Chapter 4

Other Information

Chapter 4 Other Information

§1 Display

1.1. Display Settings

MultiFonts CP/M for the QX-10 supports both green and color display monitors. Since different hardware is used in each of these, it is also necessary to set the display controller in different modes. When the system uses a green monitor in the non-MFBASIC normal mode, BIOS sets the display controller in the mixed character/graphics mode and the entire screen is set in the character display area. When the system uses a green monitor in the non-MFBASIC MF mode or MFBASIC mode, BIOS sets the display controller in the mixed character/graphics mode and the entire screen is set in the graphic display area.

In the case of a color monitor, the display controller is set in the graphic mode for all modes of operation (non-MFBASIC normal and MF modes, and MFBASIC mode).
The table below summarizes the I/O port numbers and data output to set the display controller in the different display modes.
The mode settings are made with the scroll command.

Mode / I/O port	Green monitor		Color monitor
	Non-MFBASIC normal mode	Non-MFBASIC MF mode/MFBASIC mode	Non-MFBASIC normal mode/non-MFBASIC MF mode/MFBASIC mode
Command 039H	070H	070H	070H
Parameter 1 038H	0 (Lower byte of display address)	0 (Lower byte of display address)	0 (Lower byte of display address)
Parameter 2 038H	0 (Upper byte of display address)	0 (Upper byte of display address)	0 (Upper byte of display address)
Parameter 3 038H	0 or 0F0H	0F0H	0F0H
Parameter 4 038H	019H or 03FH	07FH	03FH

Parameters 1 and 2 specify the starting address for display; i.e., the VRAM address which is displayed in the upper left corner of the screen. When parameters 1 and 2 are incremented by 1 (this is done by a 16-bit logical operation) and the commands and parameters (parameters 1 to 4) are written to the I/O ports again, results differ according to mode as follows.
For a green monitor in the non-MFBASIC normal mode, the character in the upper left corner is erased and all other characters are moved back one position. A new character then moves into the bottom right corner of the screen.
For a green monitor in the non-MFBASIC MF mode or MFBASIC mode or a color monitor in any mode, the 16-dot area at the upper left corner of the screen is erased and all other portions of the screen are moved back a corresponding amount. New data then moves into the 16-dot area at the bottom right corner of the

screen.

Thus, the scroll command both determines the screen mode and scrolls data.

All characters are displayed as bit images when a green monitor is used in the mixed character/graphics mode and the graphic section is specified or when a color monitor is used. See the description of the CONOUT routine in Chapter 1 for details.

A green monitor is provided with the QX-10 as standard equipment; however, a color monitor can also be used when the optional color monitor interface board is installed. Since BIOS also provides support for color monitors, no changes need be made in MultiFonts CP/M. Differentiation between the green and color monitors and specification of the display colors for the color monitor is as follows.

<Differentiation between green and color monitors>

When a green monitor is connected, the value present at I/O port 02CH is 0FEH, and when a color monitor is connected, the value present is 0FDH.

<Display color specification for color monitors>

Any of eight different colors--black (no color), blue, red, green, purple, light blue, yellow, and white--can be displayed. However, colors other than blue, red, and green are displayed by combining two or more of these three basic colors.

Purple	- Displayed using blue and red
Light blue	- Displayed using blue and green
Yellow	- Displayed using red and green
White	- Displayed using blue, red, and green

The colors blue, red, and green are specified by setting one bit of the corresponding I/O port to "1". (Results are not assured if more than one bit is set to "1".)

I/O port 02DH <-- 01B (01H) - Blue

I/O port 02DH <-- 010B (02H) - Green

I/O port 02DH <-- 0100B (04H) - Red

Once a color has been specified, that specification remains effective until the next specification is made. The color specification is invalid when a green monitor is used.
when the screen is not zoomed.

1.2 Zoom Function

The μPD7220 is used as the display controller in the QX-10. This controller is equipped with a zoom function. The zoom function makes it possible to magnify characters for display. However, this change in size is not reflected in hard copies taken of the display; when a hard copy is made, characters appear the same as

Zooming is done by outputting commands to I/O ports as shown below.

```
IN      A,(038H)      ;
AND     6             ;
XOR     4             ;7220 BUFFER EMPTY?
JR      NZ,$-6        ;NO.
LD      A,046H        ;ZOOM COMMAND.
OUT     (039H),A      ;
LD      A,010H        ;ZOOM PARAMETER.
OUT     (038H),A      ;
LD      A,1           ;
OUT     (03AH),A      ;
.
.
```

Here, I/O port 039H is the μPD7220 command port and 038H is the parameter and status port. The zoom command is 046H; after this command has been output, it is followed by outputting the magnification to the parameter port. The magnification must also be output to port 03AH.

The zoom magnification can be set to any value from 1 to 16. The value which is output to I/O port 038H is equal to (ratio - 1) x 16, and that which is output to port 03AH is equal to (ratio - 1).

The command sequence for magnifying characters by 10 is as follows.

```
.
.
LD      A,046H
OUT     (039H),A      ;COMMAND.
LD      A,090H
OUT     (038H),A      ;PARAMETER 1 x 10
LD      A,09H
OUT     (03AH),A      ;PARAMETER 2 x 10
.
.
```

The upper left corner of the display is used as the origin for zooming. Thus, characters are enlarged from the top left corner toward the bottom right. The reason for using the upper left corner of the display as the origin is that processing becomes quite complicated if the origin is placed at the center; i.e., it becomes necessary to change the display address for the upper left corner each time the zoom function is used. This could be done using the scroll command.

The scroll command and its parameters are as follows.

Mode / I/O port	Green monitor		Color monitor
	Non-MFBASIC normal mode	Non-MFBASIC MF mode/MFBASIC mode	Non-MFBASIC normal mode/non-MFBASIC MF mode/MFBASIC mode
Command 039H	070H	070H	070H
Parameter 1 038H	0 (Lower byte of display address)	0 (Lower byte of display address)	0 (Lower byte of display address)
Parameter 2 038H	0 (Upper byte of display address)	0 (Upper byte of display address)	0 (Upper byte of display address)
Parameter 3 038H	0F0H	0F0H	0F0H
Parameter 4 038H	03FH	07FH	03FH

See the discussion of the CONOUT routine in Chapter 1 concerning display addresses and display modes.

Although the display address can be changed independently of the zoom function, BIOS would have no way of knowing that a change had been made. Therefore, subsequent character display might not be performed properly the next time the CONOUT routine is called. Thus, if the display address is changed, it must be restored (or the screen must be cleared) before calling CONOUT. The display address is saved as two bytes in SPOS (address 0FE70H) in the BIOS common data area, and can be restored by outputting that value to an I/O port with the scroll command.

BIOS support is not provided when the zoom function is used, and cannot be utilized again until the original character size is restored. This is done by setting a magnification ratio of 1.

Magnification of characters for display by the zoom function is done entirely by hardware, and no software considerations are necessary. The effect is the same as if the display were viewed through a magnifying glass. Although portions of the image extending beyond the edge of the screen are not visible, all data in VRAM can be brought into view by executing the scroll command to move the display address up, down, left, or right. Again, results are not assured if CONOUT is called for display.

However, the CONOUT routine can be used to output an escape sequence to clear the screen. In this case, it is the user's responsibility to restore the zoom magnification to the original ratio.

§2 Memory Banks

2.1 Using the User Banks

User bank 1 can be used as a program area or data area as long as MFBASIC is not started. User bank 2 can be used in the same manner. When one of the user banks is to be used as program area, the program must be loaded into the corresponding bank. This is done as follows.

(1) Read in one sector of the subroutine program by calling BDOS from a program in the main bank.

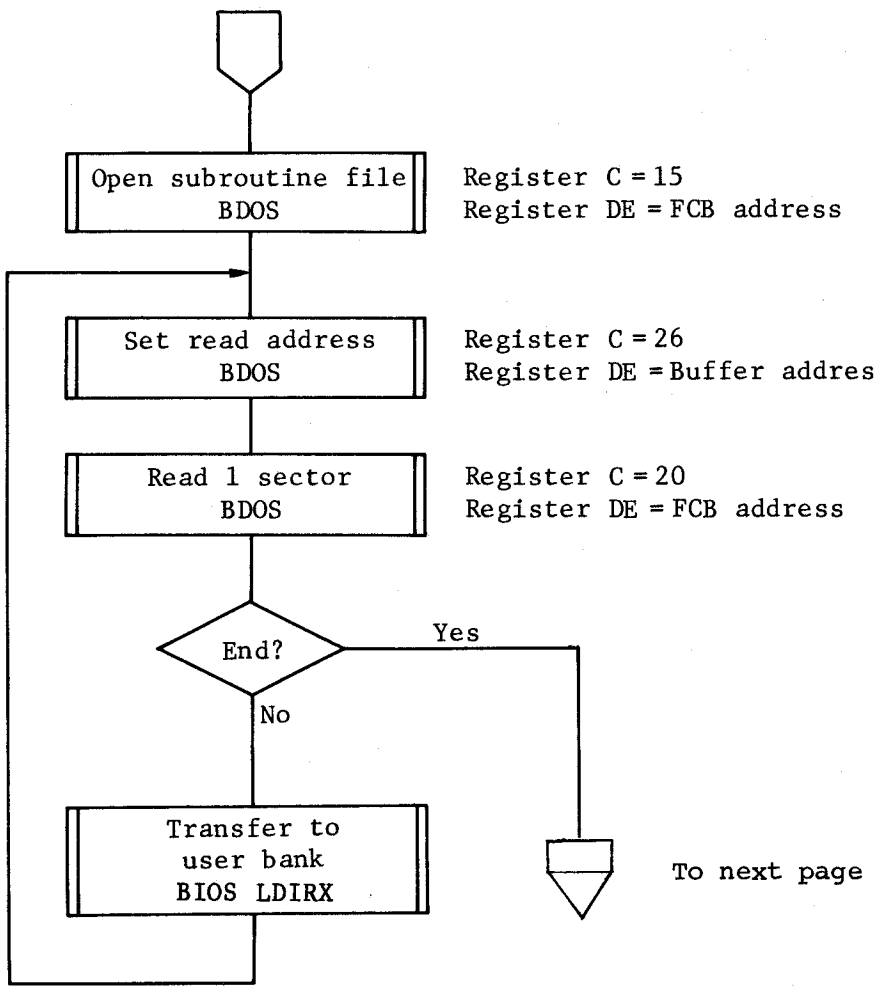
(2) Call BIOS routine LDIRX to transfer the data read to user bank 1 or 2.

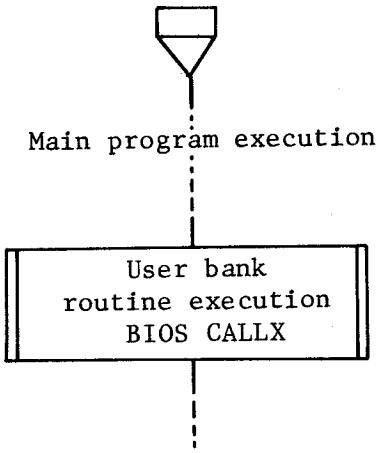
(3) Repeat steps 1 and 2 until the entire subroutine has been loaded.

(4) Start processing with the main program after subroutine program loading has been completed.

(5) Use the CALLX routine of BIOS to call the subroutine in the user bank. (CALLX makes it possible to execute subroutines in other banks.)

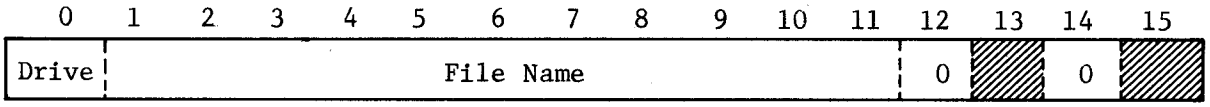
These procedures are shown as a flowchart below.

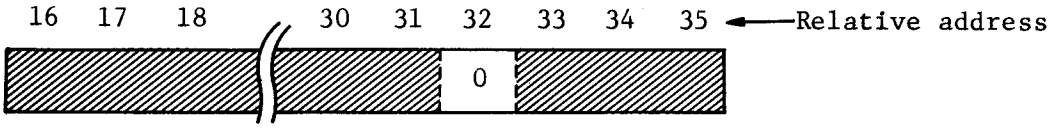




User bank subroutine call

FCB for BDOS calls





As shown above, the FCB (file control block) is composed of 36 bytes. The FCB is referenced by BDOS when a file is accessed. Specification of a file control block is mandatory whenever the user accesses a program or data file on a flexible disk.

An example of the FCB specification for opening a subroutine program file is shown below.

Byte 0: Drive number

0 - Disk currently logged in

1 - Drive A

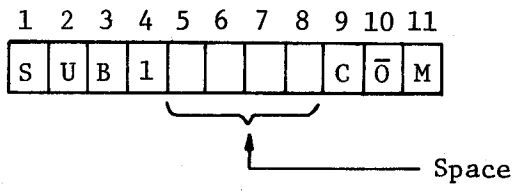
2 - Drive B

3 - Drive C

4 - Drive D

Bytes 1 - 11: Filename

Example: SUB1.COM



Only upper-case characters can be specified for the file name.

Bytes 12, 14, and 32: Set to 0.

Any data may be specified for bytes which are shaded.

When a file is opened by making a BDOS call, prepare an FCB such as that shown above before making the call.

Normally the FCB is located in the main bank area starting at address 05CH. However, it can also be prepared in a user program area.

2.2 Switching Banks

User programs are normally loaded into the main bank area from 0100F to 0DFFFH for execution. However, if memory banks are switched during program execution, program control is switched to the selected memory bank. Therefore, at the user's option it is possible to continue program execution in the selected bank. (See 2.1 "Using the User Banks" for procedures for loading programs or data into banks other than the main bank.) The sequence for switching memory banks is as follows.

(1) Save the current bank number.

(2) Set the new bank number in the BIOS common data area.

(3) Output the new bank number to the applicable I/O port.

If the above sequence is loaded into part of resident memory, there will be no problem with execution of the next instruction even if banks are switched. Therefore, every effort should be made to avoid switching banks from other than resident memory, and BIOS entries LDIRX, JUMPX, or CALLX should be used for continuing execution.

An example of coding for continuation of a program in another bank is shown below.

```
MEMBANK   EQU     0FEF0H          ;CURRENT BANK NO.
CBANK     EQU     030H            ;I/O PORT.
NBANK     EQU     018H            ;I/O PORT.
          :
          DI                      ;DISABLE INTERRUPT.
          IN      A,(CBANK)       ;KEEP CURRENT BANK NO.
          LD      (SVBANK),A      ;
          LD      A,040H          ;SELECT USER BANK.
          LD      (MEMBANK),A     ;SET NEW BANK NO.
          OUT     (NBANK),A       ;CHANGE.
          EI                      ;ENABLE INTERRUPT.
          :
SVBANK:   DS      1               ;BANK NO. SAVE AREA.
```

In the example above, MEMBANK is the BIOS control data byte in the BIOS common data area which is used for saving the current bank number. Since BIOS uses this information for switching banks, the new bank number must be set in MEMBANK.

The bank number is set as a bit pattern (see below) in which only one of the bits is set to "1". Note that operation is not assured if more than one bit of this bit pattern is "1". Also note that access to all memory other than the resident area (addresses 0E000H to 0FFFFH) will be cut off if all bits of this pattern are set to "0".

Bank	Bit pattern
Main bank	010H
System bank (BIOS)	020H
User bank 1	040H
User bank 2 (option RAM)	080H

§3 Keyboard Control

The keyboard is interfaced with the system via an NEC μPD7201. Data transfer between the keyboard and the main system is bidirectional, and is performed at 1200 bps. The signals used for transfer of data are as follows.

Pin No.	Signal
1	RxD
2	Clock
3	+12 VDC
4	TxD
5	GND
6, 7, 8	NC/GND

Data sent to the main system from the keyboard includes (1) the matrix codes of keys pressed, and (2) the LED status codes. The control and status codes for the LEDs are as shown in the table below.

LED control/status codes

Key	Control	code	Status	code
	On	Off	On	Off
CAPS LOCK	41H	40H	C1H	C0H
INSert	45H	44H	C5H	C4H
SF4	49H	48H	C9H	C8H
SF3	4BH	4AH	CBH	CAH
SF2	4DH	4CH	CDH	CCH
SF1	4FH	4EH	CFH	CEH

Only matrix codes are generated when the keys are pressed; these matrix codes are converted into the corresponding internal character codes by the system software. The momentary shift keys (GRPH, left SHIFT, right SHIFT, and CTRL) generate both make and break codes. The keyboard matrix scanning codes generated are as shown in the following table.

KEYBOARD SCAN CODES (ASCII)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0													CAPS LOCKS OFF			
1	F4	LF		CAPS LOCK		Q	2	F3					CAPS LOCK ON			
2	F5	A5	A4	(SPACE)		W	# 3	F2								
3	F6			Z	A	E	$ 4	F1								
4	F7	000		X	S	R	% 5		SHIFT (R) BRK				INS OFF			
5	F8		3	C	D	T	& 6	ESC	SHIFT (R) MAKE				INS ON			
6	F9		2	V	F	Y	' 7	! /	SHIFT (L) BRK							
7	F10	0	1	B	G	U	(8	TAB	SHIFT (L) MAKE							
8	A1	+	-	N	H	I	) 9						MF4 OFF			
9	BREAK	6	9	M	J	O	~ 0						MF4 ON			
A	PAUSE	5	8	< ,	K	P	= -		CTRL BRK				MF3 OFF			
B	SCAN DUMP	4	7	> .	L	@	∿		CTRL MAKE				MF3 ON			
C	HELP		=		+ ;	{ [			GRPH BRK				MF2 OFF			
D	A2		*		* :	}]	BS		GRPH MAKE				MF2 ON			
E	A3		/			INS	HOME						MF1 OFF			
F	SF1		,		? /	DEL	CLS						MF1 ON			

Shift Key

LED

Data sent to the keyboard from the main system controls the keyboard as follows.

(1) Auto-repeat start time setting

The auto-repeat start time (the time from the moment a key is first pressed until the auto-repeat function starts operating) can be set in 25 ms increments within the range from 300 ms to 1075 ms.

(2) Auto-repeat interval setting

The auto-repeat interval (the interval at which input is accepted from a key after the repeat function starts operating) can be set to within the range from 30 ms to 185 ms in 5 ms increments.

(3) Auto-repeat control

The repeat function can be enabled or disabled. When disabled, only one character is input each time a key is pressed, no matter how long that key is held down.

(4) Keyboard LED control

LEDs built into certain keytops can be turned on or off by specifying the corresponding LED number.

(5) LED status read

The current status of all LEDs (on or off) can be checked.

(6) SW status read

The current status of the keyboard switches (SHIFT, CTRL, and GRPH) can be checked.

(7) Keyboard enable/disable

Input from the keyboard buffer can be either enabled or disabled.

(8) Reset

The keyboard controller can be reset. The status upon reset is as follows.

- o The keyboard buffer (built into the keyboard) is cleared.
- o The repeat function is enabled.
- o The repeat starting time is set to 500 ms.
- o The repeat interval is set to 50 ms.
- o All LEDs are turned off.

The reset function also executes a keyboard diagnostic program which turns on the LEDs in sequence, then turns all LEDs on and back off again; and which scans all keys, outputting 0H if no key is pressed and 0FFH if any key is pressed.

Control commands for the eight functions described above

are summarized in the table below.

Command	Bit pattern 7	6	5	4	3	2	1	0	Example
Repeat starting time set	0	0	0	*1		n_1			00H ∿ 01FH
Repeat interval set	0	0	1	*2		n_2			020H ∿ 03FH
Repeat control	1	0	1	X	X	X	X	*3 ON/OFF	0A0H, 0A1H
LED ON/OFF	0	1	0	X	*4	LED No.		*5 ON/OFF	040H ∿ 04FH
LED status read	0	1	1	X	X	X	X	X	07FH
SW status read	1	0	0	X	X	X	X	X	080H
Keyboard enable	1	1	0	X	X	X	X	*6 ON/OFF	0C0H, 0C1H
Reset	1	1	1	X	X	X	X	*7 Diagnosis	0FEH

Either 0 or 1 may be specified for bits marked with an "x".

Notes:

*1,*2 nl and n2 specify the repeat starting time and the repeat interval. The repeat starting time is equal to 300 ms + nl x 25 ms, and the repeat interval is equal to 30 ms + n2 x 5 ms.

*3 Bit 0 controls the repeat function. The repeat function is inhibited if 0 is set in bit 0, and is enabled if 1 is set in bit 0.

*4 With the LED status read command, bits 3, 2, and 1 designate the LED to be turned on or off as follows.

LED No.=0 - CAPS LOCK LED
LED No.=2 - INS LED
LED No.=4 - MF4 LED
LED No.=5 - MF3 LED
LED No.=6 - MF2 LED
LED No.=7 - MF1 LED

*5 This bit determines whether the specified LED is turned on or off. When bit 0 is 0, the LED is turned off; when it is 1, the LED is turned on.

*6 Input from the keyboard is inhibited if bit 0 is 0 (although data currently being sent is output); input is enabled if this bit is set to 1.

*7 Specifies execution of the diagnostic program. The diagnostic program is executed when bit 0 is set to 0, and is not executed when it is set to 1.

All of these keyboard control commands are output to I/O port 010H.

Example: Sequence to light the MF3 LED

```
        .
        .
        LD      A,01000111B ;MF3 LED ON.
        OUT     (010H),A    ;
        .
```

Normally, the results of the LED status read command or the SW status read command cannot be read by the BIOS CONIN routine. Therefore, the user must read in the value as follows.

(1) Call the BIOS CONST routine to confirm that there is no data in the keyboard buffer. If there is data and it can be ignored, initialize the keyboard data pointer as follows.

```
        .
        .
KBUF    EQU     0FE2EH      ;KEYBOARD BUFFER ADDR.
INPTR   EQU     0FE21H      ;KEY PUT POINTER.
KEYPTR  EQU     0FE23H      ;KEY GET POINTER.
        .
        .
        DI                  ;DISABLE INT.
        LD      HL,KBUF     ;INITIALIZE KEY DATA.
        LD      (INPTR),HL  ;POINTER.
        LD      (KEYPTR),HL ;
        EI                  ;
        .
        .
```

(2) Output the applicable command (LED or SW status read).

(3) Call the CONST routine to wait until data is input.

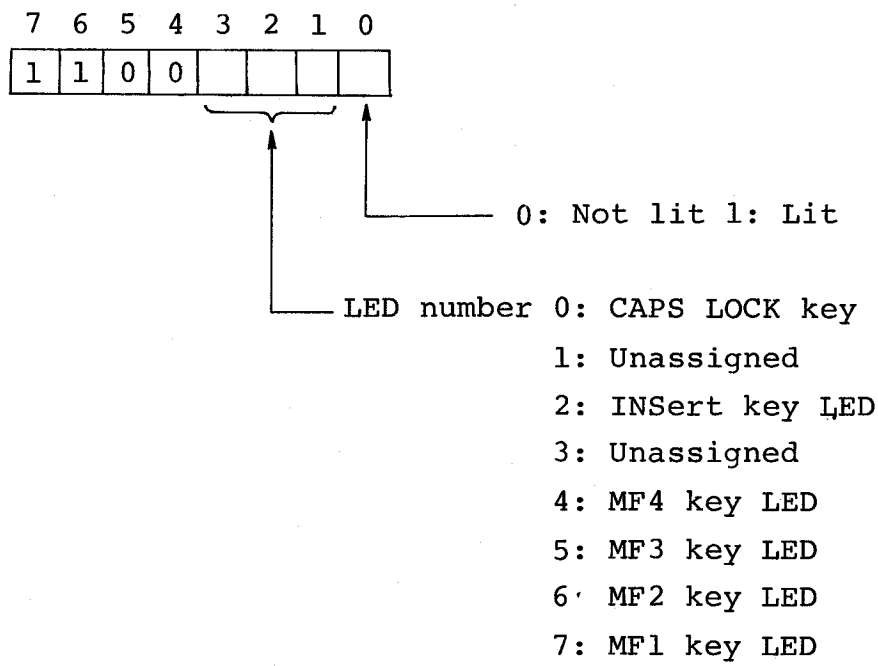
(4) Read in 8 bytes of data from KBUF (address 0FE2EH) in the BIOS common data area for each command executed, then reinitialize the keyboard buffer.

The following is an example of a program which does steps 1 to 4 above.

```
KBUF    EQU     0FE2EH
INTPTR  EQU     0FE21H
KEYPTR  EQU     0FE23H
;
CONST   EQU     0F606H
;
        .
        .
        CALL    CONST       ;CHECK INPUT STATUS.
        OR      A           ;
        CALL    NZ,KRINIT   ;IF READY
        LD      A,060H      ;READ LED STATUS.
        OUT     (010H),A    ;
        LD      HL,KBUF     ;KEY BUFFER HEAD ADDR.
        LD      B,8         ;8 TIMES.
LOOP:   PUSH    BC
        PUSH    HL
        CALL    CONST       ;CHECK STATUS.
        POP     HL
```

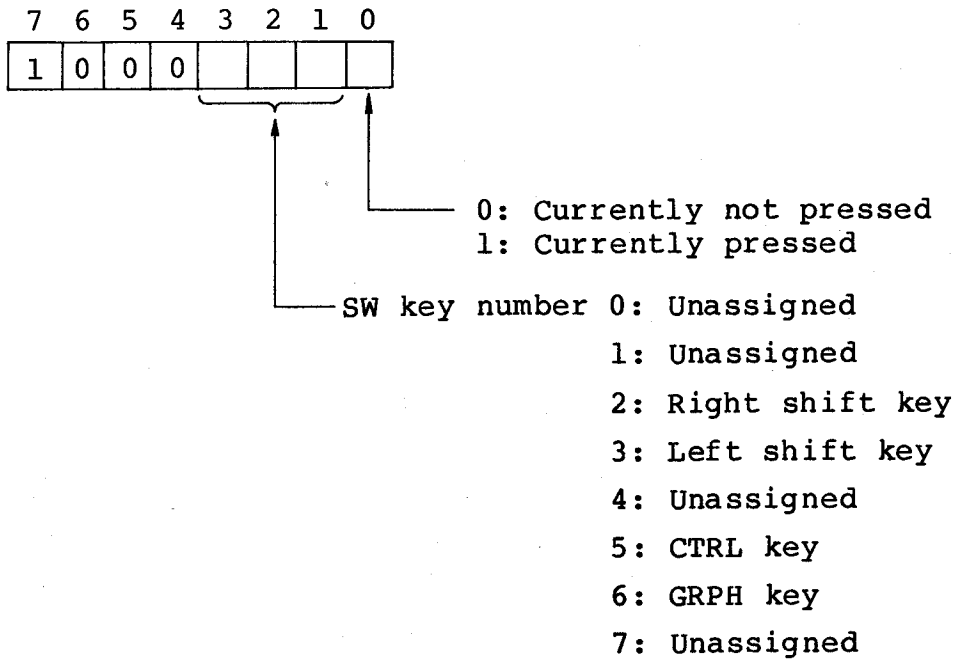
```
            POP     BC
            OR      A
            JR      Z,LOOP          ;NO DATA.
            LD      A,(HL)          ;A=LED DATA.
            .                       .
            .                       .
            INC     HL              ;
            DJNZ    LOOP            ;
            CALL    KBINIT          ;INITIALIZE POINTER.
            .                       .
            .                       .
KBINIT:     DI                      ;SUBROUTINE.
            LD      HL,KBUF         ;
            LD      (INTPTR),HL     ;INITIALIZE KEY DATA.
            LD      (KEYPTR),HL     ;POINTER.
            EI                      ;
            RET                     ;
            .
            .
```

Eight bytes of data are returned as the result of the LED status read command. These values are composed of bit patterns such as the one shown below.



Example: The value 011001011B indicates that the MF3 key LED is lit.

Eight bytes are also returned as the result of the SW data read command. These are composed of bit patterns such as the following.



Example: The value 01000011B indicates that the left shift key is currently being pressed.

The repeat starting time, repeat interval, and repeat control can also be set by setting data in KBSTRT (address 0FE4DH), KBINTVL (address 0FE4EH), and KBREPT (address 0FE4CH) in the BIOS common data area. However, the values set are different in the case of repeat control; set 0FFH to enable the repeat function and 0H to disable it.

§4 Light Pen

Depending on the display mode with which it is used, light pen sensitivity may be quite low. This is particularly true when used with other than a green monitor in the non-MFBASIC normal mode. The reason for this is as follows.

Light pen detection by the μPD7220 is performed by reading data twice and ignoring it if the results of the two reads do not match. For modes other than the non-MFBASIC normal mode (or other than a green monitor), there are 400 character addresses in the vertical direction; i.e., data addresses differ in 1-dot units. Therefore, if the light pen slips even slightly when it is pressed against the screen, the address (and particularly the vertical address) is likely to change. This means that the light pen must be held at a constant angle to the screen and kept from slipping in any direction. Further, sensitivity is particularly low with red when a color monitor is used.

This problem does not occur when the light pen is used with a green monitor in the non-MFBASIC normal mode.

The address detected by the light pen is posted by reading it in from an I/O port; however, the address read is not the same as the actual address of the screen area against which the light pen was pressed. BIOS corrects for this difference as follows.

	Green monitor		Color monitor
Display mode	Non-MFBASIC Normal mode	Non-MFBASIC MF mode/MFBASIC mode	Non-MFBASIC normal mode/Non-MFBASIC MF mode/MFBASIC mode
Correction value (2 bytes)	-5 (0FFFBH)	-2 (0FFFEH)	-3 (0FFFDH)
BIOS common data area containing the correction value	MLPBIAS (0FEF6H)	GLPBIAS (0FE9BH)	CLPBIAS (0FEF4H)

As indicated above, different 2-byte correction values are stored in the BIOS common data area for each mode. If the address read differs from the actual one by too much, these values can be changed using the DDT transient command. See the discussion of CONOUT in Chapter 1 for procedures for using DDT to change data in the BIOS common data area.

§5 RS-232C Interface

Including the optional RS-232C cards, the QX-10 supports a total of 5 RS-232C interface channels. Except with respect to maximum transfer speed, processing for both the standard and optional interfaces is performed in exactly the same manner. Further, data can be received on all five channels simultaneously as long as the following conditions are satisfied.

(a) The transfer speed must not be more than 300 bps.

(b) SI/SO control must not be used.

(c) XON/XOFF control may be specified; however, in this case the interfaces cannot be used for transfer of binary data.

(d) Every effort should be made to minimize I/O access and bank switching during processing of receive data. It is particularly important to avoid access to flexible disk drives or output of data to the screen.

BIOS routines provided for using the RS-232C interfaces include RSOPEN, RSCLOSE, RSINST, RSOUTST, RSIN, RSOUT, and RSIOX. However, except for RSIOX, these routines can only be used for control of the main board interface. Therefore, the RSIOX routine must be called for control of the optional RS-232C interfaces. Further, the receive buffer is fixed to the 512-byte area starting at address 0DA00H of the system bank for all routines other than RSIOX. This means that the RSIOX routine cannot be used together with other RS-232C control routines.

The standard RS-232C interface can be assigned to any logical device name (LST:, PUN:, RDR:, or CON:) by specifying the appropriate setting in the I/O byte. Assignment of the RS-232C interface to a logical device does not open the interface; the interface is opened only when data is input or output. The RS-232C interface is opened by resetting it (the μPD7201). Therefore, the standard RS-232C interface cannot be used by a user program or MFBASIC when it has been specified in the I/O byte. In this case, use one of the optional RS-232C interfaces for standard system I/O.

When connecting an HX-20 to the QX-10 via their RS-232C interfaces, data sent from the HX-20 to the QX-10 will be shifted by one bit (resulting in an error) if the RS-232C interface on the QX-10 side is opened before that on the HX-20 side. The reason for this is that when the HX-20's RS-232C interface is opened, noise is output during initialization of the interface when its power is turned on. The QX-10 interprets this noise as a start bit, so that subsequent data received is shifted by one bit. Therefore, be sure to open the RS-232C interface of the HX-20 prior to that of the QX-10.

All RS-232C interfaces are set to the asynchronous mode by BIOS. When any of the interfaces is to be used in the synchronous mode, jumper settings must be changed and RS-232C control must be provided.

The jumper settings for the respective modes are as follows.

Main board RS-232C interface

Mode / Jumper	Asynchronous ⇔ Synchronous	
J2	A1	B1
	A2	B2

Changing these settings for synchronous communication causes the external clock to be used in the case of the main board RS-232C interface.

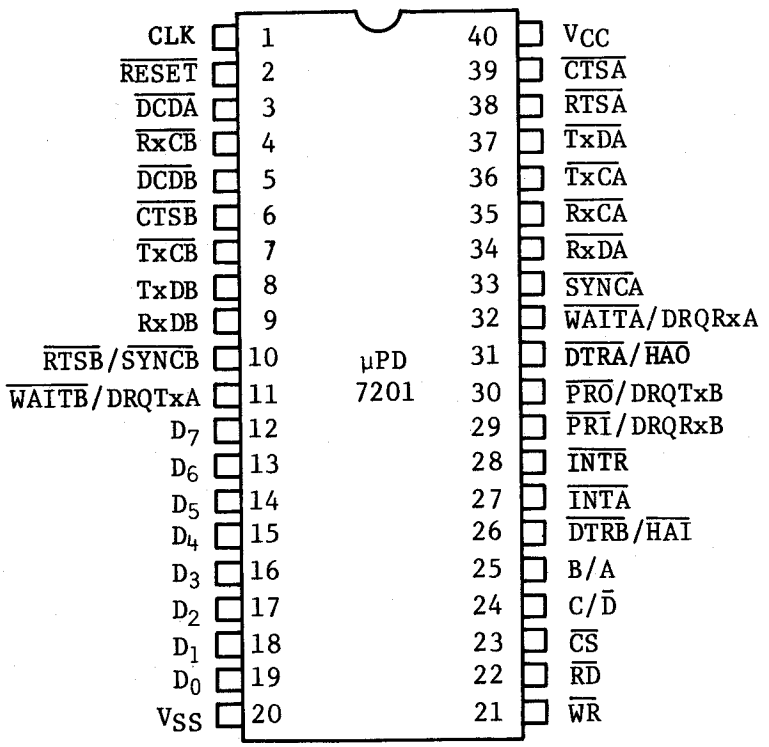
Jumper settings for the optional RS-232C interface cards are as follows.

Mode / Jumper	Asynchronous	⇔ Synchronous	
		External Send Clock	Internal Send Clock
J6	B	A	B
J7	A	B	B

When the interfaces are set to the synchronous mode, the external clock is used as the receive clock.

Settings of the various registers for synchronous communication are described below.

PIN CONFIGURATION



PIN DESCRIPTION

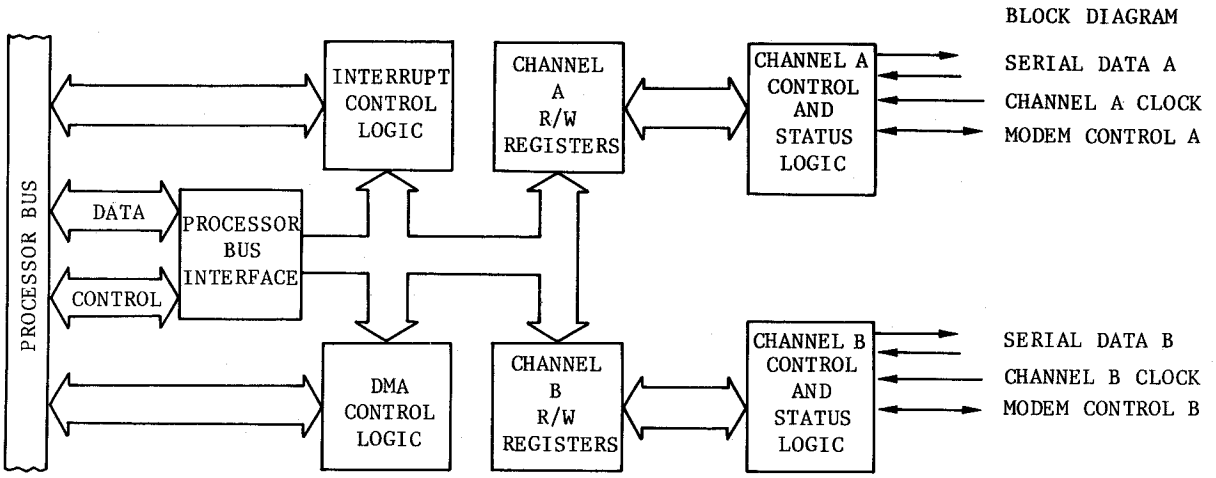
μPD7201

NO.	PIN		DESCRIPTION
	SYMBOL	NAME	
12-19	D_0-D_7	System Data Bus (bidirectional, 3-state)	The system data bus transfers data and commands between the processor and the μPD7201. D_0 is the least significant bit.
25	B/$\overline{A}$	Channel A or B Select (input, High selects Channel B)	This input defines which channel is accessed during a data transfer between the processor and the μPD7201.
24	C/$\overline{D}$	Control or Data Select (input, High selects Control)	This input defines the type of information transfer performed between the processor and the μPD7201. A High at this input during a processor write to or read from the μPD7201 causes the information on the data bus to be interpreted as a command for the channel selected by B/$\overline{A}$. A low at C/$\overline{D}$ means that the information on the data bus is data.
23	$\overline{CS}$	Chip Select (input, active Low)	A low level at this input enables the μPD7201 to accept command or data inputs from the processor during a write cycle, or to transmit (data to the processor during a read cycle.
1	CLK	System Clock (input)	The μPD7201 uses standard TTL clock.
22	$\overline{RD}$	Read (input active Low)	If $\overline{RD}$ is active, a memory or I/O read operation is in progress. $\overline{RD}$ is used with C/$\overline{D}$. B/$\overline{A}$ and $\overline{CS}$ to transfer data from the μPD7201 to the processor or the memory.
21	$\overline{WR}$	Write (input, active Low)	The $\overline{WR}$ signal is used to control the transfer of either command or data from the processor or the memory to the μPD7201.
2	$\overline{RESET}$	Reset (input, active Low)	A low $\overline{RESET}$ disables both receivers and transmitters, forces TxDA and TxDB marking, forces the modem controls high and disables all interrupts. The control registers must be rewritten after the μPD7201 is reset and before data is transmitted or received. $\overline{RESET}$ must be active for a minimum of one complete CLK cycle.
10,38	$\overline{RTSA}$, $\overline{RTSB}$	Request to Send (outputs, active Low)	When the $\overline{RTS}$ bit is set, the $\overline{RTS}$ output goes Low. When the $\overline{RTS}$ bit is reset in the Asynchronous mode, the output goes High after the transmitter is empty. In Synchronous modes, the $\overline{RTS}$ pin strictly follows the state of the $\overline{RTS}$ bit. Both pins can be used as general-purpose outputs.
10,33	$\overline{SYNCA}$,$\overline{SYNCB}$	Synchronization (inputs/outpus, active Low)	These pins can act either as inputs of outputs. In the Asynchronous Receive mode, they are inputs similar to $\overline{CTS}$ and $\overline{DCD}$. In this mode, the transitions on these lines affect the state of the Sync/Hunt status bits in Read Register 0. In the External Sync mode, these lines also act as inputs. When external synchronization is achieved. $\overline{SYNC}$ must be driven Low on the second rising edge of $\overline{RxC}$ after that rising edge of $\overline{RxC}$ on which the last bit of the sync character was received. In other words, after the sync pattern is detected, the external logic must wait for two full Receive Clock cycles to activate the $\overline{SYNC}$ input. Once $\overline{SYNC}$ is forced Low, it is wise to keep it Low until the processor informs the external sync logic that synchronization has been lost or a new message is about to start. Character assembly begins on the rising edge of $\overline{RxC}$ that immediatley precedes the falling edge of $\overline{SYNC}$ in the External Sync mode. In the Internal Synchronization mode (monosync and Bisync), these pins act as outputs that are active during the part of the receive clock ($\overline{RxC}$) cycle in which sync characters are recognized. The sync condition is not latched, so these outputs are active each time a sync pattern is recognized, regardless of character boundaries.
26,31			These outputs follow the state programmed into the DTR bit. They can also be programmed as general-purpose outputs.

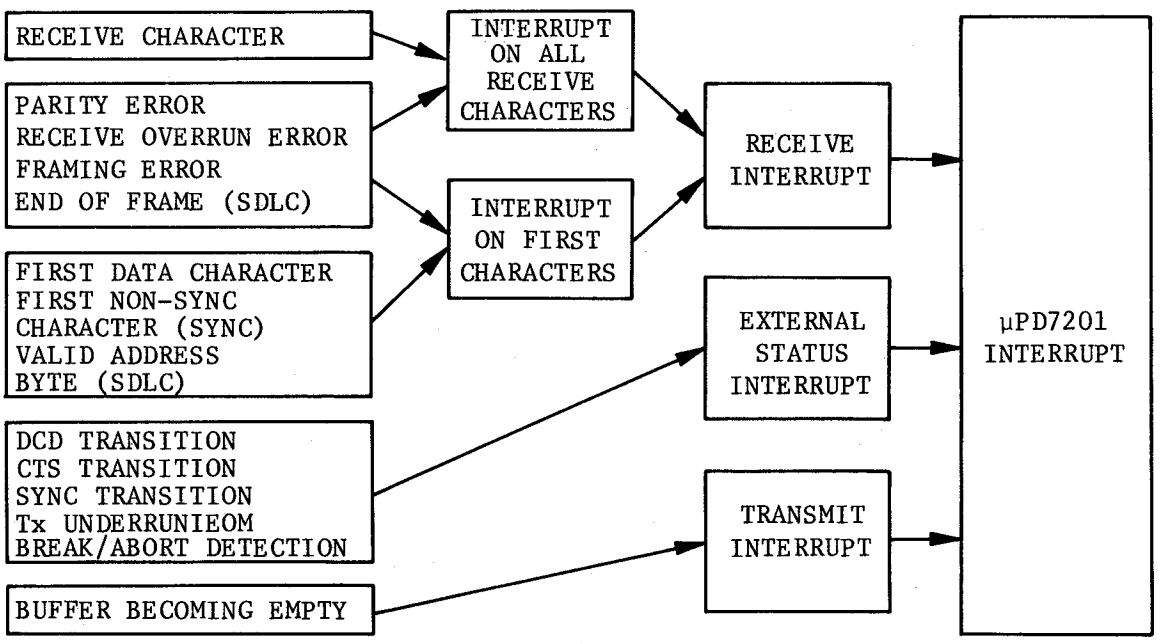
PIN DESCRIPTION
(CONT.)

NO.	PIN		DESCRIPTION
	SYMBOL	NAME	
27	$\overline{INTA}$	Interrupt Acknowledge (input, active Low)	This signal is generated by the processor and is sent to all peripheral devices. It serves to acknowledge the interrupt and to allow the highest priority interrupting device to put an 8-bit vector on the bus. INT and INTA are compatible with the fully nested option of the μPD8259A-5.
29	$\overline{PRI}$	Priority In (input, active Low)	These signals are daisy chained through the peripheral device controllers. The signal on these lines is intact until a device with a pending interrupt request is found on the chain. After that device, this signal holds off lower priority device interrupts. A higher priority device can interrupt the processing of an interrupt from a lower priority device, provided the processor has interrupts enabled. $\overline{PRI}$ is used with $\overline{PRO}$ to form a priority daisy chain when there is more than one interrupt-driven device. A low on this line indicates that no other device of higher priority is being serviced by a processor interrupt service routine. $\overline{PRO}$ is Low only if $\overline{PRI}$ is Low and the processor is not servicing an interrupt from the μPD7201. Thus, this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its processor interrupt service routine.
30	$\overline{PRO}$	Priority Out (output, active Low)	
11,29 30,32	DRQTxA, DRQTxB DRQRxA, DRQRxB	DMA Request (outputs, active High)	These signals are generated by the receiver or transmitter of Channel A and Channel B. These signals can be connected to most DMA Controllers directly and are used for handshaking during DMA transfer.
26	$\overline{HAI}$	DMA Acknowledge (input, active Low)	Typically, the HLDA signal driven from the processor is input to the $\overline{HAI}$ terminal of the highest priority μPD7201, and the $\overline{HAO}$ output of that μPD7201 is daisy chained to the $\overline{HAI}$ input of the lower priority μPD7201 and propagated downstream. $\overline{HAI}$ and $\overline{HAO}$ signals provide aknowledgement for the highest priority outstanding DMA request.
31	$\overline{HAO}$	DMA Acknowledge (output, active Low)	
28	$\overline{INT}$	Interrupt Request (output, open collector, active Low)	When the μPD7201 is requesting an interrupt, it pulls $\overline{INT}$ low.
11,32	$\overline{WAITA}$, $\overline{WAITB}$	(Outputs, open drain)	Wait lines for both channels that synchronize the processor to the μPD7201 data rate. The reset state is open drain.
6,39	$\overline{CTSA}$, $\overline{CTSB}$	Clear to Send (inputs, active Low)	When programmed as Auto Enables, a Low on these inputs enables the respective transmitter. If not programmed as Auto Enables, these inputs may be programmed as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow-risetime inputs. The μPD7201 detects pulses on these inputs and interrupts the processor on both logic level transitions. The Schmitt-trigger inputs do not guarantee a specified noise-level margin.
3,5	$\overline{DCDA}$, $\overline{DCDB}$	Data Carrier Detect (inputs, active Low)	These signals are similar to the CTS inputs, except they can be used as receiver enables.
9,34	RxDA, RxDB	Receive Data (inputs, active High)	
8,37	TxDA, TxDB	Transmit Data (outputs, active High)	
4.35	$\overline{RxCA}$, $\overline{RxCB}$	Receiver Clocks (inputs)	The Receiver Clocks may be 1, 16, 32, or 64 times the data rate in asynchronous modes. Receive data is sampled on the rising edge of $\overline{RxC}$.
7,36	$\overline{TxCA}$, $\overline{TxCB}$	Transmitter Clocks (inputs)	In asynchronous modes, the Transmitter Clocks may be 1, 16, 32, or 64 times the data rate. The multiplier for the transmitter and the receiver must be the same. Both $\overline{TxC}$ and $\overline{RxC}$ inputs are Schmitt-trigger buffered for relaxed rise- and fall-time requirements (no noise margin is specified). TxD changes on the falling edge of $\overline{TxC}$. Note that $\overline{TxC}$ and $\overline{RxC}$ in Channel B are on a common pin, $\overline{RxCB}/\overline{TxCB}$.

BLOCK DIAGRAM



INTERRUPT STRUCTURE



WR0

7	6	5	4	3	2	1	0
CRC1	CRC0	CMD2	CMD1	CMD0	PTR2	PTR1	PTR0
00 Null		000 Null			(000 WR0,RR0)		
01 Reset Rx CRC Checker		001 Send Abort (SDLC)			001 WR1,RR1		
10 Reset Tx CRC Generator		010 Reset E/S INT			010 WR2,RR2 (CH.B only)		
11 Reset Tx Underrun/EOM Bit		011 Channel Reset			011 WR3		
		100 Enable INT on Next Rx CHAR			100 WR4		
		101 Reset Tx INT/DMA Pending			101 WR5		
		110 Error Reset			110 WR6		
		111 End of INT (CH.A only)			111 WR7		

WR1

7	6	5	4	3	2	1	0
Wait Enable	0	Wait on Rx/Tx	Rx INT Mode 1	Rx INT Mode 0	Status Affects Vector	Tx INT/DMA Enable	E/S INT Enable
0 Disable	0	0 Tx	00 INT/DMA Disable		0 Fixed Vector	0 Disable	0 Disable
1 Enable		1 Rx	01 INT on First CHAR.		1 Modified Vector	1 Eable	1 Enable
			10 INT on All CHAR. (Parity included)				
			11 INT on All CHAR. (Parity included)				

WR2A

7	6	5	4	3	2 *	1	0
$\overline{RTSB}/\overline{SYNC3}$ Select	0	Vector Mode	INT Model	INT Mode 0	Priority Select	INT/DMA Model	INT/DMA Mode 0
0 $\overline{RTSB}$	0	0 Non-Vectored	00 35-1 Vectored		0 TxA-RxB	00 Both CH. INT	
1 $\overline{SYNC3}$		1 Vectored	01 35-2 Vectored		1 RxB-TxA	01 CH.A:DMA CH.B:INT	
			10 36 Vectored			10 Both CH.DMA	

* 0: RxA TxA RxB TxB E/SA E/SB
1: RxA RxB TxA TxB E/SA E/SB

High ---------- Low

WR2B	7	6	5	4	3	2	1	0
	V_7	V_4	V_5	V_4	V_3	V_2	V_1	V_0

WR3	7	6	5	4	3	2	1	0
	Rx Bits/ CHAR1	Rx Bits/ CHAR0	Auto Enable	Enter Hunt Phase	Rx CRC Enable	Adrs Search Mode	CYNC CHAR Load Inhibit	Rx Enable
Async	00 5Bits/CHAR. 01 7Bits/CHAR. 10 6Bits/CHAR. 11 8Bits/CHAR.		0 Disable 1 Enable	0	0	0	0	0 Disable 1 Enable
Sync				0 Nop 1 Re-enable	0 Disable 1 Enable		0 Nop 1 Inhibit	

WR4	7	6	5	4	3	2	1	0
	Clock Rate 1	Clock Rate 0	SYNC Mode 1	SYNC Mode 0	Stop Bits 1	Stop Bits 0	Parity Even/Odd	Parity Enable
Async	00 x1 Clock Mode 01 x16 Clock Mode 10 x32 Clock Mode 11 x64 Clock Mode		00		01 1 stop bit/CHAR 10 1 1/2 stop bits/CHAR 11 2 stop bits/CHAR		0 Odd 1 Even	0 Disable 1 Enable
Sync	00		00 8 bit SYNC 01 16 bit SYNC 11 EXT SYNC		00 SYNC Mode			

WR5	7	6	5	4	3	2	1	0
	DTR	Tx Bits/ CHAR 1	Tx Bits/ CHAR 0	Send Break	Tx Enable	CRC-16/ CCITT	RTS	Tx CRC Enable
Async	0 $\overline{DTR}$=1 1 $\overline{DTR}$=0	00 5 or Less Bits/CHAR 01 7 Bits/CHAR. 10 6 Bits/CHAR. 11 8 Bits/CHAR.		0 Marking 1 Spacing	0 Disable 1 Enable	0	0 $\overline{RTS}$=1 (all sent) 1 $\overline{RTS}$=0	0
Sync						0 CCITT-0 1 CRC-16	0 $\overline{RTS}$=1 1 $\overline{RTS}$=0	0 Disable 1 Enable

	WR7								WR6							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	SYNC Bit 15∿8								SYNC Bit 7∿0							
Async	not used								not used							
Monosync	Rx SYNC CHAR.								Tx STNC CHAR.							
Bisync	SYNC CHAR. Bit 15∿8								SYNC CHAR. Bit 7∿0							
EXT sync	not used								Tx SYNC CHAR.							

RR0	7	6	5	4	3	2	1	0
	Break/ Abort	Tx Under-run/EOM	CTS	SYNC/ Hunt	DCD	Tx Buffer Empty	INT Pending	Rx CHAR. Available
Async	1 Break Detected 0 Normal	Unknown	1 $\overline{CTS}$=0 0 $\overline{CTS}$=1	1 $\overline{SYNC}$=0 0 $\overline{SYNC}$=1	1 $\overline{DCD}$=0 0 $\overline{DCD}$=1	1 Tx Buff Empty 0 Tx Buff Full	1 INT Pending 0 No INT Pending	1 Rx CHAR Available 0 Rx Buff Empty
EXT sync	0	1 Tx Underrun /EOM 0 Not Tx Underrun				1 Tx Buff Empty 0 Tx Buff Full (With or without CRC)	(CH.A only)	
Monosync Bisync				1 Hunt Phase 0 Exit Hunt Phase				

RR1	7	6	5	4	3	2	1	0
	End of Frame	CRC/ Framing Error	Rx Overrun Error	Parity Error	Residue Code 2	Residue Code 1	Residue Code 0	All Sent
Async	0	1 Framing Error 0 No Error	1 Rx Overrun Error 0 No Error	1 Parity Error 0 No Error	Unknown			1 All Sent 0 Not All Sent
Sync		1 CRC Error 0 No Error						1

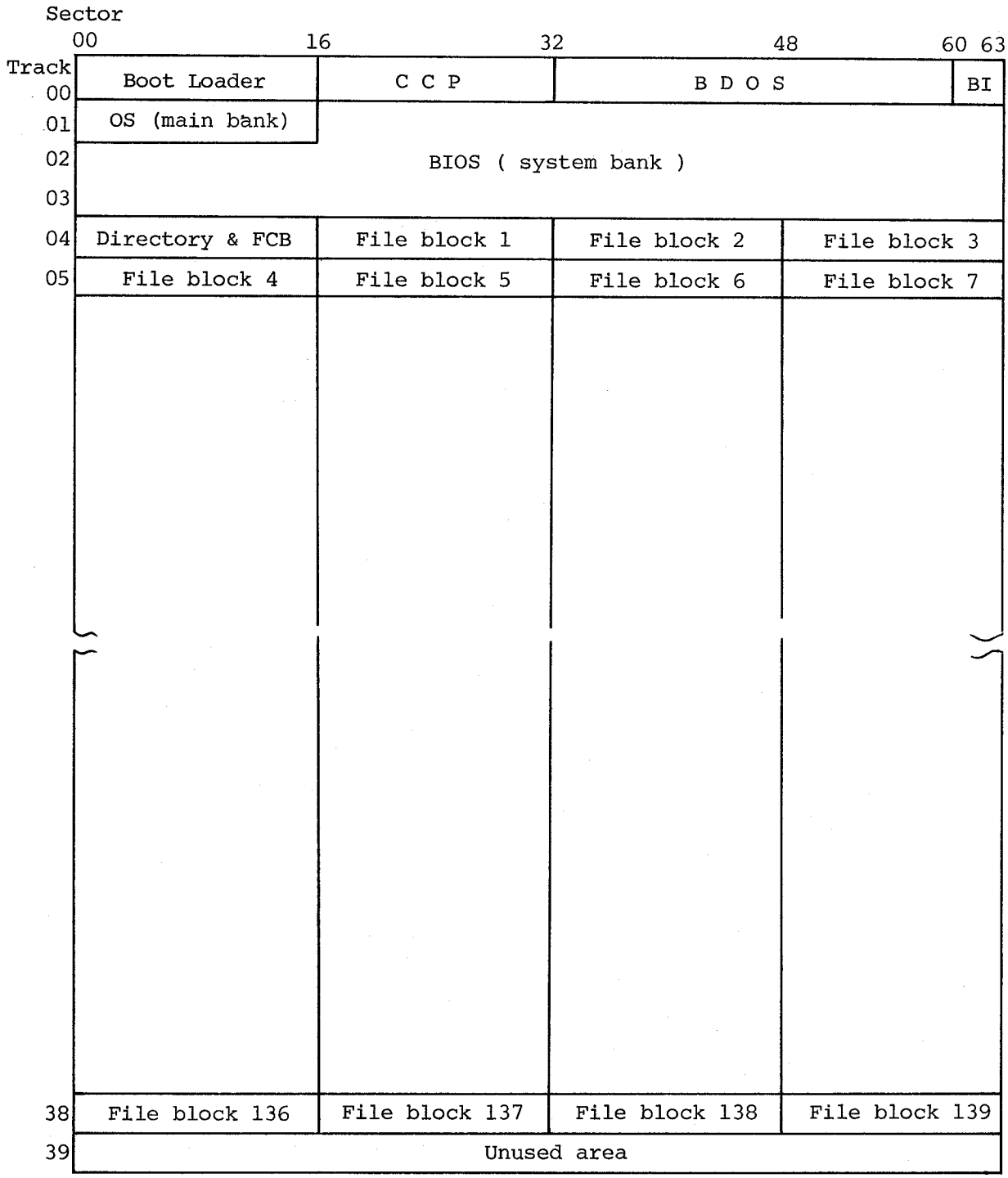
RR2	7	6	5	4	3	2	1	0
WR1 D_2=0	V_7	V_6	V_5	V_4	V_3	V_2	V_1	V_0
WR1 D_2=1 WR2A D D =00,01	V_7	V_6	V_5	1 CH. A 0 CH. B	00 Tx Buffer Empty 01 External/Status Change 10 Receive Character Available 11 Special Rx Condition		V_1	V_0

Note:

Register WR0 can be set directly, but other registers (WR1 to WR7) are set by setting the register number in WR0 and then setting the data. Further, RR0 can be read directly from the I/O port, but PR1 can only be read after its register number has been written into WR0.

§6 Flexible Disks

6.1 Flexible disk layout



Track	Sector	Page #	Memory address when loaded in QX-10	CP/M Module name
00	00,01		0F000H	Boot Loader
	02,03		0F100H	
	04,05		0F200H	
	06,07		0F300H	
	08,09		0F400H	
	10,11		0F500H	
	12,13		0F600H	
	14,15		0F700H	
00	16,17	00	0E000H	CCP
	18,19	01	0E100H	
	20,21	02	0E200H	
	22,23	03	0E300H	
	24,25	04	0E400H	
	26,27	05	0E500H	
	28,29	06	0E600H	
	30,31	07	0E700H	
00	32,33	08	0E800H	BDOS
	34,35	09	0E900H	
	36,37	10	0EA00H	
	38,39	11	0EB00H	
	40,41	12	0EC00H	
	42,43	13	0ED00H	
	44,45	14	0EE00H	
	46,47	15	0EF00H	
	48,49	16	0F000H	
	50,51	17	0F100H	
	52,53	18	0F200H	
	54,55	19	0F300H	
	56,57	20	0F400H	
	58,59	21	0F500H	

00	60,61	22	0F600H	BIOS (main bank)
	62,63	23	0F700H	
01	00,01	24	0F800H	
	02,03	25	0F900H	
	04,05	26	0FA00H	
	06,07	27	0FB00H	
	08,09	28	0FC00H	
	10,11	29	0FD00H	
	12,13	30	0FE00H	
	14,15	31	0FF00H	
01	16,17	32	0H	BIOS (system bank)
	18,19	33	100H	
	20,21	34	200H	
	22,23	35	300H	
	24,25	36	400H	
	26,27	37	500H	
	28,29	38	600H	
	30,31	39	700H	
	32,33	40	800H	
	34,35	41	900H	
	36,37	42	0A00H	
	38,39	43	0B00H	
	40,41	44	0C00H	
	42,43	45	0D00H	
	44,45	46	0E00H	
	46,47	47	0F00H	
	48,49	48	1000H	
	50,51	49	1100H	
	52,53	50	1200H	
	54,55	51	1300H	
	56,57	52	1400H	

01	58,59	53	1500H	BIOS (system bank)
	60,61	54	1600H	
	62,63	55	1700H	
02	00,01	56	1800H	
	02,03	57	1900H	
	04,05	58	1A00H	
	06,07	59	1B00H	
	08,09	60	1C00H	
	10,11	61	1D00H	
	12,13	62	1E00H	
	14,15	63	1F00H	
	16,17	64	2000H	
	18,19	65	2100H	
	20,21	66	2200H	
	22,23	67	2300H	
	24,25	68	2400H	
	26,27	69	2500H	
	28,29	70	2600H	
	30,31	71	2700H	
	32,33	72	2800H	
	34,35	73	2900H	
	36,37	74	2A00H	
	38,39	75	2B00H	
	40,41	76	2C00H	
	42,43	77	2D00H	
	44,45	78	2E00H	
	46,47	79	2F00H	
	48,49	80	3000H	
	50,51	81	3100H	
	52,53	82	3200H	
	54,55	83	3300H	
	56,57	84	3400H	

02	58,59	85	3500H	BIOS (system bank)
	60,61	86	3600H	
	62,63	87	3700H	
03	00,01	88	3800H	
	02,03	89	3900H	
	04,05	90	3A00H	
	06,07	91	3B00H	
	08,09	92	3C00H	
	10,11	93	3D00H	
	12,13	94	3E00H	
	14,15	95	3F00H	
	16,17	96	4000H	
	18,19	97	4100H	
	20,21	98	4200H	
	22,23	99	4300H	
	24,25	100	4400H	
	26,27	101	4500H	
	28,29	102	4600H	
	30,31	103	4700H	
	32,33	104	4800H	
	34,35	105	4900H	
	36,37	106	4A00H	
	38,39	107	4B00H	
	40,41	108	4C00H	
	42,43	109	4D00H	
	44,45	110	4E00H	
	46,47	111	4F00H	
	48,49	112	5000H	
	50,51	113	5100H	
	52,53	114	5200H	
	54,55	115	5300H	
	56,57	116	5400H	

03	58,59 60,61 62,63	117 118 119	5500H 5600H 5700H	
04	00 - 15			(Directory & FCB)
04 ⎩ 38	16 ⎩ 63			(Data track)

6.2 File directory structure

The file directory of the QX-10 MultiFonts CP/M disk is included in sectors 0 through 15 of track 4. This directory contains the names and locations of all files on the disk and their file attributes (R/O, system, etc.).

Disk locations from sector 16 of track 4 through sector 63 of track 38 are used as user data area. This area is divided into 139 2K-byte blocks, and files are managed based on the numbers of these blocks.

Up to 64 files can be cataloged in the file directory, which is divided into 64 32-byte blocks. Each directory block manages up to 32K bytes of a file. When the size of a file exceeds 32K bytes, additional directory blocks must be used together with the cataloged file name. When the entire user area of a disk is used for a single file, 9 directory blocks are required to manage that file.

The structure of each directory block is shown below.

us	file	type	ex	xxx	rc	file block
00	01 ..	09..	12	13	14	15 16.. 31

us: Indicates whether or not the directory block is valid.

0: Valid

0E5H: Invalid

file: File name

type: File type

ex: Indicates the block size in 16K-byte (128 sector) units. A number from 0 to 17.

xxx: Not used.

rc: Indicates the file size in sector units. A number from 0 to 128.

file block: Contains block numbers in which the file is written.

When the MSB of the first byte of the file type is referred to as t1 and MSB of the second byte as t2,

t1 = 0 R/W file

= 1 R/O file

t2 = 0 DIR file

= 1 System file

Examples:

```
00 50 49 50 20 20 20 20 20 43 4F 4D 00 00 00 3A
36 37 38 39 00 00 00 00 00 00 00 00 00 00 00 00
```

The file name is PIP.COM and its size is 58 sectors.

```
00 2E 4B 41 4E 4A 49 20 20 D3 59 53 01 00 00 80
1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29
                    .
                    .
                    .
00 2E 4B 41 4E 4A 49 20 20 D3 59 53 03 00 00 40
2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 00 00 00 00
```

The file name is .KANJI.SYS, and the file has the R/O attribute and a size of 448 sectors (128×3+64).

```
E5 54 45 53 54 20 20 20 20 41 53 4D 00 00 00 80
01 02 03 04 05 06 07 08 00 00 00 00 00 00 00 00
```

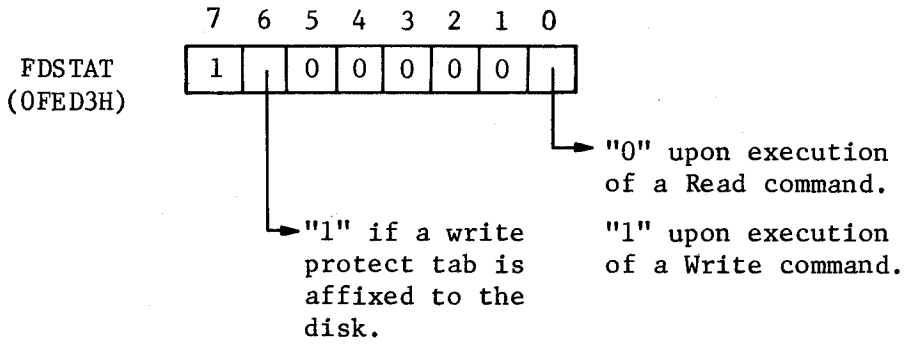
The indicated file (TEST.ASM, which has a length of 128 sectors) has been erased.

```
00 45 50 53 4F 4E 20 20 20 44 41 54 01 00 00 80
01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10
                    .
                    .
                    .
00 45 50 53 4F 4E 20 20 20 44 41 54 11 00 00 30
81 82 83 84 85 86 87 88 89 8A 8B 00 00 00 00 00
```

The file name is EPSON.DAT and the size of the file is 2224 sectors (128×17+48).

6.3 Disk error checks

If an attempt is made to access a flexible disk drive when that drive is not ready (e.g., when the drive door is open or a specified expansion drive is not connected), BIOS issues the "FDD TIME OUT" message. If the BREAK key is pressed at this time, control returns to CCP; if any other key is pressed, the access attempt is continued. To avoid this kind of situation, set 0FFH in BASIC (address 0FED4H) in the BIOS common data area. This causes the drive status to be stored in FDSTAT (address 0FED3H), after which control returns to the program attempting access.



Error processing can then be performed within the user program after checking the contents of FDSTAT.

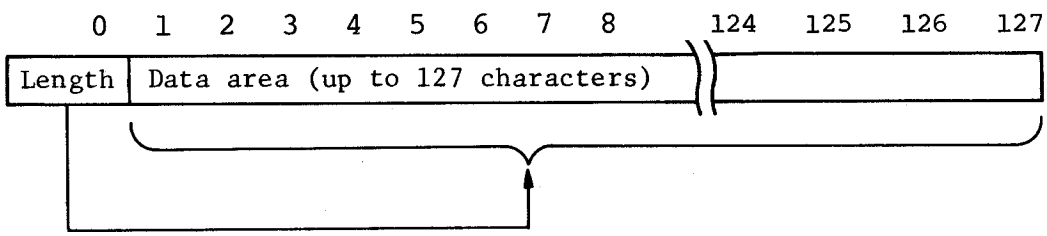
§7 Set-up for Auto-Start

Auto-start is the automatic loading of a specified command line for immediate execution into the CCP command buffer when a cold start is made. The contents of this command line are normally specified using the AUTOST transient command. However, it is also possible to specify the contents of the command line for auto-start execution after a warm or cold boot without using the AUTOST command. The auto-start command line can also be changed under certain program conditions. In such cases, the auto-start specification is changed using a program such as that shown below. The same procedure or the AUTOST command can be used again if it is necessary to restore the former contents of the auto-start command line.

```
;*************************** CONSTANT ***************************
;
BOOT      EQU     0             ;REBOOT TO SYSTEM
WBOOT     EQU     1             ;ADDRESS OF WARM BOOT
BDOS      EQU     5             ;BDOS ENTRY POINT
;
BREAK     EQU     03H
LF        EQU     0AH           ;LINEFEED
CR        EQU     0DH           ;CARRIAGE RETURN
;
CI        EQU     6             ;CONSOLE INPUT
CO        EQU     9             ;CONSOLE OUTPUT
;
SELDSK    EQU     24            ;SELECT DISK
SELTRK    EQU     27            ;SELECT TRACK
SELSEC    EQU     30            ;SELECT SECTOR
SETDMA    EQU     33            ;SET DMA ADDRESS
WRITF     EQU     39            ;WRITE DISK FUNCTION
;
SECTOR:   DB      0EH
DMAA:     DS      2             ;DMA ADDRESS
;
;
WRTDSK:   LD      B,2           ;WRITE DATA TO 1 TRACK 0E, 0F SECTOR
          LD      A,0EH
          LD      (SECTOR),A
          LD      HL,DATA
          LD      (DMAA),HL
WRT10:    PUSH    BC
          XOR     A
          CALL    SEL           ;SELECT DISK
          LD      BC,1
          CALL    TRK           ;SELECT TRACK
          LD      A,(SECTOR)
          CALL    SEC           ;SELECT SECTOR
          LD      BC,(DMAA)
          CALL    DMA           ;SET DMA ADDRESS
          CALL    WRITE         ;NON-BLOCKING WRITE TO DISK
          CP      0
          POP     BC
          JR      NZ,WRTERR     ;IF ERROR
          LD      HL,SECTOR
          INC     (HL)
```

```
            LD      HL,(DMAA)
            LD      DE,80H
            ADD     HL,DE
            LD      (DMAA),HL
            DJNZ    WRT10
            .
            .
            .
WRTERR:     .                       ;ERROR OCCURRED
            .
            .
SEL:        LD      C,A             ;SELECT DISK
            LD      DE,SELDSK
            JR      EXBIOS
TRK:        LD      DE,SELTRK       ;SELECT TRACK
            JR      EXBIOS
SEC:        LD      C,A
            LD      DE,SELSEC       ;SELECT SECTOR
            JR      EXBIOS
DMA:        LD      DE,SETDMA       ;SET DMA ADDRESS
            JR      EXBIOS
WRITE:      LD      C,1             ;NON-BLOCKING WRITE TO DISK
            LD      DE,WRITF
            JR      EXBIOS
;
EXBIOS:     LD      IX,(WBOOT)      ;ENTRY BIOS
            ADD     IX,DE
            JP      (IX)
            .
            .
DATA:       DB      DATAEND-$-1-1;DATA LENGTH < 256
            DB      'SAMPLE PARAM=1,2,3',0
DATAEND     EQU     $
```

Here, data to be written into the command line area must be in the following format.



The number of characters in the data area is indicated by "Length" and the data set in "Data area" is the command which is to be input following the entry prompt (A>). All letters specified in the data area must be upper-case letters, and 0H must be set at the end of the string as an end mark. This end mark is not included in the character count; therefore, the maximum number of characters which can be specified is 127 and the total length of the data area (including the end mark) is 128 bytes.

§8 UL/DL Programs and Conversion Utilities

8.1 UL and DL programs

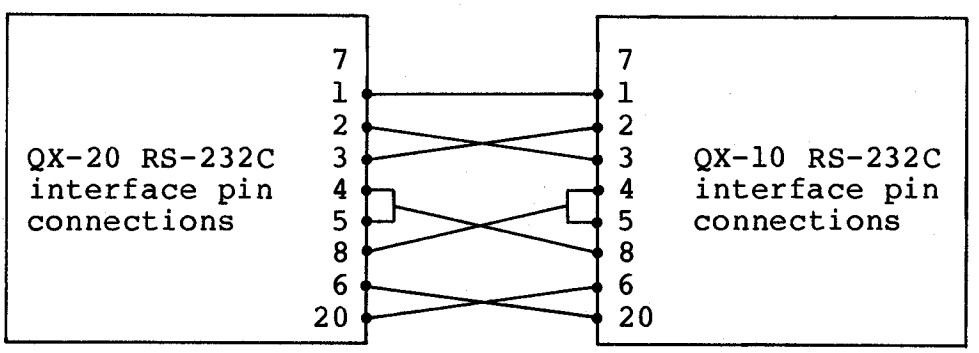
Programs developed on the QX-20 can be converted for execution on the QX-10. (This applies to any program files which operate under CP/M.) This is done using EPSON's UL (UpLoad) and DL (DownLoad) utility programs.

UL and DL programs for the QX-20 (DL, Vers. 4.0 and UL, Vers. 4.0) are distributed on double density, double sided, 96TPI flexible diskettes, and those for the QX-10 (DL, Vers. 0.2 and UL, Vers. 0.1) are distributed on double density, double sided, 48 TPI flexible diskettes.

Notes:

1. Note that the CP/M operating system is not included on the file transfer program diskettes; therefore, be sure to boot up the QX-20 and QX-10 using your CP/M system diskettes before attempting to execute the UL or DL programs.

2. The QX-20 should be connected to the QX-10 via their RS-232C interfaces as shown in the figure below.



Operating procedures for the DL and UL programs are as follows.

(1) Connect the QX-20 to the QX-10 as described above.

(2) Boot up the CP/M operating system on the DL side (the QX-10).

(3) Boot up the CP/M operating system on the UL side (the QX-20).

(4) Insert the diskette containing the DL file transfer program in drive A of the QX-10.

(5) Insert the diskette containing the UL file transfer program in drive A of the QX-20.

QX

(6) Execute the DL program from the keyboard of the QX-10.

The syntax for execution of the DL program is as follows.

>DL[_[<drive>:][<new filename>]] ↵

If the <drive> specification is omitted, the file is downloaded to the active drive. If the <newfilename> specification is omitted, the file is downloaded under the name assigned to the file on the UL side.

Examples:	>DL ↵	File downloaded to active drive under name assigned on UL side.
	>DL_B:↵	File downloaded to drive B under name assigned on the UL side.
	>DL_*.ASM ↵	File downloaded to active drive with file type changed to ".ASM".

Notes :

1. Execution of the DL program can be aborted by pressing the BREAK key.

2. If the file name under which the downloaded file is to be saved is the same as that of another file on the disk, the message "Overwrite (Y/N)?" is displayed; press the "Y" key if the current file is to be overwritten.)

(7) Execute the UL program from the keyboard of the QX-20.

The syntax for execution of the UL program is as follows.

>UL[_[<drive>:][<filename>]] ↵

If the <drive> specification is omitted, the file is uploaded from the active drive.

Examples:	>UL_A.MAC ↵	File A.MAC is uploaded.
	>DL_?.COM ↵	All .COM files are uploaded.
	>DL_B:*.* ↵	All files on the diskette in drive B: are uploaded.

Note:

Execution of the DL program can be aborted by pressing the BREAK key.

8.2 BPTRNQX conversion utility

When a QX-20 MFBASIC program file is transferred to the QX-10 using the UL/DL commands or the SAVE/LOAD commands of BASIC, its syntax and parameters can be converted for use on the QX-10 by means of the BPTRNQX conversion utility. Program statements are converted according to the following rules.

1) CIRCLE (x,y),r,a,s,e,f ⟶ CIRCLE (x,y),r,c,s',e',a

Here r, c=0 when f is PRESET; otherwise, c is omitted. The value used for s' is s x 6.283185, and that used for e' is e×6.283185.

2) PUT@ (xl,yl)-(x2,y2),A,f --> PUT@ (xl,yl),A,f

3) CSRLIN --> (CSRLIN - 1)

Procedures for using the BPTRNQX program are described below.

a) After transferring the MFBASIC program file as described in 8.1 above, start up EPSON MFBASIC on the QX-10, then LOAD and RUN program file "BPTRNQX.BAS".

b) The following message is then displayed.

```
You can translate program file of QX-20
into one of QX-10 by this program
And you ?
          YES--Y    NO--N
```

Pressing the N key in response to this message terminates the program without doing anything; pressing the Y key causes the following message to be displayed to prompt for insertion of a flexible disk.

```
Please set floppy
          OK--Y
```

Insert a flexible disk in the appropriate drive and press the Y key; the message shown below is then displayed.

```
Floppy set completed!
```

c) The program then prompts for entry of the name of the file to be converted as shown below. Enter the complete file name, including the drive name. (The file must have been saved in ASCII format; otherwise, the error message "***** ERROR *****(NOT ASCII SAVE)" is displayed and conversion program execution ends.)

Example:

```
Please input sequential file name of QX-20
(ex. B:QX20.BAS)
```

d) After a proper name has been entered, the program prompts for entry of the name to be assigned to the QX-10 program file generated as follows.

```
Please input one of QX-10 (ex. A:QX10.BAS)
```

If the same name is entered as in response to the prompt shown in (C) above, the prompt is displayed again.

e) After a proper name has been entered, the following message is displayed to request confirmation that conversion is to be performed; press the Y key to make the conversion or the N key to return to the point described in (c) above.

```
OK--Y    NO--N
```

f) The following message is displayed while the file is being

converted.

```
Running
QX-20 B:QX20.BAS -----> QX-10 A:QX10.BAS
```

Note:

If the length of a converted line exceeds 255 characters, the following message is displayed and excess characters are discarded.

```
***** ERROR ***** (LINE xx IS TOO LONG)
```

g) After conversion has been completed, the following message is displayed to ask whether there are any other program files to be converted.

```
Translate was completed
Continue ?
     YES--Y     NO--N
```

Press the Y key if another program is to be converted; otherwise, press the N key.

§9 CP/M program library installation

Many application programs are available for operation under CP/M, and you will probably wish to install many of these on the QX-10 for execution. However, there is no fixed installation procedure which will work in all cases. Some considerations are described below.

One of the most commonly utilized functions is that of cursor movement. The internal codes generated when the cursor control keys (↑ , ↓ , → , or ←) are pressed are stored in XCHUP (address 0FE58H, default 01EH), XCHDOWN (address 0FE59H, default 01FH), XCHRIGT (address 0FE5AH, default 01CH), and XCHLEFT (address 0FE5BH, default 01DH).

Further, the current horizontal position of the cursor (1 to 80) is stored in HPOS (address 0FE75H) of the BIOS common data area and the current vertical position (1 to 25 in the non-MFBASIC normal mode and 1 to 20 in the non-MFBASIC MF mode or MFBASIC mode) is stored in VPOS (address 0FE76H).

See the list of escape sequences in the Appendix for the console escape sequences.

Any type of printer can be connected; set the print size to match that of the printer used.

Screen sizes which can be specified are 80 columns x 25 lines for the non-MFBASIC normal mode, and 80 columns x 20 lines for the non-MFBASIC MF mode and MFBASIC mode. The user must switch between these three modes by changing the contents of MBFLG (address 0FE50H) and MFLG (address 0FE82H) in the BIOS common data area.

Set the data according to the program or source program to be installed. For details, see the individual installation manual.

	MFLG=0	MFLG=1
MBFLG=0	Non-MFBASIC Normal mode	Non-MFBASIC MF mode
MBFLG=1	MFBASIC mode	