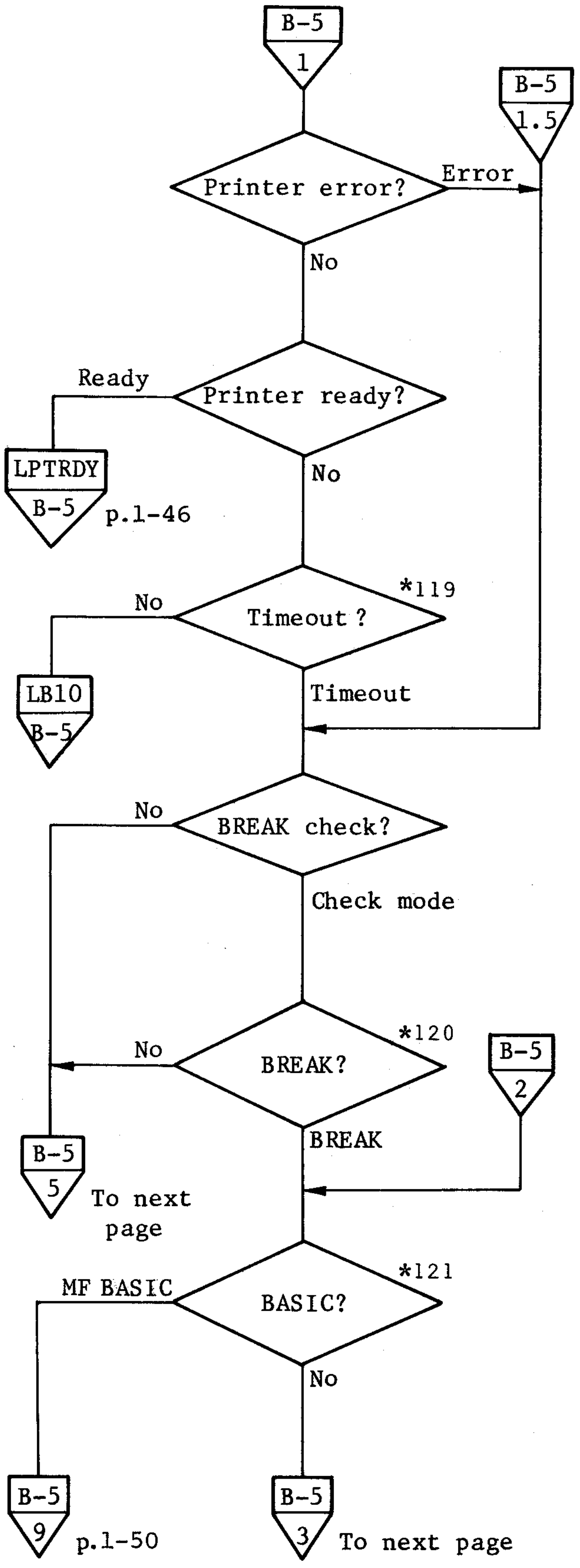
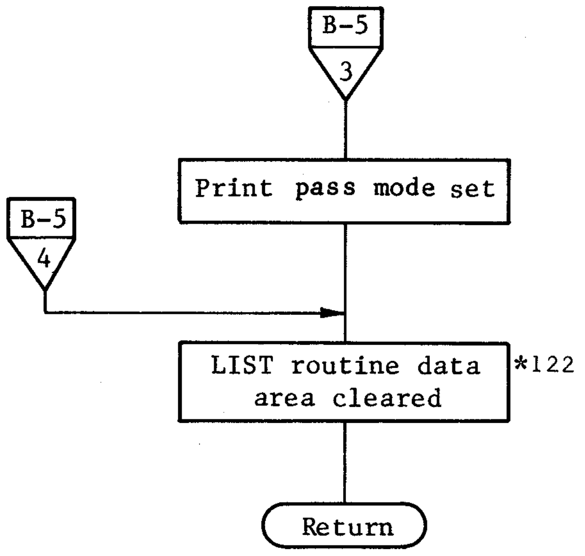


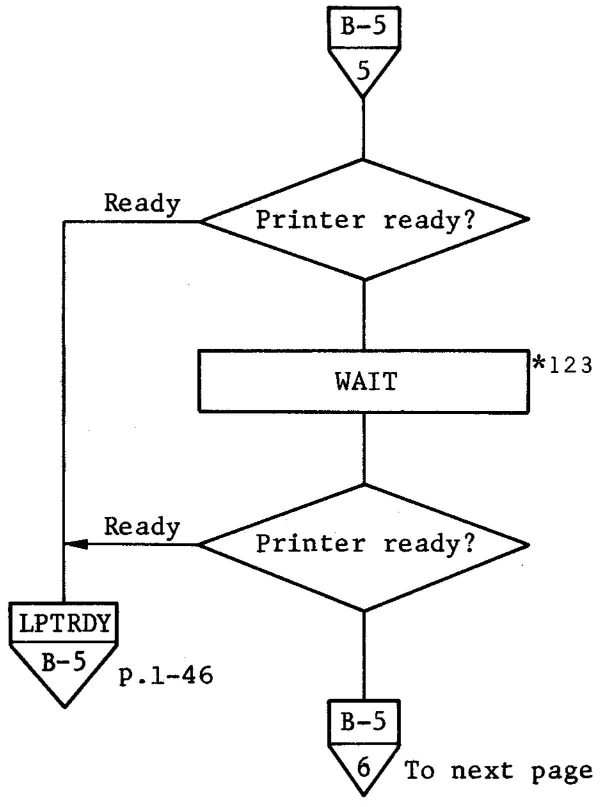
*119 Timeout check repeated 8192 times (LB10) from the preceding page.

*120 Check made to determine whether the BREAK key has been pressed.

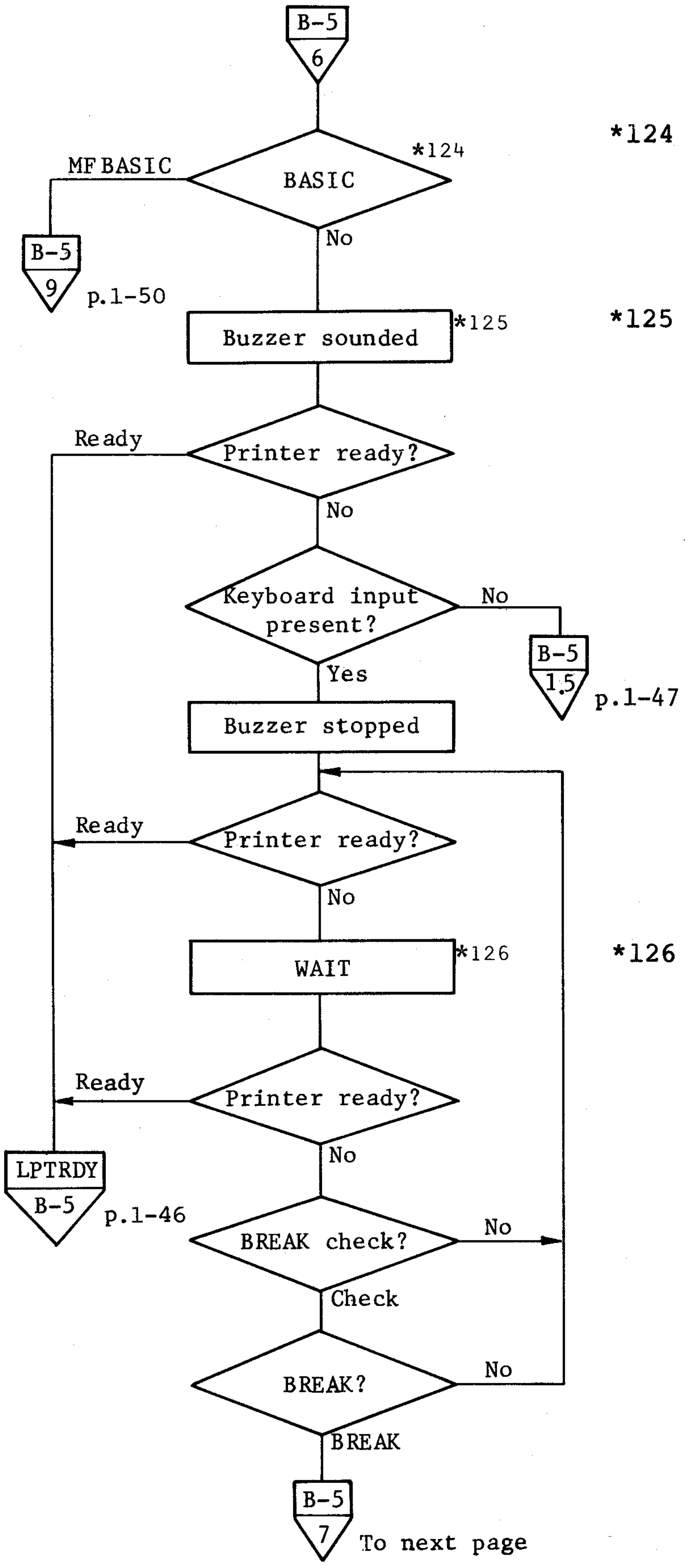
*121 Check made to determine whether MFBASIC is running.



*122 Control flags, etc., used by the LIST routine cleared.



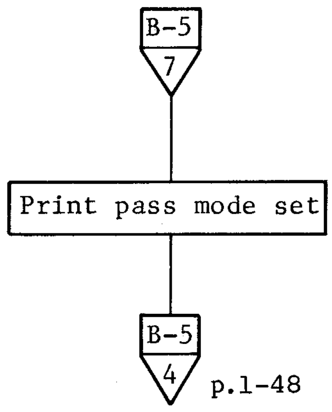
*123 Subroutine TIMER called.

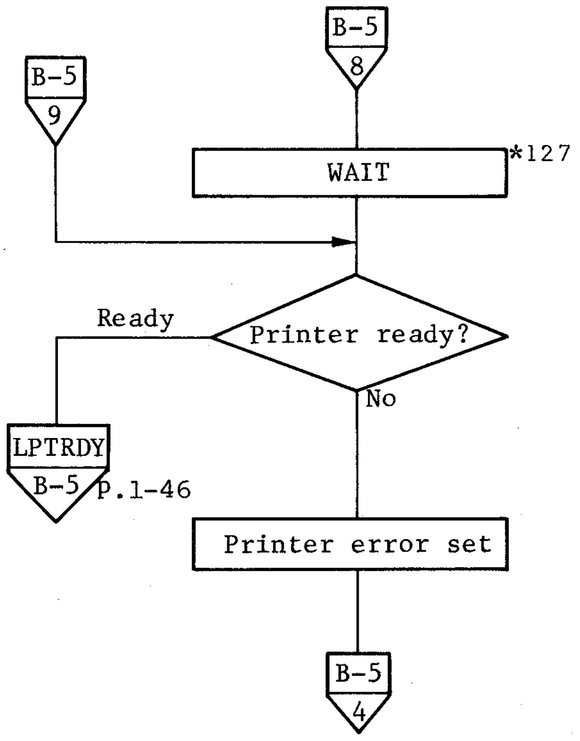


*124 Check made to determine whether MFBASIC is running.

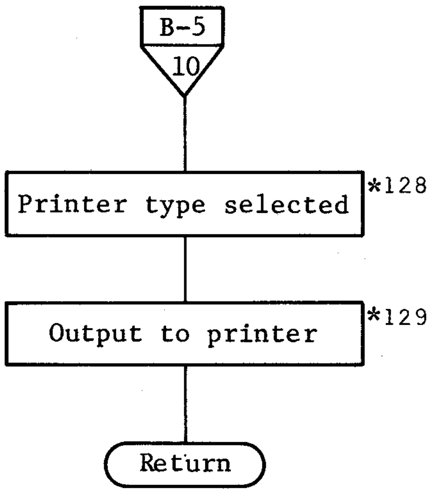
*125 BUZZON, TIMER, and BUZZOFF subroutines called.

*126 TIMER subroutine called.





*127 TIMER subroutine called.



*128 Output procedure set to that specified by the CONFIG transient command.

*129 Subroutine LPOUT called.

§5 PUNCH (Punch Output), READER (Reader Input), LISTST (List Status)

5.1 PUNCH (Address 0F612H)

5.1.1 General

This BIOS routine outputs the character code set in register C to the device currently assigned to device name "PUN:". The current "PUN:" device is determined by the setting of the I/O byte. Devices which can be assigned to "PUN:" include the printer, console, and RS-232C interface; the device assigned to "PUN:" upon completion of a cold start is the RS-232C interface.

Output by the PUNCH routine is directed to the printer, console, or RS-232C interface depending on the setting of the I/O byte; see the descriptions of the LIST, CONOUT, and RSOUT routines for details. For any other devices, this routine returns without doing anything.

All registers are changed by execution of this routine.

5.1.2 Call procedure

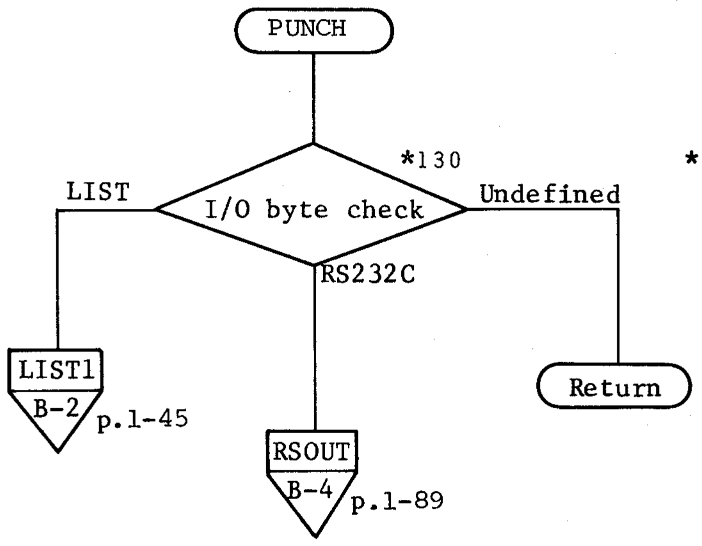
Entry parameters: Register C=Output data

Example:
```
PUNCH     EQU     0F612H      ;PUNCH ENTRY POINT.
          .
          .
          LD      HL,OUTDATA
PLOOP:    LD      A,(HL)
          CP      0
          JR      Z,OUTEND
          LD      C,A
          PUSH    HL
          CALL    PUNCH
          POP     HL
          INC     HL
          JR      PLOOP
;
OUTEND:   .
          .
          .
OUTDATA:  DB      'ABC '
```

This example outputs "ABC " to the RS-232C interface.

Return information: See LIST, CONOUT, RSOUT.

5.1.3 General flowchart



*130 Since "PUN:" is assumed to be undefined when PTP is specified as "PUN:", the routine returns without doing anything.

5.2 READER (Address 0F615H)

5.2.1 General

This routine inputs one character from the device currently assigned to "RDR:", sets that character code in register A, and returns. The device currently assigned to "RDR:" is determined by the setting of the I/O byte. Devices which can be assigned to "RDR:" include the keyboard and RS-232C interface. The default device upon completion of a cold start is the RS-232C interface. This routine returns without doing anything for devices other than the keyboard and the RS-232C interface.

All registers are changed upon execution of this routine. For details, see the descriptions of the CONIN and RSIN routines.

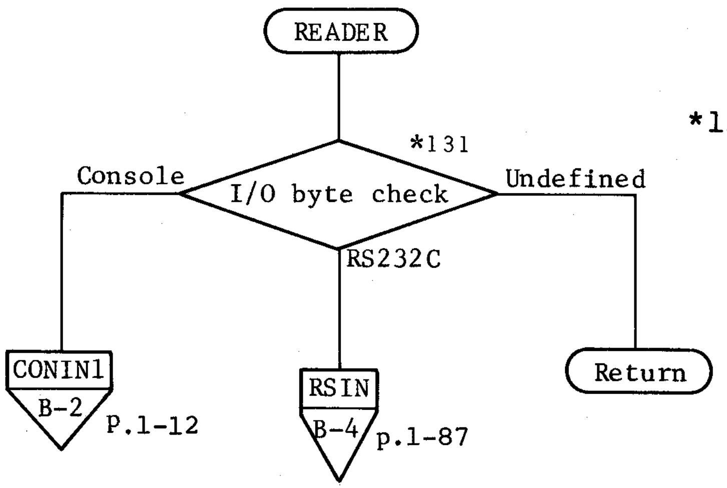
5.2.2 Call procedures

Entry parameters: None

Example:
```
READER  EQU    0F615H
        .
        .
        CALL   READER
        .
        .
```

Return information: See the descriptions of the CONIN and RSIN routines.

5.2.3 General flowchart



*131 Since "RDR:" is assumed to be undefined for devices other than the keyboard or RS-232C interface, the routine returns without doing anything.

5.3 LISTST (Address 0F62DH)

5.3.1 General

This routine checks the status of the device currently assigned to "LST:" and sets the result in register A. The device assigned to "LST:" is determined by the setting of the I/O byte.

Devices which can be assigned to "LST:" include the printer, console, and RS-232C interface. The default device upon completion of a cold start is the printer.

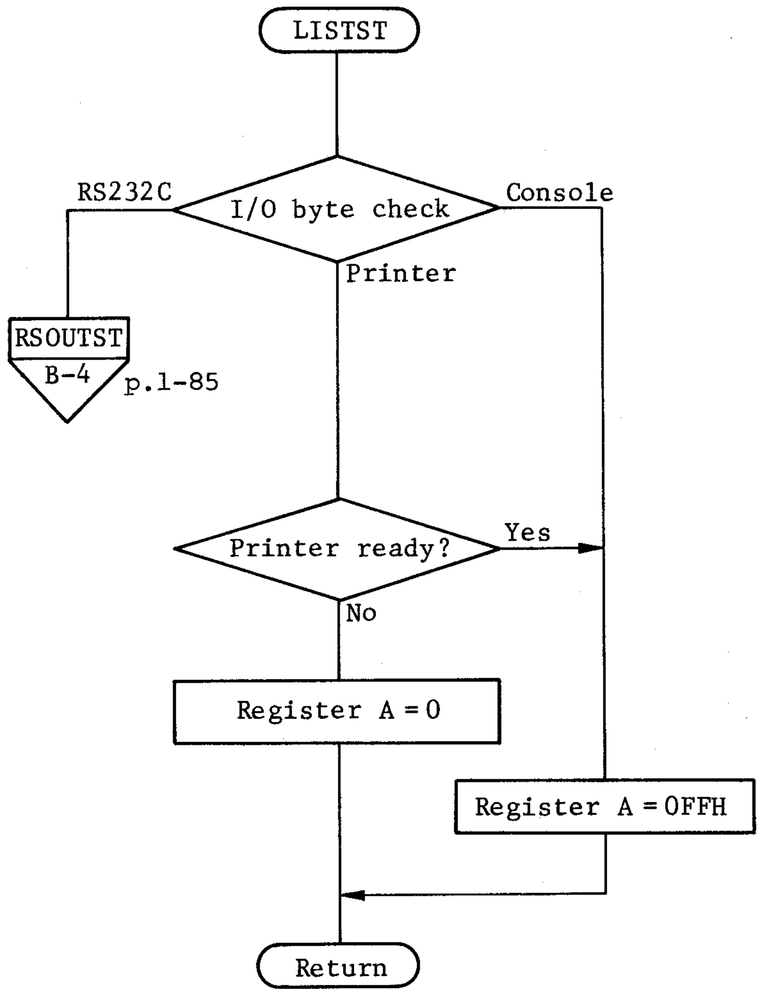
5.3.2 Call procedure

Entry parameters: None

Example:
```
LISTST   EQU     0F62DH
         .
         .
         CALL    LISTST
         CP      0
         JP      Z,BUSY
         .
         .
```

Return parameters:	Register A=0FFH - Usable
	Register A=0 - Unusable

5.3.3 General flowchart



§6 HOME, SELDSK, SETTRK, SETSEC, SETDMA, READ, WRITE, SECTRAN

This section describes BIOS routines used for flexible disk drive control.

6.1 HOME (Address: 0F618H)

6.1.1 Function

This routine zeroes the flexible disk seek track setting (SEKTRK) in the BIOS common data area, but does not actually move the heads to track 0. The contents of all registers are changed when this routine is called.

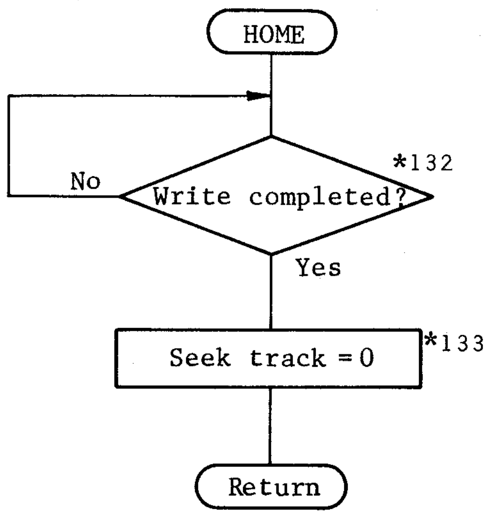
6.1.2 Call procedure

Entry parameters: None

Example:
```
HOME      EQU     0F618H
          .
          .
          CALL    HOME
          .
          .
```

Return information: None.

6.1.3 General Flowchart



*132 Checks whether any data remains to be written to the disk. This check function can be reset by interrupt.

*133 Sets SEKTRK (address: 0FC41H) to zero in the BIOS common data area.

6.2 SELDSK (Address: 0F61BH)

6.2.1 Function

This routine selects the flexible disk drive which is to be accessed. The drive number must be specified in register C before calling this routine. The contents of all registers are changed when this routine is called. The disk parameter block address is returned to register HL.

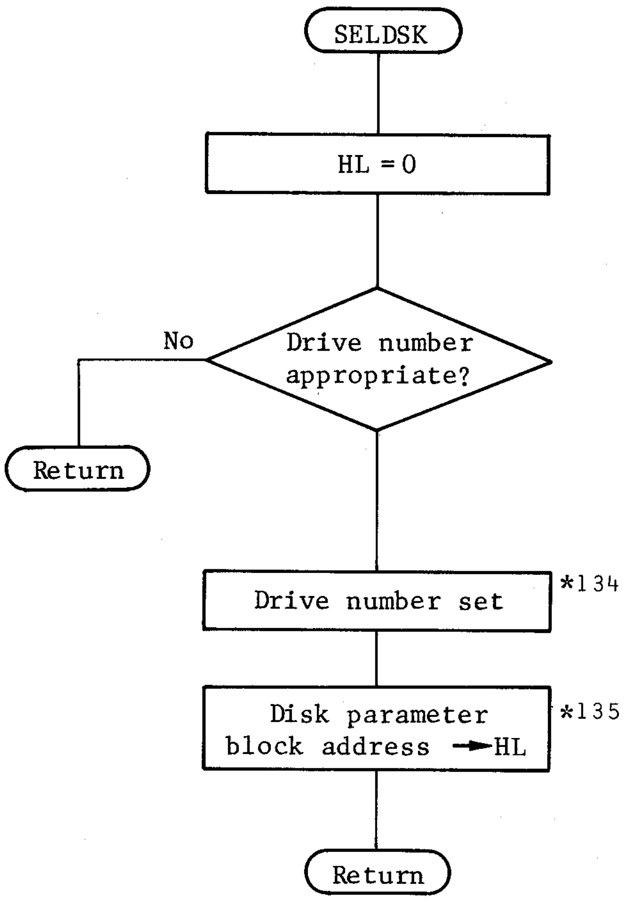
6.2.2 Call procedure

Entry parameters:	Register C = 0	Drive A
	1	B
	2	C
	3	D
	4	E (Disk image RAM)
	5	F

Example:
```
SELDSK  EQU     0F61BH
        .
        .
        LD      C,1             ;SELECT DRIVE B.
        CALL    SELDSK          ;
        .
        .
```

Return information:	HL=0	Entry parameter error
	HL≠0	Normal completion (HL contains the disk parameter block address.)

2.3 General Flowchart



*134 SEKDSK (address: 0FC40H) in the BIOS common data area

*135 For the disk parameter block, see the description of the BIOS common data area.

6.3 SETTRK (Address: 0F61EH)

6.3.1 Function

This routine sets the track number which is to be accessed. The track number must be specified in register BC before calling this routine. Only the contents of register pair HL are not changed when this routine is called.

6.3.2 Call procedure

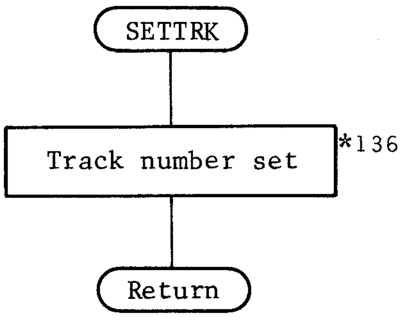
Entry parameters:	Register BC	Track number
		0 to 39 for drives A to D
		0 to 6 for drives E and F

Example:
```
        SETTRK  EQU     0F61EH
                .
                .
                LD      BC,10
                CALL    SETTRK
                .
                .
```

Return information: None.

An error will result if the WRITE or READ routine is executed when a value other than 0 to 39 is specified in register BC.

6.3.3 General flowchart



*136 SEKTRK (address: 0FC41H) in the BIOS common data area

6.4 SETSEC (Address: 0F621H)

6.4.1 Function

This routine specifies the sector to be accessed. The sector number must be specified in register C before calling this routine. Only the contents of register A are changed when this routine is called.

6.4.2 Call procedure

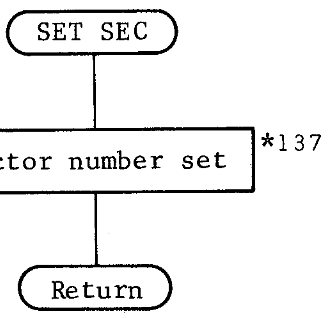
Entry parameters: Register C   Sector number (0 to 63)

Example:
```
SETSEC   EQU    0F621H
         .
         .
         LD     C,18
         CALL   SETSEC
         .
         .
```

Return information: None.

An error results if the WRITE or READ routine is executed when a value other than 0 to 63 is specified in register C.

6.3.3 General flowchart



*137 SEKSEC (address: 0FC43H) in the BIOS common data area

6.5 SETDMA (Address: 0F624H)

6.5.1 Function

This BIOS routine specifies the starting address of the 128-byte data area in which data read or to be written is stored. The address must be specified in register BC before calling this routine. Only the contents of register pair HL are changed when this routine is called.

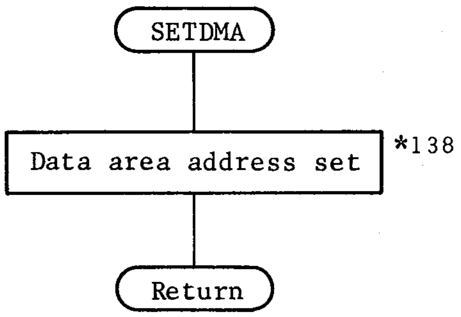
6.5.2 Call procedure

Entry parameters: BC=128-byte data area address

Example:
```
SETDMA   EQU     3*11            ;SET DMA ENTRY
BIOSADD  EQU     1               ;WARM BOOT ADDRESS
         .
         .
EXBIOS:  LD      HL,(BIOSADD);
         ADD     HL,DE           ;MAKE JUMP ADDRESS.
         JP      (HL)            ;EXECUTE BIOS
         .
         .
         LD      BC,BUF          ;SET DATA ADDRESS
         LD      DE,SETDMA       ;
         CALL    EXBIOS          ;EXECUTE SETDMA.
         .
         .
BUF:     DS      128             ;READ/WRITE BUFFER.
         .
         .
```

Return information: None

6.5.3 General Flowchart



* 138 DMAADR (address: 0FC54H) in the BIOS common data area

6.6 READ (Address: 0F627H)

6.6.1 Function

This routine reads a 128-byte data block from the disk as specified by SELDSK, SETTRK, SETSEC and SETDMA. The contents of all registers are changed when this routine is called.

6.6.2 Call Sequence

Entry parameters: The flexible disk drive, track number, sector number and data area address must be specified by the SELDSK, SETTRK, SETSEC and SETDMA routines before calling this routine.

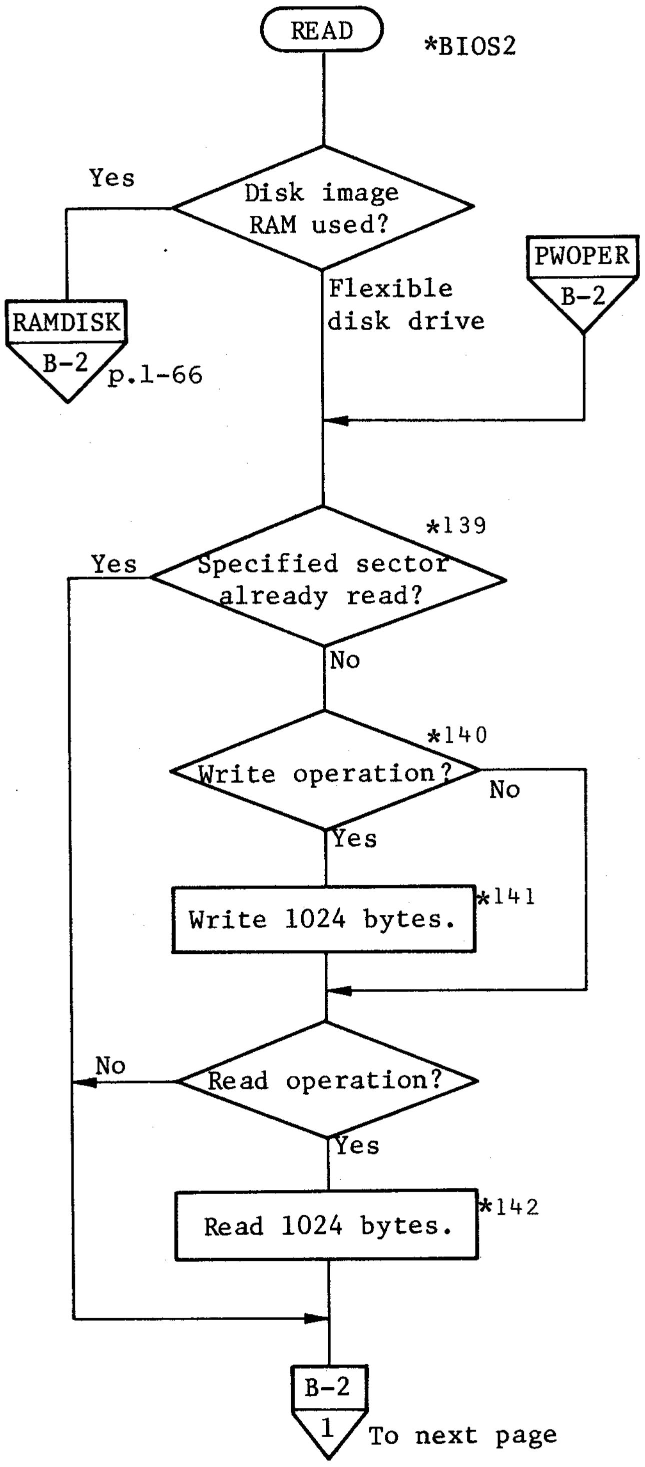
Example: The following sequence reads a 128-byte data block from track 20, sector 0 on the disk in drive B into the buffer.

```
BIOSADD  EQU    1
SELDSK   EQU    3*8
SETTRK   EQU    3*9
SETSEC   EQU    3*10
SETDMA   EQU    3*11
READ     EQU    3*12
WRITE    EQU    3*13
         .
         .
EXECBIOS:LD     HL,(BIOSADD);MAKE ENTRY POINT
         ADD    HL,DE       ;
         JP     (HL)        ;EXECUTE BIOS ENTRY
         .
         .
         .
         .
         LD     C,1         ;SELECT DRIVE B.
         LD     DE,SELDSK   ;
         CALL   EXECBIOS    ;
;
         LD     BC,20       ;TRAK = 20.
         LD     DE,SETTRK   ;
         CALL   EXECBIOS    ;
;
         LD     C,0         ;SECTOR = 0.
         LD     DE,SETSEC   ;
         CALL   EXECBIOS    ;
;
         LD     BC,DATA     ;SET DATA BUFFER.
         LD     DE,SETDMA   ;
         CALL   EXECBIOS    ;
;
         LD     DE,READ     ;READ.
         CALL   EXECBIOS    ;
;
         AND    A           ;ERROR ?
         JP     NZ,ERROR    ;ERROR OCCURRED.
         .
```

```
DATA:     DS      128          ;READ BUFFER.
```

Return information: Register A=0 Normal completion
                               ≠0 Abnormal completion

6.6.3 General Flowchart

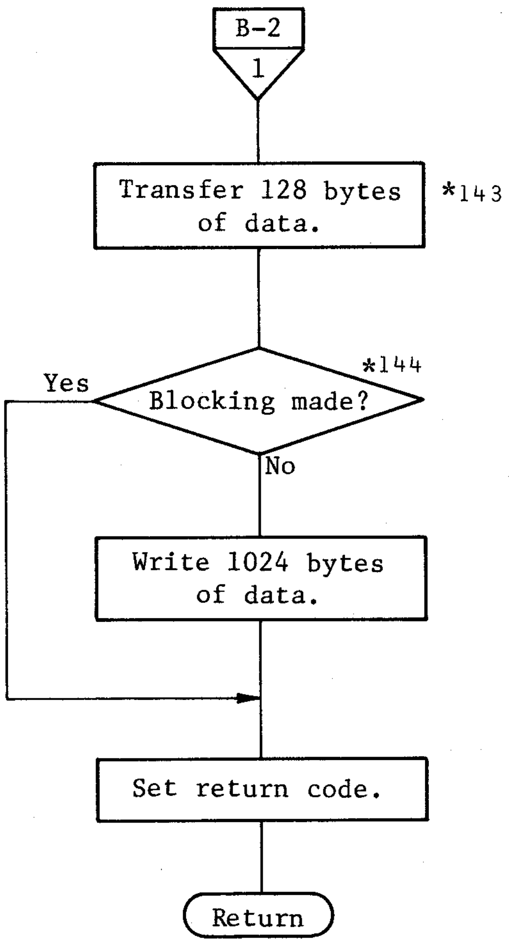


*139 Data is read/written in 1024 byte units to increase read/write speed. Data is stored in the memory block starting at 0DC00H.

*140 Checks whether any data remains to be written to the disk.

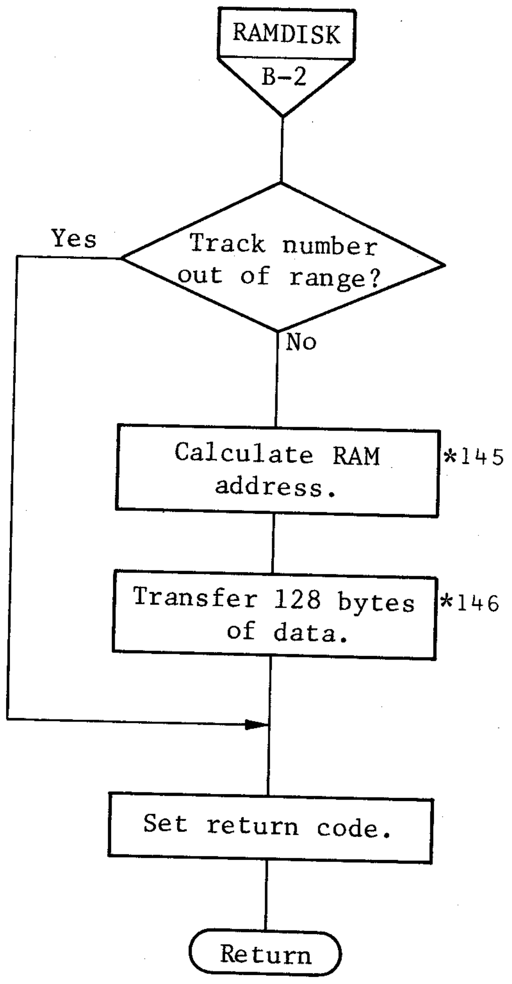
*141 Subroutine WRITEHST

*142 Subroutine READHST



*143 Subroutine LDIRX. 128 bytes are transferred from user data area to system bank during write and from system bank to user area during read.

*144 Checks whether 128 bytes are written without blocking.



*145 Calculates the address of RAM area to/from which data is written/read by the following equation. Track x 8192 + sector x 80

*146 Subroutine LDIRX. Data is transferred from the user area to RAM during a write and from RAM to the user area during a read.

6.7 WRITE (Address: 0F62AH)

6.7.1 Function

This routine writes 128-byte data to the disk based on the parameters specified by SELDSK, SETTRK, SETSEC and SETDMA. The contents of all registers are changed when this routine is called.

6.7.2 Call procedure

Entry parameters:

(1) The flexible disk drive, track number, sector number and data area address must be specified by the SELDSK, SETTRK, SETSEC and SETDMA routines before calling this routine.

(2) Register

C=0	Normal write
C=1	Write without blocking
C=2	Sequential file write

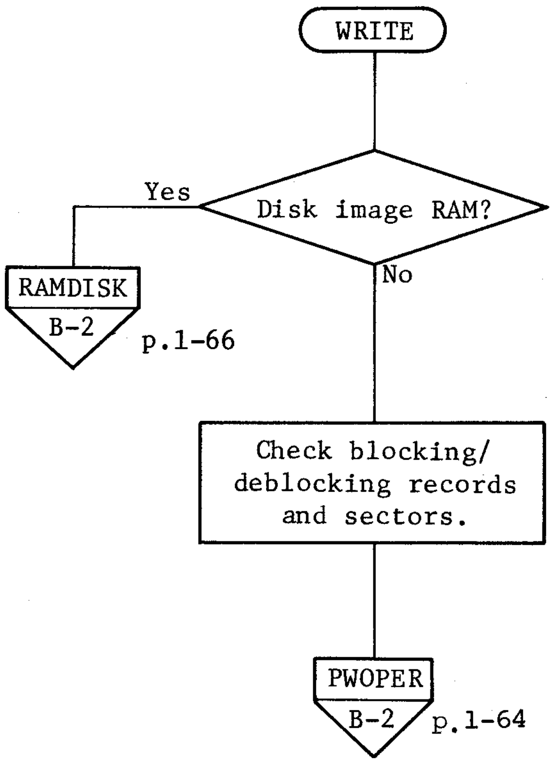
Example: The following sequence writes the same data as that read by the example shown in the preceding section.

```
        .
        .
        LD      C,0             ;NORMAL.
        LD      DE,WRITE
        CALL    EXECBIOS        ;EXEC WRITE.
        AND     A
        JP      NZ,ERROR        ;ERROR OCCURRED.
        .
        .
```

Return information: Register

A=0	Normal completion
A≠0	Abnormal completion

6.7.3 General Flowchart



6.8 SECTRAN (Address: 0F630H)

6.8.1 Function

This routine converts a logical sector number into the corresponding physical sector number.
With the QX-10, the logical sector number equals the physical sector number and therefore this routine transfers data between registers.
Only the contents of register pair HL are changed when this routine is called.

6.8.2 Call procedure

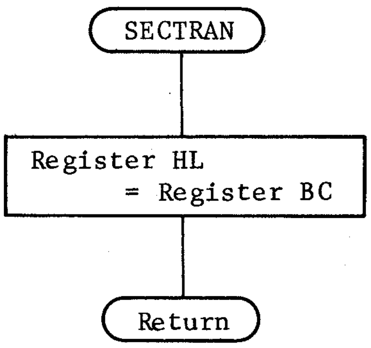
Entry parameters: Register BC = Logical sector number

Example:

```
SECTRAN  EQU    0F630H
         .
         .
         LD     BC,10
         CALL   SECTRAN
         .
         .
```

Return information: Register HL = Physical sector number

6.8.3 General Flowchart



§7 PSET (Point Set), HCOPY (Hard Copy), BEEP

7.1 PSET (Address: 0F633H)

7.1.1 Function

This routine reads 8-bit data at the specified display screen location, performs the specified logical operation on the data read and other data specified, then sets the result at the original location in the specified color (white for all colors other than black when the green monitor is used).
This routine cannot be used with a green monitor when the system is in the non-MFBASIC normal mode.

7.1.2 Call procedure

Entry parameters:

Register B = Data to be used in the logical operation.

Register C = Logical operation

C = 1:	AND
C = 2:	OR
C = 3:	XOR

No operation is performed if any other value is specified.

Register E = Color

E = 0:	Black
E = 1:	Blue
E = 2:	Red
E = 3:	Violet
E = 4:	Green
E = 5:	Light blue
E = 6:	Yellow
E = 7:	White

With the green monitor, numbers other than 0 are equivalent to 7 (white).

Register pair HL = Display address

This is not the same as the display address described in §3 "CONOUT".
One location is assigned to each block of 8 sequential dots.
Therefore, location addresses assigned to the top line of the screen are 0 to 79. (640/8=80).
The maximum number which can be specified for the display location is 31999 (07CFFH); this is the number which indicates the 8-dot block at the bottom right corner of the screen. (80 x 400 = 32000).

Example: The following sequence reverses the relationship between black and white in the character displayed at the upper right corner of the screen (with the color monitor in the non-MFBASIC normal mode).

```
PSET        EQU     0F633H
            .
            .
            LD      B,16
            LD      HL,0FFFFH      ;HL = -1
PSETLOOP:
            PUSH    BC
            LD      B,0FFH
            LD      C,3            ;REPLACE(XOR)
            LD      DE,80
            ADD     HL,DE          ;HL = ADDRESS.
            LD      E,7            ;COLOR = WHITE
            CALL    PSET
            POP     BC
            DJNZ    PSETLOOP
             .
             .
```

Return information:

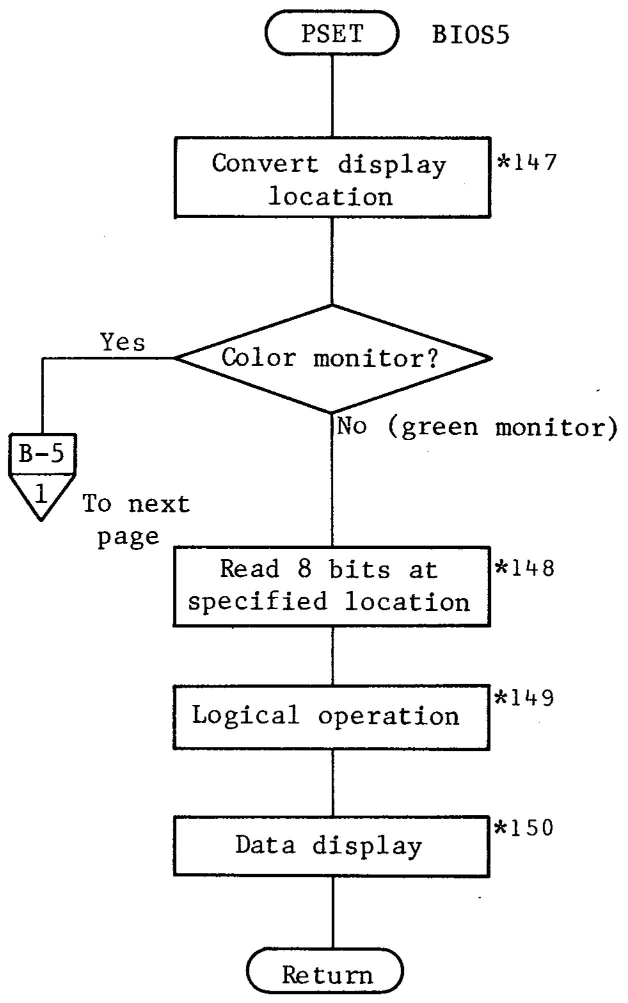
Register C = Result (result of operation with red)

Register D = Result (result of operation with green)

Register E = Result (result of operation with blue)

When the green monitor is used, the result is returned to register C only. The contents of register pair DE are changed, but the contents of register B and register pair HL are not changed.

7.1.3 General Flowchart

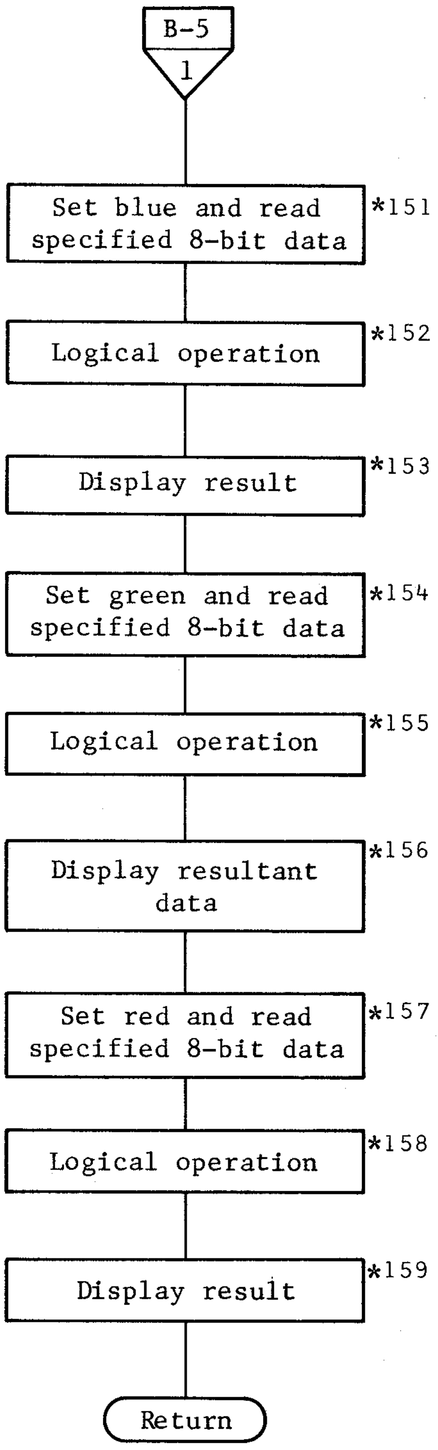


*147 Converts the 8-dot display address specified in the entry parameters into a 16-dot display address.

*148 Subroutine PREAD called.

*149 Subroutine PFUNC called.

*150 Subroutine PWRITE called.



*151 Subroutine PRREAD

*152 Subroutine PFUNC. Logical operation is not performed when blue is not specified.

*153 Subroutine PWRITE called.

*154 Subroutine PRREAD called.

*155 Subroutine PFUNC. Logical operation is not performed when green is not specified.

*156 Subroutine PWRITE

*157 Subroutine PREAD

*158 Subroutine PFUNC

*159 Subroutine PWRITE

7.2 HCOPY (Address: 0F636H)

7.2.1 Function

This routine makes a hard copy of the current screen image on the printer.
When the printer used is the MX-80, this routine is effective only when the green monitor is used in the non-MFBASIC normal mode.
With the color monitor, dots in the background color are not printed. Therefore, the hard copy does not become all black even if the background color is not black.
Combinations of mode, monitor and printer with which this routine can be used are shown below.

Printer \ CRT / Mode	Green CRT I	Green CRT II	Green CRT III	Green CRT IV	Color CRT I	Color CRT II	Color CRT III	Color CRT IV
MX-80	○	×	×	×	×	×	×	×
TYPE 2, MX-82 or MX-100	○	○	○	○	○	○	○	○
TYPE 3, MX-82 or MX-100	○	○	○	○	○	○	○	○
FX-80	○	○	○	○	○	○	○	○
FX-100	○	○	○	○	○	○	○	○
RX-80	○	○	○	○	○	○	○	○

○...Hard copy
×...No operation

I: Non-MFBASIC Normal mode
II: Non-MFBASIC MF mode
III: MFBASIC width 80 mode
IV: MFBASIC width 40 mode

When the printer is offline, the buzzer built into the QX-10 sounds when an attempt is made to print a hard copy.
Press any key to stop the buzzer and put the printer online, or press the BREAK key to stop data transfer.
Printing can be stopped at any time by pressing the BREAK key.

7.2.2 Call procedure

Entry parameters: None

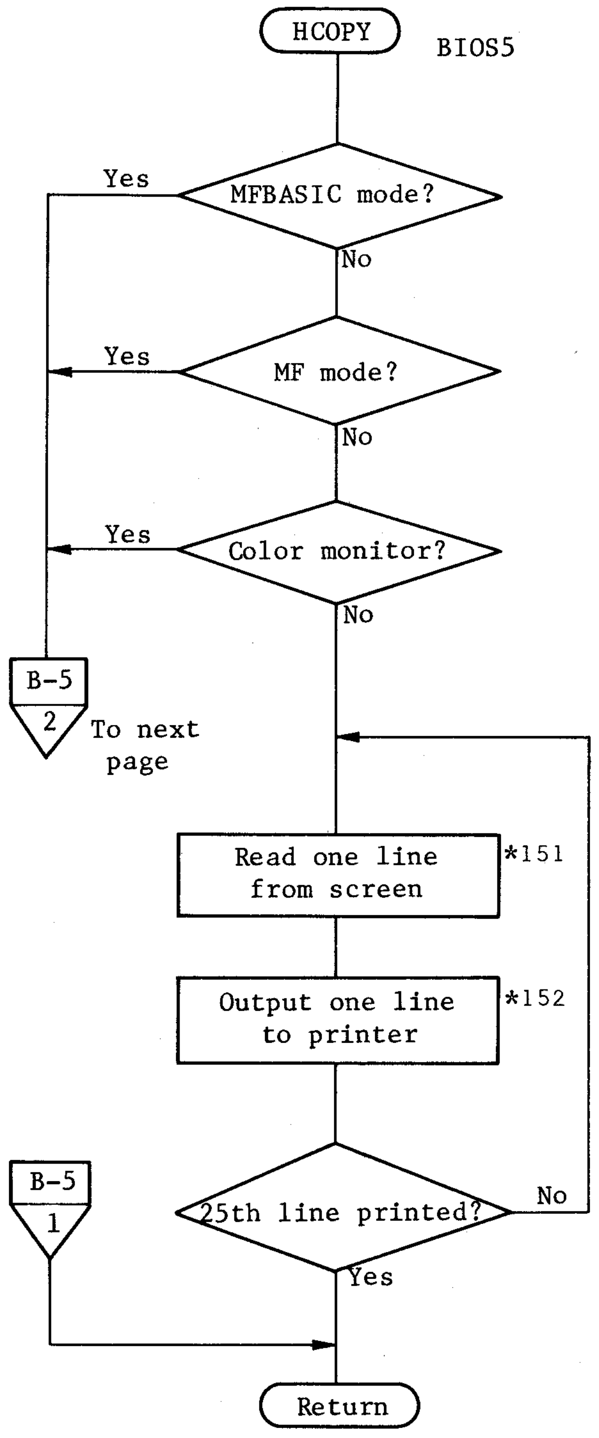
Example:

```
HCOPY   EQU     0F636H
        .
        .
        CALL    HCOPY
        .
        .
```

Return information: None

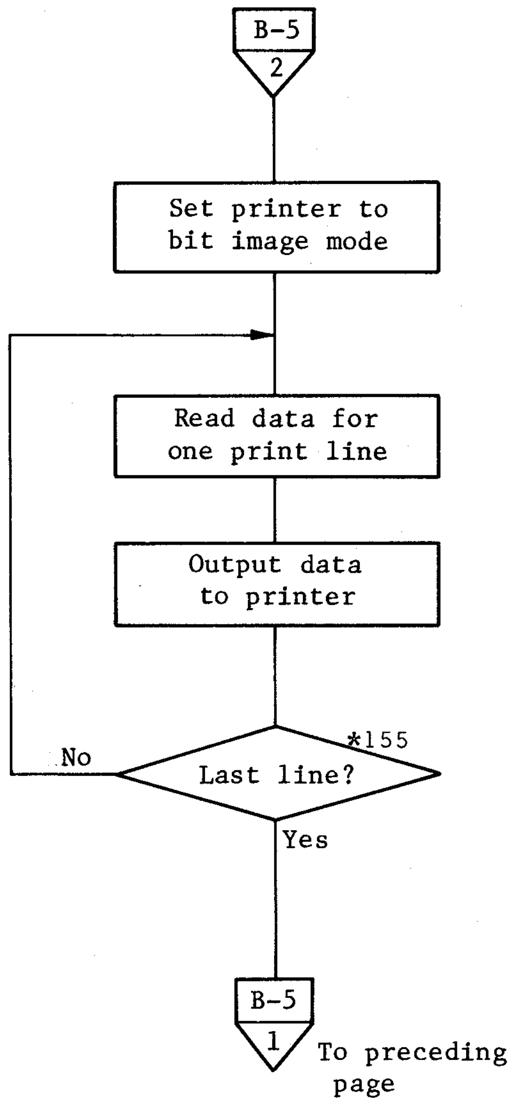
The contents of all registers are changed by execution of this routine.

7.2.3 General Flowchart



*151 Subroutines CRSW, VECTW, and MASK called.

*152 Subroutine LPOUT called.



*155 25 lines for non-MFBASIC normal mode; otherwise 20 lines

7.3 BEEP (Address: 0F639H)

7.3.1 Function

This routine controls the QX-10's built-in speaker.

3.2 Call procedure

Entry parameters:

Register C = Control data

C = 0	Speaker OFF	
C = 1 to 254 (0FEH)	Sounds the speaker for (specified value x 10) msec. The maximum period is 2.54 sec.	
C = 255	Sounds the speaker continuously.	

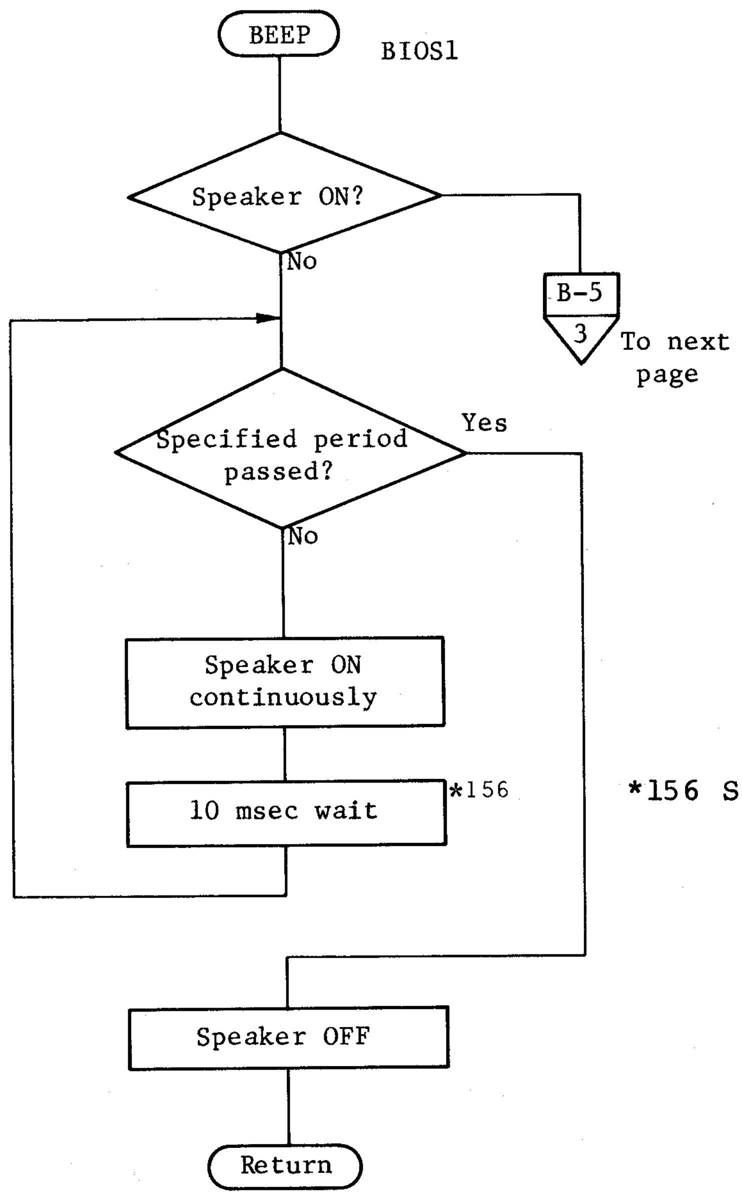
Example: The following sequence sounds the speaker for 0.1 sec.

```
BEEP:   EQU     0F639H
         .
         .
        LD      C,10
        CALL    BEEP
         .
         .
```

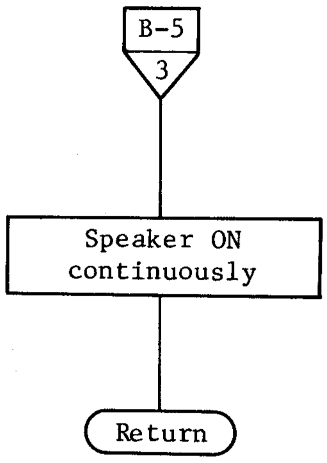
Return information: None

The contents of all registers are changed by execution of this routine.
Control is not returned to the user program as long as the speaker is still sounding.

7.3.3 General Flowchart



*156 Subroutine ST1ML called.


B-5
3
Speaker ON continuously
Return

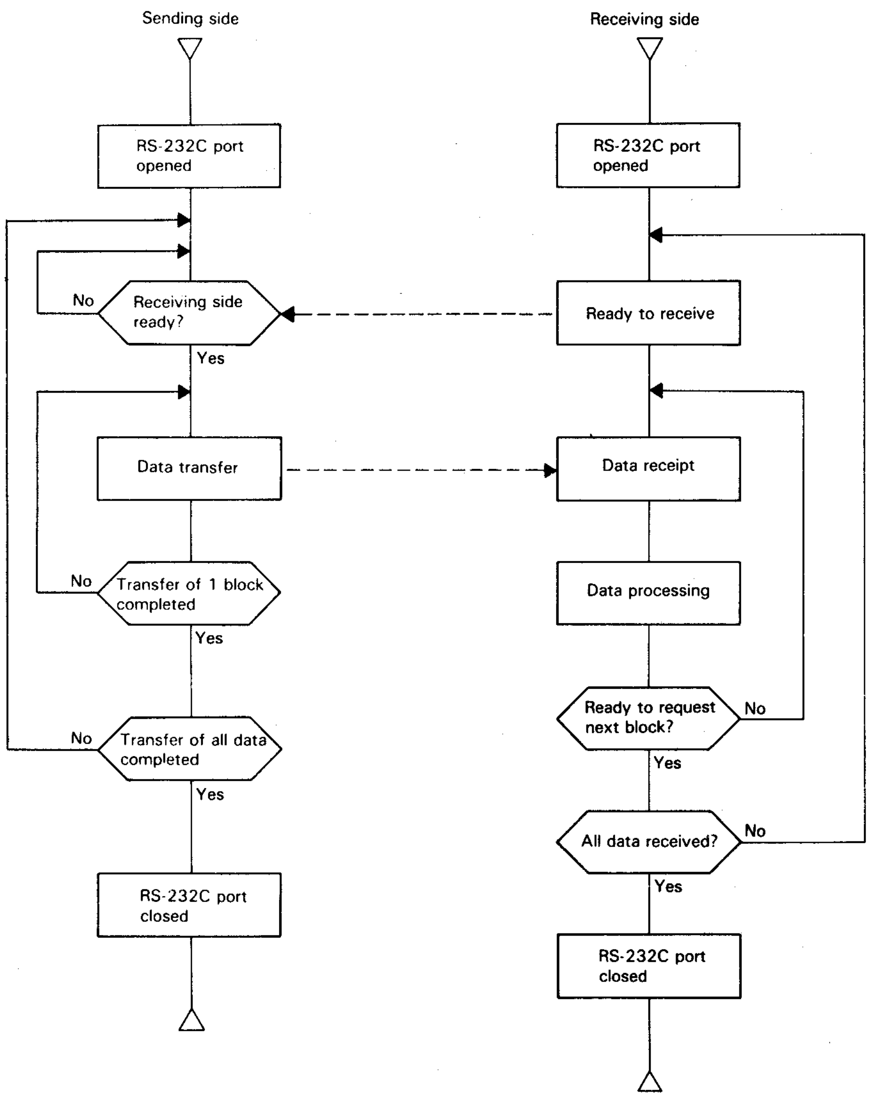
§8 RSOPEN, RSCLOSE, RSINST, RSOUTST, RSIN, RSOUT

8.1 RSOPEN (Address: 0F63CH)

8.1.1 Function

This routine initializes the main board RS-232C interface according to conditions set with the CONFIG command.
This routine cannot open the optional RS-232C interface ports.
XON/XOFF and SI/SO cannot be specified with this routine.
Refer to the "QX-10 Operation Manual" for details on the CONFIG command.

The receive buffer for the main board RS-232C interface is the 512-byte area starting at address 7000H in the system bank; the user does not need to prepare this receive buffer. Data reception is controlled by interrupt processing. Therefore, the user can perform other processing while receiving data through the RS-232C interface. However, if RS-232C interrupts are disabled, data cannot be received through the interface and the receive buffer in the interface controller will overflow (Receive Overrun). Use the following procedures to avoid this.



8.1.2 Call procedure

Entry parameters: None

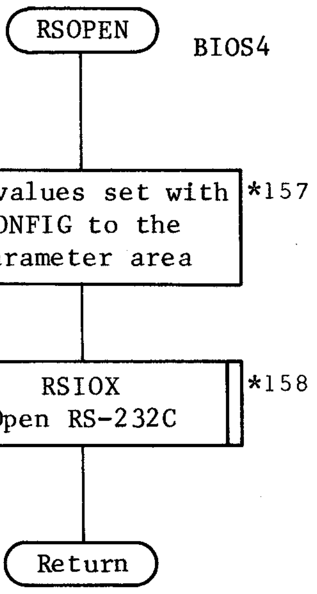
Example:

```
RSOPEN   EQU    0F63CH
         .
         .
         CALL   RSOPEN
         .
         .
```

Return information: None

The contents of all registers are changed when this routine is called. The RS-232C interface is reset when this routine is called. Therefore, data remaining in the receive buffer is discarded.

8.1.3 General Flowchart



*157 Sets parameters required for calling subroutine RSIOX (Open).

*158 BIOS entry RSIOX

8.2 RSCLOSE (Address: 0F63FH)

8.2.1 Function

This routine closes the main board RS-232C interface. The RS-232C interface is reset and cannot be used again until it has been reopened. Optional RS-232C interfaces cannot be closed with this routine.

8.2.2 Call Sequence

Entry parameters: None

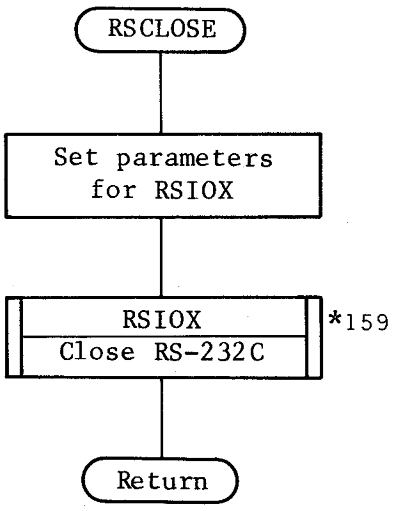
Example:

```
RSOPEN   EQU    0F63CH
         .
         .
RSCLOSE  EQU    0F63FH
         .
         .
         CALL   RSOPEN
         .
         .
         CALL   RSCLOSE
         .
         .
```

Return information: None

The contents of all registers are changed when this routine is called.

8.2.3 General Flowchart



*159 BIOS entry RSIOX (Close).

8.3 RSINST (Address: 0F642H)

8.3.1 Function

This routine checks for data in the receive buffer of the main board RS-232C interface and returns the result in register A. Optional RS-232C interfaces cannot be checked with this routine. The user can call this routine to check for receive data, then perform other processing if no data has been received.

8.3.2 Call procedure

Entry parameters: None

Example:

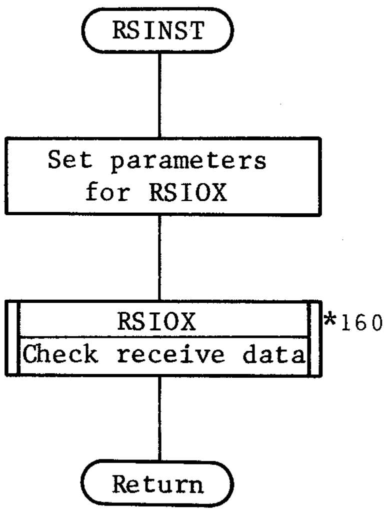
```
RSOPEN   EQU   0F63CH
RSINST   EQU   0F642H
RSCLOSE  EQU   0F63FH
         .
         .
         CALL  RSOPEN
         .
         .
         CALL  RSINST
         .
         .
         CALL  RSCLOSE
         .
         .
```

Return information:

Register	A = 0FFH:	Receive buffer contains data.
	A = 0:	Receive buffer is empty.

The contents of all other registers are also changed.

8.3.3 General Flowchart



*160 BIOS routine RSIOX (INSTS)

8.4. RSOUTST (Address: 0F645H)

8.4.1 Function

This routine checks whether the main board RS-232C interface is ready to send data and returns the result in register A. Optional RS-232C interfaces cannot be checked with this routine. To transmit data, the user must wait until the interface is ready to send. However, transmission will not be disabled unless a very low transfer rate is used.

8.4.2 Call procedure

Entry parameters None

Example:

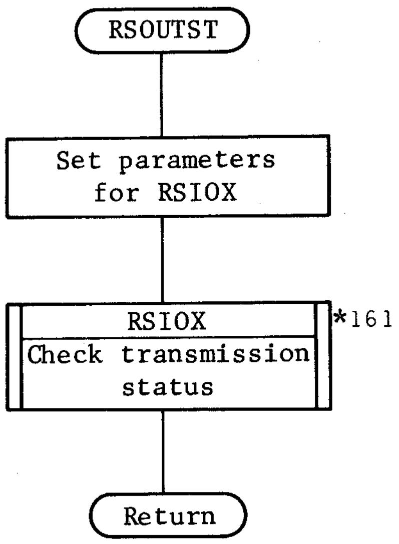
```
RSOPEN   EQU   0F63CH
RSOUTST  EQU   0F645H
RSCLOSE  EQU   0F63FH
         .
         .
         CALL  RSOPEN
         .
         .
         CALL  RSOUTST
         .
         .
         CALL  RSCLOSE
         .
         .
```

Return information:

Register A = 0FFH:	Ready to send
A = 0:	Not ready to send

The contents of other registers are also changed.

8.4.3 General Flowchart



*161 BIOS entry RSIOX (OUTST)

8.5 RSIN (Address: 0F648H)

8.5.1 Function

This routine reads one byte of data from the main board RS-232C interface and loads it into register A.
If the receive buffer contains no data, control is not returned to the calling program until data is received.
Therefore, it is recommended that this routine be used in combination with the RSINST routine.
When the character length is 7 bits or less, only significant bits are read. Therefore, the user does not need to remove insignificant bits.

8.5.2 Call procedure

Entry parameters: None

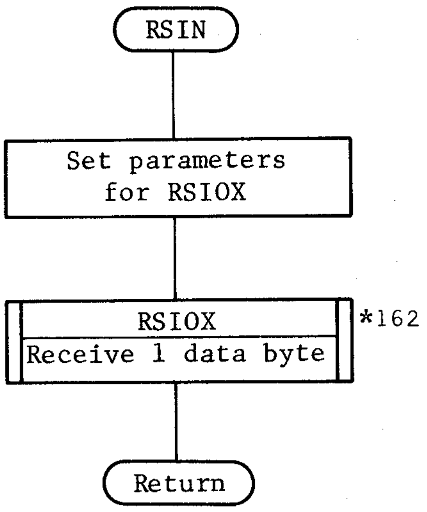
Example:

```
RSOPEN   EQU    3*19
RSINST   EQU    3*21
RSIN     EQU    3*23
RSCLOSE  EQU    3*20
BIOSE    EQU    1
         .
         .
EXRS:    LD     HL,(BIOSE)
         ADD    HL,DE
         JP     (HL)          ;EXECUTE BIOS.
         .
         .
         LD     DE,RSOPEN
         CALL   EXRS          ;OPEN
DATACHK: LD     DE,RSINST     ;
         CALL   EXRS          ;CHECK INPUT STATUS.
         AND    A             ;DATA READY?
         JR     Z,DATACHK     ;NO
         LD     DE,RSIN       ;
         CALL   EXRS          ;GET DATA.
         .
         .
```

Return information: None

The contents of all registers are changed by execution of this routine.

8.5.3 General Flowchart



*162 BIOS entry RSIOX (GET)

8.6 RSOUT (Address: 0F64BH)

8.6.1 Function

This routine sends one byte of data via the RS-232C interface. If the interface is not ready to send, this routine stands by until the interface becomes ready. Therefore, control is not returned to the calling program until data has been sent.

8.6.2 Call procedure

Entry parameters: Register C = Data to be sent

Example:

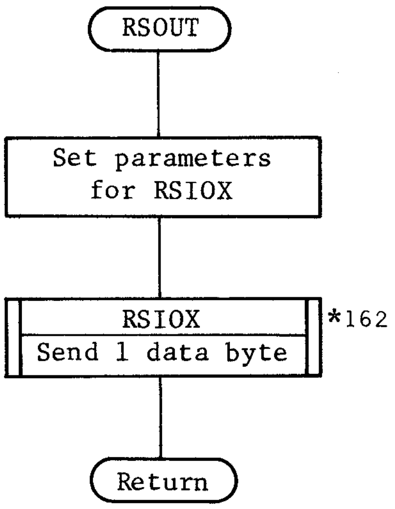
```
BIOSE     EQU     1
RSOPEN    EQU     3*19
RSOUTST   EQU     3*22
RSOUT     EQU     3*24
RSCLOSE   EQU     3*20
          .
          .
          LD      DE,RSOPEN     ;
          CALL    EXRS          ;OPEN.
          LD      HL,DATA       ;
          LD      B,(HL)        ;
          LD      A,B           ;B = DATA LENGTH
          CP      0             ;
          JR      Z,END         ;

OUT1:     INC     HL            ;
          LD      C,(HL)        ;C = OUTPUT DATA.
          PUSH    HL            ;
          PUSH    BC            ;
OUT2:     LD      DE,RSOUTST    ;
          CALL    EXRS          ;
          AND     A             ;READY?
          JR      Z,OUT2        ;NO.
          POP     BC            ;
          PUSH    BC            ;
          LD      DE,RSOUT      ;
          CALL    EXRS          ;OUTPUT.
          POP     BC
          POP     HL
          DJNZ    OUT1
END:      LD      DE,RSCLOSE
          CALL    EXRS
          .
          .
EXRS:     LD      HL,(BIOSE)
          ADD     HL,DE
          JP      (HL)
          .
          .
DATA:     DB      DATAEND-$-1
          DB      'ABCDEFG'
```

```
        DATAEND  EQU    $
                 .
                 .
```

Return information: None

The contents of all registers are changed by execution of this routine.

8.6.3 General Flowchart



*162 BIOS routine RSIOX (PUT) called.

§9 TIMDAT (Time/Data), MEMORY (Memory Bank Change)

9.1 TIMDAT (Address 0F64EH)

9.1.1 General

This routine sets or reads the date and time of the QX-10's CMOS clock, as specified by the entry parameters. The date and time settings can also be made with the CONFIG transient command or the TIME$ and DATE$ statements of MFBASIC. Once set, the CMOS clock's date and time are updated continuously until the backup battery becomes exhausted (about two weeks).

The year is set as a two digit number, and correction for leap years is made automatically. The time is indicated using the 24-hour system.

9.1.2 Call procedures

Entry parameters:	Register C = 0FFH	Date/time setting
	Register C = 0H	Date/time read
	Date area address	Contents
	0FEF8H	Year (00-99)
	0FEF9H	Month (01-12)
	0FEFAH	Day (01-31)
	0FEFBH	Hour (00-23)
	0FEFCH	Minute (00-59)
	0FEFDH	Second (00-59)
	0FEFEH	Weekday (00-06)
	0FEFFH	Reserved (00)

When 0FFH is set in register C to set the date and time, data is returned in BCD code in the area from 0FEF8H to 0FEFEH. When 0H is set in register C to read the clock, clock data is read into this data area in BCD code. For the weekday (address 0FEFEH), 0 stands for Sunday, 1 for Monday, and so forth.

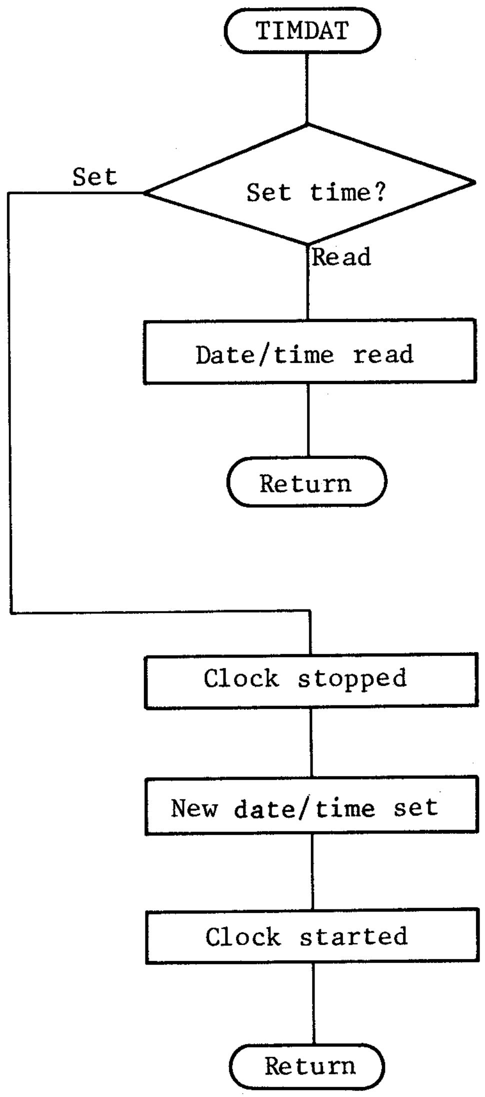
Example:
```
TIMDAT  EQU     0F64EH
        .
        .
        LD      C,0
        CALL    TIMDAT          ;READ TIME, DATE.
        .
        .
```

Return information: None.

When the date and time are read, data values are read into the common data area of BIOS (addresses 0FEF8H to 0FEFEH).

The contents of all registers are changed by execution of this routine.

9.1.3 General flowchart



9.2 MEMORY (Address 0F651H)

9.2.1 General

This routine switches to the memory bank specified in the entry parameters.
Addresses from 0H to 0DFFFH can be switched between memory banks, but addresses 0E000H to 0FFFFH are fixed to the same physical locations in memory. This area contains CCP, BDOS, and BIOS1.

When a user program is located in the area below 0DFFFH, this routine cannot be used. The reason for this is that, although control is shifted to the newly selected bank when banks are switched, normal system operation cannot be expected because there is no way of knowing what will be present at the address with which execution begins in the new bank. Therefore, programs calling this routine must be located in the memory area above 0E000H. Results are not assured if this restriction is ignored.

9.2.2 Call procedure

Entry parameters:

Register C = New bank number

C = 0	Main bank (bank containing user programs)
C = 1	User bank (disk image RAM, or open to the user)
C =-1	System bank (BIOS, etc.)
C = 2	User bank (optional)

See Appendix A for details on the memory banks.

Example:

```
MEMORY  EQU     0F651H
        .
        .
        LD      C,1             ;
        CALL    MEMORY          ;SELECT USER BANK.
        .
```

Return parameters: Register A = 0 Normal completion
Register A ≠ 0 Parameter error

When 0 is set in register A, register C contains the number of the bank selected before the routine was called. All other registers remain unchanged.