.
10	Not applicable.
11	Not applicable.

A shift character is automatically transmitted between any two characters having different "shift status" bits. In such cases, if the second character has upshifted status, an upshift character is transmitted prior to transmission of the second character; alternatively, if the second character has downshifted status, a downshift character is transmitted prior to transmission of the second character. The shift-status bits are not transmitted since the number of data bits being transmitted per character is only the low-order 5 or 6 bits if byte 3 in the communications control vector has been appropriately specified.

During reception, the 2236MXE/Option W sets the value of the shift status bit (high-order eighth bit) of each received character (before code translation) according to the most recently received shift character as follows:

High-order Bit	Meaning
1	Upshifted character.
0	Downshifted character.

NOTE:

Before code translation, each incoming character is compared to the shift characters in bytes 7 and 8 of the communications control vector. When a shift character is found, the shift status bit in the subsequent received characters is set accordingly, and the shift character is discarded. The subsequent received characters are then code translated and placed in the receive buffer.

1.11 DETECTING END-OF-RECORD CHARACTERS

The end-of-record detection feature is convenient for applications where a received data stream contains meaningful record delimiters, e.g., CR (carriage return) codes. The feature is particularly convenient for applications where there is no necessity to display the data while being received.

Any number of characters can be defined as end-of-record characters, thereby permitting the 2236MXE/Option W to divide a received data stream into records while eliminating the need for the user's program operating in the 2200 CPU to perform the task.

To activate the end-of-record detection feature, byte 6 in the communications control vector must be set to HEX(01); also, a suitably defined receive-code translation table must be loaded into the 2236MXE/Option W. To be suitably defined, the high-order bit for codes in the receive-code translation table must be set to 1 for each character defined as an end-of-record character (and set to zero for all other characters).

During reception, if end-of-record detection is enabled, the 2236MXE/Option W maintains a count of the number of end-of-record characters currently stored in the active port's receive buffer. The count is maintained in byte 5 of the communications status vector (see Table 2-2).

With an appropriate $GIO statement (see Section 2.4), the user's application program can read the status vector into the 2200 CPU and subsequently test the status information to ensure the availability of a complete record before requesting transfer of buffered data via a $GIO statement which performs data transfer from the 2236MXE/Option W to the 2200 CPU.

When data is actually transferred from the port's receive buffer to the 2200 CPU, only those characters up to (and including) the first end-of-record character are transferred. Furthermore, the high-order bit in the end-of-record character is changed from 1 to 0 when the character is transferred.

NOTE:

If the end-of-record detection feature is not needed for an application (or cannot be utilized because the high-order bit for codes in the receive-code translation table is set to 1 for a purpose other than defining an end-of-record character), byte 6 in the communications control vector should be set to HEX(00) to disable end-of-record detection.

1.12 MONITORING RECEIVED TIMEOUTS

The 2236MXE/Option W has the capability to monitor received timeouts. Byte 5 in the communications control vector is used to set the binary value of the timeout in units of 0.1 second. For example, the minimum timeout condition, 0.1 second, is specified by storing HEX(01) in byte-position-5. The maximum timeout condition, 25.5 seconds, is specified by storing HEX(FF) in byte-position-5. On the other hand, storing HEX(00) in byte-position-5 disables the monitoring feature for received timeouts.

If a timeout interval is specified, the 2236MXE/Option W maintains a received data timeout countdown in byte 6 of the communications status vector (see Section 2.3). Every 0.1 second during reception, byte 6 is decreased by 1 until it reaches 0.

1.13 SENDING AND DETECTING BREAK SIGNALS

The 2236MXE/Option W has the capability to send and detect break signals under control of the user's program operating in the 2200 CPU. Bytes 9 and 10 in the communications control vector are used to define the break signal transmission and detection intervals, respectively, in units of 10 milliseconds. For example, HEX(14) stored in byte-position-9 defines an interval equal to 200 milliseconds for transmitted break signals. Similarly, HEX(11) stored in byte-position-10 defines an interval equal to 170 milliseconds for detection of break signals.

In addition to specifying the break signal intervals in bytes 9 and 10 of the communications control vector, it is necessary to use the low-order hexdigit position in byte 2 to enable or disable the break signal as follows:

Byte 2 (Low-order Hexdigit)	Break Signal
0	Disabled
1	Enabled for Transmitted/ Received Data

Transmission of a break signal by the 2236MXE/Option W involves inverting the level of the Transmitted Data modem signal for an interval defined by byte 9 of the communications control vector.

Detection of a break signal occurs when the 2236MXE/Option W senses that the level of the Received Data modem signal is being continuously inverted for an interval at least as long as the interval defined by byte 10 of the communications control vector.

NOTE:

Detection of a break signal causes the "break signal received" bit in the status vector to be set. No other action is taken by the 2236MXE/Option W.

CHAPTER 2

PROGRAMMING TECHNIQUES

2.1 GENERAL CONSIDERATIONS

When writing a communications application program for the 2236MXE/Option W terminal processor, a programmer should organize the program in distinct modules designed to achieve initialization and communication functions. The communications module could overlay the initialization module or, if preferred, the two modules could coexist in memory.

Generally speaking, an initialization module defines the 20-byte communications control vector and assigns particular values to byte-positions denoting information such as the desired transmission rate in bits per second, the number of data bits per character, the type of parity (if any), and the number of stop bits per character. Other values in the control vector indicate whether the 2236MXE/Option W is to activate such features as break detection, monitoring of received timeouts, full or half duplex operation with or without automatic removal of null characters, and substitution of a special character for characters received with parity or framing errors. The module also supplies the $GIO statement needed to load the control vector into the 2236MXE/Option W.

An initialization module also defines transmit and receive code translation tables, if required for the application, and supplies the $GIO statements needed to load such tables into the 2236MXE/Option W. Additionally, the initialization module might define and initialize variables required by the communications module even though the variables are stored in the CPU rather than the 2236MXE/Option W memory.

2.2 SPECIFYING THE COMMUNICATIONS CONTROL VECTOR

Space is reserved in the random access memory of the 2236MXE/Option W for each port to have a communications control vector. The user configures a particular port for communications, as described in Section 2.4, and needs to define the control vector only once for a given application program. The format of the communications control vector is shown in Figure 2-1, and valid specifications for the vector are given in Table 2-1. The table is divided into two portions since the first three bytes of the vector are dual purpose while the remaining bytes are single purpose with respect to the available communications options and features.

In the application program residing in the 2200 CPU, the control vector should be defined by a one-dimensional array having 20 elements with one byte per element. For example, to represent the control vector by the array C$(), use

```
DIM C$(20)1
```

Also, as a general programming practice, all elements should be initialized to binary zero by a statement of the form

```
INIT(00) C$()
```

before assigning values to particular elements in the array.

Then, as illustrated by the following statements, individual byte-positions in the control vector can be assigned values other than binary zero to select the desired combination of options and define any special characters for the application.

Statement	Comment
C$(1) = HEX(17)	One stop bit; 300 bits per second.
C$(2) = HEX(31)	Full duplex with automatic deletion of null characters; break enabled on transmit/receive.
C$(3) = HEX(23)	Seven data bits; odd parity.
C$(4) = HEX(5E)	Substitute character for parity or framing error is an up-arrow, .
C$(5) = HEX(0A)	Timeout interval is 1 second.

Care must be exercised when defining some special characters to choose a compatible value in the first three byte-positions of the communications control vector. For example, if defining upshift and downshift characters in bytes 7 and 8, the high-order hexdigit in byte 3 must have the value 0 or 1 (since the shift feature can be used only with code sets having 5 or 6 data bits per character). See Section 1.10 and Table 2-1.

NOTE:

The communications control vector must be loaded into the 2236MXE/Option W via a $GIO statement having the appropriate microcommand sequence from Table 2-3. See Section 2.4.

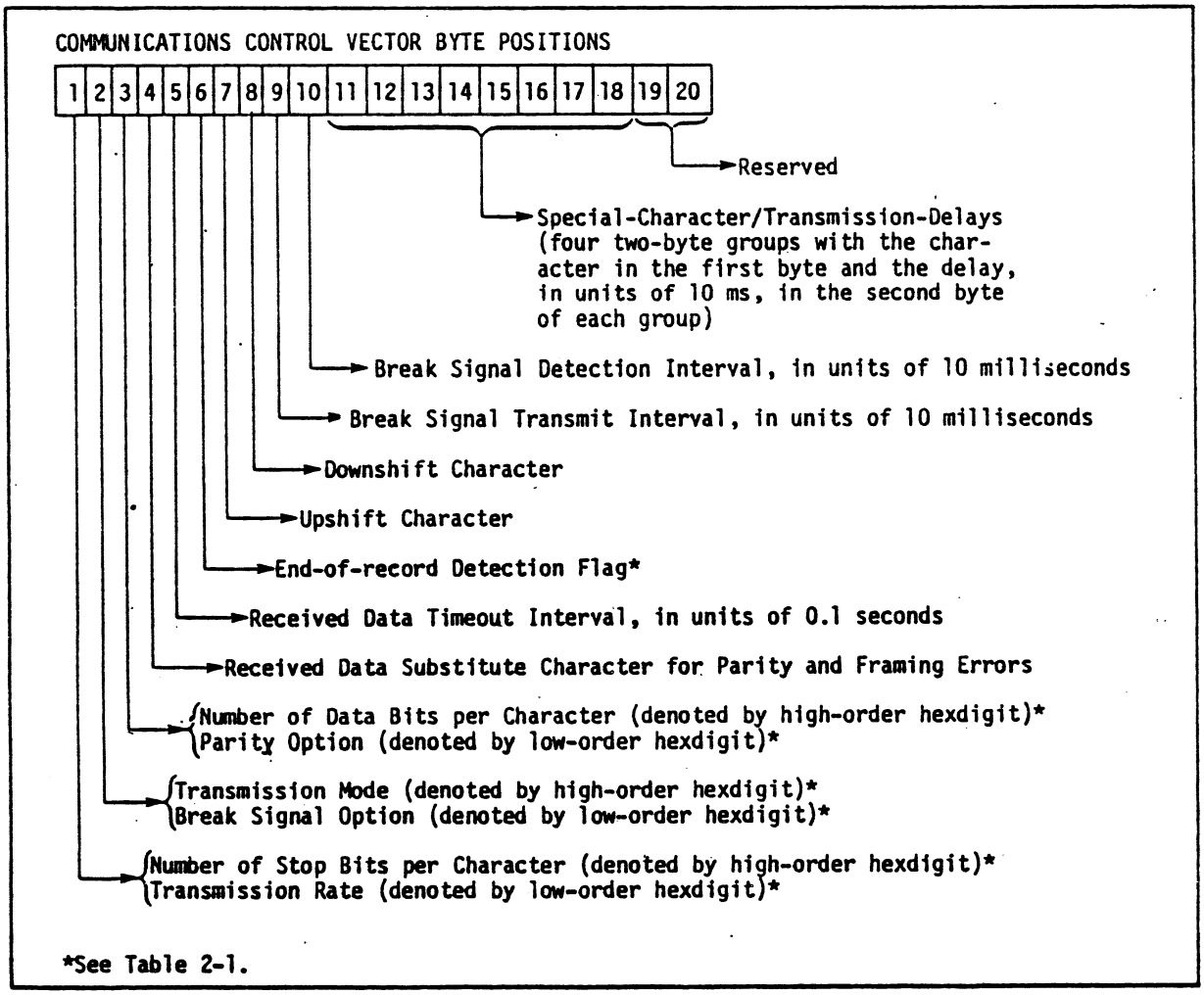

Figure 2-1. Communications Control Vector Format

Table 2-1. Valid Communications Control Vector Specifications

Byte	High-order Hexdigit	Low-order Hexdigit
1	0 = Illegal value 1 = 1 Stop bit 2 = 1.5 Stop bits 3 = 2 Stop bits	0 = 50 bits per second 1 = 75 bps 2 = 100 bps 3 = 110 bps 4 = 134.5 bps 5 = 150 bps 6 = 200 bps 7 = 300 bps 8 = 600 bps 9 = 1200 bps A = Undefined B = 2400 bps C = Undefined D = 4800 bps E = Undefined F = 9600 bps
2	0 = Half duplex 1 = Half duplex with deletion of received null characters 2 = Full duplex 3 = Full duplex with deletion of received null characters	0 = Break disabled 1 = Break enabled on transmit/receive
3	0 = 5 Data bits per character 1 = 6 Data bits 2 = 7 Data bits 3 = 8 Data bits	0 = No parity 1 = Even parity 2 = No parity 3 = Odd parity

Byte	High and Low Order Hexdigits*	Remarks
4	xy = Substitute character for parity/framing errors.	Each received character having a parity or framing error is replaced by the designated character (replacement occurs prior to code translation if translation tables are being used). See Section 1.7.

*x and y each denote any hexdigit (0 through 9, A through F). If a feature is not desired, the byte positions associated with the feature should be initiated to binary zero in the communications control vector.

Table 2-1. Valid Communications Control Vector Specifications (Continued)

Byte	High and Low Order Hexdigits	Remarks
5	xy = Timeout interval in units of 0.1 seconds.	The specification in hexadecimal notation represents the timeout interval in units of 0.1 seconds, e.g., HEX(24) = DECIMAL(36) specifies an interval of 3.6 seconds. See Section 1.12.
6	00 = Disable end-of-record detection. 01 = Enable end-of-record detection.	If enabled, the end-of-record characters must be defined via the receive-code translation table by setting the high-order bit to 1 for each code corresponding to an incoming end-of-record character. See Section 1.11.
7	xy = Upshift character.	To enable shift code insertion/deletion, the high-order hexdigit in byte 3 of the control vector must be 0 or 1 (i.e., the number of data bits per character must be 5 or 6). Also, the transmit-code translation table must identify all downshifted, upshifted, and "don't care" characters by setting the two high-order bits to 00, 01, and 10 or 11 as described in Section 1.10. The receive-code translation table must allow for the 2236MXE/Option W's automatic setting (before translation) of the high-order bit to 1 for all incoming upshifted characters.
8	xy = Downshift character.	
9	xy = Break signal transmit interval in units of 10 ms.	To enable break signal transmission/detection, the low-order hexdigit in byte 2 of the control vector must specify the polarity and the modem signals. If bytes 9 and 10 are both HEX(00), the low-order hexdigit in byte 2 should be 0. The byte 9 and 10 specifications in hexadecimal notation represent break signal transmit and receive intervals in units of 10 milliseconds, e.g., HEX(12) = DECIMAL(18) specifies a 180 ms interval. See Section 1.13.
10	xy = Break signal detection interval in units of 10 ms.	
11	xy = Special character.	Up to four special characters can be defined in bytes 11, 13, 15, and 17. The delay associated with a particular character is specified in the following byte, in hexadecimal notation representing the interval in units of 10
12	00 = No transmission delay. xy = Transmission delay in units of 10 ms.	

Table 2-1. Valid Communications Control Vector Specifications (Continued)

Byte	High and Low Order Hexdigits	Remarks
13	Same as byte 11.	milliseconds. The maximum delay, HEX(FF), is 2.5 seconds. During transmission, the 2236MXE/Option W automatically delays for the specified time following the transmission of one of the specified characters. If a transmit-code translation table is used, each special character must be defined with respect to its code after translation. If the automatic transmission delay capability is not desired, bytes 11 through 18 should each be set to HEX(00).
14	Same as byte 12.	
15	Same as byte 11.	
16	Same as byte 12.	
17	Same as byte 11.	
18	Same as byte 12.	

2.3 THE COMMUNICATIONS STATUS VECTOR

Space is reserved in the random access memory of the 2236MXE/Option W for each port to have a communications status vector whose byte and bit positions are used automatically as shown in Table 2-2. The first three bytes of the status vector are cleared automatically whenever it is read and when the port's communications control vector is loaded into the 2236MXE/Option W from the 2200 CPU. See Section 2.4.

Flags are set in particular bit positions in the first three bytes of the status vector during 2236MXE/Option W operation. In bytes 4 and 5, the current number of characters in the receive buffer and the number of end-of-record characters are maintained as binary counts. Similarly, in byte 7, the current number of characters in the transmit buffer is maintained. Byte 6, on the other hand, is similar to a real time clock whose value is initiated to the timeout interval specified in byte 5 of the communications control vector each time one of the following events occurs:

a) the communications control vector is loaded,

b) a $GIO "start receiving data" operation begins,

c) a line turnaround occurs during a $GIO "send, then receive data" operation, or

d) a character is received.

However, if the value in byte 6 of the status vector is not reset by one of these operations, the value is decreased by 1 every 0.1 second until it reaches 0.

Whenever desired, the information in the status vector can be read (transferred to the 2200 CPU) by a $GIO statement having the appropriate microcommand sequence from Table 2-3. See Section 2.4. After transfer to the 2200 CPU, status vector information can be tested, as required, by the application program.

Table 2-2. Communications Status Vector Information

Byte	Bit*	Meaning
1	1	1 = Break signal received
2	1	0 = always zero
	2	0 = always zero
	3	1 = Data Set Ready modem signal On
3	1	1 = Receive parity error detected
	2	1 = Receive buffer overrun error detected
	3	1 = Receive framing error detected
4	all	Binary count of the number of characters in the receive buffer.
5	all	Binary count of the number of end-of-record characters in the receive buffer.
6	all	Received data timeout countdown.
7	all	Binary count of the number of characters in the transmit buffer.

*Bit positions in each byte are numbered from 1 (low-order) to 8 (high-order).

2.4 2200 CPU AND 2236MXE/OPTION W INTERACTION VIA $GIO STATEMENTS

To operate the 2236MXE/Option W terminal processor in communications mode, the user's application program residing in the 2200 CPU should configure a 2236MXE/Option W port for telecommunications. This step is needed because the port can also be configured for the attachment of a terminal, as described below. The program should also include $GIO statements with a port address and suitable microcommand sequences.

Port Configuration

A port address is defined as follows:

```
/A#
```

where /A indicates the 2236MXE/Option W terminal processor and # is the system port number (terminal number). A 2200MVP or 2200LVP central processor can accommodate up to four 2236MXE terminal processors, each with four ports. For example, /A5 is the first RS-232-C port on the second 2236MXE, and /A10 is the second RS-232-C port on the third 2236MXE. However, a 2200SVP central processor can accommodate only one Option W terminal processor with three ports. /A2 is the second and /A3 is the third RS-232-C port on the Option W terminal processor.

A 2236MXE/Option W port must be configured as a TC port before it can be used for telecommunications. This is accomplished with the SELECT TC statement, defined as follows:

```
SELECT TC /A#
```

where /A# is the address of the port to be configured as a TC port. To select a port as a TC port, all of the following conditions must be true:

- The port must be on a 2236MXE/Option W terminal processor.
- The port must not be attached or assigned to any partition.
- The port must not be terminal port 1 (i.e., port address /A1).

To reconfigure the port as a terminal port, the SELECT TERMINAL statement is used.

```
SELECT TERMINAL /A#
```

If a TC port has been opened and is then reconfigured as a terminal port in the same partition, the port is automatically closed before it becomes a terminal port. (Opening and closing a TC port is discussed below.) However, if the TC port has been opened by another partition, the SELECT TERMINAL statement will produce an error.

Selecting a port to its current configuration (i.e., a TC port to be a TC port or a terminal port to be a terminal port) will not produce an error.

The $OPEN and $CLOSE statements of the BASIC-2 language can be used with any 2200 CPU input/output port except a terminal port. They can be used to lock around a TC port, after the port has been configured. (It is recommended that a TC port be opened as soon as it is configured, to prevent another user from accidentally selecting it.) The following are examples of the use of these statements:

```
$OPEN /A7
$OPEN 200, /A6
$CLOSE /A12
```

TC $GIO Statements

The format of the $GIO statements used to address a TC port is as follows:

$GIO /A# (GIO command)

where /A# is a port address and (GIO command) is the body of the $GIO statement, including the microcommand sequence. Most formats that are available within the body of the $GIO statement are legal, with the exception of address switching. Attempts to switch addresses within a $GIO statement will cause a run-time error.

A list of valid microcommand sequences for 2236MXE/Option W operations is presented in Table 2-3.

Table 2-3. Microcommand Sequences for 2236MXE/Option W and 2200MVP/LVP/SVP Interaction

2236MXE/Option W and 2200 CPU Interaction	Microcommand Sequence*	Remarks
Set communications control vector	4402 A000 440C	
Read communications status vector	4403 1020 02FF 03FF 1223 C620	
Load transmit code translation table	4404 A000 440C	
Load receive code translation table	4405 A000 440C	
Disconnect	4406	
Send break signal	4407	
Start receiving data	4408	For half or full duplex.
Transfer received data to CPU	4409 1020 02FF 03FF 1223 C620	For half or full duplex.
Send data	440A A000 440C	For half or full duplex.
Send, then receive data	440B A000 440C	For half duplex.
Stop transmitting	440C	For full duplex.
Continue transmitting	440D	For full duplex.

*A microcommand sequence can be specified directly or indirectly in a $GIO statement. If specified directly as the arg-1 component, each four-hexdigit-code can be separated from the previous one by a space for readability as shown in this table. If specified indirectly by assigning the sequence to a variable and including the variable in a statement, spaces cannot be used between the four-hexdigit-codes, e.g., A$ = HEX(4402A000440C); furthermore, the dimension of the variable must be large enough to ensure the presence of two trailing space characters which serve as the pseudo-microcommand 2020 denoting the end of the sequence. Unpredictable results may occur if at least one trailing blank does not follow an indirectly specified microcommand sequence. (See the General I/O Instruction Set Reference Manual which accompanies each Wang system having the $GIO statement in its BASIC language instruction set, or see the BASIC-2 Language Reference Manual if using a 2200MVP/LVP/SVP system.)

Brief descriptions of each of the operations in Table 2-3 follow. A sample $GIO statement is shown for each operation; however, four-hexdigit-code values used in the statement may differ and variables may be given different names in a user's program.

Set Communications Control Vector

```
$GIO SET CCV /A# (4402 A000 440C, G$) C$()
```

The communications control vector (CCV) defined by the array C$() is set (loaded) into the 2236MXE/Option W when the statement is executed. Here, A# is the address of the TC port, and G$ represents the error/status/general-purpose registers.

NOTE:

The TC port's transmit and receive buffers, as well as the communications status vector and code translation tables, are cleared automatically when the communications control vector is loaded.

Read Communications Status Vector

```
$GIO READ CSV /A# (4403 1020 02FF 03FF 1223 C620, G$) A$
```

The information currently in the communications status vector is read into the 2200 CPU and stored in the character string A$ (which must be at least 7 bytes long).

NOTE:

The error and received break indicators (i.e., bytes 1 and 3) in the communications status vector are cleared automatically after the status vector information is read into the 2200 CPU. In a program for a 2200MVP/LVP/SVP central processor, insert a line containing a $BREAK statement before each line containing a "read communications status vector" statement.

Load Transmit Code Translation Table

```
$GIO LOAD TTBL /A# (4404 A000 440C, G$) C1$()
```

The transmit code translation table is loaded into the 2236MXE/Option W from the array C1$() if such an array is previously defined in the application program. The optional transmit code translation feature is enabled only if a transmit code translation table is loaded after the communications control vector is loaded into the 2236MXE/Option W.

NOTES:

1. The transmit code translation table should be exactly 256 bytes in length and represent the codes to which System 2200 ASCII characters are to be converted prior to storage in the transmit buffer. The byte positions in the table should contain the "after translation characters" arranged in a sequence corresponding to the "before translation characters", i.e., the System 2200 ASCII characters.

2. If shift character automatic insertion/removal is in effect (i.e., the specified number of data bits per character is 5 or 6), the two high-order bits of each code in the transmit code translation table must conform to the appropriate values given in Section 1.10.

Load Receive Code Translation Table

```
$GIO LOAD RTBL /A# (4405 A000 440C, G$) C2$()
```

The receive code translation table is loaded into the 2236MXE/Option W from the array C2$() if such an array is previously defined in the application program. The optional receive code translation feature is enabled only if a receive code translation table is loaded after the communication control vector is loaded into the 2236MXE/Option W.

NOTES:

1. The receive code translation table should be exactly 256 bytes long. The byte positions in the table should contain ASCII characters (the "after translation characters" in this case) arranged in a sequence corresponding to the "before translation characters". The translation procedure is equivalent to using the binary equivalent of an incoming character's hexadecimal code as an index for a table look-up operation by which the appropriate translation character is found. For example, if the incoming non-ASCII character is a HEX(18), the binary value is 24; therefore, the corresponding ASCII character should be located in the 25th position of the receive translation table. (Keep in mind that the first position in the table corresponds to the binary value zero.)

2. If shift character automatic insertion/removal is in effect, the 256-byte receive code translation table represents two 128-byte tables. The first 128 byte positions in the table should represent the conversions for incoming downshifted characters corresponding to the hexadecimal codes HEX(00) through HEX(7F). The second 128 byte-positions should represent conversions for incoming upshifted characters corresponding to the hexadecimal codes HEX(80) through HEX(FF).

3. If end-of-record character detection is enabled, the high-order bit for codes in the receive-code translation table must be set to 1 for each character defined as an end-of-record character (and set to zero for other characters).

Disconnect

```
$GIO DISCONNECT /A# (4406, G$)
```

The 2236MXE/Option W disconnects from the line by setting the Data Terminal Ready signal to zero for a period of three to five seconds.

Send Break

```
$GIO BREAK /A# (4407, G$)
```

The 2236MXE/Option W sends a break signal if indicated by the low-order hexdigit in byte-position-2 of the communications control vector. See Table 2-1 and Section 1.13.

Start Receiving Data

```
$GIO START RCV /A# (4408, G$)
```

One "start receiving data" statement is needed to enable data reception via the 2236MXE/Option W when set for the full-duplex or half-duplex mode. If set for half-duplex mode, the transmit and receive buffers are cleared first. The 2236MXE/Option W enters the receive mode and starts receiving data. Also, the receive timeout countdown is started by initiating byte 6 of the communications status vector to the value specified as the timeout interval (if different from binary zero). After code translation (if enabled), the received data is stored in the TC port's receive buffer.

Transfer Received Data to the CPU

```
$GIO RCV /A# (4409 1020 02FF 03FF 1223 C620, G$) D$()
```

All or part (if an end-of-record character is detected) of the receive buffer characters are transferred from the 2236MXE/Option W to the 2200 CPU and stored in the array D$(). (See Section 1.11.) The $GIO data buffer, denoted here by D$(), should be at least 256 bytes long since the 2236MXE/Option W has a 256-byte receive buffer associated with each port. Bytes 9 and 10 in the error/status/general-purpose registers provided by the variable G$, i.e., arg-2 of the $GIO statement, are set to the binary representation of the number of bytes transferred whether stored or not. In a program for a 2200MVP/LVP/SVP central processor, insert a line containing a $BREAK statement before each line containing a "transfer received data" statement.

Send Data

```
$GIO SEND /A# (440A A000 440C, G$) F$()  <1, N>
```

If set for half-duplex mode, the receive buffer is cleared first. For half-duplex or full-duplex mode, bytes 1 through N of the array F$() are transferred from the 2200 CPU to the 2236MXE/Option W where they are stored in the transmit buffer after code translation (if enabled) and then transmitted.

NOTE:

The $GIO microcommand A000 implements a particular signal sequence repeatedly (once per character until each character in the arg-3 data buffer is transferred from the 2200 CPU to the 2236MXE/Option W). The $GIO syntax requires a single argument format for arg-3. Therefore, a $PACK statement should be used to pack multi-argument data into a single argument prior to executing the "send data" $GIO statement if the application requires a specially formatted buffer.

Send Then Receive Data

```
$GIO SEND RCV /A# (440B A000 440C, G$) F$()  <E>
```

This statement is applicable only for the half-duplex mode of operation and provides for automatic line turnaround. Beginning with the Eth byte, all remaining bytes of the array F$() are transferred from the 2200 CPU to the 2236MXE/Option W for storage in the transmit buffer after code translation is performed, if enabled. The 2236MXE/Option W transmits the data and then executes a "start receiving data" operation.

NOTES:

1. For half-duplex communications, the "send data" $GIO operation should be used to send all but the last bytes of data. The "send, then receive data" $GIO operation should be used to send the last bytes of data, and afterwards the "transfer received data to CPU" should be used. Use of the "send, then receive data" $GIO operation automatically implements a line turnaround procedure, thereby ensuring the 2236MXE/Option W's readiness to receive data without loss.

2. For full-duplex communications, only the "send data" and "transfer received data to CPU" $GIO operations are needed; the "send, then receive data" $GIO operation should not be used since, in full duplex mode, the 2236MXE/Option W remains in transmit and receive mode simultaneously and line turnaround does not occur.

Stop Transmitting

```
$GIO STOP SEND /A# (440C, G$)
```

This statement is applicable for the full-duplex mode of operation, in cases where the 2200 CPU must stop transmission temporarily because a control sequence is received without clearing the contents of the transmit buffer. Transmission commences when a "send data" $GIO operation or a "continue transmitting" $GIO operation is executed.

Continue Transmitting

```
$GIO CONTINUE SEND /A# (440D, G$)
```

This statement can be used to restart transmission if the transmit buffer contains data and transmission has been halted by a "stop transmitting" $GIO operation. Use of this statement when transmission is already in progress does not harm the data.

2.5 A SAMPLE PROGRAM

(To be supplied.)

APPENDIX A

ASCII CODE SET

The 2200 System character set is defined by 8-bit codes. The 8 bits in each code are designated as high-order and low-order, with the high-order bits numbered 8, 7, 6, and 5 and the low-order bits numbered 4, 3, 2, and 1. Bit number 8 is 0 and the bits 7 through 1 correspond to the ASCII (American Standard Code for Information Interchange) character set which has 128 assignment positions, as shown in Table A-1.

Wang CRTs and printers use the ASCII code set, but some units may not display all the graphic characters shown in Table A-1. In some cases, substitute graphic characters may be displayed by a Wang peripheral. For details, refer to the manual which accompanies a particular peripheral.

Table A-1. ASCII Code*

Low-order: 4-bits / High order: 4-bits	hex-digit	0_{0_0}	0_{0_1}	0_{1_0}	0_{1_1}	1_{0_0}	1_{0_1}	1_{1_0}	1_{1_1}
		0	1	2	3	4	5	6	7
0000	0	NUL 0	SOH 1	STX 2	ETX 3	EOT 4	ENQ 5	ACK 6	BEL 7
0001	1	DLE 16	DC1 17	DC2 18	DC3 19	DC4 20	NAK 21	SYN 22	ETB 23
0010	2	Space 32	! 33	" 34	# 35	$ 36	% 37	& 38	(apos.) ' 39
0011	3	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55
0100	4	@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71
0101	5	P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87
0110	6	grave accent 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103
0111	7	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119

Low-order: 4-bits / High order: 4-bits	hex-digit	1_{0_0}	1_{0_1}	1_{1_0}	1_{1_1}
		8	9	A	B	C	D	E	F
0000	0	BS 8	HT 9	LF 10	VT 11	FF 12	CR 13	SO 14	SI 15
0001	1	CAN 24	EM 25	SUB 26	ESC 27	FS 28	GS 29	RS 30	US 31
0010	2	(40	) 41	* 42	+ 43	(comma) , 44	(dash) - 45	(period) . 46	/ 47
0011	3	8 56	9 57	: 58	; 59	< 60	= 61	> 62	? 63
0100	4	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79
0101	5	X 88	Y 89	Z 90	[91	\ 92	] 93	↑ 94	(underline) _ 95
0110	6	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111
0111	7	x 120	y 121	z 122	{ 123	¦ 124	} 125	~ 126	DEL 127

*Numbers in the lower right corner of each box represent the decimal equivalent of the binary and the hexadecimal code for the character shown in the box, e.g., A = $(41)_{16} = (01000001)_2 = (65)_{10}$.

LEGEND FOR ASCII CONTROL CHARACTERS			
NUL	Null	DLE	Data Link Escape
SOH	Start of Heading	DC1	Device Control 1
STX	Start of Text	DC2	Device Control 2
ETX	End of Text	DC3	Device Control 3
EOT	End of Transmission	DC4	Device Control 4
ENQ	Enquiry	NAK	Negative Acknowledge
ACK	Acknowledge	SYN	Synchronous Idle
BEL	Bell (audible or attention signal)	ETB	End of Transmission Block
BS	Backspace	CAN	Cancel
HT	Horizontal Tabulation	EM	End of Medium
	(punched card skip)	SUB	Substitute
LF	Line Feed	ESC	Escape
VT	Vertical Tabulation	FS	File Separator
FF	Form Feed	GS	Group Separator
CR	Carriage Return	RS	Record Separator
SO	Shift Out	US	Unit Separator
SI	Shift In	DEL	Delete

APPENDIX B

SPECIFICATIONS

Power Requirements

Supplied by the 2200 CPU.

Interface Connections

2236MXE Terminal Processor

Four 25-pin, EIA RS-232-C/CCITT V.24-compatible female plugs, each for the connection of a 2236D or 2336D terminal or a DCE device.

Option W Terminal Processor

Two 25-pin, EIA RS-232-C/CCITT V.24-compatible female plugs, each for the connection of a 2236D or 2336D terminal or a DCE device.

Asynchronous Transmission Rates

50, 75, 100, 110, 134.5, 150, 200, 300, 600, 1200, 2400, 4800, or 9600 bits per second (bps).

Character Format Options

Parity: odd, even, or no parity.
Number of Data Bits: 5, 6, 7, or 8.
Number of Stop Bits: 1, 1.5, or 2.

Communication Mode

Full or half duplex.

Compatible Modems

Wang WA3451 or equivalent for full-duplex communications at 0-1200 bps. Other full-duplex or half-duplex, Wang-compatible modems from other vendors can also be used.

Null modem, available from Wang, for direct communications link.

Standard Warranty Applies

INDEX