INTELLEC® SERIES II CRT AND KEYBOARD INTERFACE MANUAL

Order Number: 122029-001

INTELLEC® SERIES II CRT AND KEYBOARD INTERFACE MANUAL

Order Number: 122029-001

Intel Corporation, 3065 Bowers Avenue, Santa Clara, CA 95051

Additional copies of this manual or other Intel literature may be obtained from:

Literature Department
Intel Corporation
3065 Bowers Avenue
Santa Clara, CA 95051

The information in this document is subject to change without notice.

Intel Corporation makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Intel Corporation assumes no responsibility for any errors that may appear in this document. Intel Corporation makes no commitment to update nor to keep current the information contained in this document.

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied.

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication or disclosure is subject to restrictions stated in Intel's software license, or as defined in ASPR 7-104.9(a)(9).

REV.	REVISION HISTORY	DATE
-001	Original issue.	8/82

PREFACE

This manual describes the CRT and keyboard interface for the enhanced Series II development system.

It is divided into four chapters and two appendixes.

Chapter 1, "Introduction," briefly describes the capabilities of the enhanced Series II CRT and keyboard interface.

Chapter 2, "Preprogrammed Keys," lists the preprogrammed keys and their associated keywords.

Chapter 3, "Console Output Functions," describes the console output function codes with examples.

Chapter 4, "Block Movement of Data to the CRT," describes the IOC commands (with examples) that allow block movement of data to the CRT.

Appendix A, "Keyboard Codes Generated," lists the hexadecimal values of the Intel keyboard characters.

Appendix B, "ASCII Character Set," lists the ASCII codes with their hexadecimal values.

Related Publications

Model 511 IOC Firmware Enhancement Kit Installation Instructions, order number 122014

Intellec Series II Microcomputer Development System Hardware Interface Manual, order number 9800555

Intellec Series II Microcomputer Development System Hardware Reference Manual, order number 9800556

Component Data Catalog, order number 210298

Notational Conventions

UPPERCASE	Indicates that characters shown in uppercase must be entered in the order shown.
italic	Indicates variable information.
[ ]	Indicate optional arguments or parameters.

CONTENTS

TABLES

ILLUSTRATIONS

CHAPTER 1
INTRODUCTION

The Series II CRT and keyboard are controlled by an Input/Output Controller Board, which is referred to as the IOC throughout this manual. All models of the Series II development systems provide cursor movement, clear screen and clear line functions, and display of the ASCII character set.

But the enhanced Series II provides much more:

- Automatic keystroke repeat function
- Preprogrammed keystroke sequences
- New console output functions, including direct cursor addressing
- Block movement of data to the CRT

These functions are provided by new IOC and keyboard firmware. (The cursor now appears as a solid, non-blinking block.)

If you have purchased the new firmware as an update package (iMDX 511), you will receive:

- Two single-density diskettes
- Two double-density diskettes
- Four 2716 PROMs (IOC CRT firmware)
- One 8741A PROM (IOC keyboard firmware)
- Two key caps (labeled FUNC)

Firmware installation is described in the *Model 511 IOC Firmware Enhancement Kit Installation Instructions*, order number 122014.

If you have purchased a new development system, the PROMS and key cap have already been integrated into your system.

CHAPTER 2
PREPROGRAMMED KEYS

The enhanced Series II IOC firmware provides numerous preprogrammed keys. When these keys and the special function key, FUNC, are pressed simultaneously, preprogrammed character sequences are transmitted as keyboard input. (The preprogrammed keys are listed below.)

For example, if you type FUNC A, the keyboard will respond as if you typed ALTER followed by a blank. Upper and lowercase are significant when typing preprogrammed keys. That is, if you type FUNC a, the keyboard will transmit alter in lowercase letters.

The HELP command, executed by typing FUNC H, displays all of the preprogrammed keys and their associated keywords on screen. When you invoke the HELP menu, the screen is cleared of current text. When you type any character, the HELP menu disappears and previous text is restored to the screen.

Keys

Quote marks are used to show significant spaces; they are not transmitted. (Remember, you must press FUNC simultaneously with the key you need.)

A	=	“ALTER ”
C	=	“COPY ”
D	=	“DIR ”
E	=	“CREDIT ”
I	=	“ATTRIB ”
J	=	“JOB ”
K	=	“DELETE ”
L	=	“:LP: ”
M	=	“LOGON ”
N	=	“ASSIGN ”
O	=	“LOGOFF ”
P	=	“:SP: ”
R	=	“RUN ”
S	=	“SUBMIT ”
T	=	“ TO ”
U	=	“ACCESS ”
X	=	“EXPORT ”

n = :F*n*: (where *n* = (0,1,...,9))

SHIFT + *n* = /JOB*n* <cr> (where *n* = (0,1,...,9))

H = HELP — displays all of the function keys and their preprogrammed keywords

Keywords

ACCESS	=	U
ALTER	=	A
ASSIGN	=	N
ATTRIB	=	I
COPY	=	C
CREDIT	=	E
DELETE	=	K
DIR	=	D
EXPORT	=	X
:F*n*:	=	*n* (where *n* = (0,1,...,9))
HELP	=	H
JOB	=	J
/JOB*n*	=	SHIFT + *n* (where *n* = (0,1,...,9))
LOGON	=	M
LOGOFF	=	O
:LP:	=	L
RUN	=	R
:SP:	=	P
SUBMIT	=	S
TO	=	T

Repeat Function

To send multiple characters, simply hold down the key that you wish to repeat and it will repeat automatically. (The key will repeat after it is held down for 0.5 seconds and will continue to repeat at a rate of 15 characters per second until it is released.)

CHAPTER 3
CONSOLE OUTPUT FUNCTIONS

New Console Output Codes

The IOC firmware modifications provide several new console output functions.

The console output functions are obtained by sending the following codes to console output.

ESC R

Clear text from current cursor position to the end of the line.

ESC S

Clear text from current cursor position to the end of the screen.

ESC T

Clear entire screen (cursor remains in current position).

ESC Y

Allows direct cursor addressing.

This command requires coordinates for the row number and column number. (Row number is the row address + 20H; column number is the column address + 20H.)

The command has the following form:

```
ESC Y rc
```

where

r and *c* — are variables for row number and column number. If a value given for *r* or *c* is too large (greater than 38H for row; greater than 6FH for column), the current coordinate is used.

Example 1

```
1BH 59H 20H 20H
```

This command moves the cursor to the upper left corner of the screen. (20H, 20H indicates the upper left corner of the screen.)

ESC W

ESC W is a generalized insert and delete line function.

The command has the following form:

```
ESC W id
```

where

i and *d* are line numbers taken from table 3-1.

The command inserts a blank line at line *i* on the screen and deletes the line at line *d*. All intervening lines scroll in the direction of the deleted line. The cursor moves to the start of the current line.

Line numbers can be presented in three ways: relative to the top of the screen, relative to the bottom of the screen, and relative to the current cursor position. Given that the CRT has exactly 25 lines, the relative-to-bottom settings are redundant. They are provided so that your cursor control sequence will be compatible with possible future terminals with other than 25 lines.

Example 1

```
1BH 57H 60H 3FH
```

This command inserts a blank line at the current cursor position.

Example 2

```
1BH 57H 3FH 60H
```

This command deletes the line at the current cursor position.

Table 3-1 lists the possible values for *i* and *d*.

Table 3-1. Line Coordinates for ESC W

Coordinate	Line
00H	Top line
01H	One below top
02H	Second below top
.	
.	
17H	23rd from top
18H	24th from top
27H	24th from bottom
.	
.	
3EH	One from bottom
3FH	Bottom line
47H	24th line above current cursor position
.	
.	
5FH	Line above current cursor position
60H	Line at current cursor position
61H	Line below current cursor position
.	
.	
78H	24th line below current cursor,position

ESC X

Allows you to set the following flags: USE$8$BITS$FLAG and PAUSE$FLAG.

The command has the following form:

```
ESC X ns
```

where

n	is the flag number.
s	is the setting (1 = true; 0 = false).

USE$8$BITS$FLAG

Eight-bit codes are used to take advantage of the field attribute features provided by the 8275 CRT controller chip. (See the *Component Data Catalog* for information on the 8275 chip.)

The values in table 3-2, when sent to the CRT, are called "attribute bytes." Wherever they are placed on the CRT screen, a blank will appear in that position. Each succeeding position, until the next attribute byte appears, will have the attribute given.

Attributes can be combined by ORing together the appropriate attribute bytes; e.g., 92H causes blinking characters and reverse video. If blinking and reverse video are enabled simultaneously, only the characters will blink.

USE$8$BITS$FLAG is set false to accommodate programs that send the eighth bit as a parity bit.

The following two ESC sequences are used to set USE$8$BITS$FLAG:

1. `1BH 58H 00H 01H`

All 8 bits of bytes received for console output are used.

2. `1BH 58H 00H 00H`

The top bit is masked away (default setting).

Table 3-2. Attribute Bytes

Attribute Byte	Attribute
80H	None
82H	Blinking characters
90H	Reverse video
A0H	Underline

PAUSE$FLAG

The following two ESC sequences are used to set PAUSE$FLAG:

1. `1BH 58H 01H 01H`

Enter PAUSE mode.

- The IOC pauses (approximately five seconds) every time it is about to scroll the screen (when 20 lines have been output without any keyboard input).
- CNTL S stops output until something other than CNTL S is typed.
- If CNTL Q is typed, or if five seconds elapse, the screen will continue scrolling.

2. `1BH 58H 01H 00H`

No PAUSE mode (default setting).

Existing Console Output Codes

This chapter documents the console output functions available with the previous IOC firmware as well as the new firmware. Codes with the same function are grouped together; i.e., ESC A and up arrow are different codes, but they perform the same function.

ESC A
up arrow

Move cursor up. If the cursor is at the top of the screen, the screen wraps and the cursor appears at the bottom of the screen.

ESC B
down arrow
CNTL J

Move cursor down. If the cursor is at the bottom of the screen, the screen scrolls.

ESC C
right arrow

Move cursor right. If the cursor is at the end of the line, it moves to the beginning of the next line. If the cursor is at the end of the screen, the screen scrolls.

ESC D
left arrow

Move cursor left. If the cursor is at the start of the line, the cursor moves to the end of the previous line. If the cursor is at the start of the screen, the cursor moves to the end of the screen.

ESC E

Clear entire screen; cursor moves to the beginning of the screen.

ESC H
HOME

Move the cursor to the top left corner of the screen.

ESC J

Clear text from the beginning of the current line (the line on which the cursor is positioned) to the end of the screen.

ESC K

Clear entire line.

CNTL G

Ring bell.

RETURN

Move the cursor to the start of the following line.

RUBOUT

Do nothing.

All other control keys are ignored.

All other keys display the ASCII character.

Table 3-3 lists the standard CRT output controls, their hexadecimal values, and functions.

Table 3-3. Standard CRT Output Codes

Code	Hexadecimal Value	Function
ESC A up arrow	1B 41 1E	Move cursor up. Move cursor up.
ESC B down arrow CNTL J	1B 42 1C 0A	Move cursor down. Move cursor down. Move cursor down.
ESC C right arrow	1B 43 14	Move cursor right. Move cursor right.
ESC D left arrow	1B 44 1F	Move cursor left. Move cursor left.
ESC H HOME	1B 48 1D	Move cursor to beginning of screen. Move cursor to beginning of screen.
RETURN	0D	Move cursor to start of line.
ESC Y *	1B 59 *r c*	Move cursor to row *r*, column *c*.
ESC K	1B 4B	Clear line.
ESC R *	1B 52	Clear text from current cursor position to end of line.
ESC J	1B 4A	Clear text from current line to end of screen.
ESC S *	1B 53	Clear text from current cursor position to end of screen.
ESC T *	1B 54	Clear screen; cursor remains in current cursor position.
ESC E	1B 45	Clear screen; cursor moves to beginning of screen.
ESC W *	1B 57 *i d*	Insert a line at line *i*; delete a line at line *d*.
ESC X *	1B 58 *n s*	Set flags: USE$8$BITS$ FLAG and PAUSE$FLAG; *n* = flag number;*s* = setting.
CNTL G	07	Ring bell.
RUBOUT	7F	

* Indicates new console output codes.

CHAPTER 4
BLOCK MOVEMENT OF DATA TO THE CRT

To allow fast block movement of data to the CRT, several new IOC commands have been implemented as the Block Write function.

The Block Write function is invoked from application programs running on an enhanced Series II. It cannot be invoked through the keyboard.

The Block Write function is initiated by sending a 0FH, 2FH, 4FH, or 6FH command to the command port of the IOC.

The commands 0FH or 4FH require the line number and column number of the position on the screen where the data will be output to the data port following a 0FH or 4FH command. (0,0 is the top left corner of the screen.)

The commands 2FH or 6FH output data at the current cursor position. Table 4-1 summarizes the differences between the commands.

Data is transmitted as follows:

1. The Block Write command (0FH, 2FH, 4FH, or 6FH) is given to the IOC command port.

2. If the command is 0FH or 4FH, the row and column number are output to the IOC data port (00H 00H = upper left corner). If the values for row number and column number are too large (greater than 18H for row number; greater than 4FH for column number), the current coordinates are used.

3. A stream of console output data bytes is output to the IOC data port. If bytes are deposited to the end of the screen, the cursor wraps around to the top of the screen. Bytes are interpreted as follows:

   - Bytes less than 80H are deposited directly in screen memory, except for carriage return and line feed, which have their usual functions.
   - Bytes between 80H and 0FDH are interpreted as follows:

     If the command is 0FH or 2FH, byte *n* is treated as (*n*—80H) blanks.

     If the command is 4FH or 6FH, the byte is an attribute byte to be deposited directly in screen memory. (See Chapter 3, section "USE$8$BITS$FLAG" for a description of attribute bytes.)
   - Byte 0FEH is dropped, but the next byte is placed in screen memory no matter what it is.
   - Byte 0FFH terminates the data and the command unless literalized as described above.

Figure 4-1 is a sample assembly language program that illustrates how to send commands and data to the IOC.

This program uses the following interface procedures to access the IOC commands:

- PCIOC transfers command byte to the IOC.
- PDIOC transfers each following data byte to the IOC.
- GDIOC retrieves data bytes from the IOC.

If there is a possibility of an interrupt procedure (or another processor) accessing the IOC, the interrupts should be disabled with a DI instruction at the beginning of the procedure and enabled with an EI instruction at the end of the procedure. Or, the interrupts can be masked at the beginning of the procedure and unmasked at the end.

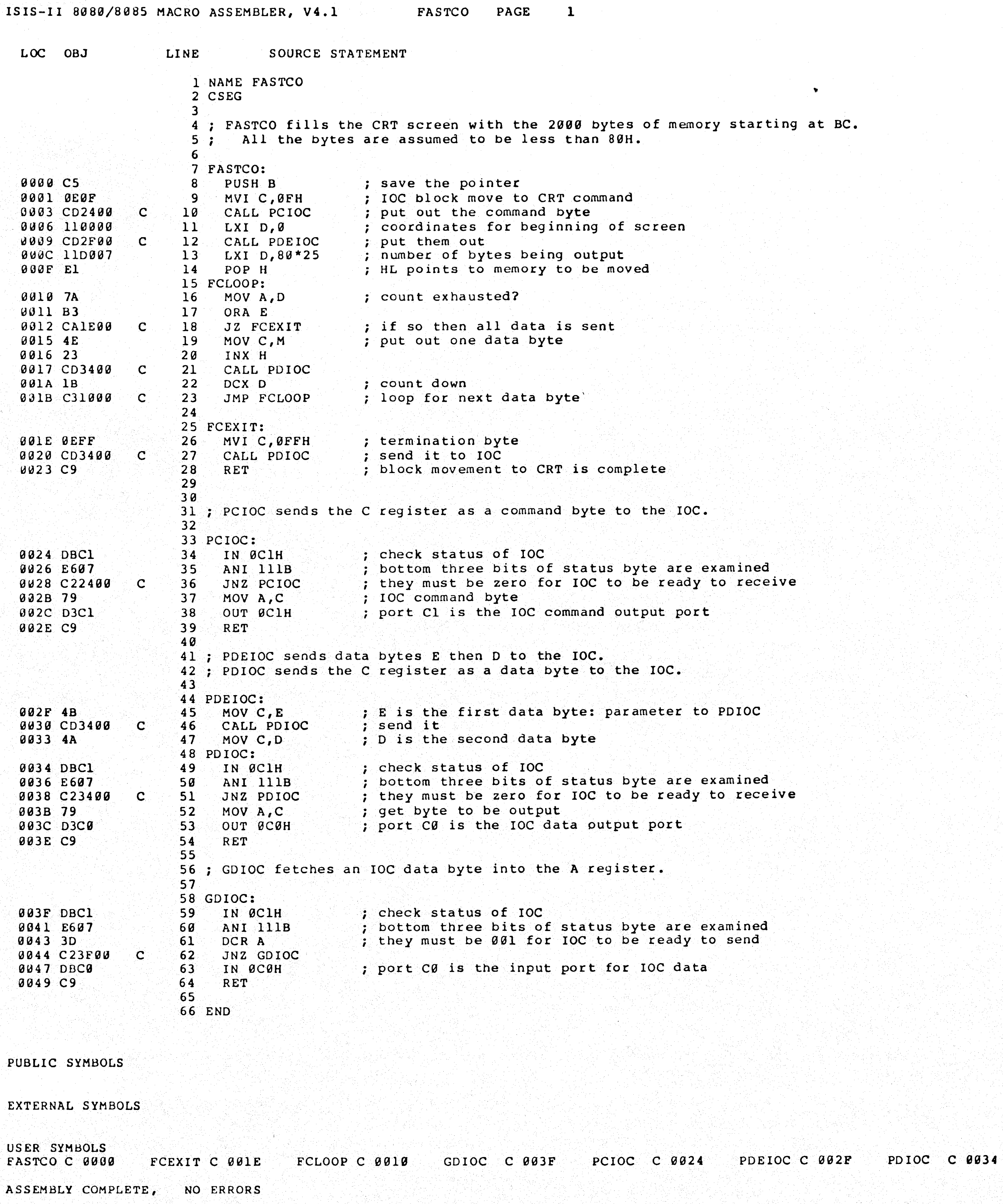
Interrupts in this category include pressing Interrupt 0 on the Series II console, since the Series II monitor performs console output.

Similarly, the Series III RUN program communicates with the IOC and therefore precludes 8086 application programs from directly accessing the IOC.

Table 4-1. Block Write Commands

Command	Requires Coordinates	Action for High Bytes
0F	Yes	(*n*−80H) blanks
2F	No	(*n*−80H) blanks
4F	Yes	Attribute character
6F	No	Attribute character

n = value of byte



```
ISIS-II 8080/8085 MACRO ASSEMBLER, V4.1        FASTCO    PAGE    1

  LOC  OBJ          LINE         SOURCE STATEMENT

                       1 NAME FASTCO
                       2 CSEG
                       3
                       4 ; FASTCO fills the CRT screen with the 2000 bytes of memory starting at BC.
                       5 ;   All the bytes are assumed to be less than 80H.
                       6
                       7 FASTCO:
 0000 C5               8   PUSH B            ; save the pointer
 0001 0E0F             9   MVI C,0FH         ; IOC block move to CRT command
 0003 CD2400    C     10   CALL PCIOC        ; put out the command byte
 0006 110000          11   LXI D,0           ; coordinates for beginning of screen
 0009 CD2F00    C     12   CALL PDEIOC       ; put them out
 000C 11D007          13   LXI D,80*25       ; number of bytes being output
 000F E1              14   POP H             ; HL points to memory to be moved
                      15 FCLOOP:
 0010 7A              16   MOV A,D           ; count exhausted?
 0011 B3              17   ORA E
 0012 CA1E00    C     18   JZ FCEXIT         ; if so then all data is sent
 0015 4E              19   MOV C,M           ; put out one data byte
 0016 23              20   INX H
 0017 CD3400    C     21   CALL PDIOC
 001A 1B              22   DCX D             ; count down
 001B C31000    C     23   JMP FCLOOP        ; loop for next data byte
                      24
                      25 FCEXIT:
 001E 0EFF            26   MVI C,0FFH        ; termination byte
 0020 CD3400    C     27   CALL PDIOC        ; send it to IOC
 0023 C9              28   RET               ; block movement to CRT is complete
                      29
                      30
                      31 ; PCIOC sends the C register as a command byte to the IOC.
                      32
                      33 PCIOC:
 0024 DBC1            34   IN 0C1H           ; check status of IOC
 0026 E607            35   ANI 111B          ; bottom three bits of status byte are examined
 0028 C22400    C     36   JNZ PCIOC         ; they must be zero for IOC to be ready to receive
 002B 79              37   MOV A,C           ; IOC command byte
 002C D3C1            38   OUT 0C1H          ; port C1 is the IOC command output port
 002E C9              39   RET
                      40
                      41 ; PDEIOC sends data bytes E then D to the IOC.
                      42 ; PDIOC sends the C register as a data byte to the IOC.
                      43
                      44 PDEIOC:
 002F 4B              45   MOV C,E           ; E is the first data byte: parameter to PDIOC
 0030 CD3400    C     46   CALL PDIOC        ; send it
 0033 4A              47   MOV C,D           ; D is the second data byte
                      48 PDIOC:
 0034 DBC1            49   IN 0C1H           ; check status of IOC
 0036 E607            50   ANI 111B          ; bottom three bits of status byte are examined
 0038 C23400    C     51   JNZ PDIOC         ; they must be zero for IOC to be ready to receive
 003B 79              52   MOV A,C           ; get byte to be output
 003C D3C0            53   OUT 0C0H          ; port C0 is the IOC data output port
 003E C9              54   RET
                      55
                      56 ; GDIOC fetches an IOC data byte into the A register.
                      57
                      58 GDIOC:
 003F DBC1            59   IN 0C1H           ; check status of IOC
 0041 E607            60   ANI 111B          ; bottom three bits of status byte are examined
 0043 3D              61   DCR A             ; they must be 001 for IOC to be ready to send
 0044 C23F00    C     62   JNZ GDIOC
 0047 DBC0            63   IN 0C0H           ; port C0 is the input port for IOC data
 0049 C9              64   RET
                      65
                      66 END

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS
FASTCO C 0000    FCEXIT C 001E    FCLOOP C 0010    GDIOC  C 003F    PCIOC  C 0024    PDEIOC C 002F    PDIOC  C 0034

ASSEMBLY COMPLETE,    NO ERRORS
```

Figure 4-1. Sample Program: FASTCO

APPENDIX A
KEYBOARD CODES GENERATED

This appendix lists all of the Intel keyboard characters and their hexadecimal values.

Key	Unshift TPWR Lock	Unshift TPWR Unlock	Shift	CNTL
A	61	41	41	01
B	62	42	42	02
C	63	43	43	03
D	64	44	44	04
E	65	45	45	05
F	66	46	46	06
G	67	47	46	07
H	68	48	48	08
I	69	49	49	09
J	6A	4A	4A	0A
K	6B	4B	4B	0B
L	6C	4C	4C	0C
M	6D	4D	4D	0D
N	6E	4E	4E	0E
O	6F	4F	4F	0F
P	70	50	50	10
Q	71	51	51	11
R	72	52	52	12
S	73	53	53	13
T	74	54	54	14
U	75	55	55	15
V	76	56	56	16
W	77	57	57	17
X	78	58	58	18
Y	79	59	59	19
Z	7A	5A	5A	1A
0 ~	30	30	7E	
1 !	31	31	21	
2 ”	32	32	22	
3 #	33	33	23	
4 $	34	34	24	
5 %	35	35	25	
6 &	36	36	26	
7 ’	37	37	27	
8 (	38	38	28	
9)	39	39	29	
- =	2D	2D	3D	
[{	5B	5B	7B	
] }	5D	5D	7D	
\ |	5C	5C	7C	
@ `	40	40	60	00
— ^	5F	5F	5E	
; +	3B	3B	2B	
: *	3A	3A	2A	
, <	2C	2C	3C	
. >	2E	2E	3E	
/ ?	2F	2F	3F	
RETURN	0D	0D	0D	
SPACE BAR	20	20	20	
ESC	1B	1B	1B	
RUBOUT	7F	7F	7F	
↑	1E	1E	1E	
↓	1C	1C	1C	
→	14	14	14	
←	1F	1F	1F	
HOME	1D	1D	1D	

FUNC Key

When the FUNC key is pressed simultaneously with any other key (or key combination), the code returned depends on whether the IOC is in local mode or on-line mode.

If the IOC is in local mode, 80H is added to the code that would have been returned if FUNC had not been pressed. The 80H bit is output to the console only if the USE$8$BITS$FLAG is set to true. (See Chapter 3, section "USE$8$BITS$FLAG.")

If the IOC is in on-line mode, the preprogrammed sequence is returned as described in Chapter 2.

To put the IOC in local mode, type:

```
CNTL SHIFT L
```

(Hold down the CNTL key and the SHIFT key, then type L.)

To put the IOC in on-line mode, type:

```
CNTL SHIFT O
```

APPENDIX B
ASCII CHARACTER SET

ASCII Character	HEX	ASCII Character	HEX
NUL	00	@	40
SOH	01	A	41
STX	02	B	42
ETX	03	C	43
EOT	04	D	44
ENQ	05	E	45
ACK	06	F	46
BEL	07	G	47
BS	08	H	48
HT	09	I	49
LF	0A	J	4A
VT	0B	K	4B
FF	0C	L	4C
CR	0D	M	4D
SO	0E	N	4E
SI	0F	O	4F
DLE	10	P	50
DC1	11	Q	51
DC2	12	R	52
DC3	13	S	53
DC4	14	T	54
NAK	15	U	55
SYN	16	V	56
ETB	17	W	57
CAN	18	X	58
EM	19	Y	59
SUB	1A	Z	5A
ESC	1B	[	5B
FS	1C	\	5C
GS	1D	]	5D
RS	1E	∧(↑)	5
US	1F	—	5F
space	20	`	60
!	21	a	61
“	22	b	62
#	23	c	63
$	24	d	64
%	25	e	65
&	26	f	66
’	27	g	67
(	28	h	68
)	29	i	69
*	2A	j	6A
+	2B	k	6B
,	2C	l	6C
-	2D	m	6D
.	2E	n	6E
/	2F	o	6F
0	30	p	70
1	31	q	71
2	32	r	72
3	33	s	73
4	34	t	74
5	35	u	75
6	36	v	76
7	37	w	77
8	38	x	78
9	39	y	79
:	3A	z	7A
;	3B	{	7B
<	3C	|	7C
=	3D	}	7D

INDEX

Within this index *f* or *ff* after a page number means *and the following page (or pages).*

intel®

Intellec® Series II CRT and Keyboard Interface Manual
122029-001

REQUEST FOR READER'S COMMENTS

Intel's Technical Publications Departments attempt to provide publications that meet the needs of all Intel product users. This form lets you participate directly in the publication process. Your comments will help us correct and improve our publications. Please take a few minutes to respond.

Please restrict your comments to the usability, accuracy, readability, organization, and completeness of this publication. If you have any comments on the product that this publication describes, please contact your Intel representative. If you wish to order publications, contact the Intel Literature Department (see page ii of this manual).

1. Please describe any errors you found in this publication (include page number).

2. Does the publication cover the information you expected or required? Please make suggestions for improvement.

3. Is this the right type of publication for your needs? Is it at the right level? What other types of publications are needed?

4. Did you have any difficulty understanding descriptions or wording? Where?

5. Please rate this publication on a scale of 1 to 5 (5 being the best rating). ____________________

NAME ______________________________ DATE ______________

TITLE ______________________________

COMPANY NAME/DEPARTMENT ______________________________

ADDRESS ______________________________

CITY ______________ STATE ______________ ZIP CODE ______________
(COUNTRY)

Please check here if you require a written reply. □

WE'D LIKE YOUR COMMENTS ...

This document is one of a series describing Intel products. Your comments on the back of this form will help us produce better manuals. Each reply will be carefully reviewed by the responsible person. All comments and suggestions become the property of Intel Corporation.

Intel Corporation
Attn: Technical Publications M/S 6-2000
3065 Bowers Avenue
Santa Clara, CA 95051

intel®

INTEL CORPORATION, 3065 Bowers Avenue, Santa Clara, California 95051 (408) **987-8080**

Printed in U.S.A.