WANG

# 8-BIT-PARALLEL I/O INTERFACE CONTROLLER USER MANUAL

(MODEL 2250/OPTION 67)

# SYSTEM 2200

# 8-BIT-PARALLEL I/O INTERFACE CONTROLLER USER MANUAL (Model 2250/Option 67)



WANG LABORATORIES, INC.
ONE INDUSTRIAL AVENUE, LOWELL, MASSACHUSETTS 01851, TEL. (617) 459-5000, TWX 710 343-6769, TELEX 94-7421

## Disclaimer of Warranties and Limitation of Liabilities

The staff of Wang Laboratories, Inc., has taken due care in preparing this manual; however, nothing contained herein modifies or alters in any way the standard terms and conditions of the Wang purchase agreement, lease agreement, or rental agreement by which this equipment was acquired, nor increases in any way Wang's liability to the customer. In no event shall Wang Laboratories, Inc., or its subsidiaries be liable for incidental or consequential damages in connection with or arising from the use of this manual or any programs contained herein.

Wang Laboratories, Inc., assumes no responsibility for wiring the male connector, furnished with this interface controller board, to any non-Wang equipment. In no event shall Wang Laboratories, Inc., or its subsidiaries be liable for any damages in connection with or arising from the wiring or operation of non-Wang equipment interfaced to a Wang system.

WANG LABORATORIES, INC.
ONE INDUSTRIAL AVENUE, LOWELL, MASSACHUSETTS 01851, TEL. (617) 459-5000, TWX 710 343-6769, TELEX 94-7421

## HOW TO USE THIS MANUAL

Information regarding the installation and operation of Wang's 8-Bit-Parallel I/O Interface Controller is provided in this manual. Chapter 1 describes the features of the controller; strobes and signal levels for control of data input and output are described in detail with respect to the 36-pin Amphenol connector which facilitates direct connection of non-Wang devices to a Wang system. The information is essential for anyone planning to wire the male Amphenol connector to the cable from a device being interfaced to a Wang central processor via the controller described herein.

Chapter 2 and several appendices describe I/O operations using BASIC language statements with built-in or customized signal sequences.

CONTENTS

## FIGURES



## TABLES

# CHAPTER 1

# CONTROLLER FEATURES

## 1.1 GENERAL INFORMATION

Wang's 8-bit-parallel I/O Interface Controller facilitates direct connection of an 8-bit-parallel non-Wang device to a Wang central processor. The device may be one of the following types:

a) input only
b) output only, or
c) input and output.

The I/O Interface Controller is available in physically different but operationally equivalent versions. One version, called the Model 2250, is attached to a mounting bar for plug-in compatibility with any one of the I/O slots in a 2200 Series CPU (Central Processing Unit) such as the 2200T or 2200VP. Another version, called Option 67, has no mounting bar and is configured to fit within the housing of compact units (such as the PCS-II and WCS/15) where the CRT, keyboard, and central processor are assembled in a single unit.

More than one Model 2250 controller can be installed in a Wang system if sufficient I/O slots are available in the 2200 Series CPU. On the other hand, only one Option 67 controller can be installed in a Wang system whose central processor resides inside the console unit.

Installation of an I/O Interface Controller is the responsibility of a Wang Customer Engineer. If a controller arrives as an addition to existing equipment, call the Wang Customer Engineer; do not attempt to install the controller -- any such attempt may void the warranty.

A 36-pin female Amphenol connector supports direct plug-in of a non-Wang device. The connector is located on the Model 2250 mounting bar or on the back panel of a unit containing an Option 67 controller. Although a 36-pin male Amphenol connector is supplied with each 8-bit-parallel I/O Interface Controller, responsibility for wiring the male connector to the cable from an external device is not assumed by Wang Laboratories.

The controller implements the sequential transfer of information one-byte-at-a-time under program control. Data transfer rates depend upon the BASIC language statement being used to control input or output operations. Although Wang systems use an 8-bit character set with 7-bit ASCII codes and the high-order bit set to zero, individual characters in any single or packed 8-bit codes (e.g., EIA, two 4-bit BCD digits, etc.) can be transmitted between a non-Wang device and a Wang system via an I/O Interface Controller. Furthermore, discrete binary information in an 8-bit format can be transmitted via the controller.

To ensure successful data transfer between a Wang system and an interfaced device, the person or persons responsible for interfacing a particular device to a Wang system must consider such factors as the following:

- the I/O specifications of the device,
- the requirements of the online application,
- the characteristics of the I/O Interface Controller, and
- the signal sequences associated with the BASIC language statements by which input or output operations may be controlled.

Appropriate wiring of the 36-pin male Amphenol connector to the cable from a non-Wang device is a prerequisite to successful online operation of a device.

Detailed descriptions of the strobes and signal levels associated with the 36-pin connector are included in this chapter. Chapter 2 contains programming considerations and gives references to sources where BASIC language I/O signal sequences are described.

> NOTE:
>
> The terms input and output, as used in this manual, are chosen from the viewpoint of a Wang central processor and an 8-bit-parallel I/O Interface Controller. Thus, a description of an input strobe or signal level translates into a description of an output strobe or signal level when considered from the viewpoint of an interfaced device.

## 1.2 INPUT/OUTPUT CIRCUITS

Input/output circuitry for the 8-bit-parallel I/O Interface Controller is TTL/DTL* compatible. A typical line receiver (input) circuit is shown schematically in Figure 1-1. A typical line driver (output) circuit is shown schematically in Figure 1-2.

*TTL - transistor-transistor-logic, DTL = diode-transistor-logic

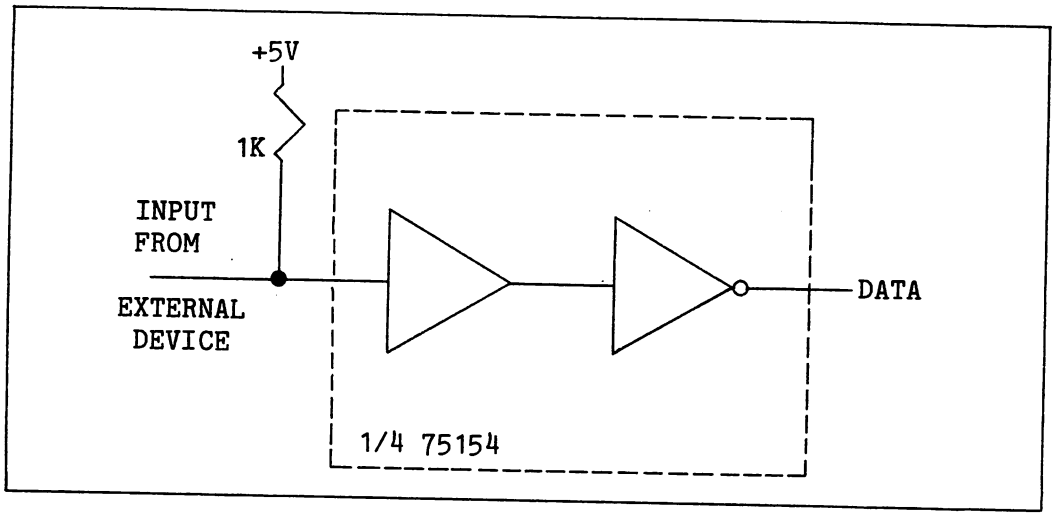



Figure 1-1. A Typical Line Receiver (Input) Circuit

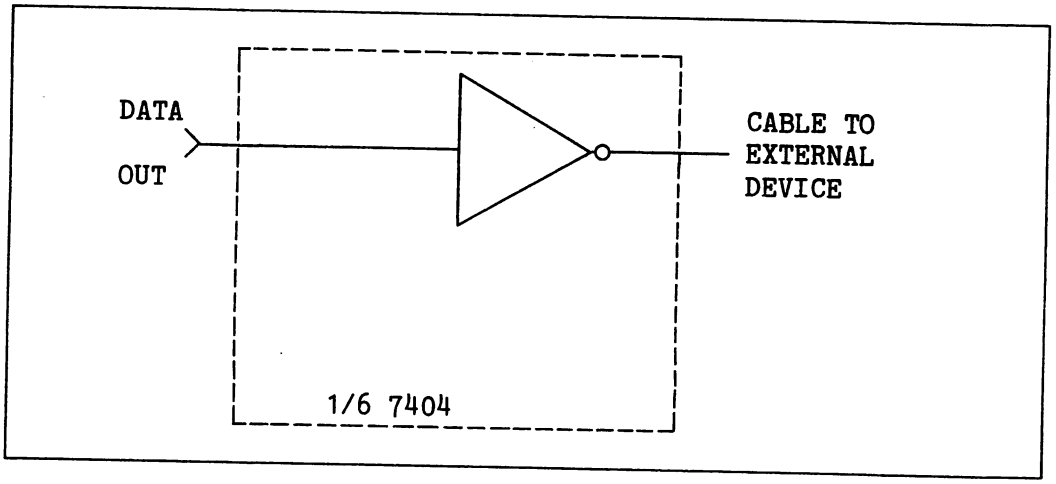



Figure 1-2. A Typical Line Driver (Output) Circuit

NOTES:

1. Use of the controller is adequate for short lines up to 100 feet in length, where environmental noise is not severe.

2. For minimum transmission error, Wang Laboratories recommends that the input/output circuitry of external devices include the same (or equivalent) line driver/receiver circuits as shown in Figures 1-1 and 1-2.

## 1.3 SIGNAL SPECIFICATIONS

Voltage levels for the controller are as follows:

Logic "0" (False) is between +2.4 and 3.6 volts.
Logic "1" (True) is between 0 and +0.4 volts.

The pulse width of output strobes from the controller is 5 microseconds, plus or minus 10%. The pulse width of input strobes from an interfaced device must lie in the range from 5 to 20 microseconds. See Figure 1-3.

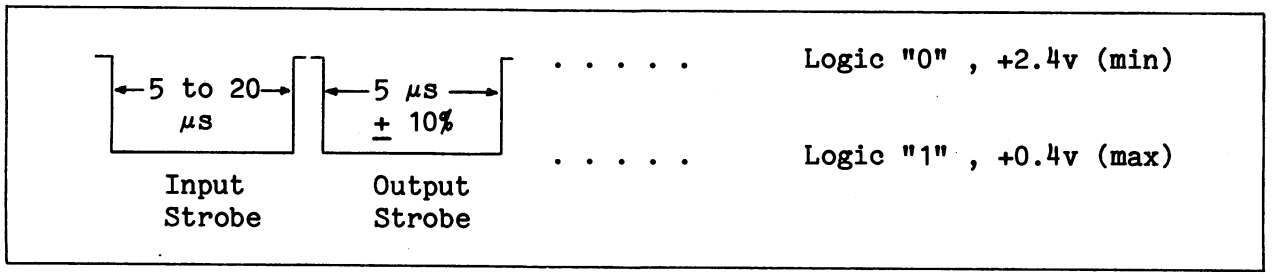



Figure 1-3. Schematic of Input and Output Strobes

NOTES:

1. In this manual, the signal levels are denoted by the abbreviations Hi and Lo, where:

   Hi = high level = logic "0".
   Lo = low level = logic "1".

2. By special request, the level of some signals on the Option 67 version of the I/O Interface Controller can be inverted by a Wang Customer Engineer. However, if such action occurs, it becomes the customer's responsibility to indicate the effective changes in the copy of this manual supplied with the controller.

## 1.4 I/O CONNECTOR PIN ASSIGNMENTS

The functional assignments for the 36-pin Amphenol connector are listed in Table 1-1. An open pin connection for an input circuit is equivalent to a high-level signal (logic "0").

Table 1-1. I/O Connector Pin Assignments

| Pin No. | Signal Mnemonic | I/O* | Remarks | Pin No. | Signal Mnemonic | I/O* | Remarks |
|---|---|---|---|---|---|---|---|
| | | | 8-Bit-Parallel, Buffered Input | | | | 8-Bit-Parallel, Buffered Output |
| 1 | $\overline{IB5}_I$ | I | 10 } High-order | 20 | $\overline{OB1}_O$ | O | 01 } Low-order |
| 2 | $\overline{IB6}_I$ | I | 20 } hexdigit | 21 | $\overline{OB2}_O$ | O | 02 } hexdigit |
| 3 | $\overline{IB7}_I$ | I | 40 } 8-4-2-1 | 22 | $\overline{OB3}_O$ | O | 04 } 8-4-2-1 |
| 4 | $\overline{IB8}_I$ | I | 80 } bit positions. | 23 | $\overline{OB4}_O$ | O | 08 } bit positions. |
| 5 | $\overline{IB1}_I$ | I | 01 } Low-order | 24 | $\overline{OB5}_O$ | O | 10 } High-order |
| 6 | $\overline{IB2}_I$ | I | 02 } hexdigit | 25 | $\overline{OB6}_O$ | O | 20 } hexdigit |
| 7 | $\overline{IB3}_I$ | I | 04 } 8-4-2-1 | 26 | $\overline{OB7}_O$ | O | 40 } 8-4-2-1 |
| 8 | $\overline{IB4}_I$ | I | 08 } bit positions. | 27 | $\overline{OB8}_O$ | O | 80 } bit positions. |
| 9 | $\overline{IBS}_I$ | I | Input Strobe from External Device | 28 | $\overline{DORB}$ | O | Data-Output-Buffer Empty/Full Level |
| 10 | $\overline{PRMS}_O$ | O | Prime Output Strobe | 29 | | | Spare |
| 11 | $\overline{ENDI}_I$ | I | End-of-Input or Special Control Level (Buffered) | 30 | | | Spare |
| 12 | $\overline{COB1}_O$ | O | 1 } 4-bits, buffered | 31 | $\overline{OBS}_O$ | O | Data Output Strobe |
| 13 | $\overline{COB2}_O$ | O | 2 } Control Information | | | | |
| 14 | $\overline{COB4}_O$ | O | 4 } Output | 32 | $\overline{CPB}_O$ | O | CPU/Controller Ready/Busy Level |
| 15 | $\overline{COB8}_O$ | O | 8 } | | | | |
| 16 | $\overline{CBS}_O$ | O | Control Output Strobe | 33 | | | ± 0 volts common |
| 17 | $\overline{IRB}$ | O | Input-Buffer Empty/Full Level | 34 | | | |
| | | | | 35 | | | |
| 18 | $\overline{ACK}$ | I | External Device Acknowledge Strobe | 36 | | | Chassis Ground |
| 19 | $\overline{RBI}$ | I | External Device Ready/ Busy Level | | | | |

*As seen by the Wang system and I/O Interface Controller

## 1.5 SIGNAL MNEMONICS AND DESCRIPTIONS

For easy reference, this section describes the signal mnemonics in alphabetical order. In other sections, functionally related signals are grouped together.

$\overline{ACK}$ -- Pin 18

External Device Acknowledge -- an input strobe from an external device requiring "strobes" longer than 5 $\mu$s. $\overline{ACK}$ resets $\overline{DORB}$ to Hi (logic "0", Ready). Generally, if $\overline{ACK}$ is used, $\overline{DORB}$ should be connected to $\overline{RBI}$ so the controller effectively provides its own ready/busy level for output operations. The leading edge of an $\overline{ACK}$ strobe should not occur before the trailing edge of an $\overline{OBS}_0$ or $\overline{CBS}_0$ strobe.

$\overline{CBS}_0$ -- Pin 16

Control Output Strobe -- a 5 $\mu$s output strobe produced when a $\overline{CBS}$ strobe from the CPU loads one byte of information into the controller's 8-bit Data Output Buffer and simultaneously loads the 4-bit Control Information Buffer with the four low-order bits of the byte.

$\overline{COB1}_0$,$\overline{COB2}_0$,$\overline{COB4}_0$,$\overline{COB8}_0$ -- Pins 12, 13, 14, 15

Control Information Output -- output levels corresponding to the 4-bit Control Information Buffer loaded by a $\overline{CBS}$ strobe. The levels are latched until a subsequent $\overline{CBS}$ strobe occurs.

$\overline{CPB}_0$ -- Pin 32

CPU Ready/Busy -- an output level indicating when the CPU and controller are ready to receive an input strobe from an external device. Hi (logic "0", Ready) indicates two conditions: (1) CPU awaiting input and (2) controller enabled. Lo (logic "1", Busy) is set by the controller when the $\overline{IB1}_I$ through $\overline{IB8}_I$ levels are strobed into the CPU, or at any time the controller is unable to accept data.

$\overline{DORB}$ -- Pin 28

Data Output Buffer Empty/Full -- an output level available for external devices requiring "strobes" longer than 5 $\mu$s. Lo (logic "1", Full) is set when the CPU strobes information via an $\overline{OBS}$ or $\overline{CBS}$ strobe into the controller's one-byte Data Output Buffer. Hi (logic "0", Empty) is reset by an $\overline{ACK}$ strobe. Caution: Since $\overline{DORB}$ is set Lo at the same time data is latched, devices using $\overline{DORB}$ should delay after its leading edge.

$\overline{ENDI}$ -- Pin 11

End-of-Input or Special-Control Level -- an optional ninth input level. If Pin 11 is not connected, $\overline{ENDI}$ is always Hi (logic "0") to indicate standard data on Pins 1 through 8. Some BASIC statements test the buffered $\overline{ENDI}$ level and execute alternative procedures if the level is Lo (logic "1") to indicate a special or termination condition when a byte of information is received. If used, the level must be available at the leading edge of the $\overline{IBS}_I$ strobe.

$\overline{IB1}_I$ through $\overline{IB8}_I$ -- Pins 1 through 8

Input Data (Buffered) -- input levels corresponding to one byte of information in 8-bit-parallel format. The levels must be available at the leading edge of the $\overline{IBS}_I$ strobe. Upon receipt of an $\overline{IBS}_I$ strobe, the levels on Pins 1 through 8 are strobed into the controller's 8-bit Input Buffer and held until $\overline{CPB}_0$ is Hi; then the information is strobed into the CPU buffer.

$\overline{IBS}_I$ -- Pin 9

Input Strobe -- an input strobe indicating the availability of eight data levels, $\overline{IB1}_I$ through $\overline{IB8}_I$, and an optional ninth level, $\overline{ENDI}$.

$\overline{IRB}$ -- Pin 17

Input Buffer Empty/Full -- an output level indicating when the controller's 8-bit Input Buffer contains information awaiting transfer to the CPU. Lo (logic "1", Full) is set by an $\overline{IBS}_I$ strobe; Hi (logic "0", Empty) is reset when the $\overline{IB1}_I$ through $\overline{IB8}_I$ levels are strobed into the CPU by the controller. $\overline{IRB}$ is also available to the CPU for testing if the controller is enabled with its even address (low-order bit "0"). See Section 2.2.

$\overline{OB1}_0$ through $\overline{OB8}_0$ -- Pins 20 through 27

Output Data (Buffered) -- output levels corresponding to one byte of information in the controller's Data Output Buffer. Usually these levels are strobed into the buffer by an $\overline{OBS}$ strobe from the CPU; however, during execution of some BASIC statements and particular $GIO microcommands, the eight levels are strobed into the buffer by a $\overline{CBS}$ strobe. The levels are latched until a subsequent $\overline{OBS}$ or $\overline{CBS}$ strobe occurs.

$\overline{OBS}_0$ -- Pin 31

Output Data Strobe -- a 5$\mu$s output strobe produced when an $\overline{OBS}$ strobe from the CPU loads one byte of information into the controller's 8-bit Data Output Buffer.

$\overline{PRMS}_0$ -- Pin 10

Prime Output Strobe -- a 5 $\mu$s output strobe generated when the RESET button on the Wang system keyboard is depressed to interrupt an operation and return control to the operator. Generally, the signal is used by peripheral devices as a reset/initialization signal.

$\overline{RBI}$ -- Pin 19

External Device Ready/Busy -- an input level from an external device, where Hi (logic "0", Ready) indicates the device is ready to receive Wang system output, and Lo (logic "1", Busy) indicates the device is not ready. $\overline{RBI}$ should be set to Lo within 2 $\mu$s of the leading edge of an $\overline{OBS}_0$ or $\overline{CBS}_0$ strobe; otherwise, a second output strobe may result from one ready level. $\overline{RBI}$ should remain Lo until the trailing edge of an $\overline{OBS}_0$ or $\overline{CBS}_0$ occurs, regardless of the speed of the device. $\overline{RBI}$ is also available to the CPU for testing if the controller is enabled with its odd address (low-order bit "1").

NOTES:

1. By using the $GIO statement to custom-tailor the signal sequence for an I/O operation, data can be sent from the CPU via a $\overline{CBS}$ or an $\overline{OBS}$ strobe, as desired.

2. $\overline{OBS}$ strobes are sent from the CPU on the output bus circuit; $\overline{CBS}$ strobes are sent on the control bus circuit. If an I/O Interface Controller is enabled at the time an $\overline{OBS}$ strobe is sent, 8-bits are strobed into the controller's Data Output Buffer and an $\overline{OBS}_0$ strobe is produced on Pin 31 of the connector. Similarly, if an interface controller is enabled at the time a $\overline{CBS}$ strobe is sent, 8-bits are strobed into the Data Output Buffer, the 4 low-order bits are simultaneously strobed into the Control Information Buffer, and a $\overline{CBS}_0$ strobe is produced on Pin 16 of the connector.

3. If the $\overline{ENDI}$ level on Pin 11 is Lo (Logic "1") when a byte of information is received, a special condition is indicated; however, the significance of the condition depends upon the BASIC statement being executed. Statements such as INPUT and KEYIN utilize the $\overline{ENDI}$ level; also, $GIO has the capability to check $\overline{ENDI}$ if microcommands with valid "Check T" codes are used.

## 1.6 INPUT DATA SIGNALS

Pins 1 through 8 in the 36-pin connector are allocated for 8-bit-parallel input from an interfaced device. The eight levels represented by the mnemonics $\overline{IB1}_I$ through $\overline{IB8}_I$ correspond to one byte of information.

Figure 1-4 shows the correspondence between the connector pins, the signal mnemonics, and the bit positions in the controller's Input Buffer. Labels for the bit positions in the one-byte buffer are defined with respect to the 8-bit character set used by the Wang system (7-bit ASCII codes and the eighth bit set to zero). However, any single or packed 8-bit codes (or discrete binary data) are acceptable as input via the controller.

When an external device sends an $\overline{IBS}_I$ strobe on Pin 9, the levels on Pins 1 through 8 are strobed into the controller's Input Buffer and held or immediately strobed into the CPU, depending upon the timing of the $\overline{IBS}_I$ strobe relative to the $\overline{CPB}_O$ level. The timing is dependent upon the signal sequence of the input operation being executed.

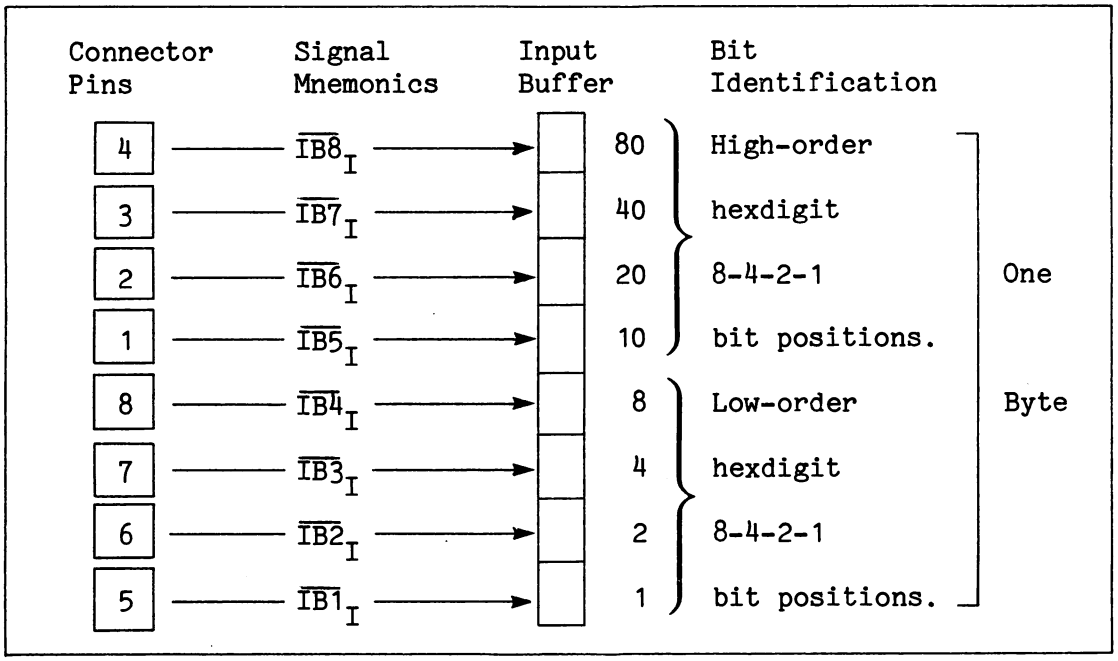



Figure 1-4. Schematic of Pin Assignments for Input Data

NOTE:

The levels $\overline{IB1}_I$ through $\overline{IB8}_I$ on Pins 1 through 8 must be available at the leading edge of an $\overline{IBS}_I$ strobe on Pin 9.

## 1.7 INPUT CONTROL SIGNALS

Strobes and signal levels available for control of input operations are summarized in Table 1-2. Also, for special applications, the levels $\overline{COB1}_O$, $\overline{COB2}_O$, $\overline{COB4}_O$, and $\overline{COB8}_O$ on Pins 12 through 15 can be used to control input (and output) operations.

Table 1-2. Input Control Signals

| Mnemonic | Pin Number | Signal Origin | Signal Type | Remarks |
|---|---|---|---|---|
| $\overline{IBS}_I$ | 9 | external device | Input strobe | Indicates one-byte of data is available on Pins 1 through 8. |
| $\overline{ENDI}$ | 11 | external device | Input level: Hi (or Open)= standard data. Lo = special or termination data. | If used, this optional, buffered, termination or special condition level must be available at the leading edge of $\overline{IBS}_I$. |
| $\overline{CPB}_O$ | 32 | CPU/ controller | Output level: Hi=controller enabled & CPU awaiting input. Lo=CPU busy. | Indicates when the CPU is ready for input via the controller. (See Note 1 on the following page.) |
| $\overline{IRB}$ | 17 | CPU/ controller | Output level: Hi=Empty. Lo=Full. | Indicates when the Input Buffer contains information awaiting transfer to the CPU; $\overline{IRB}$ is also available to the CPU for testing if the controller is enabled with its even address. (See Notes 2 and 3 on the following page.) |
| $\overline{PRMS}_O$ | 10 | CPU/ controller | Output strobe | Produced when the Wang system RESET key is depressed to interrupt an operation and return control to the operator. |

NOTES:

1. Devices using $\overline{CPB}_O$ to govern input to the controller should ignore a Hi (CPU ready) level until an $\overline{IBS}_I$ strobe is completed. Although overlapping strobes will not occur if programming techniques ensure that the CPU time interval between accepting a strobe and becoming ready for another strobe is much longer than an $\overline{IBS}_I$ strobe, the preferred technique is to have the external device hardware design protect against overlapping strobes.

2. Devices using $\overline{IRB}$ to govern input to the controller should ignore a Hi (buffer empty) level until an $\overline{IBS}_I$ strobe is completed; otherwise, overlapping strobes can cause problems which programming techniques cannot solve.

3. If the controller is enabled with its odd address (low-order bit "1") for an input operation, the $\overline{RBI}$ level on Pin 19 (not the $\overline{IRB}$ level on Pin 17) is available to the CPU for testing.

## 1.8 OUTPUT DATA SIGNALS

Pins 20 through 27 in the connector are allocated for 8-bit-parallel output from the Wang system to an interfaced device. The eight levels $\overline{OB1}_O$ through $\overline{OB8}_O$ correspond to one byte of information (8-bits) in the controller's Data Output Buffer.

Figure 1-5 shows the correspondence between the connector pins, the signal mnemonics, and the bit positions in the controller's Data Output Buffer.

During an output operation, the Data Output Buffer is loaded by either an $\overline{OBS}$ or a $\overline{CBS}$ strobe from the CPU, depending upon the BASIC language statement or $GIO microcommand being executed. Therefore, either an $\overline{OBS}_O$ strobe on Pin 31 or a $\overline{CBS}_O$ strobe on Pin 16 is produced by the controller when the Data Output Buffer contains a byte of information.

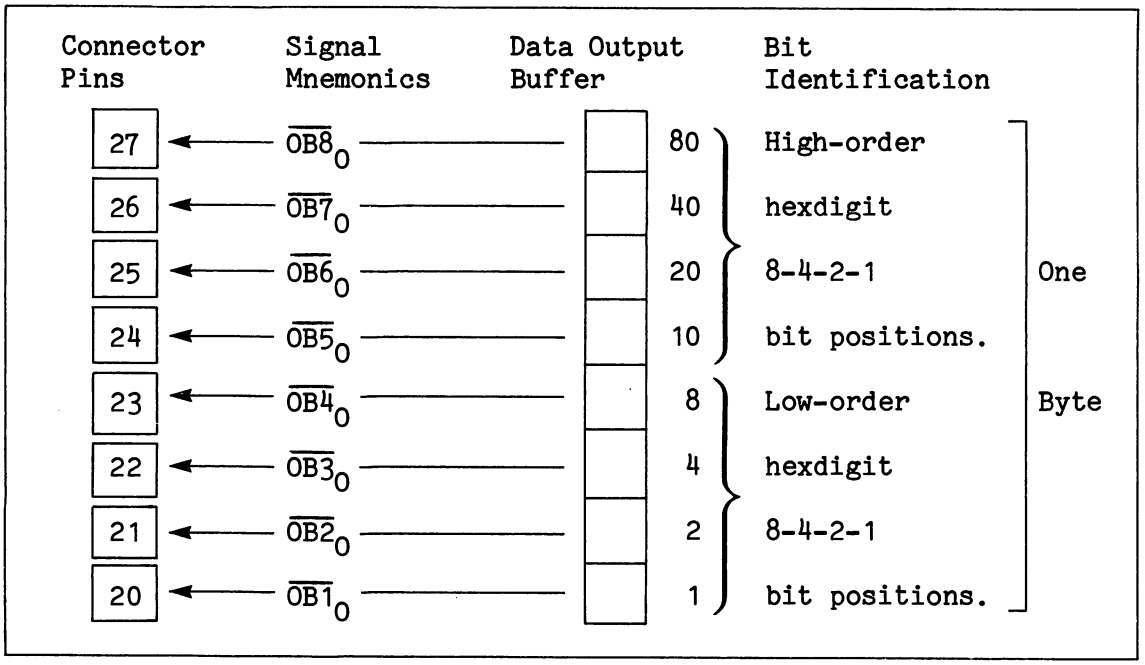



Figure 1-5. Schematic of Pin Assignments for Output Data

NOTES:

1. The eight Data Output Buffer signal levels are latched until a subsequent $\overline{OBS}$ or $\overline{CBS}$ strobe occurs.

2. When a $\overline{CBS}$ strobe from the CPU loads the Data Output Buffer, the strobe also loads the 4-bit Control Information Buffer with signal levels corresponding to the low-order hexdigit. The Control Information Buffer levels are latched until a subsequent $\overline{CBS}$ strobe occurs -- they are not altered by $\overline{OBS}$ strobes. See Section 1.5 and Table 1-3.

## 1.9 OUTPUT CONTROL SIGNALS

Strobes and signal levels available for control of output operations are summarized in Table 1-3.

Table 1-3. Output Control Signals

| Mnemonic | Pin Number | Signal Origin | Signal Type | Remarks |
|---|---|---|---|---|
| $\overline{ACK}$ | 18 | external device | Input strobe | $\overline{ACK}$ resets $\overline{DORB}$ to Hi. $\overline{ACK}$ should be used by devices designed to acknowledge each received strobe with a "handshake" or acknowledgment strobe, or by devices requiring output "strobes" longer than 5 $\mu$s. |
| $\overline{RBI}$ | 19 | external device | Input level: Hi = Ready. Lo = Busy. | $\overline{RBI}$ is available to the CPU for testing if the controller is enabled with its odd address. |
| $\overline{PRMS}_0$ | 10 | CPU/ controller | Output strobe | Produced when the Wang system RESET key is depressed to interrupt an operation and return control to the operator. |
| $\overline{OBS}_0$ | 31 | CPU/ controller | Output strobe | Produced when an $\overline{OBS}$ strobe from the CPU loads the 8-bit Data Output Buffer. |
| $\overline{DORB}$ | 28 | CPU/ controller | Output level: Hi = Empty. Lo = Full. | Set to Lo when either an $\overline{OBS}$ or $\overline{CBS}$ strobe from the CPU loads the Data Output Buffer; reset to Hi by an $\overline{ACK}$ strobe. |
| $\overline{CBS}_0$ | 16 | CPU/ controller | Output strobe | Produced when a $\overline{CBS}$ strobe from the CPU simultaneously loads the 8-bit Data Output Buffer and the 4-bit Control Information Buffer. |
| $\overline{COB1}_0$, $\overline{COB2}_0$, $\overline{COB4}_0$, $\overline{COB8}_0$ | 12, 13, 14, 15 | CPU/ controller | Output levels Hi=logic "0". Lo=logic "1". | Levels strobed into the Control Information Buffer by a $\overline{CBS}$ strobe, and latched until another $\overline{CBS}$ strobe occurs. |

# CHAPTER 2

# PROGRAMMING CONSIDERATIONS

## 2.1 INTRODUCTION

The customized nature of applications involving interfaced non-Wang devices precludes presentation of detailed programming examples; therefore, the information in this chapter is limited to fundamental considerations such as choosing a BASIC language statement for program control of an interfaced device, and using an appropriate address code for a particular statement. For the information to be completely meaningful, readers should be familiar with the Wang system being used in conjunction with an 8-bit-parallel I/O Interface Controller.

## 2.2 DEVICE ADDRESS CODES

Since Wang systems, especially the larger ones, can be configured with several input and output devices, it is usually necessary to identify the particular device to which or from which data is to be transferred. When specifying a device for an I/O operation, the BASIC syntax requires a three-hexdigit (12-bit) code of the form xyy, where

- x, the "device type" digit, usually determines which microcode routines are accessed by the system during execution of the BASIC statement, and
- yy, the "preset address" digits, represent the hexadecimal equivalent of a binary value corresponding to the setting on a device controller's 8-pole address switch.

To ensure address uniqueness for the devices in a particular system, a Wang Customer Engineer sets the address switches in conformity with a standard list when installing a system initially or adding equipment to an existing system. As indicated in Table 2-1, an Option 67 or a Model 2250 I/O Interface Controller is installed with its address switch set to correspond to the hexadecimal value 3A, i.e., the binary value 00111010. A second Model 2250 controller is installed with its address switch set to hex 3C, i.e., 00111100; a third such controller is installed with its switch set to hex 3E, i.e., 00111110.

Although an I/O Interface Controller's address switch is set to an even address (low-order bit "0"), only the seven high-order bits in an address code must match the switch setting in order to enable the controller. Thus, an odd address which differs only in the low-order bit position also enables the controller. (See Table 2-1.)

The choice of an even or odd address affects which signal is made available to the CPU on the controller's $\overline{RB}$ (Ready/Busy) line. The even address makes the $\overline{IRB}$ (Input Buffer Ready/Busy) level available to the CPU; the odd address makes the $\overline{RBI}$ (External Device Ready/Busy) level available to the CPU. For some BASIC statements, only an even or an odd address can be used; for other statements, the choice is optional and should be based upon the characteristics of the interfaced device. Furthermore, for any statement which does not check the $\overline{RB}$-line during execution, the choice is immaterial. Therefore, Section 2.4 provides brief descriptions of several BASIC language statements which may prove useful for control of non-Wang devices, and includes information about the appropriate address codes for input and output operations via an 8-bit-parallel I/O Interface Controller.

Table 2-1. I/O Interface Controller Addresses

| Switch Setting | Corresponding Addresses* | Remarks |
|---|---|---|
| 3A | Even: x3A<br>Odd : x3B | Standard for the Option 67 controller and for one Model 2250 controller (whether one or more such controllers are installed in a system). |
| 3C | Even: x3C<br>Odd : x3D | For a second Model 2250 controller in one system. |
| 3E | Even: x3E<br>Odd : x3F | For a third Model 2250 controller in one system. |

* Recommended or required values for the x-digit are included in the descriptions of BASIC statements in Section 2.4.

## 2.3 DEVICE SELECTION

When a Wang system is Master Initialized (power turned off and then on), "primary devices" are automatically selected for the eight classes (groups) of I/O operations identified by the SELECT statement I/O-class parameters CI (console input), CO (console output), INPUT, PRINT, LIST, TAPE, DISK, and PLOT. To be a primary device, a device must be plugged into a controller whose address code is one of the five default addresses for the system. For example, the keyboard whose controller is preset with the address 01 is a primary device since the address code 001 is selected automatically during master initialization for all subsequent operations governed by the parameters

CI and INPUT. Similarly, the CRT display unit whose controller is preset with the address 05 is a primary device since the address code 005 is selected automatically for all subsequent operations governed by the parameters CO, PRINT, and LIST. The disk or diskette drive corresponding to address code 310 is a primary device. The other default addresses are 10A (for the operations in the I/O-class TAPE) and 413 (for the I/O-class PLOT).

Anyone writing an application program for an interfaced device plugged into an 8-bit-parallel I/O Interface Controller must remember that the device is not a primary device. Hence, to access an interfaced device for an input or output operation, the appropriate even or odd address code of the controller must be specified in at least one of the following ways:

a) directly (using a three-hexdigit address) or indirectly (using a file number) in any BASIC statement whose syntax permits address specification, or

b) in a SELECT statement containing the I/O-class parameter governing the operation, e.g., SELECT INPUT 23A assigns the code 23A to the I/O-class parameter INPUT which governs all subsequent INPUT and KEYIN operations.

> NOTE:
>
> When an optional address is omitted in a BASIC statement, the default address is the address last selected for the I/O-class parameter associated with the operation (or the "primary" address if no selection is made after master initializing the system).

## 2.4 CHOOSING BASIC STATEMENTS

Generally speaking, several BASIC language I/O statements are feasible for program control of non-Wang devices interfaced to Wang systems via 8-bit-parallel I/O Interface Controllers. Brief descriptions of the statements, appropriate address codes, and any special wiring considerations are summarized for input operations in Table 2-2 and for output operations in Table 2-3. More detailed descriptions for the statements are given in Appendices A and B.

Table 2-2. Input Operations

| Statement | Description, Address Code, and Wiring Considerations |
|---|---|
| INPUT | Suitable for devices not requiring initialization signals. Receives ASCII data for storage in one or more numeric and/or alphanumeric variables if the incoming data stream is properly interspersed with commas and carriage return characters serving as data separators. Since the statement does not check the controller's $\overline{RB}$-line, either the even or odd address with device type code 0 or 2 (23A, 23B, 03A, or 03B) is appropriate. (See Appendix A.) |
| MAT INPUT | Similar to INPUT, except receives ASCII data for storage in one or more numeric or alphanumeric arrays. |
| DATALOAD BT | Provides one initialization strobe with a special 8-bit code which can be used or ignored by the device. Receives any 8-bit formatted character codes (or discrete binary data) by providing one request strobe per input character when the CPU via the controller senses $\overline{RBI}$ on Pin 19 is Hi; however, for devices not requiring request strobes, Pin 19 can be left open so $\overline{RBI}$ remains continuously Hi (Ready). Since the statement is designed to check the $\overline{RBI}$ level, the controller's odd address must be used; furthermore, Wang Laboratories supports only device type code 6 (e.g., 63B) in conjunction with the controller. (See Appendix A.) |
| KEYIN | Suitable for reception of a single-character "handshaking" message, e.g., ACK, ENQ, NAK. Also, if programmed in a loop, suitable for character-by-character reception from relatively slow or intermittent devices. Usually, the even address with device type 0 or 2 is preferred (23A or 03A); however, the odd address (23B or 03B) may be used for a device designed to supply its own ready/busy signal. (See Appendix A.) |
| $GIO | Suitable for customizing an input (or output) operation. Particularly useful for devices not easily controlled by BASIC statements having built-in signal sequences. Device type 0 is preferred for all $GIO operations. The controller's odd address should be used for operations whose microcommands include "WR", i.e., CPU awaits enabled device ready signal. (See Appendix C.) |

Table 2-3. Output Operations

| Statement | Description and Appropriate Address Codes |
|---|---|
| PRINT | Outputs the data currently stored in the specified numeric or alphanumeric variables, data specified via HEX functions, and also space characters if necessary to achieve any column positioning specified via TAB functions. Automatically inserts any space characters necessary to achieve zoned, packed, or mixed formats, as specified by the commas and semicolons appearing between arguments in the argument list. Also inserts any carriage return characters necessary to satisfy the currently effective line length, and supplies either a line feed or null character after each carriage return, depending upon the device type code currently in effect for the operation (see Table 2-4). The controller's odd address with device type 0, 2 or 4 should be used. (See Appendix B.) |
| PRINTUSING | Outputs data and interspersed text, as specified in the referenced image (%) statement. The controller's odd address with device type 0, 2 or 4 should be used (see Table 2-4). (Also, see Appendix B.) |
| MAT PRINT | Similar to PRINT, except the argument list can specify numeric and alphanumeric arrays as the source of data to be output. |
| HEXPRINT | Outputs the hexadecimal notation for each byte of data (including any trailing space characters) currently stored in the specified alphanumeric variables and arrays. Data representing individual elements of an array is not separated at element boundaries; however, carriage return characters are inserted, as necessary, to satisfy the currently effective line length. Furthermore, data representing successive arguments in the argument list is not separated if semicolons are used in the list. On the other hand, if a comma separates a pair of arguments, a carriage return is inserted to indicate where one argument ends and the next one begins. The controller's odd address with device type 0, 2 or 4 should be used (see Table 2-4). (Also, see Appendix B.) |
| DATASAVE BT | Outputs data without inserting any special characters to separate the values currently stored in the elements of the specified alphanumeric array. Suitable for transmitting a long data stream with no limit on the number of characters, other than the limit imposed on the specified array by the storage capacity of the CPU. Any 8-bit codes, EIA codes, or packed BCD codes stored in the alphanumeric array can be transmitted from the CPU to the controller. The controller's odd address with device type 4 must be used. (See Appendix B.) |
| $GIO | Suitable for customizing an output (or input) operation. Particularly useful for devices not easily controlled by BASIC statements having built-in signal sequences. Device type 0 is preferred for all $GIO operations. The controller's odd address should be used for operations whose microcommands include "WR", i.e., CPU awaits enabled device ready signal. (See Appendix C.) |

Table 2-4. Effect of the Device Type Code on PRINT, PRINTUSING, and HEXPRINT Statements

| Code | CPU Action |
|---|---|
| 0 | Supplies a line feed (LF) character automatically after each system-generated carriage return (CR) character. Therefore, device type "0" is appropriate for an output operation to a device which does not supply its own LF character after receiving a CR character. |
| 2 | Supplies a null character, $(00)_{16}$, automatically after each system-generated CR. Therefore, device type "2" is appropriate for an output operation to a device which supplies its own LF after receiving a CR. |
| 4 | Suppresses the character count during an output operation and thereby suppresses any CR normally generated when the character count equals the currently selected line length for the output operation. Supplies a LF after any other system-generated CR. Therefore, device type "4" is appropriate for an output operation to a printer which supplies its own CR/LF when the physical carriage width is exceeded. |

APPENDIX A

BUILT-IN SIGNAL SEQUENCES FOR DATALOAD BT, INPUT, AND KEYIN

DATALOAD BT Sequence

If the controller's odd address with device type code 6 is specified, the DATALOAD BT sequence includes the following actions:

| | | |
|---|---|---|
| 1. | Enable | The CPU strobes an 8-bit address to the I/O bus to enable the controller. (Address specification is optional in a DATALOAD BT statement. Priority is given to the specified address, if any; otherwise, the address currently selected for the I/O-class parameter TAPE is used.) |
| 2. | Test device ready | The CPU tests the $\overline{RB}$-line and waits until $\overline{RBI}$ is Hi. (Therefore, the odd address is necessary to make the $\overline{RBI}$ level available to the CPU on the controller's $\overline{RB}$-line.) |
| 3. | Output initialization strobe | The CPU sends an initialization character to the controller's Data Output Buffer via an $\overline{OBS}$ strobe. Then, the controller generates an $\overline{OBS}_0$ strobe on Pin 31 and provides eight data levels on Pins 20 through 27. Specifically, a character $(x1)_{16}$ is provided if the parameter R is specified; otherwise, $(x0)_{16}$ is sent. However, in either case, the value of x is indeterminate. (If the interfaced device does not require an initialization character, the strobe can be ignored.) |
| 4. | Test device ready | Again, the CPU tests the $\overline{RB}$-line and waits until $\overline{RBI}$ is Hi. |
| 5. | Set CPU ready | The CPU sets $\overline{CPB}$ Hi, which also sets $\overline{CPB}_0$ Hi, thereby ensuring that a received character is strobed into the CPU and not held in the controller's Input Buffer. |
| 6. | Output request strobe | The CPU/controller generates a character request strobe $\overline{CBS}_0$ on Pin 16 which may be used to initiate an $\overline{IBS}_I$ strobe and eight data levels from the device; however, the $\overline{CBS}_0$ strobe can be ignored by a device not requiring a request strobe, and the $\overline{CPB}_0$ level on Pin 32 can be used to initiate/inhibit $\overline{IBS}_I$. |
| 7. | Receive a character | Within 4 $\mu$s of the leading edge of an $\overline{IBS}_I$ strobe, the controller strobes the eight levels on Pins 1 through 8 into the CPU and sets $\overline{CPB}_0$ Lo. Alternatively, if appropriate for a device, the $\overline{CBS}_0$ strobe on Pin 16 can be used in lieu of $\overline{IBS}_I$ by wiring Pin 16 to Pin 9. |

8. Receive remaining characters — Steps 4 through 7 are repeated until any one of the following conditions occurs:

   - the N parameter (denoting the number of characters to be received) is satisfied, if specified, or
   - the S parameter (denoting a stop/termination code) is satisfied, if specified, or
   - the specified receiving variable or array becomes full.

9. Disable — The CPU disables the controller and advances program execution to the next statement.

INPUT Sequence

The INPUT sequence includes the following actions:

1. Prompt — The CPU enables the currently selected CO (console output) device to send the literal string message, if any, and a question mark (plus a space character) to indicate the system is awaiting input.

2. Enable — The CPU strobes an 8-bit address to the I/O bus to enable the controller. (Since the INPUT syntax does not permit address specification, the address currently selected for the I/O-class parameter INPUT is used.)

3. Set ready — The CPU sets $\overline{CPB}$ Hi, which also sets $\overline{CPB}_0$ Hi, thereby indicating the system is ready to receive an $\overline{IBS}_I$ strobe from the interfaced device.

4. Receive input strobe — The controller awaits an $\overline{IBS}_I$ strobe on Pin 9 and eight levels on Pins 1 through 8, representing one character in 8-bit parallel format. (The system waits indefinitely if no strobe is received.)

5. Receive one character — Within 4 $\mu$s of the leading edge of an $\overline{IBS}_I$ strobe, the controller strobes the 8-bit character directly into the CPU, and also sets $\overline{CPB}_0$ Lo. (The $\overline{ENDI}$ level is also strobed into the CPU since provision is made in the INPUT statement to handle input via a special function key which automatically sets $\overline{ENDI}$ Lo.)

6. Temporarily disable

The CPU temporarily disables the controller for echo and character processing as follows (assuming $\overline{ENDI}$ is Hi):

- The received character is compared with a special character list, where $(08)_{16}$ or $(5F)_{16}$ represents a backspace instruction which effectively removes the previously received character, and $(5C)_{16}$ represents a line erase instruction which effectively removes all currently buffered received characters; none of these codes is stored in the buffer.
- A received $(00)_{16}$ or $(7F)_{16}$ code is not stored.
- Any code above $(7F)_{16}$ may cause unpredictable action.
- An $(0D)_{16}$ carriage-return code is stored, but interrupts reception, as described in Step 9.
- Otherwise, the received character is stored in the CPU buffer, and the character count is incremented by one.

7. Echo the character

The CPU enables the currently selected CO (console output) device and sends a duplicate (an echo) of the buffered character, or any backspace or line erase action which occurs.

8. Receive subsequent characters

Steps 2 through 7 are repeated until an $(0D)_{16}$ is stored in the CPU buffer, or the character count reaches a system-related value (e.g., 191 for a 2200T central processor), whichever occurs first. If the count reaches the maximum value without an $(0D)_{16}$, the CPU disables the controller, enables the CO device to send an ERR 45 code, ignores the currently buffered data, resets the character count to zero, and returns to Step 2.

9. Process buffered characters

Upon receipt of an $(0D)_{16}$ character, the CPU processes any currently buffered characters for storage in the receiving variables specified in the INPUT statement argument list; the procedure includes the following actions:

- If no data precedes the $(0D)_{16}$, statement execution ends.
- Otherwise, the buffer is searched for a data separator comma, $(2C)_{16}$, i.e., a comma not embedded in data delimited by double or single quotation mark codes, $(22)_{16}$ or $(27)_{16}$.

(continued on next page)

- If a data separator comma is located, the current position of the argument list pointer is checked to determine which variable is to receive the data preceding the comma (or preceding the carriage return character if no other data separator is found).

- If the currently flagged receiving variable is numeric, a data validity check is made. If any illegal character is found, an ERR 29 code is sent to the CO device, the data currently being processed (and all remaining buffered data) is ignored, the argument list pointer does not move, and execution returns to Step 2. Otherwise, the data is converted to Wang's internal numeric format and stored in the receiving variable; then, the argument list pointer moves to the next receiving variable.

- If the currently flagged receiving variable is alphanumeric, data is transferred to memory character-by-character until the allocated storage area is filled, or the data separator is reached. Any excess characters are ignored; trailing space characters, $(20)_{16}$, are automatically supplied if the storage area is not filled by the received data. Similarly, if a string function (STR) is in the currently flagged argument list position, data is transferred only to the designated byte positions (excess characters are ignored or trailing space characters added). If the buffered data being transferred is delimited by single quotation marks, any uppercase codes are converted to lowercase before storage. When data transfer to the flagged variable is completed, the argument list pointer moves to the next receiving variable.

- The buffer is searched for the next data separator comma if the previously transferred data was delimited by a comma. If a comma is found, the data transfer cycle is repeated, as described, depending upon the nature of the next receiving variable. The process continues until no receiving variables remain, or the $(0D)_{16}$ character is reached.

10. Receive other characters

If the $(0D)_{16}$ code has been reached, but another receiving variable remains, Steps 2 through 9 are repeated until the last variable is reached, or an $(0D)_{16}$ is received with no other data.

11. Disable

After data is transferred to the last receiving variable, or a carriage-return without data is encountered, the CPU disables the controller and moves to the next statement in the program.

## Echo Suppression

If desired, the INPUT-data-echo normally displayed on the CRT screen can be suppressed by programming techniques. For example, assume the following:

a) The CRT echo of INPUT-data is undesirable for an application requiring a particular CRT display to remain undisturbed while data is being read from an external device.

b) The external device is an input-only device not wired to receive output from the CPU via the controller; Pin 19, $\overline{\text{RBI}}$, is open (Hi = logic "0" = Ready).

c) The external device is plugged into an 8-bit-parallel I/O Interface Controller whose address switch is set to 3A. (See Table 2-1.)

Now, consider the following programming technique. Whenever the input address of the controller is selected for execution of one or more INPUT statements, select the output address of the controller for CO-class operations. By this technique, the INPUT-data-echo is sent to the output buffer on the controller rather than to the CRT screen. However, since the external device is an input-only device with respect to the CPU and controller (by our assumption), the echo characters are ignored. The following program demonstrates the technique of echo suppression:

| Statements | Comments |
|---|---|
| 10 DIM A$(100)35 | Dimension the A$-array: 100 elements with a maximum of 35 characters per element. |
| 20 SELECT INPUT 23A, CO 23B | Select the controller for subsequent INPUT-class and CO-class operations (23B can be assigned to the INPUT parameter). |
| 30 FOR J=1 TO 100 | Set up a FOR-NEXT loop to execute Line 40 repeatedly. |
| 40 INPUT A$(J) | Receive data and store in the Jth element of the A$-array. |
| 50 NEXT J | Increment J and loop. |
| 60 SELECT INPUT 001, CO 005 | Reselect the keyboard for INPUT-class operations and the CRT for CO-class operations. |

NOTES:

1. The programming technique illustrated by Line 20 is not suitable for echo suppression if the external device receives and responds to data sent from the CPU via the controller.

2. When assigning an address, other than the CRT address, to the CO-parameter in a SELECT statement to suppress the CRT INPUT-data-echo, do not assign a nonexistent device address to the CO-parameter. If such an address is used, the system locks out when the program is run with no controller matching the address. To regain control, the operator must Master Initialize the system which clears all memory.

3. If desired, assign the address of the Line Printer (usually 215) or the Output Writer (usually 211) to the CO-parameter in the SELECT statement, and thereby obtain a hardcopy of the INPUT-data-echo.

4. Always include a program statement reselecting the CRT (usually code 005) for CO-class operations, as illustrated in Line 60. Then, when program execution is complete, the CRT displays the system-generated colon which is sent automatically to the CO-device to indicate return of system control to the CI (console input) device.

## KEYIN Sequence

The KEYIN sequence includes the following actions:

1. $IBS_I$ received/ not received — The interfaced device can send an $\overline{IBS}_I$ strobe on Pin 9, eight levels on Pins 1 through 8, and an optional $\overline{ENDI}$ level on Pin 11 at any time, without checking $\overline{CPB}_O$ on Pin 32. However, unless the device checks $\overline{IRB}$ on Pin 17, a character may be overwritten and thereby lost before its transfer to the CPU occurs.

2. Enable — The CPU strobes an 8-bit address to the I/O bus to enable the controller. (Since the KEYIN syntax does not permit address specification, the address currently selected for the I/O-class parameter INPUT is used.)

3. Test for character — The CPU tests the controller's $\overline{RB}$-level, which is either the $\overline{IRB}$ level on Pin 17 (if the address is even) or the $\overline{RBI}$ level on Pin 19 (if the address is odd).

4. If $\overline{RB}$ = Hi* — The CPU disables the controller, terminates the KEYIN operation, and advances program execution to the next statement.

5. If $\overline{RB}$ = Lo* — The procedure is as follows:

   - The CPU sets $\overline{CPB}$ Hi, which also sets $\overline{CPB}_0$ Hi.
   - The controller then strobes the eight Input Buffer levels and the $\overline{ENDI}$ level into the CPU.
   - The CPU disables the controller, stores the 8-bits of data (one byte) in the specified alpha variable, and then executes a program branch as follows:
     - to the first specified line number if $\overline{ENDI}$ is Hi, or
     - to the second specified line number if $\overline{ENDI}$ is Lo.

* With an even address in Step 3,

$\overline{RB}$ is { Hi, logic "0" / Lo, logic "1" } when $\overline{IRB}$ on Pin 17 is { Hi, logic "0", buffer empty / Lo, logic "1", buffer full };

alternatively, with an odd address in Step 3,

$\overline{RB}$ is { Hi, logic "0" / Lo, logic "1" } when $\overline{RBI}$ on Pin 19 is { Lo, logic "1", device busy / Hi, logic "0", device ready }.

Note that the Pin 19 logic, as seen by the CPU on the $\overline{RB}$-line, is inverted if an odd address is used for a KEYIN operation. For many applications suited to the KEYIN statement, the even address is preferred; however, the odd address may be used to control input from a device designed to supply its own ready/busy signal.

The following program can be used to demonstrate the conditional branches of the KEYIN statement. In the program, line 20 is executed repeatedly by an endless loop. Each time no character is found in the keyboard controller buffer, the line 30 message is displayed. However, touching alpha or numeric keys on the keyboard produces the line 50 message, while touching special function keys produces the line 70 message.

| Statements | Remarks |
|---|---|
| 10 DIM A$1 | Dimension A$ for one character. |
| 20 KEYIN A$, 50, 70 | Scan the keyboard for input. |
| 30 PRINT "*******" | |
| 40 GOTO 20 | |
| 50 PRINT "---50---" | |
| 60 GOTO 20 | |
| 70 PRINT "S.F. KEY" | |
| 80 GOTO 20 | |

By including the following line:

```
15 SELECT INPUT xyz
```

where xyz is replaced by the input address of an I/O Interface Controller, the program above can be used to demonstrate KEYIN operations for an external device plugged into the controller.

APPENDIX B

BUILT-IN SIGNAL SEQUENCES FOR PRINT, PRINTUSING, HEXPRINT, AND DATASAVE BT

The built-in signal sequences for PRINT, HEXPRINT, and DATASAVE BT statements are essentially the same. The sequences described here assume that the device controller currently selected for any one of these operations is an I/O Interface Controller whose odd address is assigned to the I/O-class parameter governing the particular operation (or specified in a DATASAVE BT statement).

Two sequences are described here:

1) a "normal" output sequence and
2) an "acknowledge" output sequence.

The normal output sequence occurs if a Ready/Busy level $\overline{RBI}$ is supplied by the external device on Pin 19 of the connector. The acknowledge output sequence occurs if the $\overline{DORB}$ level on Pin 28 is connected to Pin 19 for applications where an external device requires output "strobes" longer than the 5$\mu$s strobe normally supplied by the CPU/controller; in such cases, the external device supplies an $\overline{ACK}$ strobe via Pin 18 rather than an $\overline{RBI}$ level on Pin 19.

The normal output sequence is as follows:

1. Enable — The CPU strobes the 8-bit address code to the I/O bus to enable the controller. When enabled with its odd address, the controller makes the $\overline{RBI}$ level on Pin 19 available to the CPU on the $\overline{RB}$-line to test the external device Ready/Busy level.

2. Test ready — If the $\overline{RBI}$ level on Pin 19 is Lo (logic "1", Busy), the CPU continues a test loop until a Hi level is sensed.

3. Output one character — When $\overline{RBI}$ on Pin 19 becomes Hi (logic "0", Ready), the CPU strobes an 8-bit character into the Data Output Buffer on the controller. The controller then makes eight parallel data levels (corresponding to the character) available on Pins 20 through 27 and makes a 5$\mu$s $\overline{OBS}_0$ strobe available on Pin 31. Although the controller also sets the $\overline{DORB}$ level on Pin 28 to Lo (logic "1", Full), $\overline{DORB}$ is not utilized in the normal sequence.

4. Hold — Upon receipt of the $\overline{OBS}_0$ strobe, the external device sets the $\overline{RBI}$ level on Pin 19 to Lo (logic "1", Busy). The $\overline{RBI}$ level remains Lo while the external device is processing the character. When processing is complete, the external device resets $\overline{RBI}$ to Hi.

5. Output successive characters — Steps 2, 3, and 4 are repeated if the previous character is not the final character of the output operation. (A ready test is not made after the final character is output.)

6. Disable — The CPU disables the controller and program execution advances to the next statement.

The acknowledge output sequence is as follows:

1. Enable — The CPU strobes the 8-bit address code to the I/O bus to enable the controller. When enabled with its odd address, the controller makes the $\overline{RBI}$ level on Pin 19 available to the CPU on the $\overline{RB}$-line to test the external device Ready/Busy level. However, since Pin 28 is tied to Pin 19 (rather than using an external device ready/busy level), the CPU effectively tests the $\overline{DORB}$ level. Initially, $\overline{DORB}$ = Hi (logic "0", Empty); therefore, $\overline{RBI}$ = Hi (logic "0", Ready).

2. Test ready — Whenever the level on Pin 19 is Lo (logic "1", Busy), the CPU continues a test loop until a Hi level is sensed.

3. Output one character — When the level on Pin 19 becomes Hi (logic "0", Ready), the CPU strobes an 8-bit character into the Data Output Buffer on the controller. The controller makes the eight data levels available on Pins 20 through 27 and sets the $\overline{DORB}$ level on Pin 28 to Lo (logic "1", Full). (The controller also provides a 5$\mu$s $\overline{OBS}_0$ strobe on Pin 31; however, the strobe need not be utilized by the external device.)

4. Hold — The $\overline{DORB}$ level on Pin 28 remains Lo while the external device is receiving and processing the data levels. Since Pin 28 is tied to Pin 19, the $\overline{RBI}$ level is also Lo. When processing is complete, the external device sends an acknowledge strobe.

5. Receive acknowledge strobe — When the external device sends an $\overline{ACK}$ strobe on Pin 18, the controller resets the $\overline{DORB}$ level on Pin 28 to Hi (logic "0", Empty). Thus, the $\overline{RBI}$ level on Pin 19 is reset to Hi.

6. Output successive characters — Steps 2 through 5 are repeated if the previous character is not the final character of the output operation. (A ready test is not made after the final character is output.)

7. Disable — The CPU disables the controller and program execution advances to the next statement.

> NOTE:
>
> In Step 5 of the normal output sequence and Step 6 of the acknowledge output sequence, several system-generated characters may occur in addition to the data corresponding to the argument list of the particular BASIC statement being executed. See Tables 2-3 and 2-4.

APPENDIX C

CUSTOMIZED SIGNAL SEQUENCES FOR I/O OPERATIONS

In contrast to the statements with built-in signal sequences, the $GIO (i.e., the General Input/Output) statement fits into the framework of the high-level BASIC language, yet provides the capability to customize input and output operations by a technique similar to machine-language programming. Effectively, the user tailors a signal sequence to the specifications of a particular input or output device by choosing one or more microcommands to define the desired input or output operation.

Since a microcommand is a four-hexdigit-code denoting several fundamental operations, the available microcommands serve as "building blocks" which can be assembled in a variety of ways to define a relatively simple, or a very complex, input or output operation. In accordance with the syntax of the $GIO statement, the microcommand sequence defining a custom-tailored operation can be specified directly or indirectly as the first argument (arg-1) in a particular $GIO statement. The $GIO syntax and the available microcommands are described in detail in at least one of the manuals provided with each Wang system whose central processor includes such a statement in its instruction set. Except for the 2200VP central processor, the $GIO statement is described in the General I/O Instruction Set Reference Manual; for the 2200VP central processor, the $GIO statement is described in the BASIC-2 Language Reference Manual.

The capability to customize input and output operations via $GIO statements is almost limitless since individual microcommands can implement one or more diverse functions such as the following:

- setting a delay condition applicable to subsequent output of each character
- setting a timeout condition applicable to subsequent sensing of device-ready signals and data input
- disabling a previously enabled delay or timeout condition
- storing special characters in particular registers
- comparing registers and setting error flags or terminating the operation
- outputting immediate or indirect (stored) characters with or without awaiting an acknowledge or echo character
- outputting a device address to deselect a currently selected device and select a different device
- inputting single or multicharacter data with or without echoing or verifying the received data
- sending strobes to request each character during a multicharacter input operation

- setting CPU ready signals
- awaiting device ready signals
- terminating an operation by count or special-character-comparison
- calculating, sending, and/or saving the longitudinal redundancy check (LRC) character for a multicharacter output or input operation.

The registers (byte-positions) in the variable specified as the arg-2 component of a $GIO statement are reserved for storage of the following types of information:

- an indirect character for a single-character output operation or for comparison during a single-character input with verify operation
- a two-byte value defining a delay or a timeout interval
- a special character defining a termination condition for a multicharacter input operation
- an acknowledge or echo character received after a single-character output operation
- an LRC character calculated during a multicharacter input or output operation
- a two-byte binary count of the total number of transferred characters (whether stored or not) when a buffer overflow occurs during multicharacter input
- error/status flags (stored on a bit-by-bit basis in Register 8) indicating such conditions as buffer overflow, LRC error, echo/verify error, compare error, timeout exceeded, and termination (by count, special character, or ENDI-level).

Example

The following example illustrates some capabilities of the $GIO statement. The microcommand sequence includes operations which might be useful for data output to a punch tape unit. The microcommand 4011 sends the code $(11)_{16}$, an X-ON character, to the unit. The microcommands 1221, 7105 and 4000 send a null character to the CRT (after a delay of 25.6 milliseconds) to allow the motor on the punch unit sufficient time to reach a specified condition. The delay is disabled before the multicharacter output operation, represented by A000, begins. Finally, the microcommand 4013 sends the code $(13)_{16}$, an X-OFF character, to the unit.

$GIO /03B (0202 0300 4011 1221 7105 4000 713B 1200 A000 4013, R$) B$()

| Microcommand | Function |
|---|---|
| 0202 | Store the character $(02)_{16}$ in Register 2 (the second byte of R$). |
| 0300 | Store the character $(00)_{16}$ in Register 3. |
| 4011 | Send the character $(11)_{16}$ to the currently selected address, i.e., 3B. |
| 1221 | Set a delay condition equal to 50 microseconds multiplied by the two-byte binary value stored in Registers 2 and 3. |
| 7105 | Deselect the current address and select the address 05 (the CRT). |
| 4000 | Send the null character $(00)_{16}$ to the current address (now the CRT). |
| 713B | Deselect the current address and select the address 3B. |
| 1200 | Disable the delay specified by the microcommand 1221. |
| A000 | Output each character stored in the buffer B$(), using the following sequence:<br><br>WR = wait for a ready signal from the enabled device.<br>DATAOUT/OBS = send the next character with an OBS strobe.<br>LEND = the LRC End sequence specified by hexdigit $h_4$. Since $h_4$ = 0, the sequence is "None". |
| 4013 | Send the character $(13)_{16}$ to the currently selected address, i.e., 3B. |

NOTE:

Before an appropriate microcommand sequence can be chosen for program control of any device, the application and hardware requirements must be defined and related to the inherent features of the available microcommands. Therefore, the example given in this section should not be used to control a particular device unless the microcommands and the hardware requirements are understood thoroughly and deemed compatible.

APPENDIX D

THE CONTROLLER ADDRESS SWITCH

Usually, a System 2200 interface controller is assigned different address codes for input and output operations only if the board contains two 8-pole address switches (one located in the input channel circuitry and the other located in the output channel circuitry). Such is not the case for the 8-bit-parallel I/O Interface Controller whose design utilizes only one 8-pole address switch.

Figure D-1 shows a diagram of an 8-pole switch with a nonexistent grid added for reference purposes. Numbers on the board are bit-position indicators for setting (or verifying) the binary equivalent of a two-hexadecimal-digit address assigned to a controller.

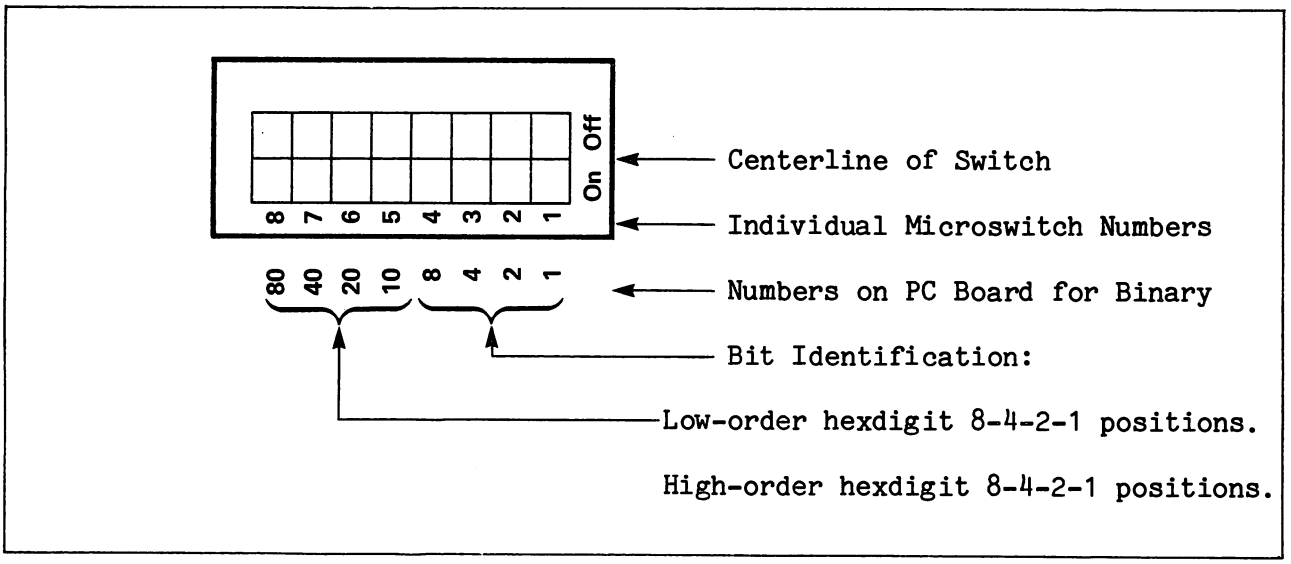



Figure D-1. Diagram of 8-Pole Address Switch

The following conditions hold for each microswitch in an 8-pole address switch:

OFF = logic "0" = High-level.
ON = logic "1" = Low-level.

Ordinarily, the ON-OFF configuration of an 8-pole address switch must correspond to the ONE-ZERO configuration of all the bits in the 8-bit binary equivalent of the last two hexdigits in the three-hexadecimal-digit (12-bit) code used for program control of a specific controller board (or one of its channels). However, only seven microswitches in the 8-pole switch on the Model 2250 or Option 67 board are connected. The Position 1 (low-order bit) microswitch is always open (logic "0") whether set ON or OFF because the controller is designed to be enabled when the seven high-order bits of an address strobe match the preset address of the board.

Table D-1 contains a list of the 8-bit binary codes equivalent to the hexadecimal codes 3A through 3F which are used for the standard address codes for Model 2250 boards (see Section 2.2). Note in Table D-1 that the seven high-order bits in the binary codes for HEX(3A) and HEX(3B) are identical. Also, the seven high-order bits in the binary codes for HEX(3C) and HEX(3D) are identical, but different from the seven identical bits in the first pair. A similar observation can be made for HEX(3E) and HEX(3F).

Table D-2 shows the ON-OFF configuration for each of the six codes in Table D-1. Note that the ON-OFF configuration of the seven high-order microswitches determines whether the controller address switch is set for the two-hexdigit code 3A, 3C, or 3E.

Once a setting of 3A (or 3B) is made on the controller address switch, either code enables the interface controller for an I/O operation. Both input and output strobes can function in either mode of operation. However, the low-order bit (e.g., "0" in 3A, "1" in 3B) determines which one of the Ready/Busy signal levels is presented to the CPU on the $\overline{RB}$-line shown in the Appendix E and F schematics.

If the low-order bit in an address strobe (whose seven high-order bits match the Position 2 through 8 microswitch configuration) is "0", the controller makes the $\overline{IRB}$ signal level on Pin 17 (the Input Buffer Ready/Busy state) available to the CPU for testing. On the other hand, if the low-order bit in the address strobe is "1", the controller makes the $\overline{RBI}$ level on Pin 19 (the External Device Ready/Busy state) available to the CPU for testing.

Table D-1. Hexadecimal and Binary Equivalents for the Address Switch

| Unit | Hexadecimal | Binary Equivalent |
|---|---|---|
| 1 | 3A<br>3B | 00111010<br>00111011 |
| 2 | 3C<br>3D | 00111100<br>00111101 |
| 3 | 3E<br>3F | 00111110<br>00111111 |

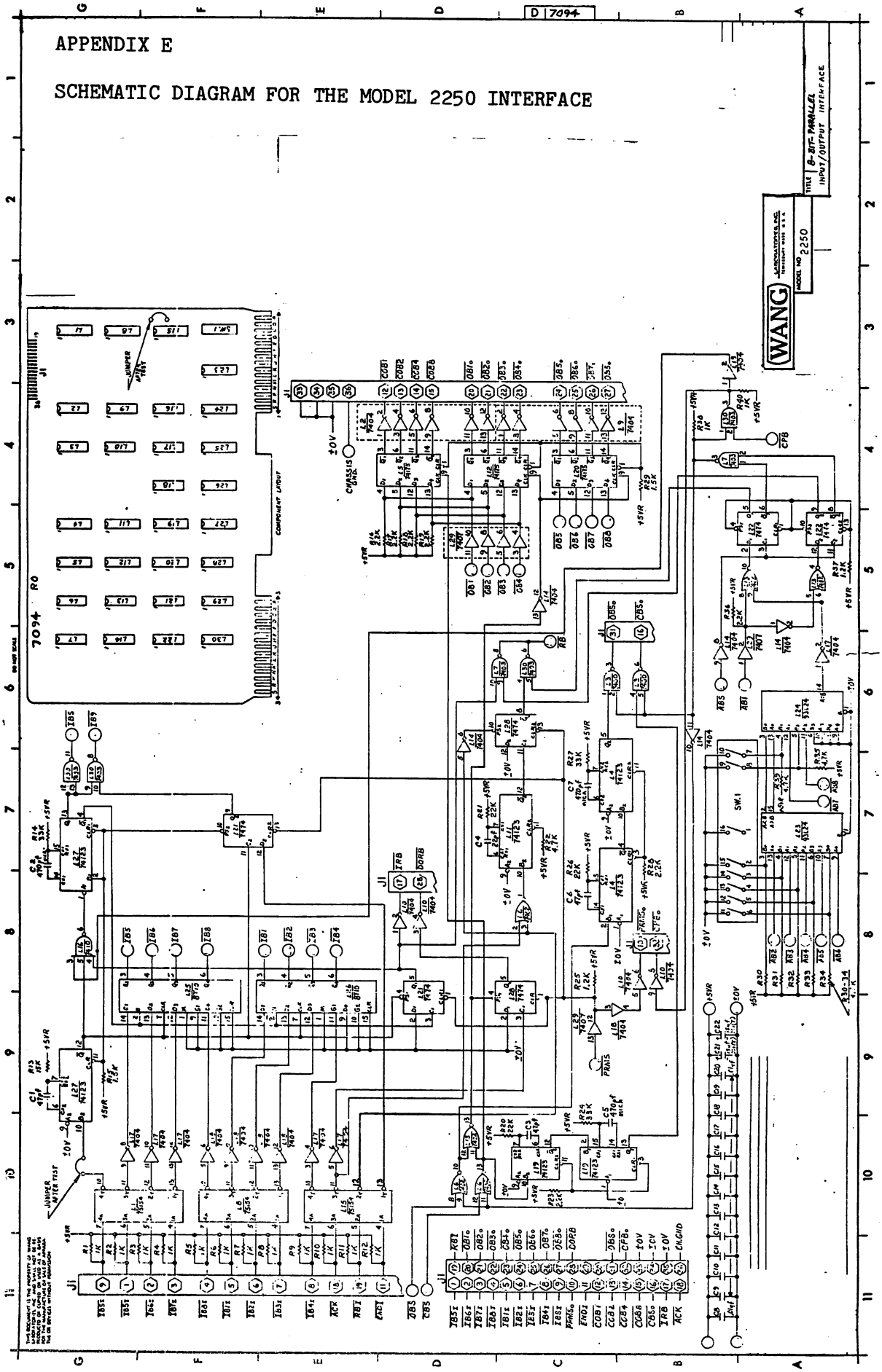
Table D-2. ON-OFF Configuration For Address Switch

| Unit | I/O Codes | ON-OFF Configuration* |
|---|---|---|
| 1 | 3A/3B | FFNNNFNX |
| 2 | 3C/3D | FFNNNNFX |
| 3 | 3E/3F | FFNNNNNX |

*F = OFF, N = ON, X = either OFF or ON.

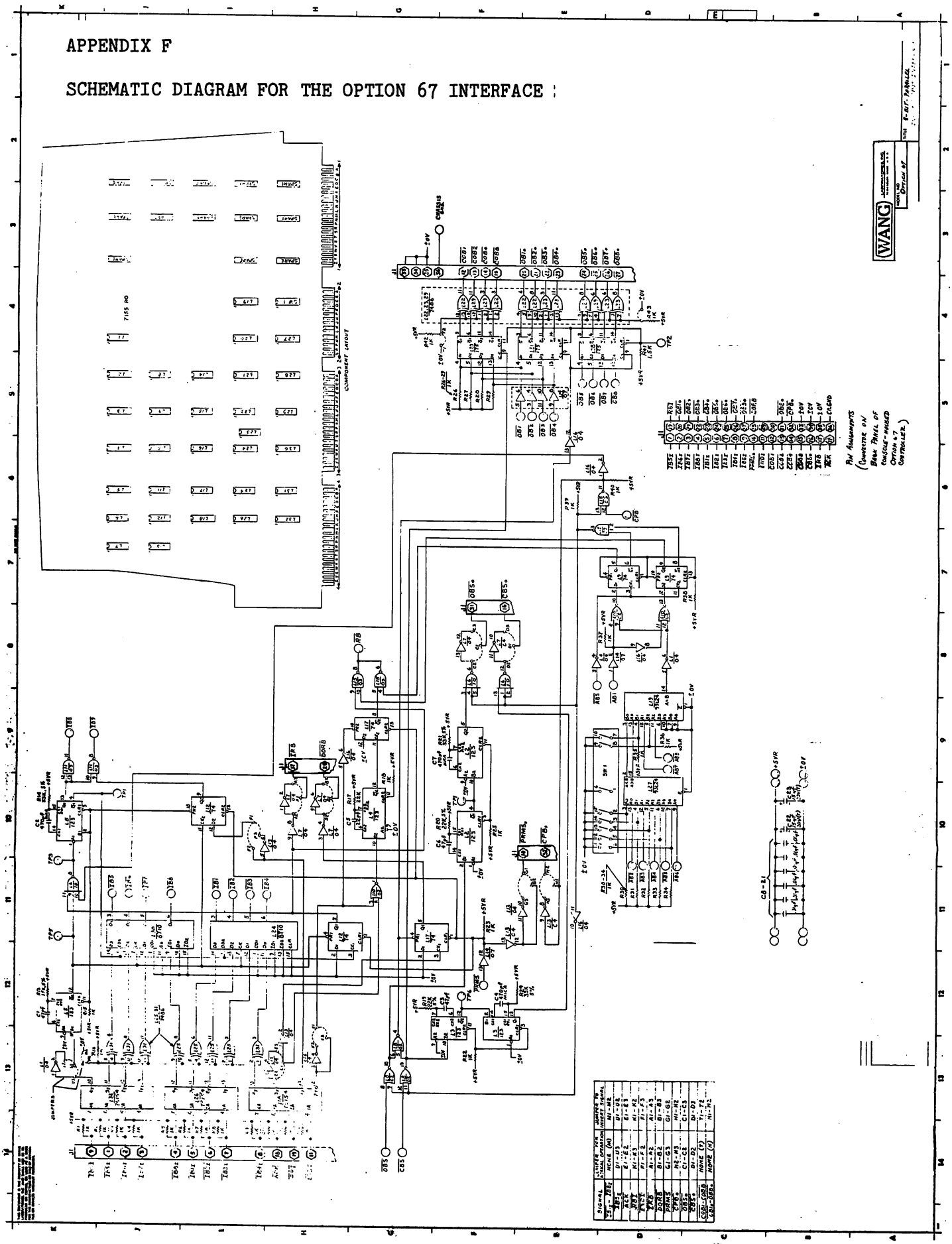
# APPENDIX E

## SCHEMATIC DIAGRAM FOR THE MODEL 2250 INTERFACE



APPENDIX F

SCHEMATIC DIAGRAM FOR THE OPTION 67 INTERFACE :



APPENDIX G

SIGNAL LEVEL INVERSION (OPTION 67 ONLY)

As indicated in Section 1.3 (see Note 2), signal level inversion can be performed, upon request, by a Wang Customer Engineer. The following signals may be inverted, individually or in groups, in any combination:

$\overline{ACK}$

$\overline{CBS}_O$

$\overline{COB1}_O$ through $\overline{COB8}_O$ (in a group of four)

$\overline{CPB}_O$

$\overline{DORB}$

$\overline{ENDI}$

$\overline{IB1}_I$ through $\overline{IB8}_I$ (in a group of eight)

$\overline{IBS}_I$

$\overline{IRB}$

$\overline{OB1}_O$ through $\overline{OB8}_O$ (in a group of eight)

$\overline{OBS}_O$

$\overline{PRMS}_O$

$\overline{RBI}$

Unless a signal is inverted, the normal operation described in Section 1.5 applies.

| NOTE: |
| --- |
| Signal inversion by anyone other than a Wang Customer Engineer voids the warranty. |

APPENDIX H

SPECIFICATIONS

| | |
|---|---|
| Input/Output Circuitry: | TTL/DTL compatible. |
| Voltage Levels: | Logic "0" (false) between +2.4 and 3.6 volts.<br>Logic "1" (true) between 0 and +0.4 volts. |
| Strobes: | Input strobe pulse width must be 5 to 20 microseconds.<br>Output strobe pulse width is 5 microseconds $\pm$ 10%. |
| Power Requirements: | Supplied by the central processor. |
| Electrical Connection: | A 36-pin female Amphenol connector is attached to the interface controller. A 36-pin male Amphenol connector, to be wired to the cable from a non-Wang device, is furnished as an accessory. |
| Data Transfer: | Sequential transfer of 8-bit-parallel information, under program control. |

Standard Warranty Applies.

## EQUIPMENT MAINTENANCE

It is recommended that your equipment be serviced annually. A Maintenance Agreement is available to assure this servicing automatically. If no Maintenance Agreement is acquired, any servicing must be initiated by the customer.

A Maintenance Agreement protects your investment and offers the following benefits:

1. Preventive Maintenance -

   Your equipment is inspected for worn parts, lubricated, cleaned, and updated with any engineering changes. The service minimizes "downtime" by anticipating repairs before they are necessary.

2. Fixed Annual Cost -

   You issue only one purchase order for service for an entire year and receive one annual billing. More frequent billing can be arranged, if desired.

Further information regarding Maintenance Agreements can be obtained from your local Sales-Service Office.

> NOTE:
>
> Wang Laboratories, Inc. does not honor guarantees or Maintenance Agreements for any equipment modified by a user. Damage to equipment incurred as a result of user modification is the financial responsibility of the user.

INDEX

To help us to provide you with the best manuals possible, please make your comments and suggestions concerning this publication on the form below. Then detach, fold, tape closed and mail to us. All comments and suggestions become the property of Wang Laboratories, Inc. For a reply, be sure to include your name and address. Your cooperation is appreciated.

---

700-3156B

TITLE OF MANUAL **8-BIT-PARALLEL I/O INTERFACE CONTROLLER**

COMMENTS:

Fold

Fold

(Please tape, Postal regulations prohibit the use of staples.)

WANG

Fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 16 LOWELL, MA

POSTAGE WILL BE PAID BY ADDRESSEE

**WANG LABORATORIES, INC.**
**ONE INDUSTRIAL AVENUE**
**LOWELL, MASSACHUSETTS 01851**

**Attention: Technical Writing Department**

Fold

Cut along dotted line.

Printed in U.S.A.
13-1019

## United States

**Alabama**
Birmingham
Mobile

**Alaska**
Anchorage

**Arizona**
Phoenix
Tucson

**California**
Culver City
Fountain Valley
Fresno
Inglewood
Sacramento
San Diego
San Francisco
Santa Clara
Ventura

**Colorado**
Englewood

**Connecticut**
New Haven
Stamford
Wethersfield

**District of Columbia**
Washington

**Florida**
Miami
Hialeah
Jacksonville
Orlando
Tampa

**Georgia**
Atlanta
Savannah

**Hawaii**
Honolulu

**Idaho**
Idaho Falls

**Illinois**
Chicago
Morton
Park Ridge
Rock Island
Rosemont

**Indiana**
Indianapolis
South Bend

**Kansas**
Overland Park
Wichita

**Kentucky**
Louisville

**Louisiana**
Baton Rouge
Metairie

**Maryland**
Rockville
Towson

**Massachusetts**
Billerica
Boston
Burlington
Chelmsford
Lawrence
Littleton
Lowell
Tewksbury
Worcester

**Michigan**
Kentwood
Okemos
Southfield

**Minnesota**
Eden Prairie

**Missouri**
Creve Coeur

**Nebraska**
Omaha

**Nevada**
Las Vegas
Reno

**New Hampshire**
Manchester

**New Jersey**
Toms River
Mountainside
Clifton

**New Mexico**
Albuquerque

**New York**
Albany
Buffalo
Fairport
Lake Success
New York City
Syracuse

**North Carolina**
Charlotte
Greensboro
Raleigh

**Ohio**
Cincinnati
Cleveland
Middleburg Heights
Toledo
Worthington

**Oklahoma**
Oklahoma City
Tulsa

**Oregon**
Eugene
Portland

**Pennsylvania**
Allentown
Camp Hill
Erie
Philadelphia
Pittsburgh
Wayne

**Rhode Island**
Cranston

**South Carolina**
Charleston
Columbia

**Tennessee**
Chattanooga
Knoxville
Memphis
Nashville

**Texas**
Austin
Dallas
Houston
San Antonio

**Utah**
Salt Lake City

**Vermont**
Montpelier

**Virginia**
Newport News
Norfolk
Richmond

**Washington**
Richland
Seattle
Spokane
Tacoma

**Wisconsin**
Brookfield
Madison
Wauwatosa

## International Offices

**Australia**
Wang Computer Pty., Ltd.
Adelaide, S.A.
Brisbane, Qld.
Canberra, A.C.T.
Darwin N.T.
Perth, W.A.
South Melbourne, Vic 3
Sydney, NSW

**Austria**
Wang Gesellschaft, m.b.H.
Vienna

**Belgium**
Wang Europe, S.A.
Brussels
Erpe-Mere

**Canada**
Wang Laboratories (Canada) Ltd.
Burnaby, B.C.
Calgary, Alberta
Don Mills, Ontario
Edmonton, Alberta
Hamilton, Ontario
Montreal, Quebec
Ottawa, Ontario
Winnipeg, Manitoba

**China**
Wang Industrial Co., Ltd.
Taipei
Wang Laboratories Ltd.
Taipei

**France**
Wang France S.A.R.L.
Paris
Bordeaux
Lyon
Marseilles
Nantes
Strasbourg
Toulouse

**Great Britain**
Wang (U.K.) Ltd.
Richmond
Birmingham
London
Manchester
Northwood Hills

**Hong Kong**
Wang Pacific Ltd.
Hong Kong

**Japan**
Wang Computer Ltd.
Tokyo

**Netherlands**
Wang Nederland B.V.
IJsselstein
Gronigen

**New Zealand**
Wang Computer Ltd.
Auckland
Wellington

**Panama**
Wang de Panama (CPEC) S.A.
Panama City

**Singapore**
Wang Computer (Pte) Ltd.
Singapore

**Sweden**
Wang Skandinaviska AB
Stockholm
Gothenburg
Malmo

**Switzerland**
Wang A.G.
Zurich
Basel
Geneva

**Wang Trading A.G.**
Zug

**United States**
Wang International Trade, Inc.
Lowell, Mass.

**West Germany**
Wang Laboratories, GmbH
Frankfurt
Berlin
Cologne
Dusseldprf
Essen
Freiburg
Hamburg
Hannover
Kassel
Munich
Nurnberg
Saarbrucken
Stuttgart

## International Representatives

Abu-Dhabi
Argentina
Bahrain
Bolivia
Brazil
Canary Islands
Chile
Colombia
Costa Rica
Cyprus
Denmark
Dominican Republic
Ecuador
Egypt
El Salvador
Finland
Ghana
Greece
Guatemala
Haiti
Honduras
Iceland
India
Indonesia
Ireland
Israel
Italy
Jamaica
Japan
Jordan
Kenya
Korea
Kuwait
Lebanon
Liberia
Malaysia
Malta
Mexico
Morocco
Nicaragua
Nigeria
Norway
Paraguay
Peru
Phillippines
Portugal
Saudi Arabia
Scotland
Spain
Sri Lanka
Sudan
Syria
Thailand
Turkey
United Arab Emirates
Venezuela

WANG LABORATORIES, INC.
ONE INDUSTRIAL AVENUE, LOWELL, MASSACHUSETTS 01851, TEL. (617) 459-5000, TWX 710 343-6769, TELEX 94-7421

Printed in U.S.A.
700-3156B