TELECOMMUNICATIONS INPUT OR MULTICHARACTER INPUT WITH MULTI-SPECIAL-CHARACTER FEATURE

(A sequence inside [ ] is optional, execution depends upon action atom for the incoming character)

Code	Signal Sequence
$F0h_3h_4$	(CPB, IBS, [WR, ECHO1/OBS], [WR, ECHO2/OBS], [SAVE DATA]), REPEAT
$F1h_3h_4$	(CPB, IBS, [ECHO1/OBS], [ECHO2/OBS], [SAVE DATA]), REPEAT
$F4h_3h_4$	(CPB, IBS, [WR, ECHO1/CBS], [WR, ECHO2/CBS], [SAVE DATA]), REPEAT
$F5h_3h_4$	(CPB, IBS, [ECHO1/CBS], [ECHO2/CBS], [SAVE DATA]), REPEAT
$F8h_3h_4$	(CPB, IBS, MASK, [WR, ECHO1/OBS], [WR, ECHO2/OBS], [SAVE DATA]), REPEAT
$F9h_3h_4$	(CPB, IBS, MASK, [ECHO1/OBS), [ECHO2/OBS], [SAVE DATA]), REPEAT
FCh_3h_4	(CPB, IBS, MASK, [WR, ECHO1/CBS), [WR, ECHO2/CBS], [SAVE DATA]), REPEAT
FDh_3h_4	(CPB, IBS, MASK, [ECHO1/CBS), [ECHO2/CBS], [SAVE DATA]), REPEAT

$\underbrace{h_3h_4}$ ← Action atom for any incoming character not matching a special character.

LEGEND (for $Fh_2h_3h_4$ microcommands only)

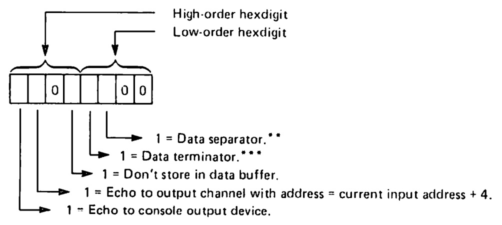
Mnemonic	Operation
CPB	CPU sets Ready/Busy signal level to Ready.
ECHO1/CBS	CPU sends echo of received character with CBS strobe to CO device.
ECHO1/OBS	CPU sends echo of received character with OBS strobe to CO device.
ECHO2/CBS	CPU sends echo of received character with CBS strobe to output channel of input device.*
ECHO2/OBS	CPU sends echo of received character with OBS strobe to output channel of input device.*
IBS	CPU awaits input strobe from enabled device.*
MASK	Set high-order eighth bit of received character to zero.
REPEAT	Repeat sequence in parentheses until valid termination condition detected.
SAVE DATA	Save received character in data buffer.
WR	CPU awaits Ready signal from enabled device.
	*An $Fh_2h_3h_4$ microcommand ignores any preceding address strobe of the form $7h_2h_3h_4$ and uses the address specified by the $GIO statement.

REGISTER USAGE

(for $GIO statements having an $Fh_2h_3h_4$ microcommand)

Register (Byte)	Bit Position	Use
1	all	Automatic storage of h_3h_4, specified in the microcommand (with h_4 set to 0). The stored value is the action atom for any input character not matching a character in the special character list.
2, 3, 4, 5	all	Not used.
6	all	Automatic storage of character received with ENDI-level = 1.
7	all	Not used.
8	01	1 = Buffer overflow.
	02	1 = Element overflow.
	04	Not used.
	08	Not used.
	10	1 = Timeout.
	20	1 = ENDI-level termination.
	40	1 = Terminator-character termination.
	80	1 = Separator received for last element.
9, 10	all	Automatic storage of the count of elements used for incoming data.
11, . . .		Storage of special character list (atom, character, atom, character, etc.). The list must end with HEX (2020).

Action Atom Definition


High-order hexdigit
Low-order hexdigit
0
0 0
1 = Data separator.**
1 = Data terminator.***
1 = Don't store in data buffer.
1 = Echo to output channel with address = current input address + 4.
1 = Echo to console output device.

**A separator denotes the end of an input "line"; the next received character is stored as the first character in the next element of the $GIO buffer. (If a separator is received for the last element, the microcommand is terminated.)

***A terminator denotes the end of a data stream; the microcommand is terminated.

THE $GIO STATEMENT

General Form:

$\text{\$GIO [comment]} \begin{bmatrix}\#f \\ /xyy\end{bmatrix} \underbrace{\text{(microcommand sequence,}}_{\text{argument 1}}\ \underbrace{\text{error/status registers)}}_{\text{argument 2}}\ \underbrace{[\text{data buffer}]}_{\text{argument 3}}$

Example: 100 $GIO WRITE /03B (6C01 4400 A206 8601, R$) B$() <5,90>

- The microcommand sequence must be one or more groups of four hexdigits ($h_1h_2h_3h_4$) representing valid codes from the microcommand categories recognized by the System 2200. The microcommand sequence can be specified directly, as shown in the example, or indirectly by an alphanumeric variable into which the appropriate hexdigit values have been previously stored.
- The error/status/general-purpose registers must be represented by an alphanumeric variable at least 10 bytes long (12 or more bytes are needed if an $Fh_2h_3h_4$ microcommand is used). The byte-positions in the alphanumeric variable are called "registers" to emphasize the multi-purpose usage of the variable.
- The data buffer is needed only if the microcommand sequence includes a multicharacter data transfer microcommand of the form $Ah_2h_3h_4$, $Bh_2h_3h_4$, $Ch_2h_3h_4$ or $Fh_2h_3h_4$. The $GIO buffer can be represented by an alpha variable, a string (STR) function, an alpha array designator, or a modified alpha array designator (i.e., an alpha array designator having a field expression specifying the portion of the array to be used for data output or input). The field expression format is as follows:

<s, n>	for any $GIO statement not having an $Fh_2h_3h_4$ microcommand
<s, m, e>	for a $GIO statement having an $Fh_2h_3h_4$ microcommand

where:

s = starting byte
n = number of consecutive bytes
m = number of bytes per element
e = number of elements.

MICROCOMMAND CATEGORIES

Code		Operation	
$0h_2h_3h_4$, $1h_2h_3h_4$		Control	no data buffer required
$4h_2h_3h_4$	(h_2 = 0 through 7)	Single character output	
$5h_2h_3h_4$	(h_2 = 0 through 7)	Single character output with acknowledge	
$6h_2h_3h_4$	(h_2 = 0 through F)	Single character output with echo	
$7h_2h_3h_4$	(h_2 = 1 or 3)	Single address strobe	
$8h_2h_3h_4$	(h_2 = 0, 2, 8, A)	Single character input with verify	
$86h_3h_4$		Single character input	
$9h_2h_3h_4$	(h_2 = 2, 3, 6, 7)	Single character input with echo	
$Ah_2h_3h_4$	(h_2 = 0, 1, 2, 4, 5, 6)	Multicharacter output	data buffer required
$Bh_2h_3h_4$	(h_2 = 0, 1, 4, 5,)	Multicharacter output with acknowledge	
$Bh_2h_3h_4$	(h_2 = 2, 3, 6, 7)	Multicharacter output with echo	
$Bh_2h_3h_4$	(h_2 = 8, 9, C, D)	Multicharacter output (each character requested)	
BAh_3h_4		Multicharacter verify	
$Ch_2h_3h_4$	(h_2 = 2, 6)	Multicharacter input	
$Ch_2h_3h_4$	(h_2 = 0, 1, 4, 5)	Multicharacter input with echo	
$Ch_2h_3h_4$	(h_2 = 8 through F)	Multicharacter input (each character requested)	
$Fh_2h_3h_4$	(h_2 = 0, 1, 4, 5, 8, 9, C, D)	Telecommunications input	

Codes 0... through 9... which do not require a data buffer can be used any number of times in a microcommand sequence.
Only one code A... through F... from a category which requires a data buffer can be used in a given microcommand sequence.

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.
700-3782A
10-80-5M

SYSTEM 2200

$GIO

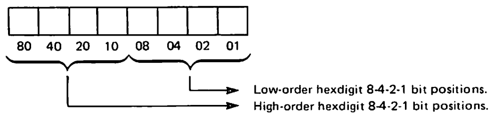
MICROCOMMANDS

WANG

REGISTER USAGE
(for $GIO statements not having an $Fh_2h_3h_4$ microcommand)

Register (Byte)	Bit Position*	Use
1	all	General-purpose or storage of a special termination character.
2, 3, 4	all	General-purpose.
5	all	General-purpose or automatic storage of an LRC character.
6	all	General-purpose or automatic storage of an ENDI-level=1 character.
7	all	General-purpose.
8	01	1 = Buffer overflow.
	02	1 = LRC error.
	04	1 = Echo/Verify error.
	08	1 = Compare error.
	10	1 = Timeout.
	20	1 = ENDI-level termination.
	40	1 = Special character termination.
	80	1 = Count termination.
9, 10	all	Automatic storage of the count of transferred characters for a multicharacter output or input microcommand.

*Bit position labels for status code (register 8) are as follows:


80 40 20 10 08 04 02 01
Low-order hexdigit 8-4-2-1 bit positions.
High-order hexdigit 8-4-2-1 bit positions.

LEGEND (for all microcommands except $FH_2h_3h_4$)

Mnemonic	Operation
ABS	CPU sends an "address bus strobe" with an immediate or indirect address to deselect the current address and select the specified address.
CBS	CPU sends a CBS strobe to the enabled device.
CHECK T, T1, T2	CPU checks for the termination condition specified by h_3.
CPB	CPU sets its Ready/Busy signal to Ready.
DATAOUT	CPU sends out next character from $GIO buffer.
ECHO	CPU sends echo of received character to enabled device.
IBS	CPU awaits input strobe from enabled device.
IMM	Immediate character is HEX(h_3h_4), specified by the microcommand.
IND	Indirect character is in the register specified by h_3.
LEND	CPU executes the LRC end sequence specified by h_4.
OBS	CPU sends an OBS strobe to the enabled device.
REPEAT	CPU repeats the sequence in parentheses for each character in a multicharacter input or output operation.
SAVE	CPU saves received character in the register specified by h_4.
SAVE DATA	CPU saves received character in the $GIO buffer.
SAVE LRC	CPU saves calculated LRC character in register 5.
SEND LRC	CPU sends calculated LRC character to enabled device.
TERM	CPU terminates $GIO statement if compared characters are not equal.
VERIFY	CPU compares received character; if unequal, the echo/verify error bit (bit 04 in register 8) is set to 1.
WR	CPU awaits Ready signal from enabled device.
W5	CPU waits 5 microseconds.

CONTROL MICROCOMMANDS

Code	Operation
$0h_2h_3h_4$	Store character HEX (h_3h_4) in register h_2.
$11h_3h_4$	Copy contents of register h_3 to register h_4.
1200	Disable previously set delay/timeout condition.
$12h_3$1	Introduce a delay* before each subsequent output strobe (except ABS); the interval in units of 50 microseconds is specified in binary in registers h_3 and h_3 + 1, where $1 \leq h_3 \leq 6$. Maximum delay interval = HEX (FFFF) ~ 3.3 seconds.
$12h_3$2	Initiate a timeout* before sensing each subsequent ready signal or input strobe; the interval in units of 1 millisecond is specified in binary in registers h_3 and h_3 + 1, where $1 \leq h_3 \leq 6$. Maximum timeout interval = HEX (FFFF) ~ 65.5 seconds. If a timeout interval is exceeded, set error bit (bit 10 in register 8) and terminate.
$14h_3h_4$	If contents of register $h_3 \neq$ contents of register h_4, set compare error bit (bit 08, register 8) to 1.
$15h_3h_4$	If contents of register $h_3 \neq$ contents of register h_4, set compare error bit (bit 08, register 8) and terminate.
$16h_3h_4$	If complemented status (register 8) code AND $h_3h_4 \neq$ HEX(00), terminate (i.e., terminate if any bit specified by the mask h_3h_4 is equal to zero).
$17h_3h_4$	If status (register 8) code AND $h_3h_4 \neq$ HEX(00), terminate (i.e., terminate if any bit specified by the mask h_3h_4 is equal to one).

*Delay and timeout conditions are mutually exclusive. Also, neither a delay nor a timeout can be in effect during execution of a multicharacter output with echo microcommand of the form $B2h_3h_4$, $B3h_3h_4$, $B6h_3h_4$ or $B7h_3h_4$; if so, a false indication of an echo error may occur.

SINGLE CHARACTER OUTPUT AND ADDRESS STROBE

Code	SIGNAL SEQUENCE	Character To be sent	Character To be saved
	Single Character Output		
$40h_3h_4$	WR, OBS/IMM	HEX (h_3h_4)	
$41h_3h_4$	OBS/IMM	HEX (h_3h_4)	
$42h_3$0	WR, OBS/IND	from register h_3	
$43h_3$0	OBS/IND	from register h_3	
$44h_3h_4$	WR, CBS/IMM	HEX (h_3h_4)	
$45h_3h_4$	CBS/IMM	HEX (h_3h_4)	
$46h_3$0	WR, CBS/IND	from register h_3	
$47h_3$0	CBS/IND	from register h_3	
	Single Character Output with Acknowledge		
$50h_3h_4$	WR, OBS/IMM, W5, CPB, IBS	HEX (h_3h_4)	
$51h_3h_4$	OBS/IMM, W5, CPB, IBS	HEX (h_3h_4)	
$52h_3h_4$	WR, OBS/IND, W5, CPB, IBS, SAVE	from register h_3	in register h_4
$53h_3h_4$	OBS/IND, W5, CPB, IBS, SAVE	from register h_3	in register h_4
$54h_3h_4$	WR, CBS/IMM, W5, CPB, IBS	HEX (h_3h_4)	
$55h_3h_4$	CBS/IMM, W5, CPB, IBS	HEX (h_3h_4)	
$56h_3h_4$	WR, CBS/IND, W5, CPB, IBS, SAVE	from register h_3	in register h_4
$57h_3h_4$	CBS/IND, W5, CPB, IBS, SAVE	from register h_3	in register h_4
	Single Character Output with Echo		
$60h_3h_4$	WR, OBS/IMM, W5, CPB, IBS, VERIFY	HEX (h_3h_4)	
$61h_3h_4$	OBS/IMM, W5, CPB, IBS, VERIFY	HEX (h_3h_4)	
$62h_3h_4$	WR, OBS/IND, W5, CPB, IBS, SAVE, VERIFY	from register h_3	in register h_4
$63h_3h_4$	OBS/IND, W5, CPB, IBS, SAVE, VERIFY	from register h_3	in register h_4
$64h_3h_4$	WR, CBS/IMM, W5, CPB, IBS, VERIFY	HEX (h_3h_4)	
$65h_3h_4$	CBS/IMM, W5, CPB, IBS, VERIFY	HEX (h_3h_4)	
$66h_3h_4$	WR, CBS/IND, W5, CPB, IBS, SAVE, VERIFY	from register h_3	in register h_4
$67h_3h_4$	CBS/IND, W5, CPB, IBS, SAVE, VERIFY	from register h_3	in register h_4
$68h_3h_4$	WR, OBS/IMM, W5, CPB, IBS, VERIFY, TERM	HEX (h_3h_4)	
$69h_3h_4$	OBS/IMM, W5, CPB, IBS, VERIFY, TERM	HEX (h_3h_4)	
$6Ah_3h_4$	WR, OBS/IND, W5, CPB, IBS, SAVE, VERIFY, TERM	from register h_3	in register h_4
$6Bh_3h_4$	OBS/IND, W5, CPB, IBS, SAVE, VERIFY, TERM	from register h_3	in register h_4
$6Ch_3h_4$	WR, CBS/IMM, W5, CPB, IBS, VERIFY, TERM	HEX (h_3h_4)	
$6Dh_3h_4$	CBS/IMM, W5, CPB, IBS, VERIFY, TERM	HEX (h_3h_4)	
$6Eh_3h_4$	WR, CBS/IND, W5, CPB, IBS, SAVE, VERIFY, TERM	from register h_3	in register h_4
$6Fh_3h_4$	CBS/IND, W5, CPB, IBS, SAVE, VERIFY, TERM	from register h_3	in register h_4
	Address Strobe		
$71h_3h_4$	ABS/IMM	HEX (h_3h_4)	
$73h_3$0	ABS/IND	from register h_3	

Note: Codes of the form 7... can be used repeatedly in a sequence to deselect the current device address and select another.

SINGLE CHARACTER INPUT

Code	Signal Sequence	Verify Character	Character To Be Saved
	Single Character Input		
8600	CPB, IBS		in register h_4
$860h_4$	CPB, IBS, SAVE		
	Single Character Input with Verify		
$80h_3h_4$	CPB, IBS, VERIFY/IMM	HEX(h_3h_4)	
$82h_3h_4$	CPB, IBS, SAVE, VERIFY/IND	in register h_3	in register h_4
$88h_3h_4$	CPB, IBS, VERIFY/IMM, TERM	HEX(h_3h_4)	
$8Ah_3h_4$	CPB, IBS, SAVE, VERIFY/IND, TERM	in register h_3	in register h_4
	Single Character Input with Echo		
9200	CPB, IBS, WR, ECHO/OBS		
$920h_4$	CPB, IBS, SAVE, WR, ECHO/OBS		in register h_4
9300	CPB, IBS, ECHO/OBS		
$930h_4$	CPB, IBS, SAVE, ECHO/OBS		in register h_4
9600	CPB, IBS, WR, ECHO/CBS		
$960h_4$	CPB, IBS, SAVE, WR, ECHO/CBS		in register h_4
9700	CPB, IBS, ECHO/CBS		
$970h_4$	CPB, IBS, SAVE, ECHO/CBS		in register h_4

MULTICHARACTER OUTPUT
(A sequence in parentheses is repeated for each character in the data buffer)

Code	Signal Sequence	Check T	Lend
	Multicharacter Output		
$A00h_4$	(WR, DATAOUT/OBS), REPEAT, LEND		h_4
$A10h_4$	(DATAOUT/OBS), REPEAT, LEND		h_4
$A20h_4$	high speed version of $A00h_4$; no timeout or delay		h_4
$A40h_4$	(WR, DATAOUT/CBS), REPEAT, LEND		h_4
$A50h_4$	(DATAOUT/CBS), REPEAT, LEND		h_4
$A60h_4$	SCAN DATA BUFFER, CALCULATE LRC, LEND		h_4
	Multicharacter Output with Acknowledge		
$B0h_3h_4$	(WR, DATAOUT/OBS, W5, CPB, IBS, CHECK T), REPEAT, LEND	h_3	h_4
$B1h_3h_4$	(DATAOUT/OBS, W5, CPB, IBS, CHECK T), REPEAT, LEND	h_3	h_4
$B4h_3h_4$	(WR, DATAOUT/CBS, W5, CPB, IBS, CHECK T), REPEAT, LEND	h_3	h_4
$B5h_3h_4$	(DATAOUT/CBS, W5, CPB, IBS, CHECK T), REPEAT, LEND	h_3	h_4
	Multicharacter Output with Echo		
$B2h_3h_4$	(WR, DATAOUT/OBS, W5, CPB, IBS, VERIFY, CHECK T), REPEAT, LEND	h_3	h_4
$B3h_3h_4$	(DATAOUT/OBS, W5, CPB, IBS, VERIFY, CHECK T), REPEAT, LEND	h_3	h_4
$B6h_3h_4$	(WR, DATAOUT/CBS, W5, CPB, IBS, VERIFY, CHECK T), REPEAT, LEND	h_3	h_4
$B7h_3h_4$	(DATAOUT/CBS, W5, CPB, IBS, VERIFY, CHECK T), REPEAT, LEND	h_3	h_4
	Multicharacter Output with each Character Requested		
$B8h_3h_4$	(CPB, IBS, CHECK T, WR, DATAOUT/OBS), REPEAT, LEND	h_3	h_4
$B9h_3h_4$	(CPB, IBS, CHECK T, DATAOUT/OBS), REPEAT, LEND	h_3	h_4
BCh_3h_4	(CPB, IBS, CHECK T, WR, DATAOUT/CBS), REPEAT, LEND	h_3	h_4
BDh_3h_4	(CPB, IBS, CHECK T, DATAOUT/CBS), REPEAT, LEND	h_3	h_4
	Multicharacter Verify		
$BAh_3$0	(CPB, IBS, VERIFY, CHECK T), REPEAT	h_3	

Valid "Check T" Codes

Termination Condition \ Microcommand	B0.. B1.. B4.. B5..	B2.. B3.. B6.. B7..	B8.. B9.. BC.. BD..	BA.0
None (go to next microcommand)	0	0	0	8
Terminate if verify unequal		1		9
Terminate if ENDI level logic '1'	2	2	2	A
Terminate on either condition		3		B

Valid "Lend" Codes

LRC End Sequence \ Microcommand	A..	B0.. through B7..	B8.. B9.. BC.. BD..
None (go to next microcommand)	0	0	0
WR, SEND LRC/OBS, SAVE LRC	2	2	
SEND LRC/OBS, SAVE LRC	3	3	
SAVE LRC	4	4	4
WR, SEND LRC/CBS, SAVE LRC	6	6	
SEND LRC/CBS, SAVE LRC	7	7	

MULTICHARACTER INPUT
(A sequence in parentheses is repeated until a valid termination condition occurs)

Code	Signal Sequence	Check T	Lend
	Multicharacter Input		
$C22h_4$	(CPB, IBS, no timeout or delay, CHECK ENDI, SAVE DATA), REPEAT, LEND	2	h_4
$C6h_3h_4$	(CPB, IBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
	Multicharacter Input with Echo		
$C0h_3h_4$	(CPB, IBS, WR, ECHO/OBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
$C1h_3h_4$	(CPB, IBS, ECHO/OBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
$C4h_3h_4$	(CPB, IBS, WR, ECHO/CBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
$C5h_3h_4$	(CPB, IBS, ECHO/CBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
	Multicharacter Input with Each Character Requested		
$C8h_3h_4$	(WR, OBS, W5, CPB, IBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
$C9h_3h_4$	(OBS, W5, CPB, IBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
CAh_3h_4	(CPB, WR, OBS, IBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
CBh_3h_4	(CPB, OBS, IBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
CCh_3h_4	(WR, CBS, W5, CPB, IBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
CDh_3h_4	(CBS, W5, CPB, IBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
CEh_3h_4	(CPB, WR, CBS, IBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4
CFh_3h_4	(CPB, CBS, IBS, CHECK T1, SAVE DATA, CHECK T2), REPEAT, LEND	h_3	h_4

Valid "check T" Codes

	Termination Conditions (order of checking from left to right)		
h_3	ENDI-level = 1 (when character received)	Special Character Received (matches char. in reg. 1)	Character Count Equals Buffer Length
0		check (save in buffer, include in LRC)	
1		check (do not save char.)	
2	check (save char. in reg. 6)		
3	check (save char. in reg. 6)	check (do not save char.)	
4			check
5		check (do not save char.)	check
6	check (save char. in reg. 6)		check
7	check (save char. in reg. 6)	check (do not save char.)	check

Valid "Lend" Codes

h_4	LRC End Sequence
0	None (go to next microcommand).
1	Calculate LRC and save.
2	Calculate LRC, save, compare with ENDI character, and set LRC error bit (use only if h_3 = 2).