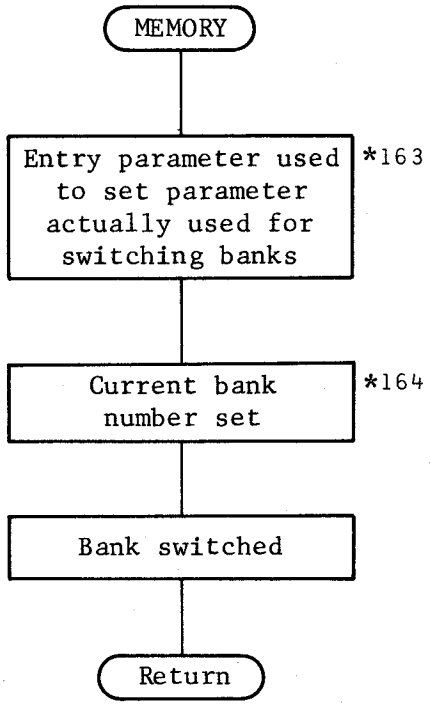
9.2.3 General flowchart



*163 Subroutine SMBANK called to set register A according to the contents of register C.

Register C	Register A
0	10H
1	40H
-1	20H
2	80H

*164 Used as the return information.

§10 RSIOX (RS-232C Input/Output Execute)

10.1 RSIOX (Address: 0F654H)

10.1.1 General

This routine controls operation of up to five RS-232C interface channels (the main board interface and those on option cards) according to the entry parameters. Any of 10 control functions may be specified in the entry parameters. These functions are summarized below.

a) OPEN - Initializes the RS-232C interface and resets interrupt masks, enabling the interface for use.

b) CLOSE - Resets the RS-232C interface and sets interrupt masks. Afterwards, the interface cannot be used until it has been reopened.

c) INSTS - Checks for data in the receive buffer.

d) OUTST - Checks whether or not transmission is possible.

e) GET - Reads in one byte of data from the receive buffer.

f) PUT - Sends one byte of data.

g) CTLIN - Reads the status of the RS-232C control lines.

h) SETCTL - Sets the RS-232C control lines.

i) ERSTS - Check the error status and resets error flags.

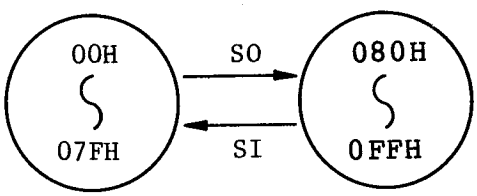
j) SENS - Checks whether the RS-232C interface is usable.

The optional RS-232C interface cards each include two interface channels. Each of these channels can be used independently, and can be set for transfer speeds of from 50 to 19200 bps. (An error will result if an attempt is made to set the main board interface for a transfer speed of 19200 bps.)

Further, XON/XOFF protocol and SI/SO (shift in/shift out) control are provided.

When SI/SO control is specified, 8-bit character codes are converted to 7-bit code for transmission as follows.

[During transmission]



However, the SI code must always be sent before sending the CR or LF codes, then an SI or SO code must be sent before sending the next character code.

---, SI, CR, SI, '6', 'M', ---

---, SI, LF, SO, 'M', ---

(Sample in MF mode)

[During reception]

o After receiving the SI code, bit 7 of all subsequent characters received is set to 0.

o When the SO code is received, subsequent codes 020H-07EH are converted to 0A0H-0FEH, and codes 0H-01FH and 07FH are left unchanged.

04DH, SO, 036H, 04DH,---------, SI, 036H,

---, '6', 'M', 'M', ---

(Sample in MF mode)

Processing is as follows when XON/XOFF control is specified.

When the rate of data transfer is so great that the receiving side cannot process data as fast as it is received, XOFF is sent to the sending side to temporarily stop transmission when the receive buffer becomes 3/4 full. The receive side continues reading in data from the receive buffer during this time, then sends XON to the sending side to resume transmission when the amount of data in the receive buffer has been reduced to 1/4 of its capacity. This makes it possible to prevent the receive buffer from overflowing.

When the sending side receives XOFF, it temporarily disables transmission, then reenables transmission when XON is received. However, codes which control data transmission in this manner are not passed to the user.

CTRL-S (013H) is used as the XOFF code, and CTRL-Q (011H) is used as the XON code.

Note:

When SI/SO or XON/XOFF control is used, data processing is governed by special codes. Since these special codes may be included in binary data such as machine language programs, proper data transmission cannot be expected when XON/XOFF or SI/SO control is used.

10.1.2 Call procedure

a) OPEN: Opens the RS-232C interface.

Entry parameters: Register B=01xH

The value specified for x determines which channel is opened as follows.

x=0: Main board RS-232C interface

1: Option card 1, channel A

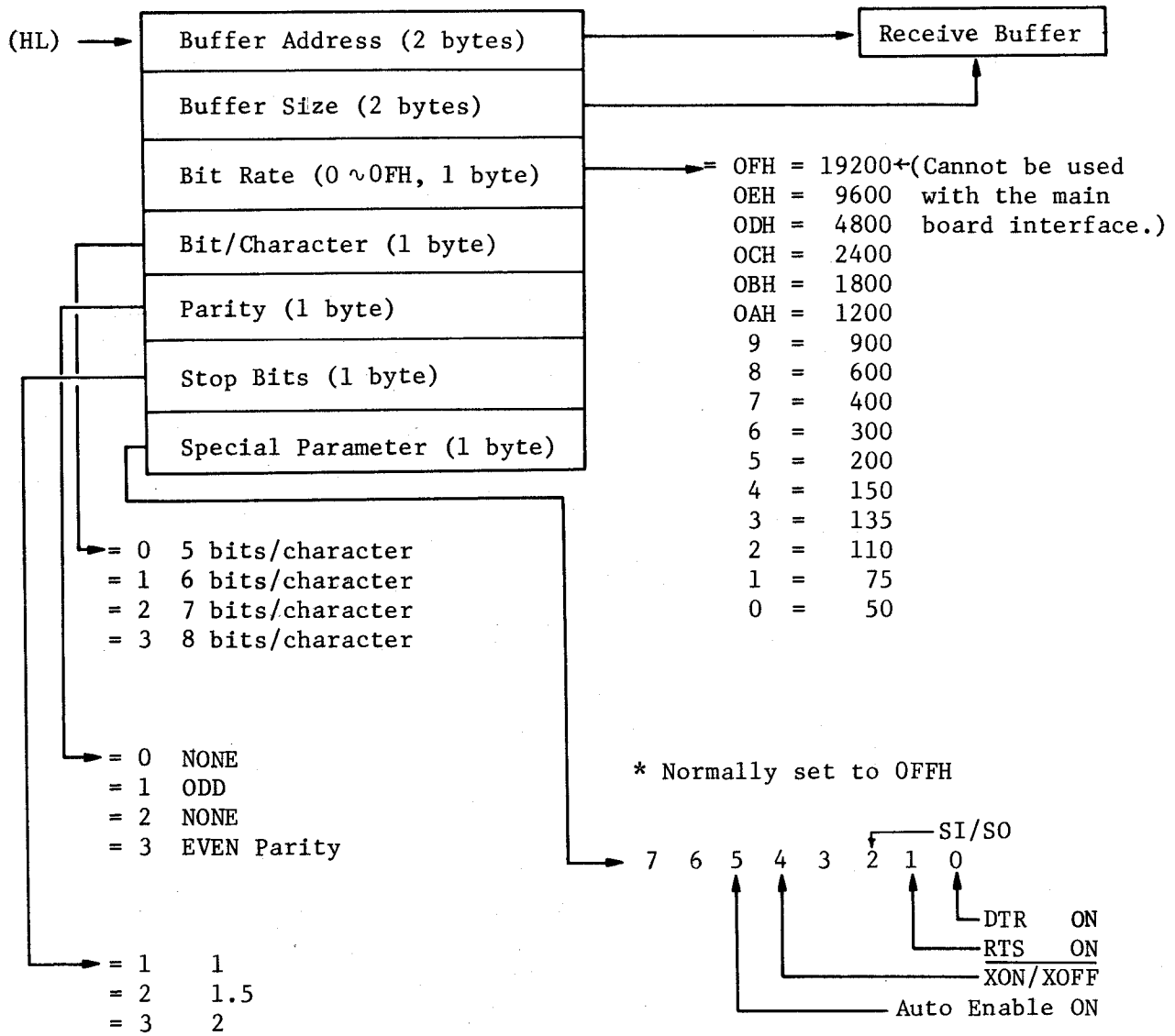
2: Option card 1, channel B

3: Option card 2, channel A

4: Option card 2, channel B

See below concerning assignment of device addresses to option cards 1 and 2.

Register HL=Parameter block address

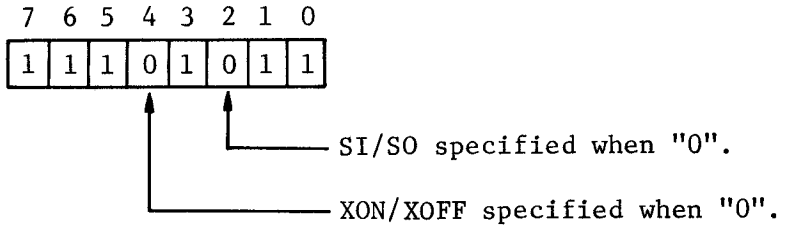


The optional RS-232C interface card consists of a single board which is equipped with two channels. These are designated channel A and channel B, with channel A corresponding to the channel with the lower I/O port address and channel B corresponding to the channel with the higher address.

Up to two optional RS-232C interface cards can be installed in the QX-10 concurrently. When this is done, the jumper settings on one of the cards must be changed. The card on which the jumper settings are changed becomes option card 2, and that on which they are not changed is option card 1.

See the RS-232C I/F card User Manual for the settings of the jumpers.

SI/SO and XON/XOFF control become effective when corresponding bits of the special parameter byte are reset to "0".



The Auto Enable, RTS, and DTR signals are set when corresponding bits are set to "1".

Example of parameter setting

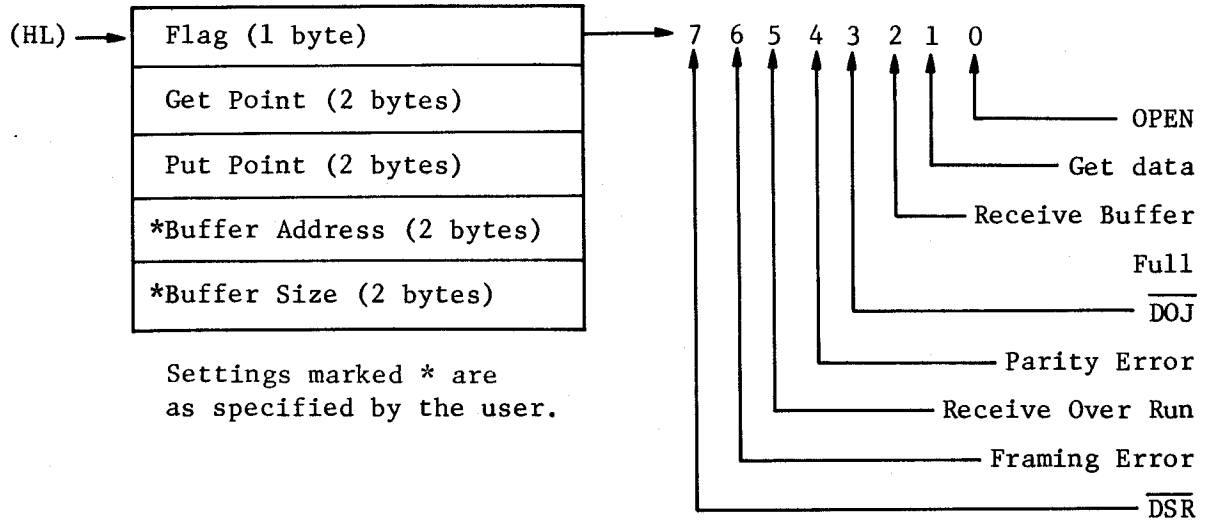
The parameter block is set as follows for a transfer rate of 9600 bps, a word length of 8 bits/character, 1 stop bit, no parity, and XON/XOFF control.

```
PARM:     DW      BUFFER          ;BUFFER ADDRESS.
          DW      2048            ;BUFFER SIZE.
          DB      0EH             ;9600 BPS.
          DB      3               ;8BIT/CHAR.
          DB      0               ;NO PARITY
          DB      1               ;STOP BIT=1
          DB      011101111B      ;XON.
;
BUFFER:   DS      2048            ;RECEIVE DATA BUFFER.
```

A parameter error will occur if a transfer rate of 19200 bps is specified for the main board RS-232C port or if an incorrect value is set in register B.

Register A =	0	Normally opened	Z flag = 1
	1	Option board not installed	0
	2	Interface busy (being used by another program)	0
	4	Parameter error	0

Register HL = Not changed



The contents of all other registers are changed when the interface is opened.

The 1-byte flag in the return information indicates the status of the RS-232C interface hardware and the software. Bits 3 to 7 of the flag indicate the hardware status, and bits 0 to 2 indicate the software status. The meanings of the hardware status bits are as follows.

Bit 3: Indicates the staus of DCD ("0" when H and "1" when L).

Bit 4: When "1", indicates that a hardware error was detected.

Bit 5: When "1", indicates that the hardware receive buffer (3 bytes) has overflowed.

Bit 6: Set to "1" when the bit rate specified does not match that of the other device.

Bit 7: Indicates the status of DSR ("1" when H and "0" when L).

Bits 4 to 6 are flags which indicate the occurrence of RS-232C interface errors. The settings of these flags remain unchanged until they are reset using the ERSTS function.

Execution of the OPEN function enables the specified RS-232C interface for communication.

Sample call

The following is an example of an RSIOX OPEN call to open option card 1, channel B with the parameters described in the example given earlier.

```
RSIOX   EQU   0F654H
        .
        .
        LD    B,012H
```

```
        LD      HL,PARM
        CALL    RSIOX
        JP      NZ,ERROR
        .
        .
```

b) CLOSE: Closes the RS-232C interface.

Entry parameters:

Register B = 02xH (see the description of the OPEN function for the meaning of x.)

Example: The following example closes the RS-232C interface used in the example given for the OPEN function.

```
        .
        .
        LD      B,022H
        CALL    RSIOX
        .
        .
```

Return information: None

Any remaining receive data is ignored and the contents of all registers are changed.

c) INSTS: Checks for receive data in the receive buffer.

Entry parameters:

Register B = 03xH (see the explanation of the OPEN function for the meaning of x.)

Register HL = Starting address of the area used for storing return information. The return information is the same as for the OPEN function; a 9-byte area must be reserved for storing this information. Once the HL register has been specified for the OPEN function, it need not be reset as long as it is not changed.

Example:

```
            .
            .
CHKDATA:    LD      B,032H      ;CHECK INPUT STATUS.
            CALL    RSIOX       ;
            JP      NZ,ERROR    ;ERROR OCCURRED.
            AND     A           ;DATA READY?
            JR      Z,CHKDATA   ;NO.
            .
            .
```

Return information:

Z flag = 1: Normal completion

Register A = 0FFH Receive buffer contains data.

Register A = 0H Receive buffer is empty.

Register BC = LOC (number of bytes of receive data)

Register HL = Return information (see the explanation of the OPEN function)

Contents of all other registers are changed.

Z flag = 0: Abnormal completion

Register A = 3 Specified channel has not been opened.

Register A = 4 Parameter error; an incorrect parameter was set in register B.

Contents of other registers are also changed.

d) OUTST: Checks whether transmission is enabled.

Entry parameters:

Register B = 04xH (see the explanation of the OPEN function for the meaning of x.)

Register HL = Starting address of the area used for storing return information. A 9-byte area must be reserved for storing the return information.

Example:

```
            .
            .
CHKOUT:  LD      B,042H
         CALL    RSIOX          ;CHECK OUTPUT STATUS.
         JP      NZ,ERROR       ;ERROR.
         AND     A              ;READY?
         JR      Z,CHKOUT       ;NO.
            .
            .
```

Return information

Z flag = 1: Normal completion

Register A = 0FFH Transmission enabled.

Register A = 0H Transmission disabled.

Register HL = Return information.

(See the explanation of the OPEN function for the return information.)

The contents of other registers are also changed.

Z flag = 0: Abnormal completion

Register A = 3 Interface not open.

Register A = 4 Parameter error.

The contents of other registers are also changed.

e) GET: Reads in one byte of data from the receive buffer.

Entry parameters:

Register B = 05xH (see the explanation of the OPEN function for the meaning of x.)

Register HL = Starting address of the area used for storing return information. A 9-byte area must be reserved for storing the return information.

Example:

```
            LD      B,012H      ;
            LD      HL,PARM     ;
            CALL    RSIOX       ;OPEN.
            JP      NZ,ERROR    ;IF ERROR.
            LD      IX,DATA     ;IX = DATA STACK AREA.
            LD      DE,2048     ;DE = DATA LENGTH.
CHKDATA:    PUSH    DE          ;
            PUSH    IX          ;
            LD      B,032H      ;
            CALL    RSIOX       ;CHECK INPUT STATUS.
            POP     IX          ;
            POP     DE          ;
            JP      NZ,ERROR    ;IF ERROR.
            AND     A           ;DATA READY?
            JR      Z,CHKDATA   ;NO.
GETDATA:    PUSH    BC          ;
            PUSH    DE          ;
            PUSH    IX          ;
            LD      B,052H      ;
            CALL    RSIOX       ;GET DATA.
            POP     IX          ;
            POP     DE          ;
            POP     BC          ;
            JP      NZ,ERROR    ;IF ERROR.
            LD      (IX),A      ;STACK RECEIVE DATA.
            INC     IX          ;
            DEC     BC          ;LOC-1.
            DEC     DE          ;DATA LENGTH-1
            LD      A,B         ;
            OR      C           ;RECEIVE DATA END?
            JR      NZ,GETDATA  ;NO.
            LD      A,D         ;
            OR      E           ;RECEIVE COMPLETE?
            JR      NZ,CHKDATA  ;NO.
```

Return information:

Z flag = 1: Normal completion

Register A = Receive data

Register HL = Return information.

See the explanation of the OPEN function for the return information.

The contents of other registers are also changed.

Z flag = 0: Abnormal completion

Register A = 3 Interface not open.

Register A = 4 Parameter error incorrect value set in register B.

The contents of other registers are also changed.

Note:

If a word length of fewer than 8 bits/character is specified when an interface is opened, invalid bits such as parity bits are cut before the character is passed to the user; therefore, the user need not cut invalid bits in advance. The user also need not take into consideration internal processing of SI and SO codes.
If no receive data is present (if the value returned in register A is 0), this routine does not return to the user program until data is received.
Further, normal reception of binary data such as machine language programs will not be possible if either XON/XOFF or SI/SO control is specified when the interface is opened. See 1., "General" for details.

f) PUT: Sends one byte of data.

Entry parameters:

Register B = 06xH (see the explanation of the OPEN function for the meaning of x.)

Register C = Send data

Register HL = Starting address of the area used for storing return information. A 9-byte area must be reserved for storing the return information.

Example:

```
        LD      B,012H      ;
        LD      HL,PARM     ;
        CALL    RSIOX       ;OPEN.
        JP      NZ,ERROR    ;IF ERROR.
        LD      DE,2048     ;PUT DATA LENGTH.
        LD      IX,DATA     ;DATA ADDRESS.
CHKOUT: PUSH    DE          ;
        PUSH    IX          ;
        LD      B,042H      ;
```

```
                CALL    RSIOX           ;CHECK OUTPUT STATUS.
                POP     IX              ;
                POP     DE              ;
                JP      NZ,ERROR        ;IF ERROR.
                AND     A               ;READY?
                JR      Z,CHKOUT        ;NO.
                LD      B,062H          ;
                LD      C,(IX)          ;
                PUSH    DE              ;
                PUSH    IX              ;
                CALL    RSIOX           ;PUT!
                POP     IX              ;
                POP     DE              ;
                JP      NZ,ERROR        ;IF ERROR
                DEC     DE              ;
                LD      A,D             ;
                OR      E               ;DATA END?
                JR      Z,PUTEND        ;YES.
                INC     IX              ;
                JR      CHKOUT          ;
        ;
        PUTEND: .
                .
                .
```

Return information:

Z flag = 1: Normal completion

Register HL = Return information.

See the explanation of the OPEN function for the return information.

The contents of other registers are also changed.

Z flag = 0: Abnormal completion

Register A = 3 Interface not open.

Register A = 4 Parameter error

An incorrect value was set in register B.

The contents of other registers are also changed.

Note:

Binary data (codes from 0H to 0FFH) such as machine language programs will not be properly received if either XON/XOFF or SI/SO control is specified when the interface is opened. See 10.1.1 above for details.

g) CTLIN: Reads the current status of the control lines.

Entry parameters: Register B = 07xH (see the explanation of the OPEN function for the meaning of x.)

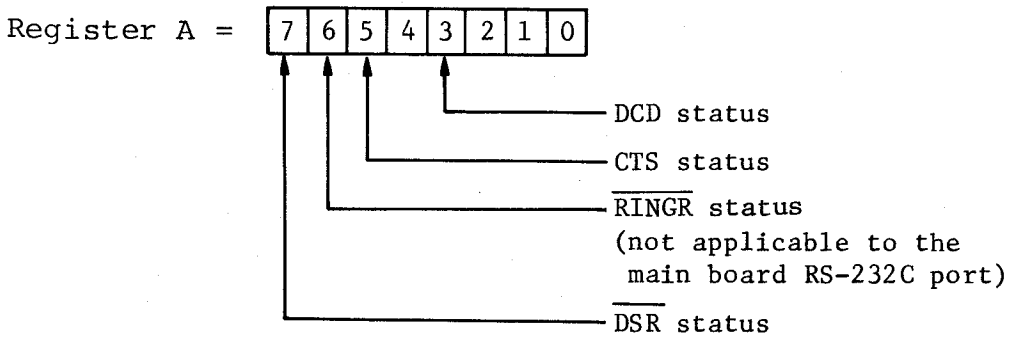
Example:

```
        .
        .
        LD      B,072H
        CALL    RSIOX
        JP      NZ,ERROR
        .
        .
```

Return information:

Z flag = 1: Normal completion

Register A: A value indicating the states of the control lines is set as follows.



DCD is H when "1" and L when "0".

CTS is H when "1" and L when "0".

RINGR is L when "1" and H when "0".

DSR is L when "1" and H when "0".

The contents of other registers are also changed.

Z flag = 0: Abnormal completion

Register A = 3 Specified channel not open.

Register A = 4 Parameter error Incorrect value was set in register B.

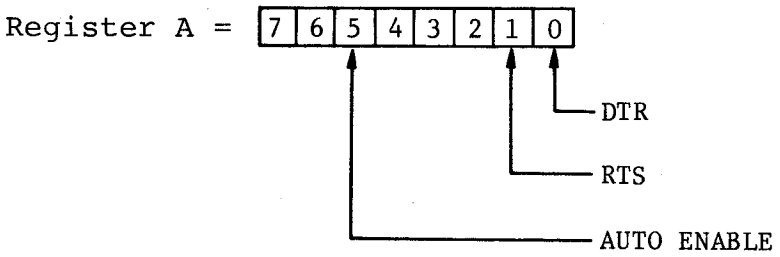
The contents of other registers are also changed.

h) SETCTL: Sets the control lines.

Entry parameters:

Register B = 08xH (See the explanation of the OPEN function for the meaning of x.)

Register C = Control line settings.



Control lines are set when corresponding bits are set to "1".

```
Example:    LD      B,082H      ;
            LD      C,02H       ;RTS=1
            CALL    RSIOX       ;
            .
            .
```

Return information:

Z flag = 1: Normal completion

The contents of other registers are also changed.

Z flag = 0: Abnormal completion

Register A = 3 Specified channel not open.

Register A = 4 Parameter error

An incorrect value was set in register B.

The contents of other registers are also changed.

i) ERSTS: Sets the error status in register A and clears the error flags.

Entry parameters:

Register B = 09xH (see the explanation of the OPEN function for the meaning of x.)

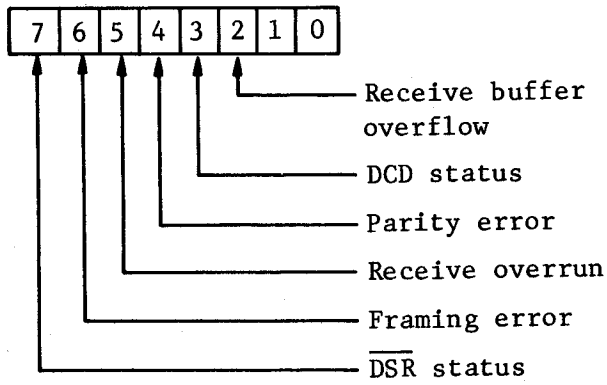
```
Example:    .
            .
            LD      B,092H
            CALL    RSIOX
            JP      NZ,ERROR
            .
            .
```

Return information:

Z flag = 1: Normal completion

The error status is set in register A as follows.

Register A =



See page 1-98 for the meanings of the individual bit settings. These settings are maintained until they are cleared. The contents of other registers are also changed.

Z flag = 0: Abnormal completion

Register A = 3 Interface not open.

Register A = 4 Parameter error
An incorrect value was set in register B.

The contents of other registers are also changed.

j) SENS: Checks whether the RS-232C interface channel specified in the entry parameter is usable.

Entry parameters:

Register B = 0FxH (see the explanation of the OPEN function for the meaning of x.)

Example:

```
        LD      B,0F2H          ;
        CALL    RSIOX           ;
        JP      NZ,BUSY         ;BUSY OR NOT READY.
        LD      B,012H
        LD      HL,PARM
        CALL    RSIOX
        .
        .
BUSY:   CP      2               ;BUSY?
        JR      Z,BUSY1         ;YES.
        :                       ;UNATTACHED.
BUSY1:  .
        .
```

Return information:

Z flag = 1: Specified channel usable.

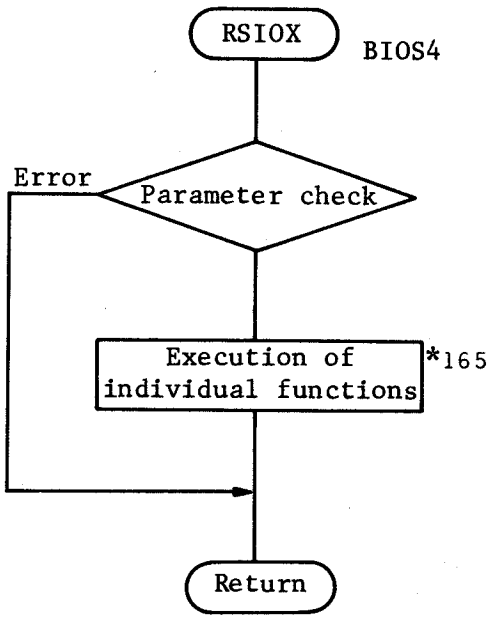
Z flag = 0: Specified channel not usable.

Register A = 1 Option card not installed.

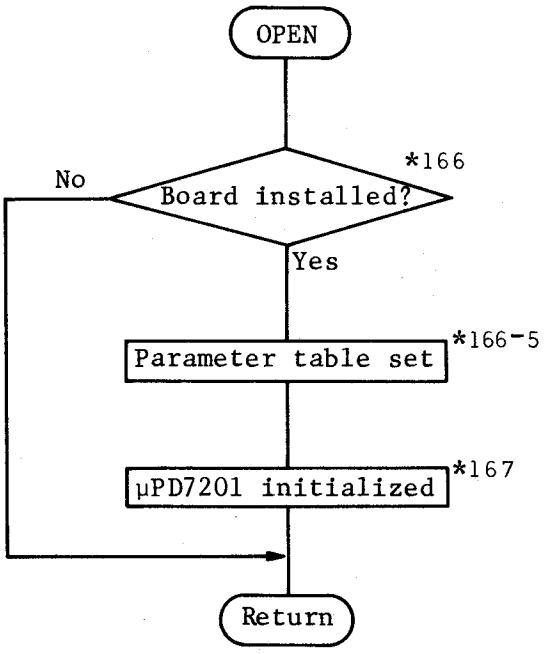
Register A = 2 Channel being used by another program.

The contents of other registers are also changed.

10.1.3 General flowchart



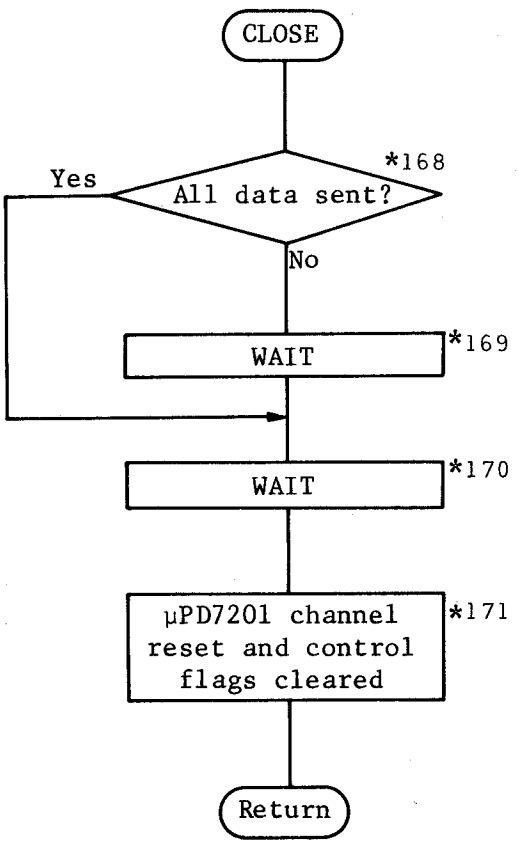
*165 The various functions are executed by means of multiple jumps. See the next page.



*166 Subroutine RSCHK called to check whether the option card is installed when an optional interface channel is specified.

*166-5 Data for individual control lines set in control table for specified channel.

*167 μPD7201 RS-232C controller set according to specified parameters.

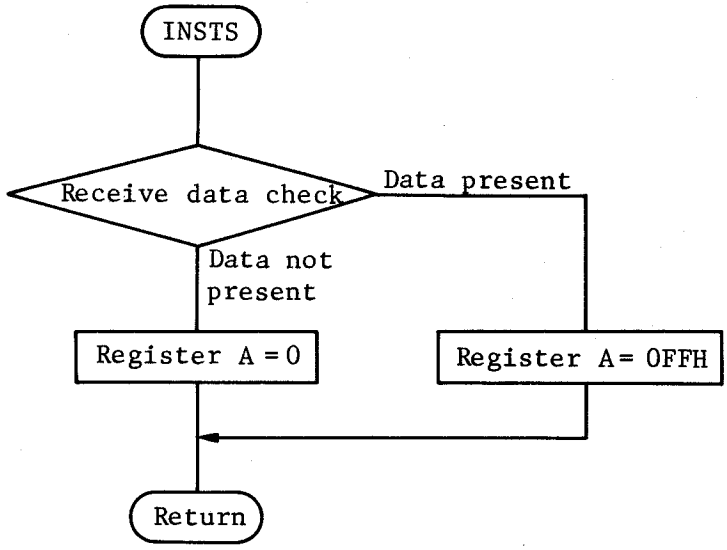


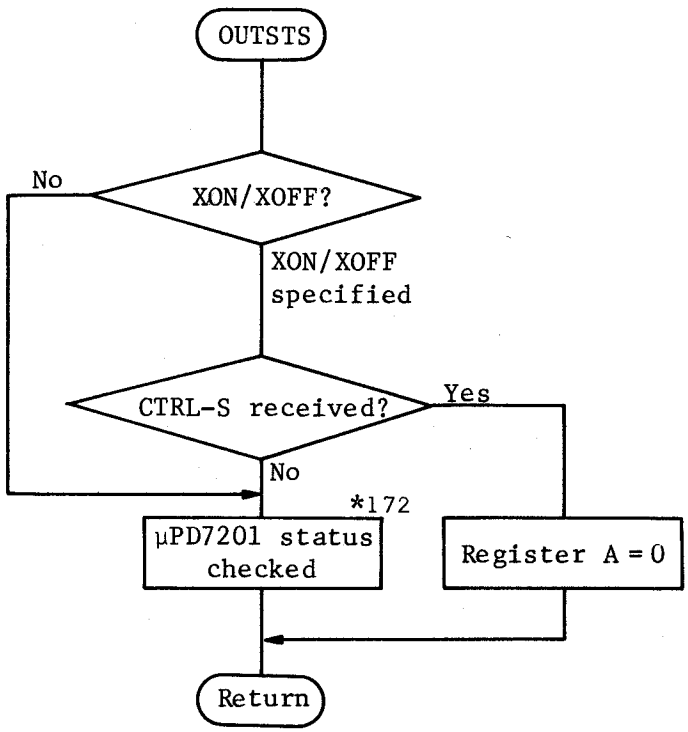
*168 Check made for unsent data in the send buffer of the μPD7201.

*169 450 msec wait.

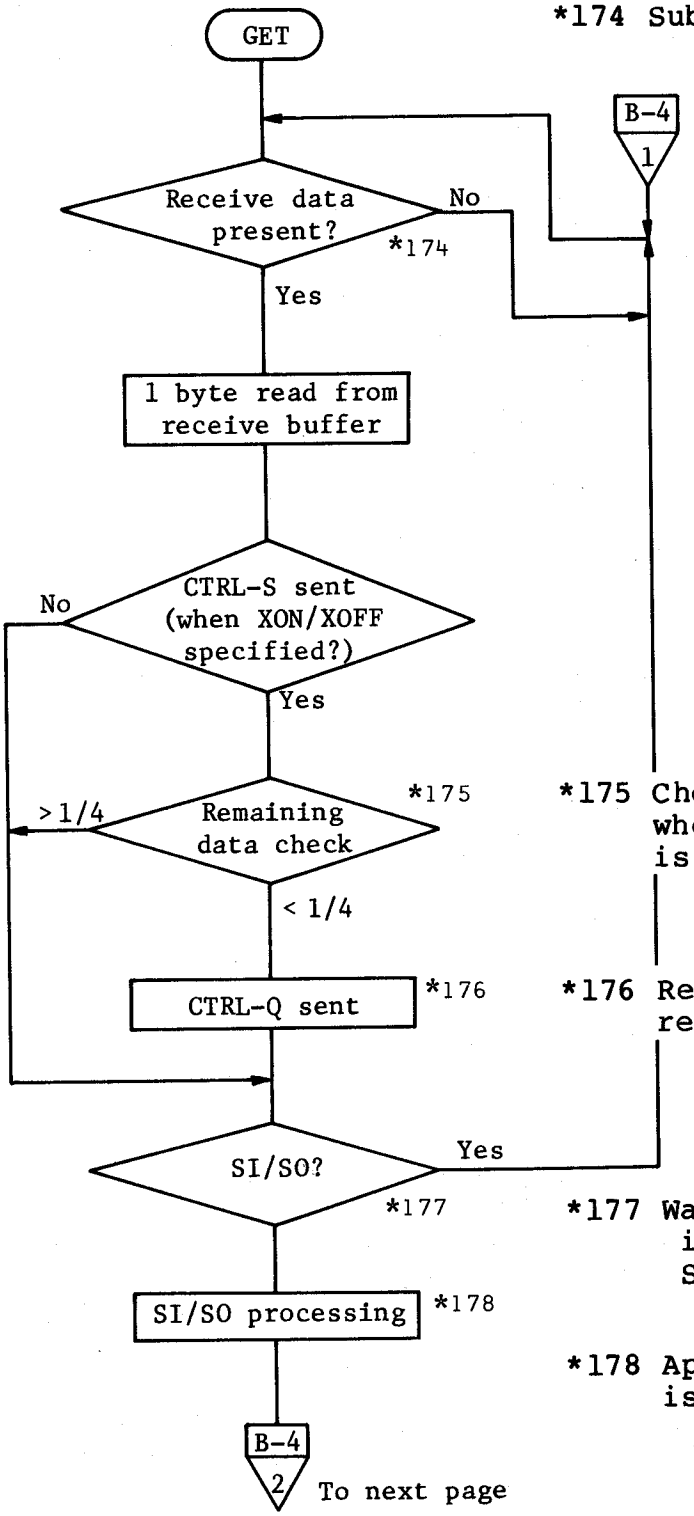
*170 50 msec wait.

*171 Channel specified by parameter reset and control flags cleared.





*172 Subroutine WRSTS called to check the status of the μPD7201 send buffer.



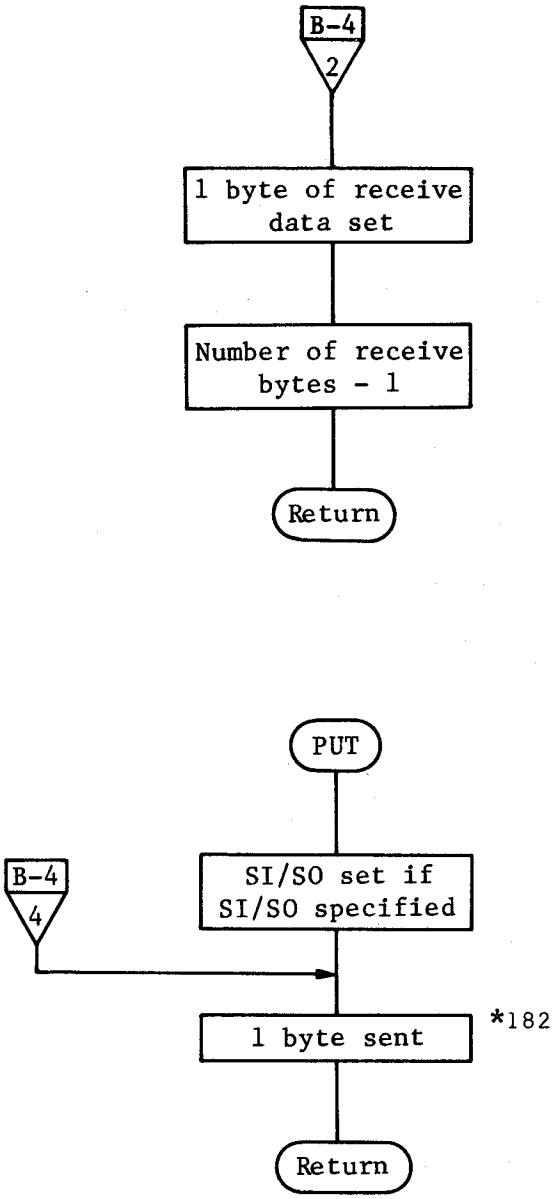
*174 Subroutine RDSTS called.

*175 Check made to determine whether the receive buffer is less than 1/4 full.

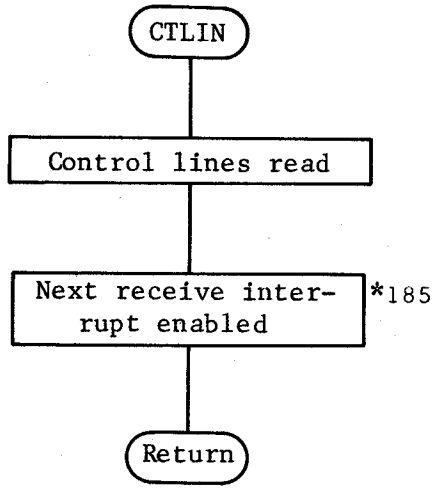
*176 Resumption of transmission requested.

*177 Waits until the next code is received (only when SI/SO is specified).

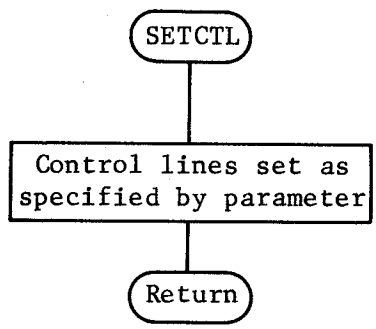
*178 Applicable only when SI/SO is specified.

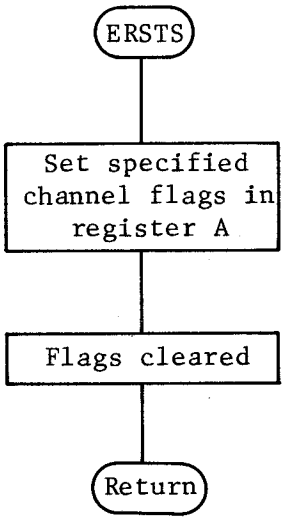


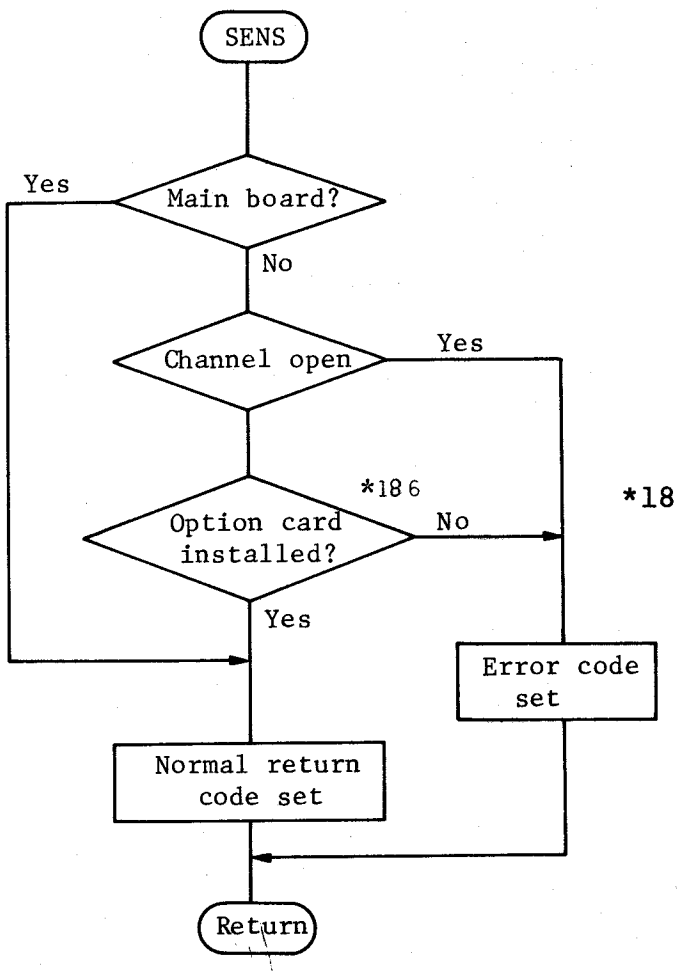
*182 Subroutine PUTSET called.



*185 Enable Interrupt on Next Rx Character command sent to the μPD7201.







*186 Subroutine RSCHK called.

§11 LIGHTPEN, MASKI (Mask Set/Reset)

11.1 LIGHTPEN (Address: 0F657H)

11.1.1 Function

This BIOS routine checks for light pen input and reads the horizontal and vertical input positions.
No data is returned if the light pen is not connected. When the light pen is pressed against the display screen, a display interrupt is issued to the CPU so that the light pen interrupt processing routine is activated. The routine reads the screen location at which the light pen is pressed. Subsequent interrupts are disabled until the data from that location is read. (However, the routine does not hang up).
The following procedures are required to use the light pen.

a) Unmask light pen interrupts by calling MASKI.

b) Check for light pen input, and read and process input data.

c) Mask light pen interrupts by calling MASKI. This is necessary because the priority of light pen interrupts is higher than those from the flexible disk drives, printer, calender clock, software timer #2 and options 3 to 7, and lower level interrupts will not be accepted if the light pen is accidentally pressed against something (e.g., a desk). Therefore, it is recommended that the light pen interrupt mask be set when light pen processing is not being performed.

The display location returned by the LIGHTPEN routine is indicated as follows.

a) With a green monitor in the non-MFBASIC normal mode

Horizontal location: 0 to 79

Vertical location: 0 to 24

Therefore, the locations of the four corners of the screen are indicated as follows.

	Horizontal	Vertical
Upper left	0	0
Upper right	79	0
Lower left	0	24
Lower right	79	24

b) With a green monitor in the non-MFBASIC MF mode or MFBASIC mode, or with a color monitor

Horizontal location: 0 to 39

Vertical location: 0 to 399

In this case, the horizontal location is read in 16-dot units and the vertical location is read in 1-dot units.

Therefore, the locations of the four corners of the screen are indicated as follows.

	Horizontal	Vertical
Upper left	0	0
Upper right	39	0
Lower left	0	399
Lower right	39	399

The light pen's sensitivity to red is very low; therefore, use of the light pen for detecting red should be avoided.
The LIGHTPEN routine has two functions.

a) Checks whether the light pen is being pressed against the screen.

b) Reads location data.

11.1.2 Call procedure

a) To check whether the light pen is pressed

Entry parameters: Register C = 2

Return information:

Z flag = 1: Normal completion

Register A = 0FFH — The light pen is being pressed and location data has been read.

Register A = 0 — The light pen is not being pressed and no data has been read.

The contents of other registers are also changed.

Z flag = 0: Abnormal completion

Register A = 1: Parameter error
Incorrect value was specified in register C.

The contents of other registers are also changed.

b) To read light pen data

Entry parameters: register C = 3

Return information:

Z flag = 1: Normal completion

Register A = 0 — The light pen has not been pressed and no location data has been read.

Register A ≠ 0: — Register pairs BC and DE contain location data.

Register pair BC: Horizontal location

Register pair DE: Vertical location

The contents of other registers are also changed.

Z flag = 0: Abnormal completion

Register A = 1: Parameter error
Incorrect value was specified in register C.

The contents of other registers are also changed.

Example)

```
BASE     EQU    1              ;
LIGHTPEN EQU    28             ;
MASKI    EQU    29             ;
         .
         .
         LD     BC,03BH        ;B=0,C=03BH
         LD     DE,MASKI       ;
         CALL   BIOS           ;ENABLE LIGHT PEN.
         JP     NZ,ERROR       ;IF ERROR.
CHKLP:   LD     C,2            ;
         LD     DE,LIGHTPEN    ;
         CALL   BIOS           ;
         JP     NZ,ERROR       ;
         AND    A              ;DATA READY?
         JR     Z,CHKLP        ;NO.
         LD     B,3            ;
         LD     DE,LIGHTPEN    ;
         CALL   BIOS           ;GET LIGHT PEN ADDR.
         JR     NZ,            ;
         .
         LD     B,020H         ;DISABLE LIGHT PEN.
         LD     DE,MASKI       ;
         CALL   BIOS           ;
         .
         .
```

The following expressions are used to convert location data read into the display address.

a) With a green monitor in the non-MFBASIC normal mode:

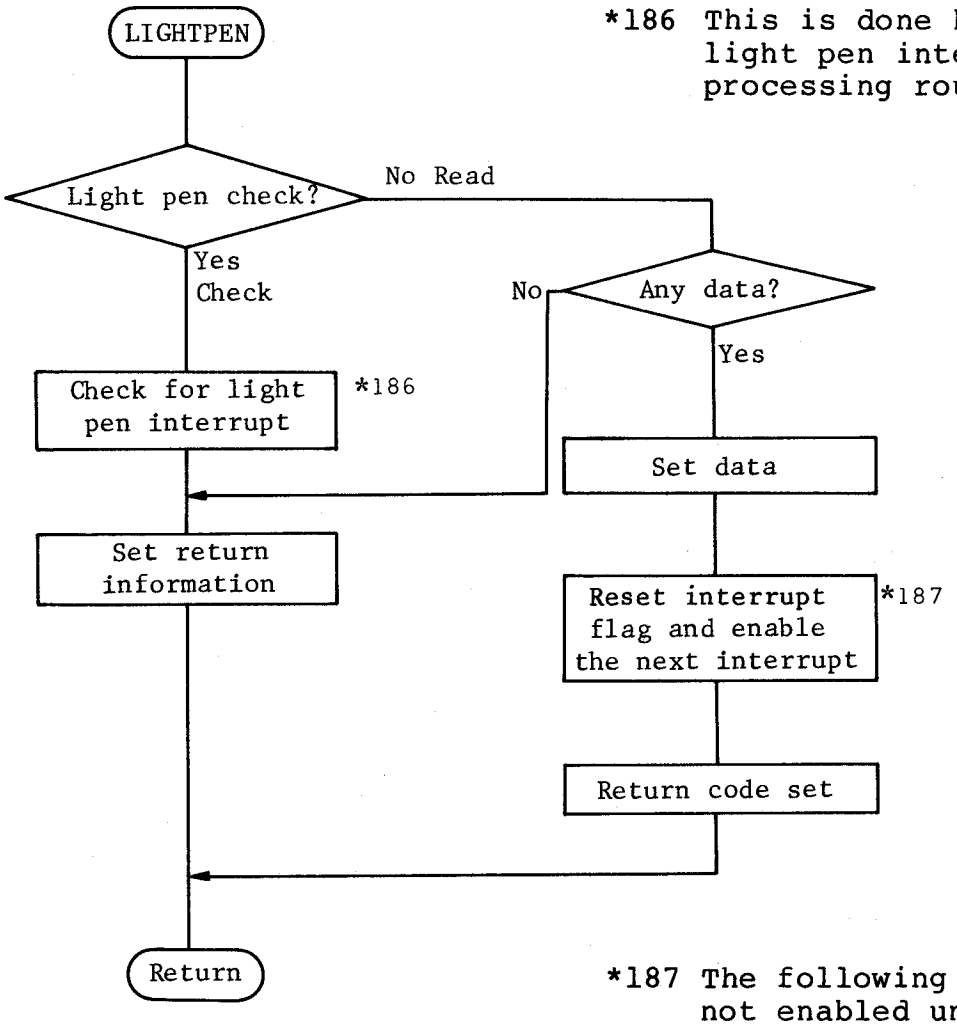
Address = (contents of 2 bytes at 0FE70H)
+ (contents of register DE x 80)
+ (contents of register BC)

The 2 bytes at 0FE70H are SPOS of the BIOS common data area. This contains the display address of the upper left corner of the screen (that is, the starting address of the current V-RAM).

b) With a green monitor in the non-MFBASIC MF mode or MFBASIC mode or with color monitor:

Address = (contents of 2 bytes at 0FE70H)
+ (contents of register DE x 40)
+ (contents of register BC)

11.1.3 General flowchart



*186 This is done by the light pen interrupt processing routine.

*187 The following interrupt is not enabled unless any data is written to I/O port address 03BH.

11.2 MASKI (Address: 0F65AH)

11.2.1 Function

This routine sets or resets the interrupt mask for the device specified by the entry parameters.
The following devices can be specified.

a) Software timer 1

b) Software timer 2

c) Light pen (option)

d) IEEE-488 I/F card (option)

e) Optical fiber I/F card (option)

f) A/D, D/A converter card (option)

g) RS232C I/F card (option)

h) MultiFonts CG ROM card

To perform interrupt processing for any of these devices, the corresponding interrupt mask must be reset (unmasked). Devices "d" through "h" are inserted in option slots and the interrupt level differs according to the slot in which the device is installed. However, the MASKI routine resets the mask regardless of which slot contains the card. Furthermore, the slot number can be determined from the interrupt level returned in the return information.
If the specified card is not installed, no option card error is returned and the mask is not reset.
The interrupt mask can be set by specifying the information returned when the mask is reset as entry parameters.
The routine does the following to check whether or not the specified device is installed.

(1) Resets all interrupt masks for option devices.

(2) Writes data to the I/O port address to which the specified device is assigned. (See to Appendix D for the I/O port addresses.) The I/O port address is returned to register C when this routine is called to reset the interrupt mask.

(3) When the device is installed, an interrupt occurs; otherwise, no interrupt occurs.

11.2.2 Call Sequence

(1) Mask Reset (enable interrupt)

Entry parameters:

Register B = 0

Register C = Device

0AH : Software timer 1

0BH : Software timer 2

03BH : Light pen

091H : IEEE-488 I/F card

098H : Optical fiber I/F card

0A3H : A/D-D/A converter card

0ACH : RS-232C option 1

0CCH : RS-232C option 2

0FDH : MultiFonts CG ROM card

Example: Resetting the light pen interrupt mask

```
MASKI    EQU    0F65AH
         .
         .
         LD     BC,03BH       ;REG B=0,REG C=03BH
         CALL   MASKI         ;ENABLE LIGHT PEN.
         JP     NZ,ERROR      ;IF ERROR.
         .
         .
```

Return information:

Z flag = 1: Normal completion

Register A = Detailed return information

1 : Option interrupt level 1, slot unknown

2 : Option interrupt level 2, slot unknown

3 : Option interrupt level 3, slot 1

4 : Option interrupt level 4, slot 2

5 : Option interrupt level 5, slot 3

6 : Option interrupt level 6, slot 4

7 : Option interrupt level 7, slot 5

9 : Software timer 1 (fast)

0DH : Software timer 1 (slow)

020H : Light pen

The contents of register A are used as an entry parameter when this routine is called to set the mask. The contents of other registers are changed.

Z flag = 0: Abnormal completion

Register A = Error information

1 : Specified card not installed.

4 : Parameter error
Device specification is incorrect.

The contents of other registers are changed.

(2) Mask set (disabling interrupt)

Entry parameters:

Register B = Interrupt level (return information from mask reset.

1 :	Option interrupt level 1, slot unknown
2 :	Option interrupt level 2, slot unknown
3 :	Option interrupt level 3, slot 1
4 :	Option interrupt level 4, slot 2
5 :	Option interrupt level 5, slot 3
6 :	Option interrupt level 6, slot 4
7 :	Option interrupt level 7, slot 5
9 :	Software timer 1 (fast)
0DH :	Software timer 1 (slow)
020H :	Light pen
0FFH :	Sets interrupt masks of all option interrupt levels and light pen except software timers 1 and 2.

Example: Setting and resetting the optical fiber I/F card interrupt mask

```
MASKI    EQU    0F65AH
         .
         .
         LD     BC,098H        ;
         CALL   MASKI          ;ENABLE OPTICAL I/F
         JP     NZ,ERROR       ;IF ERROR.
         LD     (ILEVEL),A     ;KEEP INT. LEVEL.
         .                     ;
         LD     A,(ILEVEL)     ;PIC INT. LEVEL.
         LD     B,A            ;SET MASKI PARAM.
         CALL   MASKI          ;DISABLE OPTICAL I/F.
         JP     NZ,ERROR       ;IF ERROR.
         .
```

Return information:

Z flag = 1: Normal completion

The contents of other registers are also changed.

Z flag = 0: Abnormal completion

Register A contains detailed error information.

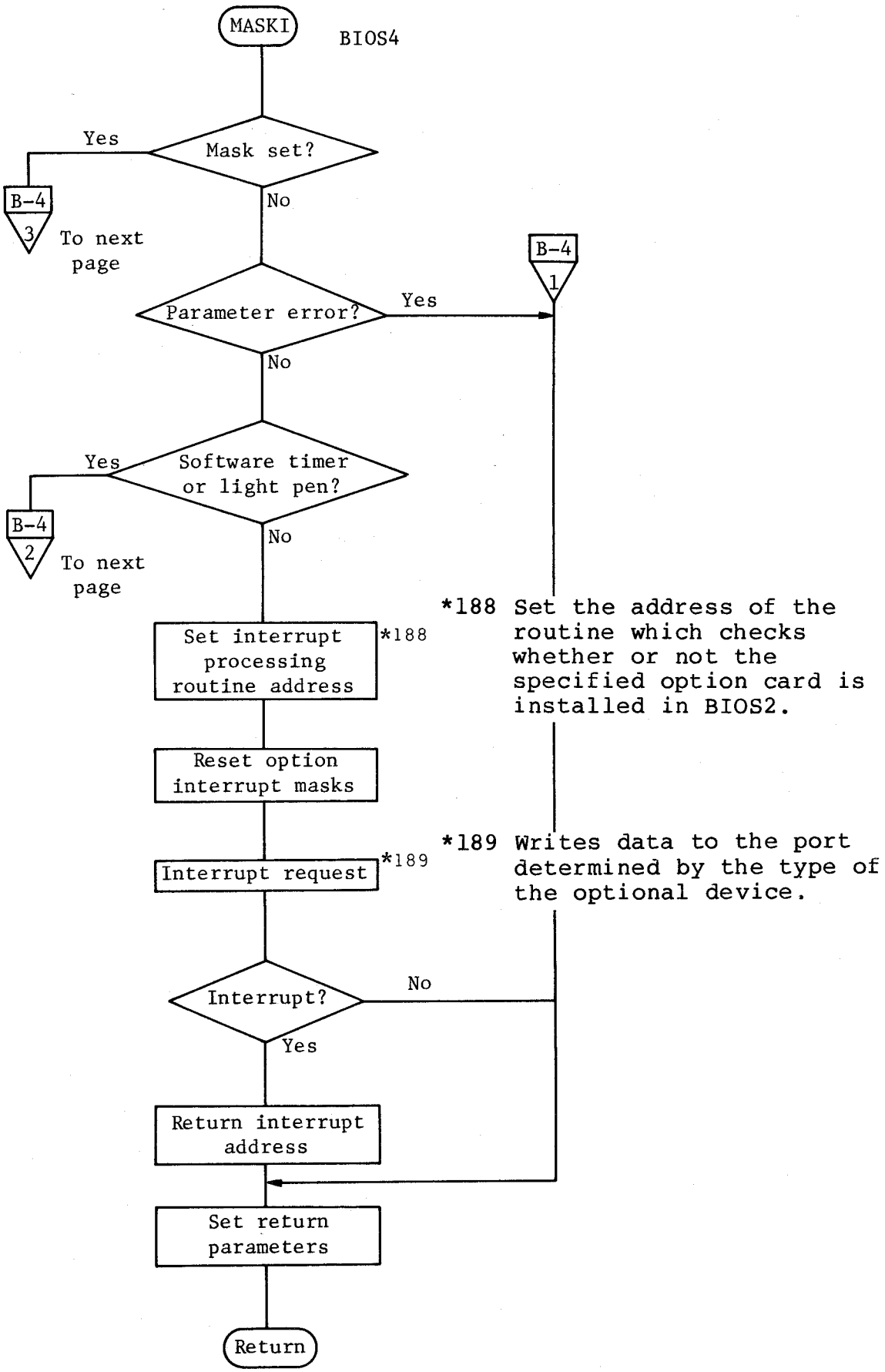
A = 4: Parameter error
The value in register B is incorrect.

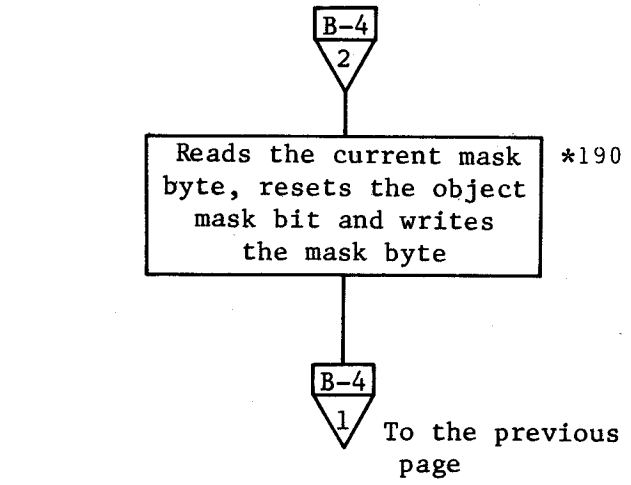
The contents of other registers are also changed.

Note:

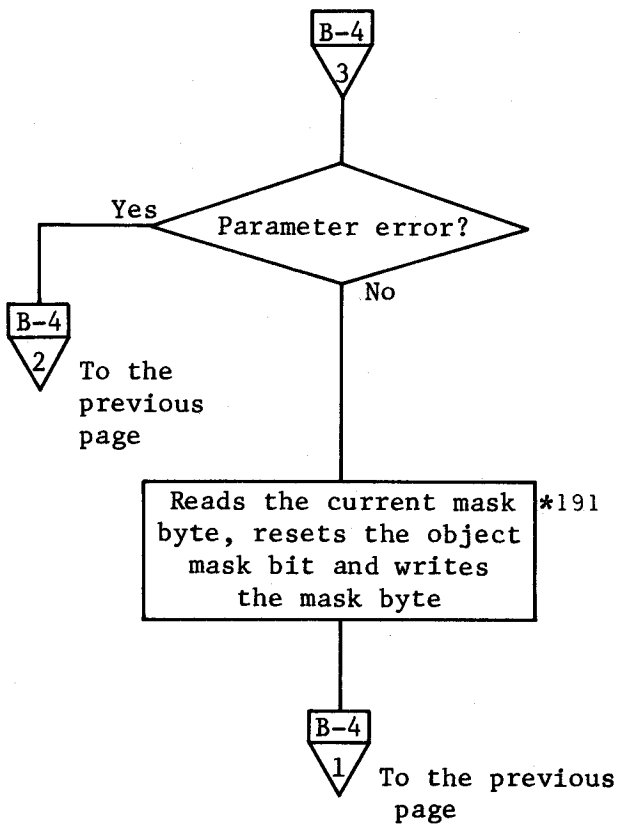
When 0FFH is specified as the input parameter in register B, masks for all option interrupt levels and light pen can be set at once. However, software timer 1 or 2 is masked by calling MSKI after setting each parameter. Masks are not set if a parameter error occurs.

11.2.3 General flowchart





*190: Master interrupt controller (for option levels 1 and 2, software timer 1 and light pen) is assigned to I/O port 09H and the slave (for option levels 3 to 7) is assigned 0DH.



*191 See *190.

§12 LOADX, STORX, LDIRX, JUMPX, CALLX

12.1 LOADX (Address 0F65DH)

12.1.1 General

This routine sets the contents of the specified address in the C register from the memory bank specified in the entry parameter, then returns. This makes it possible to reference data in a memory bank other than that in which the calling program is running. If the specified memory bank RAM is not installed, this routine generates an error and returns.

12.1.2 Call procedure

Entry parameters:

Register C: Number of the memory bank containing the data to be referenced.

- 0 : Main bank
- 1 : User bank #1
- -1 : System bank
- 2 : User bank #2 (option RAM)

Register HL : Data address

Example: The following program reads the contents of address 0100H of a user bank.

```
LOADX   EQU     0F65DH
        .
        .
        LD      C,1             ;SELECT USER BANK.
        LD      HL,0100H        ;ADDRESS=0100H
        CALL    LOADX           ;READ DATA.
        OR      A               ;CHECK RETURN CODE.
        JP      NZ,ERROR        ;IF ERROR.
        .
        .
```

Return information:

Register A = 0 Normal completion

Register C = Data

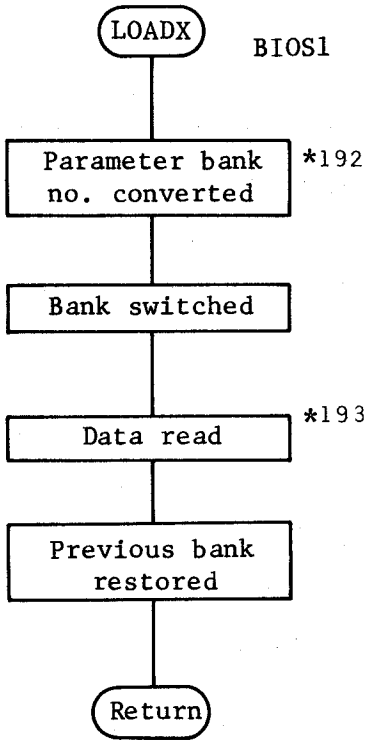
Register A ≠ 0 Abnormal completion

Parameter error (value set in C is not correct) or RAM is not installed.

Except for registers A and C, no registers are changed by calling this routine.

This routine is used when the I/O byte in the main bank is read by BIOS in the system bank.

12.1.3 General flowchart



*192 Subroutine SMBANK called to convert the parameter bank number into a hardware bank number.

*193 Data read from the address specified in the entry parameter.

12.2 STORX (Address 0F660H)

12.2.1 General

This routine stores the byte of data in register A in the address of the memory bank specified in the entry parameters. This makes it possible to transfer data from the bank in which the calling program is running to another memory bank. If the specified memory bank RAM is not installed, this routine generates an error and returns.

12.2.2 Call procedure

Entry parameters:

Register A = Data to be written

Register C = Number of the memory bank into which data is to be written

- 0: Main bank
- 1: User bank #1
- -1: System bank
- 2: User bank #2 (option RAM)

Register HL = Address in the specified memory bank

Example: The following program writes the code 0FFH into address 0200H of a user bank from the main bank.

```
STORX   EQU     0F660H
        .
        .
        LD      A,0FFH          ;SET WRITE DATA.
        LD      C,1             ;SELECT USER BANK.
        LD      HL,0200H        ;ADDRESS=0200H
        CALL    STORX           ;WRITE.
        OR      A               ;CHECK RETURN CODE.
        JR      NZ,ERROR        ;IF ERROR.
```

Return information:

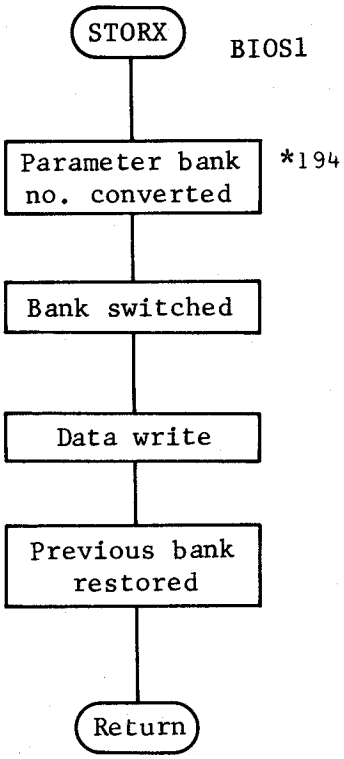
Register A = 0 : Normal completion

Register A ≠ 0 : Abnormal completion

Parameter error (value set in C not correct) or RAM is not installed.

Except for register A, no registers are changed by calling this routine. This routine is used by BIOS when transferring data from the system bank to a user program in the main bank.

12.2.3 General flowchart



*194 Subroutine SMBANK called to convert the parameter bank number into a hardware bank number.

12.3 LDIRX (Address 0F663H)

12.3.1 General

This routine transfers data from one memory bank to another according to the bank number, data address, and number of data bytes specified in the entry parameters. Since the registers are not sufficient for all the parameter information, a memory bank is established in the BIOS common data area.

This routine makes it possible to transfer programs from the main bank to a user bank by passing control to the loaded program after transfer. By doing this, it allows programs which are too large for the main bank to be executed.

However, programs loaded are handled as subroutines, and it is the responsibility of the user's main program to load the program into memory. Linkage with subroutines is accomplished by calling them with the CALLX routine (described later). For details, see 2.1, "Using the User Banks" in Chapter 4 for details.

12.3.2 Call procedure

Entry parameters:

Register BC = Number of bytes of data to be transferred

Register DE = Starting address of the area to which data is to be transferred

Register HL = Starting address of the area from which data is to be transferred.

Parameter specifications in BIOS common area:

MBANKS (address 0FEF1H, 1 byte) = No. of the memory bank containing the data to be transferred

MBANKD (address 0FEF2H, 1 byte) = No. of the memory bank to which the data is to be transferred

The memory bank numbers are specified as follows.

Bank No. 0 : Main bank

1 : User bank #1

-1 : System bank

2 : User bank #2 (option RAM)

Example: The following program transfers 4096 bytes of data from address 05000H in the main bank to the area beginning at address 0100H of the user bank.

```
LDIRX    EQU    0F663H
MBANKS   EQU    0FEF1H
MBANKD   EQU    0FEF2H
         .
         .
         LD     BC,01000H     ;DATA LENGTH=4096
         LD     DE,0100H      ;SEC. ADDR.
         LD     HL,05000H     ;DST. ADDR.
         LD     A,0           ;
         LD     (MBANKS),A    ;SEC. BANK NO.
         INC    A             ;
         LD     (MBANKD),A    ;DST. BANK NO.
         CALL   LDIRX         ;
         OR     A             ;CHECK RETURN CODE.
         JP     NZ,ERROR      ;IF ERROR.
         .
         .
```

Return information:

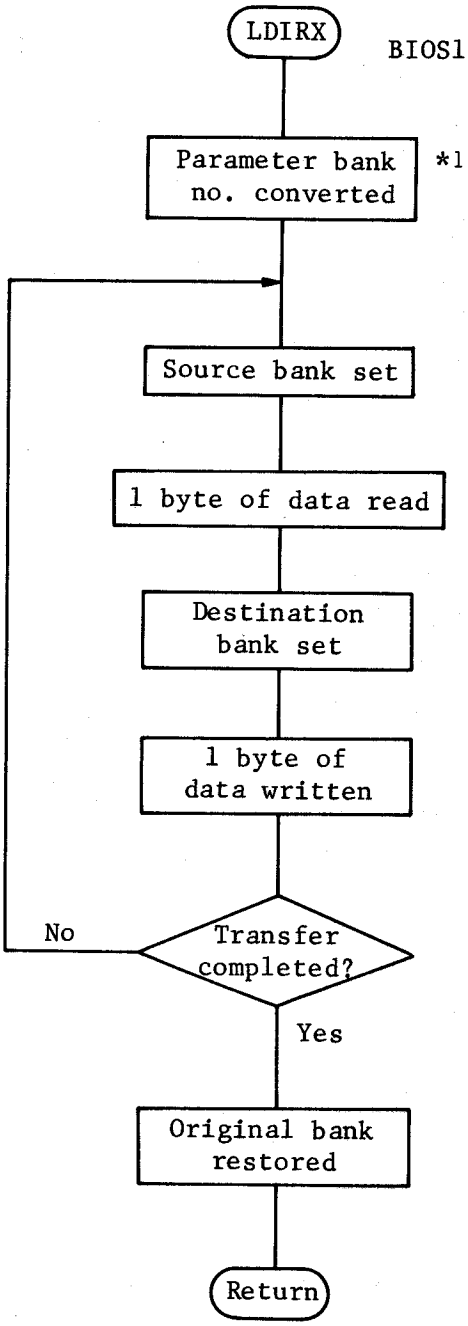
Register A = 0 : Normal completion

Register A ≠ 0 : Abnormal completion

Parameter error (incorrect bank number specified) or option RAM not installed.

Contents of other registers are also changed.

12.3.3 General flowchart



*195 Subroutine SMBANK called to convert the parameter bank number to a hardware bank number.

12.4 JUMPX (Address 0F666H)

12.4.1 General

This routine causes execution to jump to the address of the memory bank specified by the entry parameters without changing the contents of any of the registers. It is used when a subroutine call is not required. However, if the stack is to be used by the program to which the jump is made, it must be reset.

This routine transfers control by means of the JP instruction, not the CALL instruction.

12.4.2 Call procedure

Entry parameters

Register IX: Jump address

MBANKD: (address 0FEF2H in BIOS common data area)
Jump destination memory bank number.

The memory bank numbers are as follows.

MBANKD = 0 : Main bank

MBANKD = 1 : User bank #1

MBANKD = -1 : System bank

MBANKD = 2 : User bank #2 (option RAM)

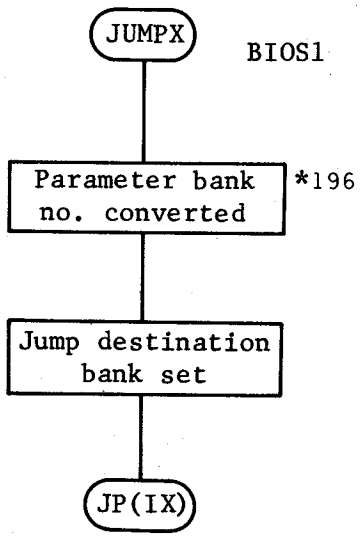
Example: The following causes execution to jump to address 0500H in a user bank.

```
JUMPX   EQU     0F666H
        .
        .
        LD      A,1             ;SELECT USER BANK.
        LD      (0FEF2H),A      ;
        LD      IX,0500H        ;SET JUMP ADDR.
        JR      JUMPX           ;JUMP!
        .
        .
```

Return information: None (control is not returned).

The contents of all registers remain as they were when control was passed to JUMPX.

12.4.3 General flowchart



*196 Subroutine SMBANK called.

12.5 CALLX (Address 0F669H)

12.5.1 General

This routine makes a subroutine call to the address in the memory bank specified in the entry parameters. When making a subroutine call with the CALLX routine, the following conditions must be satisfied in order to assure that the call is made and that control returns to the calling program properly after subroutine execution has been completed.

(1) The stack pointer must be located in the common data area between addresses 0E000H and 0FFFFH in memory. However, since BIOS1 is resident between addresses 0F600H to 0FEFFH, this area is not available for use.

(2) Two bytes are required for the stack; be sure to leave enough memory available for this purpose.

(3) The number of the bank containing the program called must be set in the BIOS common data area.

12.5.2 Call procedure

Entry parameters

Register IX: Address called

MBANKD (address 0FEF2H in the BIOS common data area):
Number of the bank containing the calling program.

The memory bank number is specified as follows.

MBANKD = 0 : Main bank
1 : User bank #1
-1 : System bank
2 : User bank #2 (option RAM)

Example: The following makes a subroutine call to address 0100H in a user bank.

```
CALLX    EQU    0F669H
MBANKD   EQU    0FEF2H
STACK    EQU    0
SUB1     EQU    0100H
         LD     SP,STACK      ;SET STACK POINTER.
         LD     IX,SUB1       ;SET SUBROUTINE ADDER.
         LD     A,1           ;SET BANK NUMBER.
         LD     (MBANKD),A    ;
         CALL   CALLX         ;EXEC. SUBROUTINE.
         .
         .
```

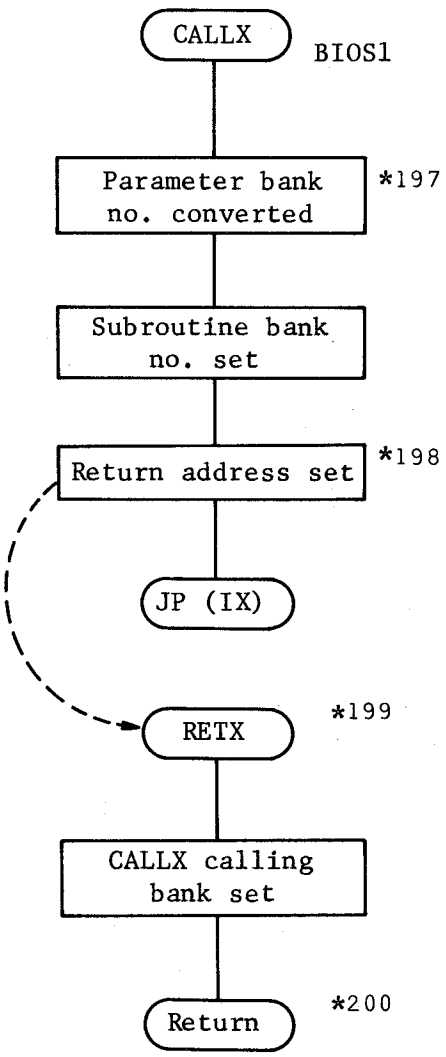
Return from subroutine SUB1 to the user program is made by an ordinary RET instruction.

Return information:

Return information depends on the subroutine called. When control passes to the user subroutine, all registers other than IY are set with the values they contained before the call was made. Therefore, return information can be set in any registers except IX and IY.

Data in MBANKD (address 0FEF2H) is lost when CALLX is executed; there, this data must be set again the next time the CALLX routine is executed.

12.5.3 General flowchart



*197 Subroutine SMBANK called.

*198 Sets the address to which control is to be returned by the RET instruction in the subroutine.

*199 Return from the user subroutine by the RET instruction.

*200 Return to the point from which CALLX was called.

§13 GETPFK, PUTPFK

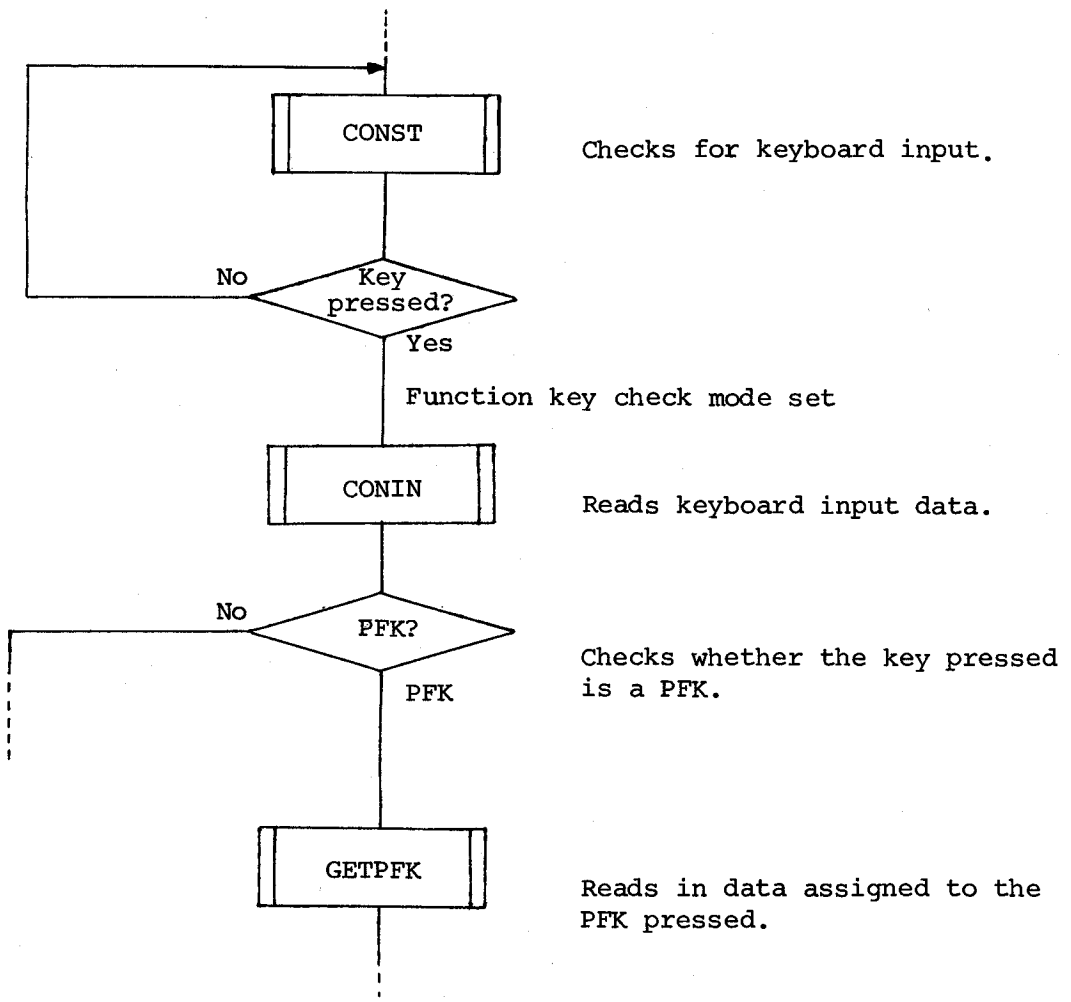
13.1 GETPFK (Address 0F66CH)

13.1.1 General

This routine returns the character string assigned to the programmable function key with the PFK number specified in the entry parameter. Although programmable function keys are ordinarily used for storing commonly used commands, this function allows them to be used for holding data (up to 15 bytes) for processing by the program. Thus, commonly used data can be input just by pressing the applicable function key; this helps to reduce the amount of keyboard work required.

Data assigned to a programmable function key should be read by the GETPFK routine, and not CONIN. The reason for this is that CONIN cannot normally differentiate between input by the PFKs and other keys. Further, if PFKs assignments are made using the PFKSET routine, the assignments will be effective whenever the system is started up. Also, remember that different mode settings are used for reading the PFK assignments for the MFBASIC mode and other modes. The MFBASIC mode is that in which the MFBASIC PFK assignments are read.

The following sequence is used for reading PFK key data.



13.1.2 Call procedure

Entry parameters:

Register B = Mode setting as follows

0 = Reads PFK assignments for other than the MFBASIC mode.

1 = Reads PFK assignments for the MFBASIC mode.

Register C = PFK number (0 to 9)

Register HL = Starting address of the buffer into which data is to be read.

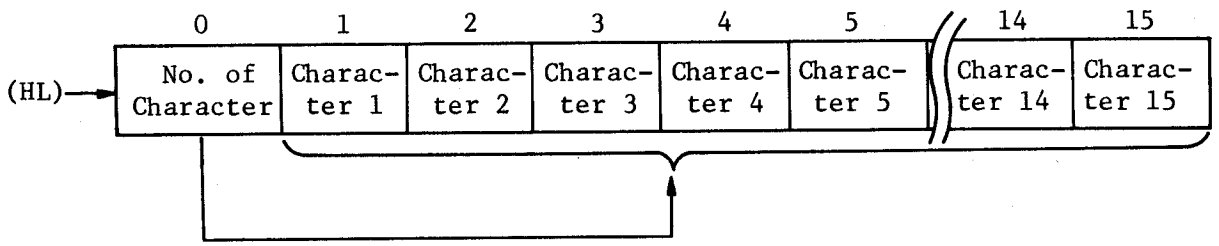
Example: The following sequence reads the contents of PFK 2 in the MFBASIC mode.

```
GETPFK   EQU    0F66CH
         .
         .
         LD     B,1          ;MFBASIC
         LD     C,1          ;PFK NO. 2
         LD     HL,BUFF      ;BUFFER ADDRESS.
         CALL   GETPFK       ;
         .
         .
BUFF:    DS     16
         .
         .
```

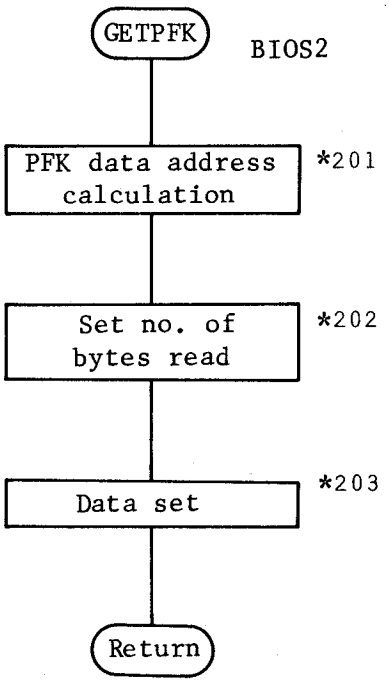
Return information: None

Contents of all registers except HL are changed.

Result:



13.1.3 General flowchart



*201 For other than the MFBASIC mode, the starting address of the PFK data is 0A100H; for the MFBASIC mode, it is 0A200H (these addresses are in the system bank).

*202 Subroutines SRMBANK and LOADX called.

*203 Subroutines SRMBANK and LDIRX called.

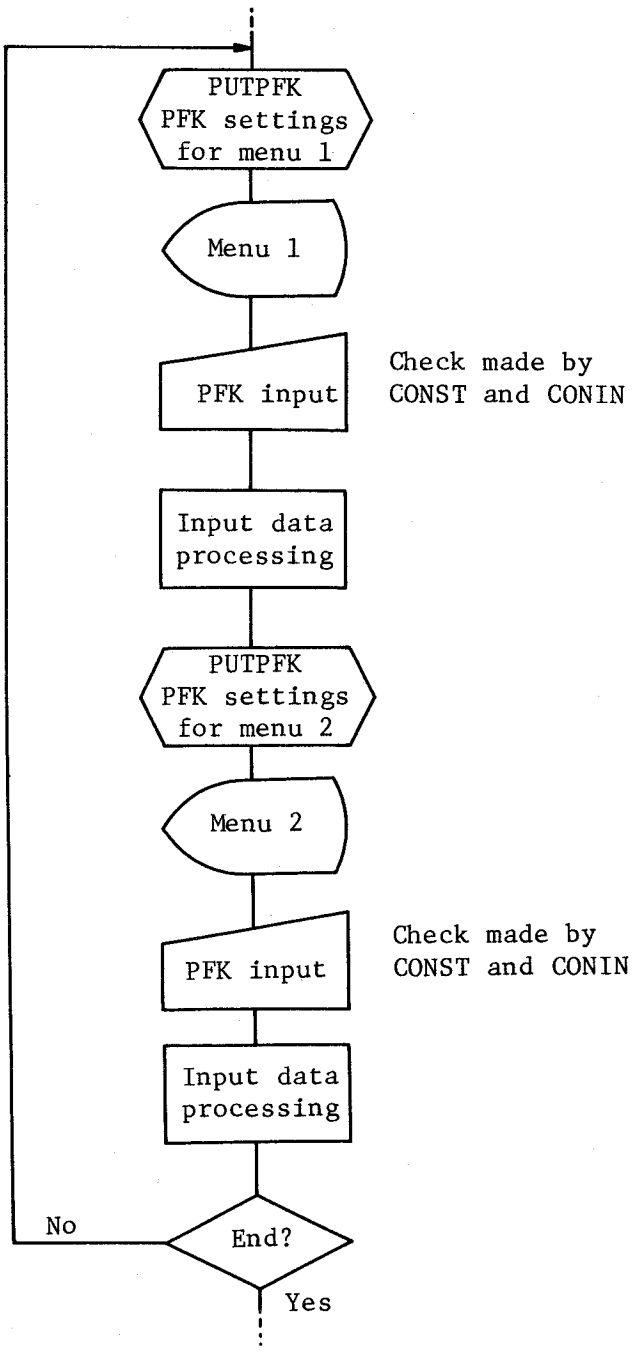
13.2 PUTPFK (Address 0F66FH)

13.2.1 General

This routine assigns character strings to the programmable function keys. Since calling this routine makes it possible to change the PFK assignments dynamically, it can be used as if there were an unlimited number of programmable function keys. Therefore, it is extremely useful when a number of menu screens are used.

A maximum of 15 characters may be assigned to any given PFK with this routine. These characters can be read in with the GETPFK routine described previously.

An example of processing procedures for use with menus is shown below.



13.2.2 Call procedure

Entry parameters:

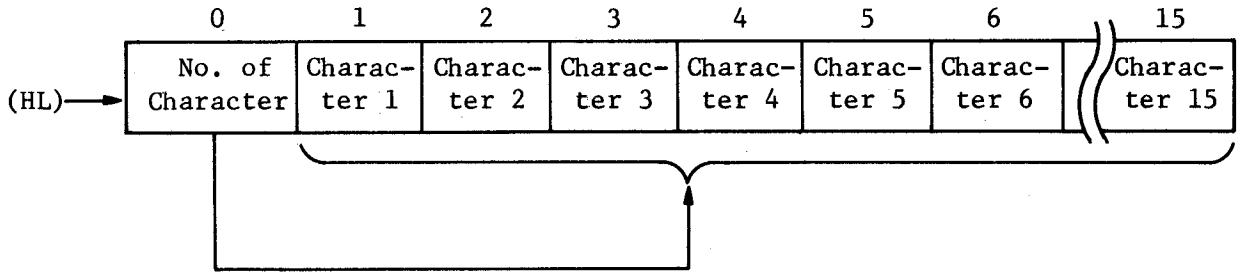
Register B = Mode setting as follows

0 = Makes PFK assignments for other than the MFBASIC mode.

1 = Makes PFK assignments for the MFBASIC mode.

Register C = PFK number (0 to 9)

Register HL = Starting address of data to be assigned to PFK.
(The data must be in the format shown below, and must not include more than 15 characters.)



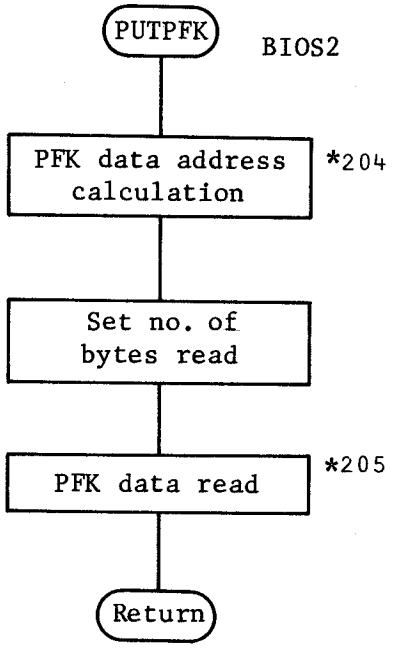
Example: The following assigns the character string "RETURN" to PFK10 for the non-MFBASIC mode.

```
PUTPFK    EQU     0F66FH
          .
          .
          LD      B,0             ;NON MFBASIC MODE
          LD      C,9             ;PFK 10
          LD      HL,PFDATA       ;DATA ADDRESS
          CALL    PUTPFK          ;SET PFK.
          .
          .
PFDATA:   DB      PFDEND-$-1      ;DATA LENGTH
          DB      'RETURN'        ;PFK DATA
PEDEND    EQU     $               ;
          .
          .
```

Return information: None

Contents of all registers other than HL are changed.

13.2.3 General flowchart



*204 For other than the MFBASIC mode, the starting address of the PFK data is 0A100H; for the MFBASIC mode, it is 0A200H (these addresses are in the system bank).

*205 Subroutines SRMBANK and LDIRX called.