SECTION 2 SOFTWARE

CHAPTER 1 GENERAL

CHAPTER 1 GENERAL

1.1 Descriptive Expressions Used in This Manual

The HX-20 incorporates two HD6301 microprocessors. One of the microprocessors has a 64K-byte memory area to control the entire HX-20 components and is called the master MCU (Micro Computer Unit). The other plays an auxiliary role. Namely, it controls I/O devices such as the microprinter, cassettes, etc., and is called the slave MCU. Each MCU has a CPU, a ROM, a RAM, a serial I/O port, a parallel I/O port, and timer function. Fig. 1-1 shows the arrangement of the registers in the CPU.

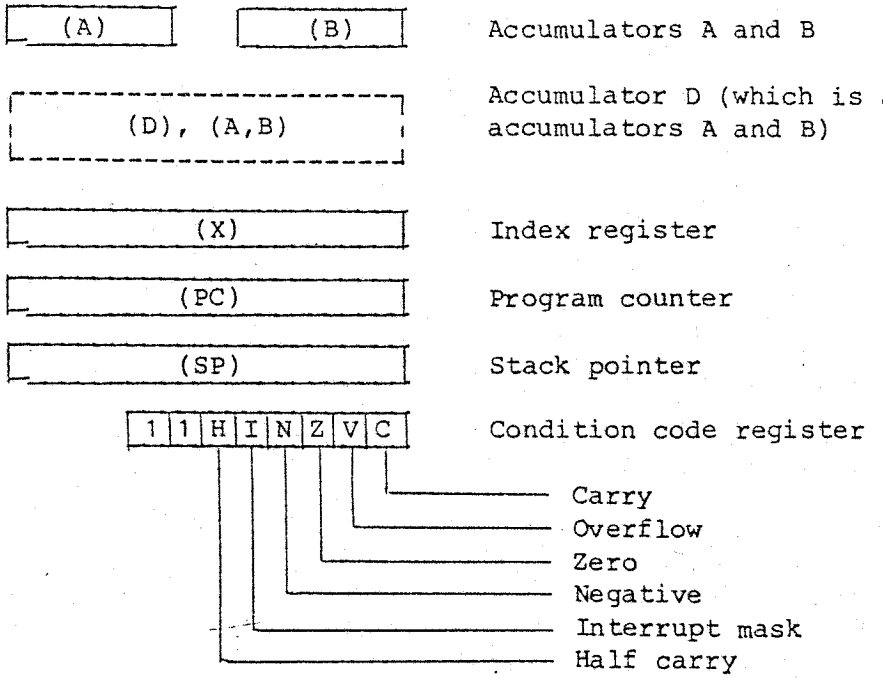


Fig. 1-1 Arrangement of CPU Registers

The registers are identified by symbols: (A) for accumulator A, (B) for accumulator B, (D) or (A,B) for accumulator D, (X) for the index register, (PC) for the program counter, and (SP) for the stack pointer. For the condition code register, (H), (I), (N), (Z), (V), and (C) are used to indicate the respective bits as shown in Fig. 1-1. As shown in Fig. 1-2, bit positions are indicated from the bit lowest place with the lowest place value (or weighting) at the extreme right such as bit 0, bit 1 ... This bit with the lowest place value is called LSB (Least Significant Bit), while the bit with the highest place value (at the extreme left) is called MSB (Most Significant Bit).

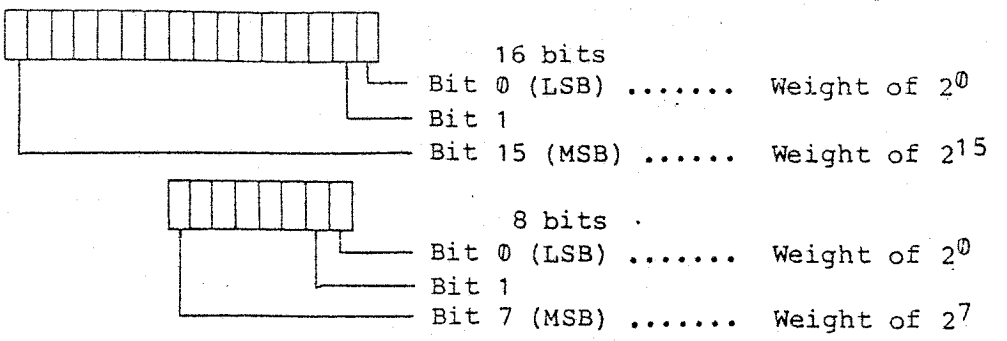


Fig. 1-2 Bit Positions

As various number systems are in use, such as binary, decimal, hexadecimal, etc., the base or radix of a number system is placed as a subscript to the lower right of a number to identify whether the number is decimal, hexadecimal, octal, or binary as shown in Fig. 1-3.

1001_2 (Binary 1001)
10_8 (Octal 10)
10_{10} (Decimal 10)
10_{16} (Hexadecimal 10)

Fig. 1-3 Number Notation (with Base m)

Unless otherwise specified, the hexadecimal notation is used throughout this manual to express the contents of memory and registers which are represented by binary numbers. Characters represented by ASCII codes may be enclosed by single quotation marks (ex: 'ABCD'). Symbol Δ represents a space.
The MCU is provided with the internal registers shown in the following Table. The registers are abbreviated as follows.

Table 1-1 Internal Registers

Address	Register	Abbreviation
00	Port 1 Data direction register	DDR1
01	Port 2 Data direction register	DDR2
02	Port 1 Data register	PORT 1
03	Port 2 Data register	PORT 2
04	Port 3 Data direction register	DDR3
05	Port 4 Data direction register	DDR4
06	Port 3 Data register	PORT 3
07	Port 4 Data register	PORT 4
08	Timer control and status register	TCSR
09	Counter (high-order bytes)	FRC
0A	Counter (low-order bytes)	
0B	Output compare register (high-order bytes)	OCR
0C	Output compare register (low-order bytes)	
0D	Input capture register (high-order bytes)	ICR
0E	Input capture register (low-order bytes)	
0F	Port 3 control and status register	
10	Rate and mode control register	RMCR
11	Transmit/receive control and status register	TRCSR
12	Receive data register	RDR
13	Transmit data register	TDR
14	RAM control register	
15 to 1F	Reserved	

Table 2 lists the abbreviations for the respective bits of each internal register.

Table 2 Bits of Internal Registers and Their Abbreviations

Address register	Bit	Abbreviation
02 Port 1	0	P10
	1	P11
	2	P12
	3	P13
	4	P14
	5	P15
	6	P16
	7	P17
03 Port 2	0	P20
	1	P21
06 Port 3	0	P30
	1	P31
	2	P32
	3	P33
	4	P34
	5	P35
	6	P36
	7	P37
07 Port 4	0	P40
	1	P41
	2	P42
	3	P43
	4	P44
	5	P45
	6	P46
	7	P47
08 TCSR	0 (Output level)	OLVL
	1 (Input edge)	IEDG
	2 (Enable timer overflow interrupt)	ETOI
	3 (Enable output compare interrupt)	EOCI
	4 (Enable input capture interrupt)	EICI
	5 (Timer overflow flag)	TOF
	6 (Output compare flag)	OCF
	7 (Input capture flag)	ICF
11 TRCSR	0 (Wake up)	WU
	1 (Transmit enable)	TE
	2 (Transmit interrupt enable)	TIE
	3 (Receive enable)	RE
	4 (Receive interrupt enable)	RIE
	5 (Transmit data register empty)	TDRE
	6 (Overrun framing error)	ORFE
	7 (Receive data register full)	

1.2 Sample Program Format
Table 1-3 shows the standard format of a sample program.

Table 1-3 Standard Format of Sample Program

Column number	Description	
1 to 5	Error message flag	
6 to 10	Source line number (decimal)	
11	Location counter section flag (see Note 1.)	
13 to 16	Location counter value (hexadecimal)	
18 to 19	Operation code (hexadecimal)	
21 to 28	Non-branch instructions	
	21 to 22	First byte of operand
	23 to 24	Second byte of operand
Part which depends on instructions	26	Section flag of operand (See Note 2.)
	Branch instructions	
	21 to 22	Relative address of branch destination
	24 to 27	Location address of branch destination
	28	Branch error warning flag ("*" is displayed if relative address is in the range $70 to $90.)
30 to 31	M1, M2, M3, M4: Macroexpansion statements (The digit following "M" indicates the nesting level.) IF: Statement skipped by the IF control instruction	
35 to 40	Label field	
42 to 47	Operation field	
49 to 56	Operand field	
58 to last column	Comment field	

```
          *** 6301 CROSS MACROASSEMBLER   VER1.0 ***                                    09/24/82   09:33:20      PAGE   1
Heading
---> ERR   SEQ     LOC  OBJECT      PROGRAM  MULT16       *** 16 BITS UNSIGNED MULTIPLY ***

         00001                                    NAM    MULT16
         00002                                    TTL    *** 16 BITS UNSIGNED MULTIPLY ***
         00003                             *
         00004                                    OPT    PAGE=55
         00005A  1000                             ORG    $1000
         00006                             *
         00007                             * 16 BIT UNSIGNED MULTIPLY (16 BIT RESULT)
         00008                             *  REENTRANT CODE (USES 6 BYTES ON STACK)
         00009                             *
         00010                             *  A,B TIMES  X  RESULT  A,B
         00011                             *
         00012A  1000 37                   MPY16  PSH B
         00013A  1001 36                          PSH A
         00014A  1002 3C                          PSHX
         00015A  1003 30                          TSX
         00016                             *  STACK NOW LOOKS LIKE
         00017                             *  +0  MS BYTE MULTIPLICATION          *     A B
         00018                             *  +1  LS BYTE                         *   * C D
         00019                             *  +2  MS BYTE MULTIPLIER              *  -----------
         00020                             *  +3  LS BYTE                         *        BD
         00021A  1004 A6 02    A                  LDA A  2,X      *  A * D      *     AD
         00022A  1006 E6 01    A                  LDA B  1,X      *             *     BC
         00023A  1008 3D                          MUL             *             *   AC
         00024A  1009 37                          PSH B           *             *  -----------
         00025A  100A A6 03    A                  LDA A  3,X      *  B * C      *    X Y Z
         00026A  100C E6 00    A                  LDA B  0,X      *
         00027A  100E 3D                          MUL             *
         00028A  100F 37                          PSH B           *
         00029A  1010 A6 03    A                  LDA A  3,X      *  B * D
         00030A  1012 E6 01    A                  LDA B  1,X      *
         00031A  1014 3D                          MUL             *
         00032A  1015 30                          TSX             *
         00033A  1016 AB 00    A                  ADD A  0,X
         00034A  1018 AB 01    A                  ADD A  1,X
         00035A  101A 38                          PULX            * CLEAN UP STACK
         00036A  101B 38                          PULX
         00037A  101C 38                          PULX
         00038A  101D 39                          RTS
         00039              0000   A              END
***** TOTAL ERRORS       0
```

Annotations: Title; Heading; Line No.; Location; Operation code; Operand; Label field operation; Operand field; Ccomment field

Fig. 4 Example of Program List Format

1.3 How to Read Subroutine Lists

The subroutine lists in each chapter contain the subroutine names, entry points, descriptions of subroutines, and parameters. The parameters shown are divided into those to be output for subroutine call and those to be input for subroutine return. In describing the CPU registers, symbols are used: (A) for accumulator A, (B) for accumulator B, and (X) for the index register. For the condition code register, (C) stands for carry, (N) for a negative flag, and (Z) for a zero flag. Details of registers are listed under "Registers retained". "Subroutines referenced" lists the subroutines called in the course of execution. The I/O routines normally use addresses 0050 to 0077 as a work area. The actual locations used are represented as variables. (See Chapter 13.)

"(C): Abnormal I/O flag" appears quite often in the description of parameters at the time of subroutine return. This indicates that the I/O operation has not been performed correctly due to a drop in voltage, the power switch being turned OFF, or the BREAK key being pressed. (C)=1 indicates abnormal I/O operation and (C)=0 indicates normal I/O operation.

```
        *** 6301 CROSS MACROASSEMBLER   VER1.0 ***                              10/21/82    09:41:24

ERR   SEQ   LOC  OBJECT      PROGRAM  MULT16     --- 16 BITS UNSIGNED MULTIPLY ---


     00001                              NAM     MULT16
     00002                              TTL     --- 16 BITS UNSIGNED MULTIPLY ---
     00003                         *
     00004                         * FILE NAME  'EX$3'  BY K.A
     00005                              OPT     PAGE=55
     00006                              OPT     LOAD
     00007                         *
     00008A 1000                        ORG     $1000
     00009                         *
     00010                         * 16 BIT UNSIGNED MULTIPLY (16 BIT RESULT)
     00011                         *  REENTRANT CODE (USES 6 BYTES ON STACK)
     00012                         *
     00013                         *  A,B TIMES  X  RESULT  A,B
     00014                         *
     00015A 1000 37                MPY16  PSH B
     00016A 1001 36                       PSH A
     00017A 1002 3C                       PSHX
     00018A 1003 30                       TSX
     00019                         *  STACK NOW LOOKS LIKE
     00020                         *  +0  MS BYTE MULTIPLICATION          *       A B
     00021                         *  +1  LS BYTE                         *     * C D
     00022                         *  +2  MS BYTE MULTIPLIER              *  -----------
     00023                         *  +3  LS BYTE                         *         BD
     00024A 1004 A6 02     A              LDA A  2,X       *  A * D       *       AD
     00025A 1006 E6 01     A              LDA B  1,X       *      -       *       BC
     00026A 1008 3D                       MUL              *              *    AC
     00027A 1009 37                       PSH B            *              *  -----------
     00028A 100A A6 03     A              LDA A  3,X       *  B * C       *     X Y Z
     00029A 100C E6 00     A              LDA B  0,X       *
     00030A 100E 3D                       MUL              *
     00031A 100F 37                       PSH B            *
     00032A 1010 A6 03     A              LDA A  3,X       *  B * D
     00033A 1012 E6 01     A              LDA B  1,X       *
     00034A 1014 3D                       MUL              *
     00035A 1015 30                       TSX              *
     00036A 1016 AB 00     A              ADD A  0,X
     00037A 1018 AB 01     A              ADD A  1,X
     00038A 101A 38                       PULX             * CLEAN UP STACK
     00039A 101B 38                       PULX
     00040A 101C 38                       PULX
     00041A 101D 39                       RTS
     00042          0000   A              END
***** TOTAL ERRORS     0
```

CHAPTER 2 INPUT FROM KEYBOARD

2.1 General

The keyboard, connected to the master MCU, has 8 lines each of which fetches 10 data. In other words, the keyboard is an 8 x 10 matrix structure. The pressed key can be found by inputting the data for each line. Interrupt IRQ1 occurs each time a key is pressed. The keyboard matrix incorporates the Printer ON/OFF and DIP switches in addition to the alphanumeric keys.
Key input is processed by interrupts and input data is stored in the 8-byte key stack. A power ON key stack, which stores data to be input automatically from the keyboard when the power is turned ON, is also provided. The contents of the power ON key stack are first fetched and the data in the key stack is input when the power ON key stack becomes empty. The contents of the power ON key stack can be restored by turning the power ON (reset).

2.2 I/O Ports for Keyboard Input

Table 2.1 shows the I/O ports related to the keyboard input.

Table 2.1 I/O Ports Related to Keyboard Input

Address	Bit position	Definition
20	0	Output Specifies scanning of L0 line. 0: Scanning enabled. 1: Scanning is not performed.
	1	Output L1
	2	Output L2
	3	Output L3
	4	Output L4
	5	Output L5
	6	Output L6
	7	Output L7
22	0	Input Scan result D0 0: ON 1: OFF
	1	Input Scan result D1 0: ON 1: OFF
	2	Input Scan result D2 0: ON 1: OFF
	3	Input Scan result D3 0: ON 1: OFF
	4	Input Scan result D4 0: ON 1: OFF
	5	Input Scan result D5 0: ON 1: OFF
	6	Input Scan result D6 0: ON 1: OFF
	7	Input Scan result D7 0: ON 1: OFF
26	4	Output Key input interrupt mask 0: Mask 1: Mask open
28	0	Input Scan result D8
	1	Input Scan result D9
P15		Input Key input interrupt flag 0: A keyboard input interrupt has occurred. 1: No keyboard input interrupt has occurred.

2.3 Key Scan

As shown in Fig. 2-1, ten data can be input from each of the eight lines connected to the keyboard. Line L0 inputs data from keys 0, 1, 2, 3, 4, 5, 6, 7, the PF1 key and DIP switch 1. In the same way, data from keys @, A, B, C, D, E, F, G, the PF3 key and DIP switch 3 are input through line L2. This means that to input all of the data from the keyboard, lines L0 to L7 must be selected in turn and the data fetch operation repeated eight times.

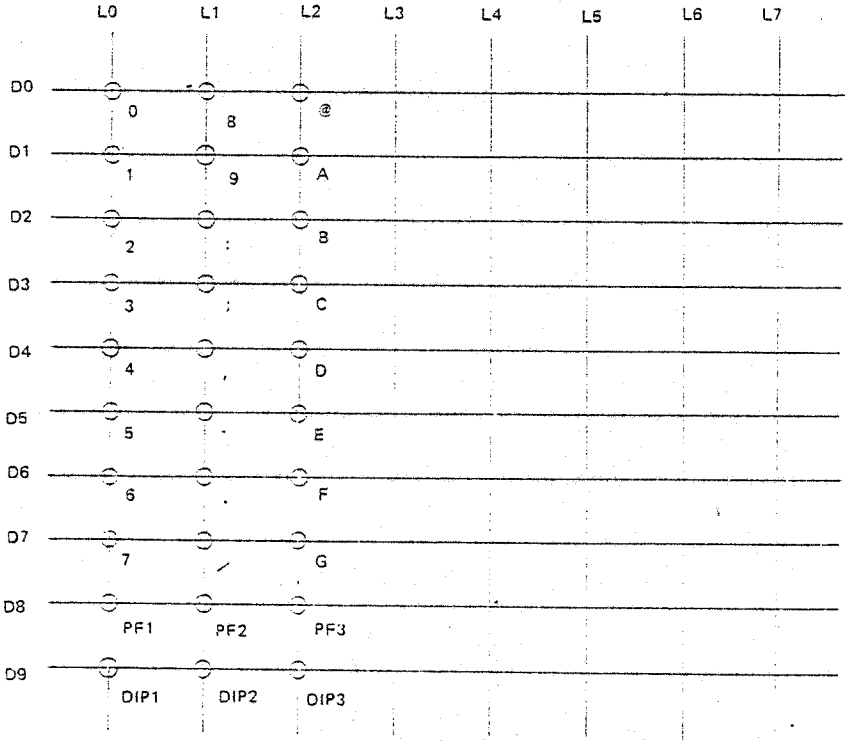


Fig. 2-1 Key Matrix

In some cases, data may not be input correctly from the keyboard due to this circuit configuration. For example, when keys 1, 8 and 9 are pressed, the circuit recognizes key 0 as having been pressed. That is, if key 9 is pressed while holding down keys 1 and 8, the input is recognized as 1, 8, 0. There are several such combinations which will cause incorrect data to be received. Key scan is performed by the following procedure.

(1) Close key input interrupt mask
P264 (bit 4 at address 26) is an IRQ1 key input interrupt mask. As an interrupt occurs if the key is pressed (i.e., the line to scan is specified and the key on the line is pressed) if this mask is open, the interrupt is disabled.

(2) Specify line to scan
There are 8 lines, L0 to L7, and any line can be specified for input. When line L0 is specified, the data on the line L0 can be input. If all lines L0 to L7 are specified, any key can be detected. The bit 0 at address 20 specifies line L0. When the

value of this bit is '0', scan is enabled and when '1', scan is not performed. The value FE (line L0 is scanned and the other lines are not scanned) is output first to address 20.

(3) Fetch data

When the contents of the address 22 are input, the data in D0 through D7 can be obtained. When the contents of address 28 are input, the data in D8 and D9 (bit 0, bit 1) can be obtained. (There is a wait of several tens of microseconds to obtain correct data after the line is specified.) Now, input from keys 0 to 7, PF1 and DIP switch 1 is enabled.

(4) Scan lines

Lines L1 through L7 are sequentially scanned and procedures 2 and 3 above are repeated. In this way all the data from the keyboard as well as the DIP switch values can be input. Table 2.2 shows the arrangement of the key matrix. Table 2.3 shows the arrangement of the keyboard matrix.

2.4 Keyboard input interrupt

An IRQ1 interrupt is enabled when the keyboard data is input. The following procedure is used to issue an IRQ1 interrupt.

(1) Specify the key line

The line where an interrupt occurs when a key is pressed is specified. Set '0' in address 20 to specify the key scan line. Once '0' is set, an interrupt occurs when any key is pressed. Note that the keys and switches on D9 such as DIP switches 1 to 4, shift key, control keys, and Printer ON/OFF switch are excluded from the keyboard input interrupt.

(2) Open the keyboard input interrupt mask

Write '1' to bit 5 of address 26 (P265) where the keyboard input interrupt mask is performed.

(3) Open the CPU interrupt mask

The CPU interrupt mask is opened by a CLI command.

(4) Confirm the keyboard interrupt

If the P15 is '0' when an IRQ1 interrupt occurs, it indicates the occurrence of the key input interrupt.

2.5 Timing of Key Input Process

An IRQ1 interrupt occurs when a key is input. Sampling (OCR interrupt) is performed using the MCU free running counter.

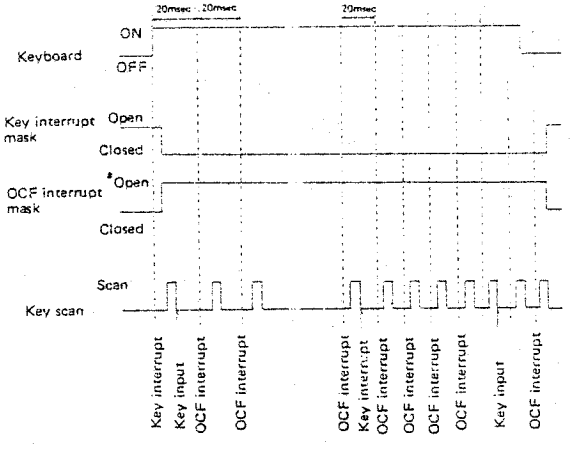


Fig. 2-2 Timing of Key Input

After a key is pressed as shown in Fig. 2-2, the Output Compare Register (OCR) issues interrupts at intervals of 20 msec (the key interrupt mask is closed) and auto repeat process is performed by key scan. If the same key is pressed continuously for a certain number of key scans after issuance of an OCF interrupt, it is assumed that the key has been newly pressed. The received data from the keyboard is stored in the First In, First Out (FIFO) key stack.

2.6 Automatic Key Input at Power ON

The 18-byte variable KYISTK contains key input data that can be specified by a monitor K command during reset (refer to Memory Map). When the value of the variable KYISFL is 0A, the KYISTK contains key input data. When the value is 0B, key input data is currently being fetched from the KYISTK. If the value of the KYISFL is 0B when the subroutine KEYIN (to fetch the key input data from the key stack) is called, the value obtained from the KYISTK is used as the key input data.

2.7 Key Input Subroutines

The following subroutines for key input are provided.

(1) INITKY: Initializes the keyboard and sets the default value.
(2) KEYSCN: Performs the key scan operation and obtains input data from the pressed key.
(3) KEYIN: Fetches one character from the key input buffer.
(4) KEYSTS: Obtains the number of characters in the key input buffer.
(5) KYSSTK: Specifies the automatic input key data.

2.8 Sleep Function

The MCU is provided with a sleep function to reduce power consumption when it is not functioning. The sleep mode is entered during execution of the KEYIN subroutine to wait for key input when the key input buffer is empty.

2.9 Special Keys

(1) BREAK key
When the BREAK key is pressed, the data is not taken into the key stack, but the I/O operation is cancelled. (Subroutine BREKIO is called.) Then, the break process is performed to the slave MCU and bit 7 of variables MIOSTS (address 007D) and SIOSTS (address 007C) become ON. The data input from the power ON key stack is cancelled. If bit 7 of the variable RUNMOD (address 007B) is '1', control returns from the key input interrupt. When bit 7 is '0', the subroutine is called starting at the address specified by address (0120, 0121). The default values of address (0120, 0121) is (FF, B2). The subroutine RSTRIO (re-start of I/O operation) is executed at the entry point of the address FFB2 and control jumps to the menu routine. In addition to the BREAK key, the specified subroutines are executed when the MENU, PAUSE, CTRL/PF3, CTRL/PF4 and CTRL/PF5 keys are pressed. Any addresses can be specified.

(2) MENU key
When the MENU key is pressed and bit 7 of the variable RUNMOD is '1', the code FC is input to the key stack and control returns from the interrupt. When bit 7 is '0', control returns from the interrupt after executing the subroutine starting at the address specified by address (0122, 0123).

(3) PAUSE key
When the PAUSE key is pressed, bit 6 of the variable MIOSTS becomes '1'. Then, control returns from the interrupt if bit 7 of the variable RUNMOD is '1'. If bit 7 is '0', control returns from the interrupt after executing the subroutine starting at the address specified by address (0124, 0125).

(4) CTRL/PF3, CTRL/PF4 and CTRL/PF5 keys
When the CTRL/PF3, CTRL/PF4 or CTRL/PF5 keys are pressed, control returns from the interrupt after executing the corresponding subroutine starting at the address specified by the address (0126, 0127), (0128, 0129) and (012A, 012B). If, for example, (FF, 10) (the entry point of Monitor) is written to the address (0126, 0127), the Monitor will be executed when the CTRL/PF3 keys are pressed.

2.10 Key Input Modes

The current key input mode (numeric and uppercase, shift, etc.) is indicated by the 1-byte variable KEYMOD. The address of this variable is (FFE4, FFE5). The current data in this address is 0169. Referring to the contents in the address, the current mode, (in this case, the keyboard mode) can be recognized. To force-set a certain mode, change the contents of the current address to those of the mode to be set. The following three modes are available.

Bit 1: Numeric mode
Bit 2: CAPS mode (lowercase letter mode is assumed when bit 2 is '1'.)
Bit 4: Graphic mode

Only one of these bits may be '1' or all of them may be '0'.
The current mode is indicated by the bit which is '1'.

2.11 Changing Constants

The constants on the RAM are the following.
Key stack size, time interval until the second key input is accepted for auto repeat, time interval until the third or subsequent key input is accepted for auto repeat, sampling interval of key scan and power ON key stack. The default values for these constants are set when the keyboard is initialized. Values set after the default values have been set are used. (For details, refer to Memory Map).

Key scan

The keyboard value read by key scan is assigned to variable NEWKTB (10 bytes, starting address: FFD0, FFD1). Fig. 2-3 shows the format of the keyboard values read by key scan.

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
NEWKTB-0	D7(L0)	D8(L0)	D8(L0)	D8(L0)	D8(L0)	D8(L0)	D8(L0)	D8(L0)
NEWKTB-1	D7(L1)	D8(L1)	.....					D0(L1)
NEWKTB-2	D7(L2)	D8(L2)	.....					D0(L2)
NEWKTB-3	D7(L3)	D8(L3)	.....					D0(L3)
NEWKTB-4	D7(L4)	D8(L4)	.....					D0(L4)
NEWKTB-5	D7(L5)	D8(L5)	.....					D0(L5)
NEWKTB-6	D7(L6)	D8(L6)	.....					D0(L6)
NEWKTB-7	D7(L7)	D8(L7)	.....					D0(L7)
NEWKTB-8	D8(L7)	D8(L6)	D8(L5)	D8(L4)	D8(L3)	D8(L2)	D8(L1)	D8(L0)
NEWKTB-9	D9(L7)	D9(L6)	D9(L5)	D9(L4)	D9(L3)	D9(L2)	D9(L1)	D9(L0)

Fig. 2-3 Key Scan Values

In this case, the DIP switches and the PRINTER ON/OFF switch are set according to the values at address 7F. (In other words, software specification takes precedence over the key scan specification.)

2.12 Tables

Table 2.2 Key Matrix

Line / Return data	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7
D 0	0 0 0	0 8 8	1 0 @	1 8 H	2 0 P	2 8 X	3 0 RET	3 8 CLR
D 1	0 1 !	0 9 9	1 1 A	1 9 I	2 1 Q	2 9 Y	3 1 SPACE	3 9 SCRN
D 2	0 2 "	0 A :	1 2 B	1 A J	2 2 R	2 A Z	3 2 TAB	3 A BREAK
D 3	0 3 #	0 B ;	1 3 C	1 B K	2 3 S	2 B [		3 B PAUSE
D 4	0 4 $	0 C ,	1 4 D	1 C L	2 4 T	2 C ¥		3 C DEL
D 5	0 5 %	0 D -	1 5 E	1 D M	2 5 U	2 D]	3 5 NUM	3 D HENK
D 6	0 6 &	0 E .	1 6 F	1 E N	2 6 V	2 E ^		
D 7	0 7 '	0 F /	1 7 G	1 F O	2 7 W	2 F ←	3 7 CAPS	
D 8	4 0 PF1	4 1 PF2	4 2 PF3	4 3 PF4	4 4 PF5	4 5 PAPER		
D 9	4 8 DIP1	4 9 DIP2	4 A DIP3	4 B DIP4		4 D SHIFT	4 E CTRL	4 F PRINT

Note: The hexadecimal values in the above Table indicate the positions of the key layout on the keyboard matrix table.

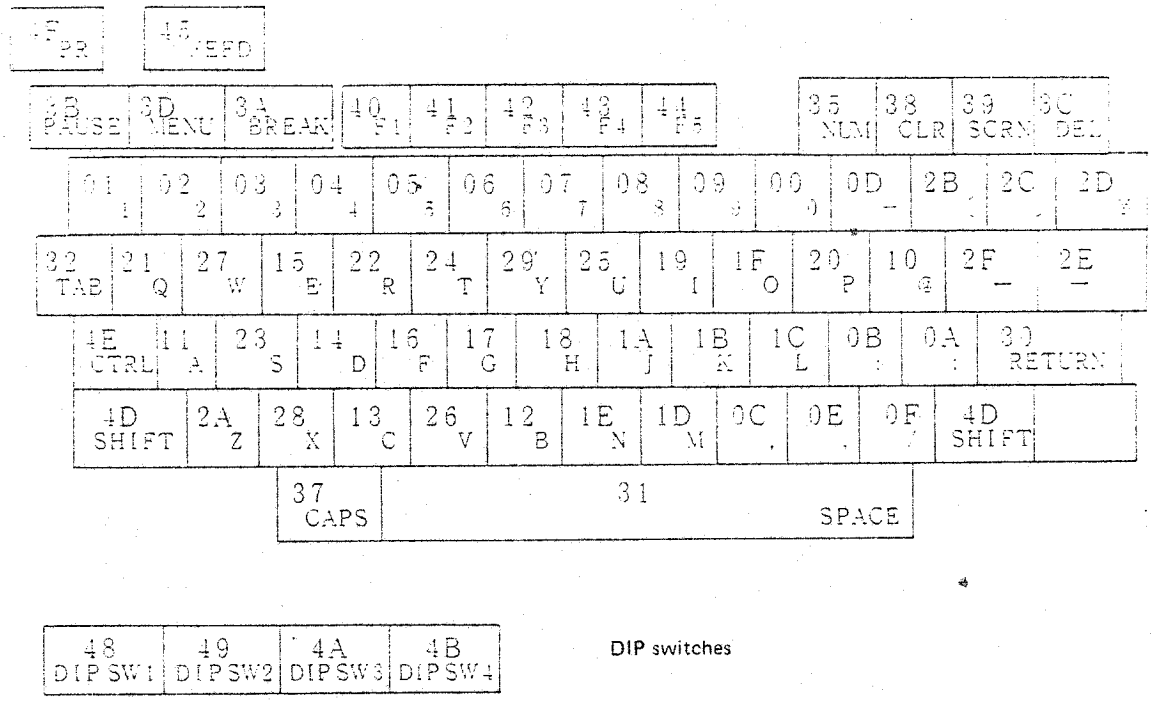


Fig. 2-4 Arrangement of the Keyboard Matrix

Note: The hexadecimal values in the above Table indicate the positions of the keys on the key table.

2.13 Keyboard Input Subroutines

Name of Subroutine	Entry Point	Description
INITKY	FFA0	Initializes key input. Sets the initial values (including default values). Specifies the vectors jumped to when the BREAK, MENU, PAUSE, CTRL/PF3, CTRL/PF4, and CTRL/PF5 keys are pressed. Specifies default values for timing such as sampling time, etc. Specifies the number of key stack data, and the key stack used when the power is turned ON.
		°Parameters: At Entry None At Return None °Registers retained None °Subroutines referenced None °Variables used None
KEYSTS	FF9D	Inputs the number of key stack data (Excluding the key stack used when the power is turned ON.)
		°Parameters: At Entry None At Return (C): Abnormal I/O flag (A): Number of key stack data (in bytes) (Z): According to the value of (A) °Registers retained (B) and (X) °Subroutines referenced None °Variables used None Note: When a PF key is pressed, the number of stack data increases by 2.
KEYIN	FF9A	Inputs one character from the key stack. If no data exists in the key stack, this subroutine lets the MCU sleep and waits until data is input from the keyboard. If data exists in the key stack when the power is turned ON, this data is recognized as input data. If both key stack data and keyboard data are available, the key stack data take precedence over keyboard data.
		°Parameters: At Entry None At Return (C): Abnormal I/O flag (A,B): Character code. 1-byte codes are stored in (A) and 2-byte codes ((A)=FE) are stored in (A,B).

Name of Subroutine	Entry Point	Description
		°Registers retained (X) °Subroutines referenced None °Variables used None
KEYSCN	FF6A	Scans the key matrix. The result of the key scan is stored in NEWKTB (10 bytes). Note that the DIP switches and Printer ON/OFF switch are set according to the value of the variable SDIPS2. The contents of NEWKTB are: Bit 7 Bit 6...Bit 0 NEWKTB+0: D7 D6 .... D0..... L0 NEWKTB+1: D7 D6 .... D0..... L1 . . . . NEWKTB+8: L7 L6 .... L1..... D8 NEWKTB+9: L7 L6 .... L1..... D9
		°Parameters: At Entry None At Return None °Registers retained None °Subroutines referenced None °Variables used K0 and K1 (The values of these variables are retained.)
KYSSTK	FF22	Inputs data to the key stack when the power is turned ON. The size of the key stack is 18 bytes. If more than 18 bytes of data are input, the excess data is ignored.
		°Parameters: At Entry (X): Starting address of character strings (B): Number of characters (0 to 18: the key stack is cleared when the number is 0.) °Parameters: At Return None °Registers retained None °Subroutines referenced None °Variables used None

2.14 Keyboard Work Area

Address (from)(to)	Variable name	Bytes	Description
0140 0140	KSTKSZ	1	The size of the key stack. The default value is 8. May be specified in the range 1 to 15. If the value is '1', input of PF keys is not accepted.
0141 0141	KICNT1	1	Time until the first key input is accepted for auto repeat. The unit depends on sampling time. The default value for sampling time is 20 msec. The default value is 40 (800 msec).
0142 0142	KICNT2	1	Time until the second or subsequent key input is accepted for auto repeat. The unit are the same as those of KICNT1. The default value is 6 (120 msec).
0143 0144	KICNTM	2	Sampling time. The unit 1 equals approx. 1.6 µsec. The default value is 12,288 (20 msec).
0145 014E	NEWKTB	10	Value of the key scan matrix. The status after the key scan is stored in this area. '1' denotes the ON condition. Bit 0 at the first address of the work area corresponds to 00 of the key matrix table and bit 7 corresponds to 07. In this manner, bit 7 of the last address corresponds to 4F.
014F 0158	OLDKTB	10	The value of the previous key scan. The previous value of NEWKTB is stored in this variable.
0159 0162	CHKKTB	10	Stores the data for the position of the newly pressed key after key scan.
0163 0164	KYISAD	2	Address of automatic key input when power is turned ON. Set to 016F at reset.
0165 0165	KYISFL	1	Flag indicating whether or not data exists in the key stack when power is turned ON. When this flag is 0A, data exists in the key stack but the fetch operation ends. When the flag is 0B, data is currently being fetched from the stack. If the flag is other than 0A and 0B, no data exists in the key stack.
0166 0166	KYISCN	1	The number of data in the automatic input key stack. Value is in the range 0 to 255.
0167 0167	KYISPN	1	The number of data input from the automatic key input. The number is in the range 0 to the value specified by KYISCN.
0168 0168	STKCNT	1	The number of data in the input key stack. The number is in the range 0 to the value specified by KSTKSZ.
0169 0169	KEYMOD	1	Input key modes. This address indicates the uppercase, numeric modes, etc.

Address (from)	(to)	Variable name	Bytes	Description
				Bit 1: Numeric mode when this bit is '1'. Bit 2: Lowercase mode when this bit is '1'. Bit 3: Unused Bit 4: Graphic mode when this bit is '1'. Bit 5: SHIFT mode when this bit is '1'. Bit 6: The CTRL key has been pressed when this bit is '1'. Bit 7: Indicates that a special keys such as the BREAK, PAUSE, MENU or one of the PF keys has been pressed when this bit is '1'. One of bits 0 through 4 must be ON or all bits must be OFF.
016A	016A	ONKFLG	1	Indicates the key input status. 00: Inhibits key reception. Waits until the pressed key is released. FF: Key input enabled. 01: Auto repeat function
016B	016B	KPRFLG	1	For auto repeat, this variable indicates the number of times the same key input has been received. When this value equals that of KICNT1 or KICNT2, the pressed character is taken to be input once.
016C	016C	KEYRPT	1	Indicates the auto repeat key position on the matrix. Refer to the Matrix Table.
016D	016E	CKEYRD	2	Input key code A PF key is 2 bytes.
016F	0180	KYISTK	18	Work area for the power ON key stack
0181	0188	CHRSTK	8	Work area for the key stack
0189	018F		7	This area is secured for expansion of the key stack.

CHAPTER 3 LIQUID CRYSTAL DISPLAY (LCD)

3.1 General

The Liquid Crystal Display (LCD) has a resolution of 120 horizontal dots and 32 vertical dots and LCD controllers which enable the specification of data for each dot.
6 LCD controllers together control the LCD, each of which controls an area 40 horizontal by 16 vertical dots.
Normally, a single character is displayed in a matrix of 6 horizontal by 8 vertical dots. Alphanumeric characters, however, are actually formed in a matrix 5 by 7 dots as spaces are left between characters on the screen.

3.2 Functions of LCD Controllers

As abovementioned, the 6 controllers together control the LCD.
The dot areas controlled by each controller are shown in Table 3-1.

Table 3-1 Dot Area Controlled by Each LCD Controller

	0 – 39	40 – 79	80 – 119
0 – 7	Controller 1 (bank 0)	Controller 2 (bank 0)	Controller 3 (bank 0)
8 – 15	Controller 1 (bank 1)	Controller 2 (bank 1)	Controller 3 (bank 1)
16 – 23	Controller 4 (bank 1)	Controller 5 (bank 0)	Controller 6 (bank 0)
24 – 31	Controller 4 (bank 1)	Controller 5 (bank 1)	Controller 6 (bank 1)

As shown in the table, each controller is responsible for an area of above of 40 by 16 dots.

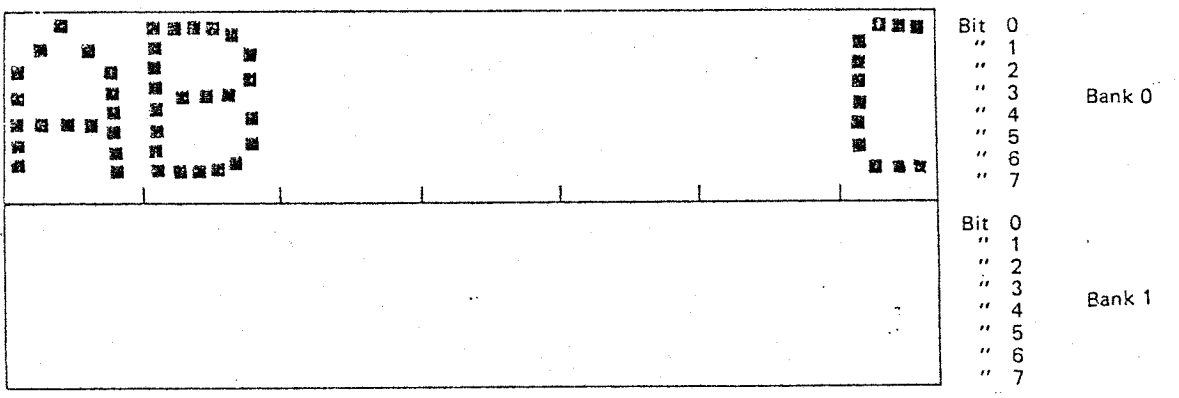


Fig. 3-1 Displayed Contents of Each LCD Controller

Each controller has data addresses 0 to 27_{16} in the row direction.
Data consists of 8 bits (Fig. 3-1).

3.3 I/O Ports for Display and Input

Table 3-2 I/O Ports Related to LCD Controllers

	Address	Bit position	Description
Master MCU	26	0 1 2	Output Selection of LCD driver 1-6: Controllers 1 to 6 selected. 0: No controller is selected.
		3	Output Selection of data or command for LCD driver 0: Data 1: Command
	28	7	Input BUSY signal of LCD controller 0: Busy
	2A		Output Serial clock to LCD controller
	2B		Output Serial clock to LCD controller
	2A	0	Output Output data to LCD controller
		1	Output Output data to LCD controller
		2	Output Output data to LCD controller
		3	Output Output data to LCD controller
		4	Output Output data to LCD controller
		5	Output Output data to LCD controller
		6	Output Output data to LCD controller
		7	Output Output data to LCD controller

3.4 Data Display Procedure

Data is displayed on the LCD by the following procedure.

(1) Selection of controller

One of the 6 controllers is selected by specifying an appropriate value in the bits 0 to 2 of address 26 using subroutine WRTP26. If 0 is specified, no controller is selected. System default is 0 for power conservation.

(2) Command setting

The bit 3 of address 26 is a bit used to select either data or command for the selected controller. When this bit is set to "1", a command is selected. This data/command selection may be performed simultaneously with the controller selection described in (1) above. Set a command in address 2A and confirm that the LCD controller is ready (when the bit 7 of address 28 is "1"). Then apply 8 serial clock pulses to the controller. (Address 2A is read 8 times. Since address 2B is also for serial clock input, 8 serial clock pulses are given to the controller upon execution of "LDD$2A" 4 times.)

(3) Data

When the bit 3 of address 26 is set to "0", data is selected. The data setting procedure is the same as the command setting described above. Depending on the type of command, data must be continuously output for display.

3.5 Input of Display Data
The bit indicating that the controller is busy (i.e., bit 7 of address 28) becomes the input data for display.

3.6 Display Subroutines
The HX-20 has the following three subroutines for character display:

(1) DSPLCN: Displays n characters of data (ASCII code) on the physical screen.
(2) DSPLCH: Displays one character of data (ASCII code) on the physical screen.
(3) DISPIT: Displays one character of data (ASCII code) on the physical screen.

3.7 Coordinates on the LCD
(x,y) indicates the coordinates on the LCD. x is the coordinate in the horizontal directions (columns) and y is the coordinate in the vertical direction (rows). (0,0) indicates that the positions of both the vertical and horizontal coordinates are the upper left edge of the LCD. The values of x,y coordinates on the text screen must be in the range shown below.

$0 \leq x \leq 19$ and $0 \leq y \leq 3$

The values of x,y coordinates on the graphic screen must be in the range shown below.

$0 \leq x \leq 119$ and $0 \leq y \leq 31$

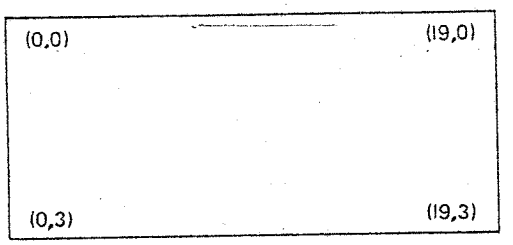


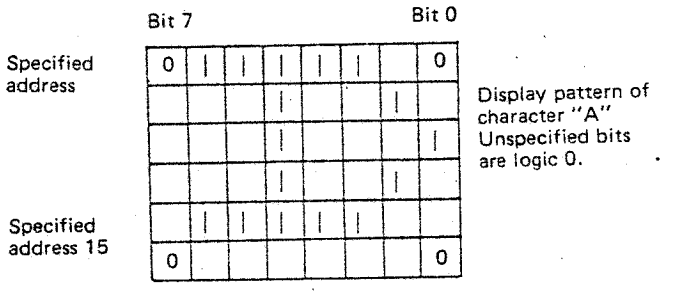
Fig. 3-2 Coordinates on the LCD (Text Screen)

3.8 LCD Subroutines

Subroutine name	Entry point	Description
DSPLCN	FF49	Displays or clears n characters on the physical screen.
		Parameters: At Entry (B): Number of characters displayed The screen is cleared when (B) is 0. (x): Starting address of data packet This parameter need not be specified when (B) is 0. Data packet Byte 0: x coordinate (0 to 19_{10}) of the display position of the first character. Byte 1: y coordinate (0 to 3_{10}) of the display position of the first character. Byte 2: Character code (ASCII) Byte 3: Character code (ASCII) Byte n+1: Character code (ASCII) At Return None Registers retained None Subroutines referenced 'DSPLCH' Variables used None
DSPLCH	FF4C	Displays one character on the physical screen. The display data is first written into the screen buffer PSBUF.
		Parameters: At Entry (A): ASCII character code (X): Display position on LCD (high, low)=x coordinate (0 to 19), y coordinate (0 to 3) At Return None Registers retained (A), (B), and (X) Subroutines referenced CHRGEN, LCDMOD and DATMOD Variables used None
DISPIT	FF5B	Displays one character on the physical screen. The display data is not written into the screen buffer (PSBUF).
		Parameters at entry and return, registers retained, subroutines referenced and variables used are the same as those for subroutine DSPLCH.

Subroutine name	Entry point	Description
CHRGEN	FF67	Generates the character pattern. A character pattern (6x8 dots) is provided for display of the character specified by the ASCII code on the LCD. Certain character patterns may change according to the value set by the DIP switch for different countries.

Parameters:
At entry
(A): Character code
(X): Starting address where 6-byte character display pattern is stored
At Return
Character display pattern (specified address)



Registers retained
None
Subroutines referenced
None
Variables used
None
Others
Re-entrant

3.9 Screen Routine Work Area

Address	Variable name	No. of bytes	Description
220 26F	PSBUF	80	Positions of data (ASCII codes) displayed on the physical screen represented in (column, row) format as follows: (0,0), (1,0) ... (19_{10}, 0) (0,1) ... (19_{10}, 3)
270 271	SCRTOP	2	Starting address of the virtual screen buffer
272 273	SCRBOT	2	Ending address of the virtual screen buffer
274 275	DISTOP	2	Starting position of the physical screen on the virtual screen The address of position (0,0) of the physical screen in the physical screen buffer
276 276	VSCRX	1	Virtual screen width indicated as the maximum values of x coordinate.
277 277	VSCRY	1	Virtual screen length indicated as the maximum value of y coordinate.
278 278	CURX	1	x coordinate of the cursor position (on the physical screen)
279 279	CURY	1	y coordinate of the cursor position (on the physical screen)
27A 27A	LRMODE	1	Scroll step x (left and right)
27B 27B	UDMOD	1	Scroll step y (up and down)
27C 27C	CURMRG	1	Scroll margin (1 through 10)
27D 27D	SSPEED	1	Vertical scrolling speed (0 to 9) When the scrolling speed is set at 8, there is a 130-msec wait between each scroll. This wait is increased in 130-msec increments for each setting: 7,6,5 ... 0. When a scrolling speed of 9 is specified, there is no wait between vertical scrolls.
27E 27E	DISPX	1	x coordinate (0 to 19_{10}) of the character to be displayed on the physical screen by a virtual screen routine
27F 27F	DISPY	1	y coordinate (0 to 3) of the character to be displayed on the physical screen by a virtual screen routine
280 280	DISSTS	1	Indicates the display status. Bit 0: Indicates whether or not left and right scrolling is permitted. 1: Scrolling disabled 0: Scrolling enabled Bit 1: Not used Bit 2: Not used Bit 3: Not used Bit 4: Indicates whether or not there is a wait in vertical scrolling. 1: Wait executed

Address	Variable name	No. of bytes	Description
			Bit 5: Cursor ON/OFF switch Determines whether the cursor ('_' below the character) will be displayed on the physical screen. 1: Cursor ON 0: Cursor OFF Bit 6: Indicates cursor ON/OFF status. 1: Cursor ON 0: Cursor OFF Bit 7: Flag to indicate whether or not the entire screen is to be rewritten 0: Display for only one character 1: Rewrites for entire screen. Note: All screen data must be checked and rewritten even if only one character is to be displayed. However, since doing this adversely affects operating speed, this switch is used to reduce the amount of screen data checked and rewritten. Bits 5 and 6 The following two operations are required to display a character at the cursor position and then move the cursor to the next position: 1. Display of the character at the cursor position The cursor is turned OFF and the character is displayed at the cursor position. 2. Cursor movement A space character is displayed where the cursor is ON. Bit 5 is used to control cursor ON/OFF condition and bit 6 determines whether the cursor will be displayed on the screen.
281 285		5	Used as temporary work area.
286 28B	CHRPTN	6	Contains the character font (result of subroutine CHRGEN).

```
      *** 6301 CROSS MACROASSEMBLER   VER1.0 ***                          10/21/82    09:43:01

ERR   SEQ   LOC  OBJECT      PROGRAM  LCD        -----LCD DRIVER ROUTINE-----

    00001                                NAM    LCD
    00002                          *
    00003                          *  LCD DRIVER'S ROUTINE
    00004                                TTL    -----LCD DRIVER ROUTINE-----
    00005                          * FILE NAME    'EX59'   BY  K.A
    00006                                OPT    LOAD
    00007                                OPT    PAGE=55
    00008                          *
    00009                          * I/O PORT
    00010                          *  $28
    00011                          *       7:R LCD DRIVER BUSY (0:BUSY 1:READY)
    00012                          *
    00013                          *  $26
    00014                          *       0:W LCD COMMAND/DATA 1
    00015                          *       1:W LCD COMMAND/DATA 2
    00016                          *       2:W LCD COMMAND/DATA 4
    00017                          *       3:W LCD COMMAND/DATA SELCTION (0:DATA 1:COMMAND)
    00018                          *       4:W KEY BOARD INTERRUPT MASK (0:CLOSE 1:OPEN)
    00019                          *       5:W PERIPHERAL CONTROL (TO SERIAL)
    00020                          *       6:W TO PLUG IN 1
    00021                          *       7:W TO PLUG IN 2 AND SLAVE P40
    00022                          *
    00023                          * SUBROUTINE ENTRY POINT
    00024          FED4   A        WRTP26 EQU    $FED4
    00025          FF67   A        CHRGEN EQU    $FF67
    00026                          * WORK AREA
    00027          007D   A        MIOSTS EQU    $7D       * MAIN I/O STASTUS
    00028                          *                       * BIT 0: ON READ/WRITE TO LCD  1:READING/WRITIN
    00029          0286   A        CHRPTL EQU    $286      * WORK AREA TO STORE CHARACTER PATTERN
    00030          0220   A        PSBUF  EQU    $220      * CHARACTER CODES ON PHISICAL SCREEN.
    00031          0280   A        DISSTS EQU    $280      * DISPLAY STATUS
    00032                          *                       *   BIT 5: CURSOR ON WITH CHARACTER PATTERN FL
    00033                          *                       *          (1:ON)
    00034                          *                       *
    00035                          *                       *
    00036A 1000                          ORG    $1000
    00037                          *
    00038                          * DISPLAY ONE CHARACTER TO REAL LCD SCREEN
    00039                          *  ROUTINE 'DISPIT': DISPLAY 1 CHARACTER TO LCD WITHOUT BUFFER
    00040                          *          'DISPCH': DISPLAY ONE CHARACTER TO LCD AND WRITE TO BUFFER
    00041                          * ON ENTRY
    00042                          *   (A): ASCII CHARACTER CODE
    00043                          *   (X): LCD POSITION  (HIGH:COLUMN, LOW:LINE)
    00044                          * ON EXIT
    00045                          *   (X): NEXT DATA POSITION (IF ILLEGAL ADDRESSING, CHANGE TO LEGAL)
    00046                          * REGISTER PRESERVE    A,B
    00047                          *
    00048                          *
    00049                          *
    00050                          * CHECK DISPLAY POSITION AND GENERATE CHARACTER FONT
    00051                          *  ON ENTRY
    00052                          *   (X):DISPLAY ADDRESS (HIGH BYTE:COLUMN,  LOW BYTE:LINE)
    00053                          *   (A):DISPLAYED CHARACTER
    00054                          *  ON EXIT
    00055                          *   (A,B):MODEFIED POSITION
```

```
ERR   SEQ   LOC  OBJECT      PROGRAM  LCD        -----LCD DRIVER ROUTINE-----

     00056                          *  REGISTER PRESERVE (X) <--- (A,B)
     00057                          *
     00058           1000  A        DPCHEK EQU    *
     00059A 1000 72 017D   A               OIM    #$1,MIOSTS * SET FLAG   LCD IS ON WRITING
     00060A 1003 37                        PSH B
     00061A 1004 36                        PSH A
     00062A 1005 3C                        PSHX
     00063A 1006 CE 0286   A               LDX    #CHRPTL    * SET CHARACTER PATTERN TO 'CHRPTL'
     00064A 1009 BD FF67   A               JSR    CHRGEN
     00065A 100C 32                        PUL A
     00066A 100D 33                        PUL B
     00067A 100E 38                        PULX                * NOTE.  (X)  <---> (A,B)
     00068A 100F 81 13     A               CMP A  #19         * IS COLUMN LIMIT OUT OF RANGE ?
     00069A 1011 25 02 1015                BCS    NONOVX      * NO.
     00070A 1013 86 13     A               LDA A  #19         * YES. LIMIT= 19
     00071A 1015 C4 03     A        NONOVX AND B  #3
     00072                          *
     00073A 1017 39                        RTS
     00074                          *
     00075                          **
     00076                          * DISPLAY ONE CHARACTER TO REAL SCREEN WITHOUT WRITING TO SCREEN BUFFE
     00077                          * ON ENTRY
     00078                          *   (A): CHARACTER (ASCII CODE)
     00079                          *   (X): DISPLAY ADDRESS (HIGH:CLOLUMN, LOW:LINE)
     00080                          *
     00081                          *  ENTRY POINT
     00082           1018  A        DISPIT EQU    *
     00083A 1018 8D E6 1000                BSR    DPCHEK
     00084A 101A 3C                        PSHX                * SAVE VALUE OF (A,B)
     00085A 101B 20 15 1032                BRA    NONSET
     00086                          *  ENTRY POINT
     00087           101D  A        DSPLCH EQU    *
     00088A 101D 8D E1 1000                BSR    DPCHEK
     00089                          *
     00090A 101F 3C                        PSHX                * SAVE (A,B)
     00091                          *
     00092A 1020 37                        PSH B
     00093A 1021 36                        PSH A
     00094A 1022 86 14     A               LDA A  #20
     00095A 1024 3D                        MUL
     00096A 1025 30                        TSX
     00097A 1026 EB 00     A               ADD B  0,X         * ADDRESS OFFSET <--- (B)*WIDTH + (A)
     00098A 1028 A6 02     A               LDA A  2,X
     00099A 102A CE 0220   A               LDX    #PSBUF      * (X) <--- PHISICAL SCREEN BUFFER ADDRESS
     00100A 102D 3A                        ABX
     00101A 102E A7 00     A               STA A  0,X
     00102A 1030 32                        PUL A
     00103A 1031 33                        PUL B
     00104                          *CALCULATE ADDRESS IN LCD DRIVER
     00105           1032  A        NONSET EQU    *
     00106A 1032 37                        PSH B               * SAVE LOCATION POINTER (X,Y)
     00107A 1033 36                        PSH A
     00108                          *                            * ALREADY SAVE FOUR BYTES
     00109                          *                            *  STACK+0: COLUMN,   STACK+1:LINE
     00110                          *                            *  STACK+2: (A)       STACK+3:(B)
```

```
ERR   SEQ   LOC  OBJECT      PROGRAM  LCD        -----LCD DRIVER ROUTINE-----

     00111A 1034 37                        PSH B
     00112A 1035 48                        ASL A            * (A) <-- (A) * 6  (DOT COLUMN)
     00113A 1036 16                        TAB
     00114A 1037 48                        ASL A
     00115A 1038 1B                        ABA
     00116A 1039 33                        PUL B
     00117                          *
     00118                          *  WORK USE  STACK
     00119                          *    STACK     00: DOT COUNTER (1 CHARACTER = 6 DOT LINES)
     00120                          *              01,02: CHARACTER FONT ADDRESS
     00121                          *              03,04: DOT COLUMN, LINE
     00122                          *
     00123A 103A 37                        PSH B
     00124A 103B 36                        PSH A
     00125                          *
     00126                          * ***** LCD DRIVE RUOTINE ******  (X):CHARACTER PATTERN TOP ADDRESS
     00127                          * SET CHARCTER TO DRIVER
     00128A 103C CE 0286  A                LDX    #CHRPTL
     00129A 103F 3C                        PSHX
     00130A 1040 5F                        CLR B
     00131A 1041 37                        PSH B
     00132A 1042 30                        TSX
     00133          1043  A        DISCHL EQU    *          * SET COUNTER.
     00134A 1043 EC 03     A                LDD    3,X
     00135A 1045 8D 56 109D                 BSR    LCADDR     * GET CHIP NO. & DATA ADDRESS.(DATADD,CHIPNO)
     00136A 1047 16                         TAB               * SAVE LCD ADDRESS TO (B)
     00137A 1048 86 64     A                LDA A  #$64       * SELECT WRITE MODE.
     00138A 104A 8D 76 10C2*                BSR    LCDMOD     * SET COMMAND
     00139A 104C 17                         TBA
     00140A 104D 8A 80     A                ORA A  #$80       * SET DATA ADDR TO LCD DRIVER.(AUTO INCREMENT)
     00141A 104F 8D 71 10C2*                BSR    LCDMOD
     00142A 1051 CC 0800   A                LDD    #$0800     * SET DATA MODE CODE FOR 'WRTP26' ROUTINE
     00143A 1054 BD 10DC   A                JSR    DATMOD     * LCD DRIVER: ENTER DATA MODE (NOT COMMAND)
     00144A 1057 E6 00     A        WRTLOP LDA B  0,X        * GET 8 BITS PATTERN
     00145A 1059 EE 01     A                LDX    1,X        * (B): DOT POSITION (0 - 5)
     00146A 105B 3A                         ABX
     00147A 105C A6 00     A                LDA A  0,X
     00148A 105E C1 05     A                CMP B  #5         * LAST DOT (6 TH): WITHOUT CURSOR
     00149A 1060 27 09 106B                 BEQ    DISC20
     00150A 1062 F6 0280   A                LDA B  DISSTS     * CURSOR ON ?
     00151A 1065 C5 20     A                BIT B  #$20
     00152A 1067 27 02 106B                 BEQ    DISC20
     00153A 1069 8A 80     A                ORA A  #$80
     00154A 106B 8D 55 10C2         DISC20 BSR    WITDAT     * WRITE ONE BYTE BIT PATTERN
     00155A 106D 30                         TSX
     00156A 106E 6C 00     A                INC    0,X        * COMPLETE TO WRITE 6 BYTES ?
     00157A 1070 A6 00     A                LDA A  0,X
     00158A 1072 81 06     A                CMP A  #6
     00159A 1074 27 0E 1084                 BEQ    ENDDIC     * YES . END
     00160A 1076 6C 03     A                INC    3,X
     00161A 1078 A6 03     A                LDA A  3,X        * INCREMENT DATA ADDRESS
     00162A 107A 81 28     A                CMP A  #40        * BOUNDARY OF DRIVER = 40.
     00163A 107C 27 C5 1043                 BEQ    DISCHL
     00164A 107E 81 50     A                CMP A  #80
     00165A 1080 27 C1 1043                 BEQ    DISCHL     * CHIP LAST ADD ?
```

```
ERR   SEQ   LOC  OBJECT      PROGRAM  LCD        -----LCD DRIVER ROUTINE-----

     00166A 1082 20 D3 1057              BRA     WRTLOP     *
     00167                         *
     00168                         *
     00169A 1084 CC 0F08  A      ENDDIC LDD     #$0F08     * CHIP SELECT OFF. COMMAND MODE
     00170A 1087 8D 53 10DC              BSR     DATMOD     *
     00171A 1089 31                      INS                * RECOVER STACK POINTER  (+5)
     00172A 108A 38                      PULX
     00173A 108B 38                      PULX
     00174A 108C 32                      PUL A              * RESTORE POSITION ON LCD
     00175A 108D 33                      PUL B
     00176A 108E 4C                      INC A
     00177A 108F 81 14    A              CMP A  #20         * NEXT POINTER.
     00178A 1091 26 04 1097              BNE    DIC100
     00179A 1093 4F                      CLR A
     00180A 1094 5C                      INC B
     00181A 1095 C4 03    A              AND B  #$03
     00182          1097  A      DIC100 EQU    *
     00183A 1097 71 FE7D  A              AIM    #$FF-$1,MIOSTS * LCD FLAG   LCD NOT BUSY
     00184A 109A 38                      PULX
     00185A 109B 18                      XGDX               *RETURN NEXT DISPLAY POINT.
     00186A 109C 39                      RTS
     00187                         *
     00188                         *
     00189                         *
     00190                         * SELECT LCD DRIVER AND CALCULATE BANK AND ADDRESS POINTER
     00191                         * NOTE. SET TO $26 DIRVER ADDRESS, BUT NOT SET TO LCD DRIVER, ONLY RETU
     00192                         *       LCD ADDRESS TO (A).
     00193                         * ON ENTRY
     00194                         *   (A): DOT LINE COLUMN POSITION (00 - 119)
     00195                         *   (B): LINE (0 - 3)
     00196                         * ON EXIT
     00197                         *   (A): DOT POINTER IN THE LCD DRIVER
     00198                         *   (B): CHIP NO. (BIT3=1)
     00199                         * REGISTER PRESERVE X
     00200                         *
     00201          109D  A      LCADDR EQU    *
     00202A 109D 3C                      PSHX               * SAVE (X)
     00203A 109E 37                      PSH B              * STACK VALUE OF (B)
     00204A 109F 30                      TSX
     00205A 10A0 5F                      CLR B
     00206A 10A1 80 28    A      LCAD10 SUB A  #40         * (A) <--- ADDRESS AND BANK
     00207A 10A3 5C                      INC B              * (B) <--- CHIP NO.
     00208A 10A4 24 FB 10A1              BCC    LCAD10
     00209A 10A6 8B 28    A              ADD A  #40         * GET START ADDRESS.   (B): 1-3
     00210A 10A8 6B 0100  A              TIM    #$1,0,X     * CHECK BANK (ODD LINE NO. = BANK 1)
     00211A 10AB 27 02 10AF              BEQ    LCAD20
     00212A 10AD 8A 40    A              ORA A  #$40
     00213          10AF  A      LCAD20 EQU    *
     00214A 10AF 6B 0200  A              TIM    #$2,0,X     * CHECK DRIVER CHIP (LINE >=2,  4-6)
     00215A 10B2 27 02 10B6              BEQ    LCAD30
     00216A 10B4 CB 03    A              ADD B  #3          * CHIP IS 4 , 5 OR 6
     00217A 10B6 31                LCAD30 INS
     00218A 10B7 38                      PULX
     00219A 10B8 36                      PSH A
     00220A 10B9 CA 08    A              ORA B  #$08        * BIT3= DATA MODE BIT (1:COMMAND)
```

```
ERR   SEQ   LOC  OBJECT      PROGRAM  LCD        -----LCD DRIVER ROUTINE-----

00221A 10B3 86 0F    A              LDA A  #$0F      * SET CHIP NO.
00222A 10B5 BD FED4  A              JSR    WRTP26    * SELECTED DRIVER CHIP, AND ENTER COMMAND MODE
00223                        *
00224A 10C0 32                      PUL A            * SET DATA ADDRESS TO DRIVER
00225A 10C1 39                      RTS
00226                        *
00227                        *
00228          10C2  A       WITDAT EQU    *
00229          10C2  A       LCDMOD EQU    *
00230A 10C2 37                      PSH B
00231A 10C3 16                      TAB
00232A 10C4 36                      PSH A
00233A 10C5 07                      TPA
00234A 10C6 0F                      SEI
00235A 10C7 3C                      PSHX
00236A 10C8 D7 2A    A              STA B  $2A       * SET ADD OR MODE.
00237A 10CA 7D 0028  A       LCDM10 TST    $28       * 7 BIT IS LCD BUSY FLAG.
00238A 10CD 2A FB 10CA              BPL    LCDM10    * WAIT.
00239A 10CF DE 2A    A              LDX    $2A       * LDD SEND 2 PULSES ,SO 2 BIT SHIFT
00240A 10D1 DE 2A    A              LDX    $2A
00241A 10D3 DE 2A    A              LDX    $2A
00242A 10D5 DF 2A    A              LDX    $2A
00243A 10D7 38                      PULX
00244A 10D8 06                      TAP
00245A 10D9 32                      PUL A
00246A 10DA 33                      PUL B
00247A 10DB 39                      RTS
00248                        *
00249                        *
00250                        * AFTER CHECK LCD BUSY, CALL 'WRTP26'
00251                        * ON ENTRY, (SAME AS 'WRTP26')
00252                        *  (A):TARGET BIT POSITION
00253                        *  (B): DATA
00254                        *
00255          10DC  A       DATMOD EQU    *
00256A 10DC 7D 0028  A              TST    $28       * 7 BIT IS LCD BUSY FLAG.
00257A 10DF 2A FB 10DC              BPL    DATMOD    * WAIT.
00258A 10E1 7F FED4  A       DTMD10 JMP    WRTP26    * SET INTERRUPT MASK.
00259                        *
00260                        *
00261                        **
00262                        *
00263                        * INITIALIZE LCD ROUTINE
00264                        *   DRIVER INITIALIZE AND CLEAR CURSOR ON FLAG
00265                        * ON ENTRY PARAMETER NONE
00266                        *
00267A 10E4 86 10    A       INITLC LDA A  #$10      * SFF (SET FRAME FREQUENCY) COMMAND
00268A 10E6 8D 0F 10F7              BSR    STRALL    * SET IT FOR EACH DRIVER.
00269A 10E8 86 1E    A              LDA A  #$1E      * ACCA : SMM (SET MULTIPLEXING MODE) COMMAND
00270A 10EA 8D 03 10F7              BSR    STRALL    *   NOTE. 1 ST DRIVER: SMM VALUE=$1E
00271                        *                                  2ND - 6 TH DRIVER: SMM=$1C
00272A 10EC 86 08    A              LDA A  #$08      * ACCA : DISP OFF COMMAND. ACCB : CHIP NO.
00273A 10EE 8D 07 10F7              BSR    STRALL
00274                        *
00275A 10F0 7F 0280  A              CLR    DISSTS    * CLEAR DISPLAY STATUS FOR NON CURSOR CLEAR.
```

```
ERR   SEQ   LOC  OBJECT      PROGRAM  LCD        -----LCD DRIVER ROUTINE-----

     00276                          *
     00277A 10F3 8D 18 110D                BSR    LCDCLR     * CLEAR SCREEN .
     00278                          *
     00279A 10F5 86 09    A                LDA A  #$09       * DISPLAY ON COMMAND.
     00280                          *
     00281                          *
     00282                          ******* SET COMMAND ALL DRIVERES.
     00283                          * SET COMMAND TO LCD DRIVER
     00284                          * ON ENTRY
     00285                          *   (A): COMMAND FOR LCD DRIVER
     00286                          * ON EXIT
     00287                          * REGISTER PRESERVE  X
     00288                          * NOTE. THIS ROUTINE MUST BE CALL ONLY 'INITLC.
     00289                          *       BECAUSE COMMAND WILL BE CHANGED.
     00290                          *
     00291A 10F7 5F                STRALL CLR B              * (B): DRIVER NUMBER
     00292A 10F8 5C                STRA10 INC B
     00293A 10F9 37                       PSH B
     00294A 10FA CA 08    A               ORA B  #$8
     00295A 10FC 36                       PSH A
     00296A 10FD 86 0F    A               LDA A  #$0F       * SELECT DRIVER AND COMMAND MODE
     00297A 10FF BD FED4  A               JSR    WRTP26
     00298A 1102 32                       PUL A
     00299A 1103 8D BD 10C2               BSR    LCDMOD     * SET COMMAND TO DRIVER
     00300A 1105 84 FD    A               AND A  #$FF-$2    * TO CHANGE '1E' (SMM COMMAND) COMMAND TO '1C'
     00301A 1107 33                       PUL B             *  OTHER COMMANS ARE $10, $08, $09.(NOT EFFECT
     00302A 1108 C1 06    A               CMP B  #6
     00303A 110A 26 EC 10F8               BNE    STRA10
     00304A 110C 39                       RTS
     00305                          *
     00306                          *
     00307                          * CLEAR LCD SCREEN
     00308                          * ON ENTRY
     00309                          *   PARAMETER NONE
     00310                          * ON EXIT
     00311                          *   (X):0
     00312                          *  REGISTER PRESERVE NONE
     00313                          *
     00314          110D  A         LCDCLR EQU    *
     00315A 110D CE 0000  A               LDX    #0         * POINTER SET.
     00316A 1110 86 20    A         LCDC10 LDA A  #$20       * SET SPACE CODE.
     00317A 1112 BD 101D  A               JSR    DSPLCH
     00318A 1115 08                       INX               * IX HAS DISPLAY POINTER.
     00319A 1116 09                       DEX
     00320A 1117 26 F7 1110               BNE    LCDC10     * NOT END.
     00321A 1119 39                       RTS
     00322                          *
     00323                          *
     00324                          * DISPLAY CHARACTER STRING TO LCD
     00325                          * ON ENTRY
     00326                          *   (B): NEMBER OF CHARCTER STRING (0 - 80)
     00327                          *   (X): ADDRESS OF DATA PACKET
     00328                          *       DATA PAKET:(ADDRESS X), (ADDRESS Y), DATA1,.....
     00329                          * ON EXIT
     00330                          *   PARAMETER  NONE
```

```
ERR   SEQ   LOC  OBJECT      PROGRAM  LCD        -----LCD DRIVER ROUTINE-----

     00331                          * REGISTER PRESERVE NONE
     00332                          * ENABLE REENTRANT
     00333                          *  WORK USE  : STACK , STACK + 0,1 :LOCATION OF CHARACTER ON LCD
     00334                          *                        2,3 :ADDRESS OF STORED CHARACTER
     00335                          *                        4   :DISPLAYED CHARACTER NUMBER
     00336A 111A 5D                 DSPLCN TST B            * IF (B)=0, CLEAR SCREEN.
     00337A 111B 27 F0 110D                BEQ    LCDCLR
     00338A 111D 37                        PSH B
     00339A 111E 3C                        PSHX
     00340A 111F EE 00     A               LDX    0,X      * GET LOCATION OF DISPLAY
     00341A 1121 3C                        PSHX            *
     00342A 1122 5F                        CLR B           * COUNTER OF DISPLAYED CHARACTER
     00343A 1123 30                        TSX
     00344A 1124 EE 02     A        DSPL10 LDX    2,X
     00345A 1126 3A                        ABX             *
     00346A 1127 A6 02     A               LDA A  2,X      * (A) <--- DISPLAYED CHARACTER
     00347A 1129 38                        PULX            * (X): LOCATION ON LCD
     00348A 112A BD 101D   A               JSR    DSPLCH   * DISPLAY ONE CHARACTER TO SCREEN.
     00349A 112D 3C                        PSHX
     00350A 112E 5C                        INC B
     00351A 112F 30                        TSX
     00352A 1130 E1 04     A               CMP B  4,X      * FINISHED ?
     00353A 1132 26 F0 1124                BNE    DSPL10
     00354A 1134 38                        PULX
     00355A 1135 38                        PULX
     00356A 1136 33                        PUL B
     00357A 1137 39                        RTS
     00358                          *
     00359          0000   A               END
***** TOTAL ERRORS     0
```