OCTAGON SYSTEMS CORP.

5150 W 80th Ave., Westminster, Co. 80030 (303) 426-8540

SYS-1 Operator's Manual

Revision 4

(c) 1982 Octagon Systems Corp.

TABLE OF CONTENTS [redacted]

SYS-1 OPERATION 1
- Description
- Edge connections
- Memory map
- Initialization

PROGRAMMING 4
- Strings
- Vectored interrupts
- Faster execution
- Program pointers
- Arrays
- Error codes

UTILITY LIBRARY 7
- Menu
- Move BASIC program
- Recovery demonstration
- Copy memory
- Hex dump
- Decimal to hexidecimal conversion
- Serial line listing
- Utility library program listing

INPUT/OUTPUT 13
- Programming the 8255
- Output drive
- Bit masking
- Baud rate programming

APPLICATIONS 15

TESTING 16

COMPONENT LAYOUT REV 2 18

PARTS LIST REV 2 19

SCHEMATIC REV 2 21

COMPONENT LAYOUT REV 3 23

PARTS LIST REV 3 24

SCHEMATIC REV 3 25

SYS-1 OPERATION

The SYS-1 board is a complete computer system requiring only a power supply for operation. With the addition of a terminal SYS-1 is a complete development system. This board is ideally suited for program development and control applications. SYS-1 is engineered for ease of use both in programming and in custom applications.

SYS-1 includes 4K of onboard RAM space and 4K of EPROM space. The EPROM programmer included onboard requires +25 VDC for programming 2716 type EPROMS. Maximum EPROM space of 6K is obtained by substituting 2K of EPROM for the upper 2K of RAM.

SYS-1 supports 24 programmable Input / Output lines. These are organized as three ports of 8 bits each, the status of which reflect the status of the data bus at the time of input or output. In addition to these, two direct flag outputs and one high level sense input are supported. The flag outputs are latched according to the BASIC statement STAT. The sense input vectors an interrupt directly into the BASIC program similiar to a GOSUB.

Addditional I/O in the form of an RS-232 port to a terminal completes the system. All data, address, and control lines are brought out to a 44 contact edge connector for expansion.

J1	DESCRIPTION	J1	DESCRIPTION
A	Power, +5 VDC	1	Power, +5 VDC
B	D0	2	D1
C	D2	3	D3
D	D4	4	D5
E	D6	5	D7
F	A0	6	A1
H	A2	7	A3
J	A4	8	A5
K	A6	9	A7
L	A8	10	A9
M	A10	11	A11
N	A12	12	A13
P	A14	13	A15
R	NWDS	14	NRDS
S	NHOLD	15	RST
T	F2	16	F3
U	SB	17	NUPS
V	-	18	NRST
W	-	19	-
X	-	20	+25 VDC see note 1
Y	-	21	-5 VDC out, see note 2
Z	Common	22	Common

Note 1. The +25VDC supply is required only for programming EPROMs. This supply must be free from over shoot during power on, power off and loading. See Appendix A for recommended circuit.

Note 2. If an external negative supply (-3 VDC to -12VDC) is available this pin can be an input. Remove U11

The memory map for the SYS-1 is as follows:

LOCATION	FUNCTION
0000-09FF	Internal ROM (Interpreter)
0A00-0FFF	24 I/O lines (8255)
1000-17FF	RAM U1
1800-1FFF	RAM U2
2000-3FFF	Unused
4000-7FFF	Baud rate select
8000-87FF	Auto start EPROM U4
8800-8FFF	SYS-1 Utility library U3
9000-97FF	EPROM programmer
A000-BFFF	Unused
C000-FFBF	Baud rate select
FFC0-FFFF	Internal RAM

SYS-1 communicates directly with your terminal programs and commands are entered from the keyboard during the development phase. In order to operate properly the terminal must be set to 4800 baud. If your terminal cannot communicate at this rate go to the section "Baud Rate Programming" for instructions to change the SYS-1 baud rate. Additionally, the terminal must be set for full duplex, online operation, with no parity. If you have a CAPS LOCK switch, it should be on.

SYS-1 is ready to communicate via the keyboard as soon as power is applied. Proper operation is evidenced by the interpreter displaying the prompt . This indicates the computer system has initialized properly.

Program entry can begin after the new program location (in RAM) has been declared:

```
>NEW #1100
```

The symbol for hexadecimal notation is #. Although RAM begins at 1000H (hex), the interpreter requires the first 100H bytes. The RAM space must next be cleared for new program entry:

```
> NEW
```

A program can now be entered from the terminal

```
> 10 PRINT "HELLO"
> 20 GO TO 10
> RUN
```

The utility library (LIB-1A) is stored in EPROM beginning at location 8800H. This program can be run at any time without disturbing RAM based programs by executing:

```
NEW#8800
```

When finished, the program being developed is recovered by executing:

```
NEW #1100
RUN
```

PROGRAMMING

The first source of programming information should be the NSC Tiny BASIC user's manual. There are, however, several points that may be enlarged upon:

The break key is used to terminate a running program. Control C is required to terminate an input statement or a program line entry.

RND(a,b) does not generate a truly random sequence if the difference between a and b is one.

The @ operator replaces both PEEK and POKE in memory assignment statements.

Using the Boolean operators with operands different by greater than 8000H will produce an apparently invalid result due to the integer bounds.

The first statement of a program designed to auto start must be CLEAR if that program has any PRINT statements.

LET is optional. TO in GOTO is optional. THEN in IF/THEN is optional. STEP in FOR/TO/STEP is optional. All spaces are optional and should be deleted if speed is desired. PRINT can be abbreviated PR.

The INC and DEC instructions are of the form A=INC(#17FF)

If during program development an attempt is made to insert a line into the program exactly 100H bytes from the end of the program the interpreter mistakenly moves entire program in RAM the length of the attempted insertion. The program can be completely recovered by executing a NEW(addr) statement where (addr) is the old program location plus the length of the line you attempted to insert. You can use the hex dump routine to locate the new beginning.

Strings are stored sequentially as ASCII characters in memory. The string begins at the address assigned when the string is entered and ends at the first carriage return (OD) thereafter.

```
>$#1300="HELLO THERE"
```

stores a string of 12 ASCII characters starting at location 1300 H and ending at 130D where carriage return is stored. The storage location does not have to be directly assigned,

```
>A=#1300:$A="HOW ARE YA?"
```

starts string storage at the location in memory contained in variable A. The interpreter supports a string copy command.

```
>B=#1500:$B=$A
```

copies the string at 1300 sequentially into locations starting at 1500. The string copy command takes the first source memory contents and copies it into the first destination address.

The interrupt enable statement ON must be accompanied by a STAT statement enabling the interrupts at the hardware level. The first bit of the status register is the enable bit.

```
>STAT=STAT OR 1
```

This should be executed in the main program loop as the interrupt processing subroutine disables the interrupt.

Faster execution times can be obtained by using the following guidelines:
1) Frequently called line numbers and subroutines should be located at the beginning of the program as this is where the interpreter starts searching.
2) All spaces and remarks should be removed. A space takes about 30 microseconds to execute, a remark several milliseconds.
3) The string copy statement copies at the rate of one character in 20 microseconds. The @ operator takes about 5 milliseconds per character.
4) Abbreviations for statements should be employed.
5) As always, well structured programs are the key to speedy executiion.

There are three two byte program pointers that will assist in recovering from a crashed program. These are the beginning of program pointer stored at FFD4 (LSB) and FFD5 (MSB), the end of program pointer at FFD6 and FFD7, and the end of RAM pointer at FFD8 and FFD9.

The command NEW #1200 stores 00 in FFD4 and 12 in FFD5. A non destructive RAM test is performed and the first 100 H bytes of RAM are allocated for the interpreter. The address of the last byte of RAM is stored in locations FFD8 and FFD9.

The command NEW duplicates the beginning of program address in the end of program address location and stores an end of program marker (7F) in that location. This effectively erases RAM. However, the program is still there as evidenced by a memory dump. The program can be recovered by replacing the 7F with any ASCII decimal number (a line number) and restoring the location of the old end of program marker in locations FFD6 and FFD7.

An array can be created in RAM by reserving and zeroing enough space for the array size:

```
FOR I=#1F00 TO #1FFF : @I=0 : NEXTI
```

The variable A is stored in the two bytes 1000H (lsb) and 1001H (msb). If A contains the value of the Ith element of the array:

```
@(#1F00*I*2)=@#1000:@(#1F01*I*2)=@(#1001)
```

will store the value of A in the array. The reverse:

```
@#1000=@(#1F00*I*2):@#1001=@(#1F01*I*2)
```

will put the contents of the Ith element of the array into the variable A.

The following is a description of the error messages:

1	Out of memory space
2	Statement used improperly
3	Unexpected character (after legal statement)
4	Syntax error
5	Value (format) error
6	Ending quote missing from string
7	GO target line not found
8	RETURN without previous GOSUB
9	Expression FOR-NEXT, DO-UNTIL or GOSUB nested too deeply
10	NEXT without previous matching FOR
11	UNTIL without previous DO
12	Division by zero

THE UTILITY LIBRARY

THE SYS-1 library of utilities is designed to enable those generally unfamiliar with the SYS-1 to perform the common tasks of moving a program from EPROM into RAM, modifying it, debugging it, and finally making a new EPROM copy of the finished program. The utilites use menu selection and prompting to reduce the need for the operator to remember key numbers or commands. The SYS-1 Utility library is located at 8800H, thus:

```
 >NEW #8800
```

will cause execution to begin. Remember, SYS-1 will begin execution of an EPROM based program automatically when initialized to a memory location containing the start of a program.

The program presents its menu:

```
LIB-1A *OCTAGON SYSTEMS CORP* (C) 4/82

SELECT
------
<1> MOVE BASIC PROGRAM
<2> COPY MEMORY
<3> HEX DUMP
<4> DEC/HEX CONVERT
<5> LLIST  SERIAL
?
```

Entry of "1" to "5" will cause execution of that subroutine. Entry of anything else will cause the menu to be presented again for reselection. By not crashing with an ERROR statement, the program is as friendly as possible when an out of range number is entered.

<1> MOVE BASIC PROGRAM

Selection of this routine enables the operator to move a program from one location in memory to another, principally from RAM to EPROM and back again.

The routine begins by requesting:

```
SOURCE
<0>         EXIT TO MENU
<>         RAM(#1100)
<2>         EPROM
<ADDRESS> OTHER
```

Since the SYS-1 allows any number of programs to exist in memory at once, the routine needs to know where to find the program to be moved. Entry of "0" is the cancel command, returning you to the menu. Entry of a "1" indicates that your program begins at 1100H, the beginning of RAM allotted for program storage. Entry of a "2" means that the source is the EPROM located in the rightmost 24 pin socket. This socket is mapped at 8000H. Entering any other number indicates the beginning address of your program. The routine will check to insure that the address you gave exists in the SYS-1 memory space. If it does not "NO SOURCE" is printed, meaning that the source you entered is not within the RAM bounds (1100 to 1FFF) or the EPROM bounds (8000 to 8FFF). Once the source is established the routine asks for the destination.

```
DESINATION
<1>        RAM(#1100)
<2>        EPROM
<ADDRESS>  OTHER
?
```

Entry of a "1" will move the program from the source location into RAM beginning at 1100H. A "2" will invoke the EPROM programmer. If neither of these are desired, entry of any other address in the SYS-1 memory range will determine the destination. The program first determines the length of the program to be moved by counting the bytes. This may take several seconds. The number of bytes to be moved is then printed. If the number of bytes is greater than 2048 and you have selected the EPROM programmer, the program prints:

```
NO FIT <CONT>
```

A 2716 EPROM contains only 2048 bytes. If you want to program only that portion of the program that will fit type CONT. The routine will then instruct you to turn on the 25 volt supply. This supply should be 25 +/-1 volt. After the 25 volt supply is energized, press the ENTER key and the programming process begins. As each line is programmed it is printed on the CRT screen. If a discepancy is found as each byte is checked after it is programmed, the programming stops and "BAD BIT" is printed. When the programming is completed the message "TURN OFF +25" is printed.

The location of the EPROM socket is 8000H for reading but the same socket is mapped at 9000H for writing (programming). The reason for this is that had you inadvertently left the 25 volt supply on and typed NEW #8000 you would have destroyed the program in the as the interpreter tests to see if it is RAM. The first statement of any program designed to run automatically should be CLEAR. Without it the interpreter goes into never never land at the first PRINT statement. If this happens, a hardware reset will restore operation.

After the program has been moved into RAM the routine asks:

```
SELECT
------
<0>    RETURN TO MENU
<1>    GO TO DESTINATION
```

Entry of a "0" will recall the menu. Entry of a "1" will cause the internal program pointers to be set so that the current program is the one just moves and the end of program pointer is set so that the program can be modified.

Three pointer values of two bytes each define the RAM area. When NEW #1100 is executed the beginning of program pointer is set to location #1100. The memory locations FFD4 and FFD5 contain 00 and 11. Typing NEW causes the same value to be duplicated in the end of program pointer registers FFD6 and FFD7. The NEW statement also causes a 7F to be stored in location 1100. This is the end of program marker. Thus, if in the course of programming you typed NEW, RAM is not erased even though it appears to be. The program can be completely recovered by first entering an ASCII digit in the first location wiping out the end of program marker.

```
@#1100=#31
```

Next the end of program pointer registers must be restored. Executing the screen dump will identify the last byte of the program - the 7F end of program marker. Enter the two byte value of the memory location in registers FFD6 (lsb) and FFD7 (msb). This will completely recover from a hardware or software reset. The listing on the next page illustrates recovery of a lost program.

```
  >NEW#1100

>  N

>10 PRINT"DEMO"
>RUN
DEMO

>LIST
10 PRINT"DEMO"

>NEW

>LIST

>RUN

> NEW #8800
 LIB-1A *OCTAGON SYSTEMS CORP* (C) 4/82

SELECT
------
<1> MOVE BASIC PROGRAM
<2> COPY MEMORY
<3> HEX DUMP
<4> DEC/HEX CONVERT
<5> LLIST SERIAL
? 3

    HEX DUMP"
<0>        EXIT TO MENU
START AT? #1100
1100 7F 30 20 50 52 49 4E 54 22 44 45 4D 4F 22 0D 7F     .0 PRINT"DEMO"..

STOP AT 37
>PRINT@#FFD6,@#FFD7
 0   17

>@#FFD6=15:@#1100=#31

>NEW#1100

>LIST
10 PRINT"DEMO"

>20 PRINT"OK"
>LIST
10 PRINT"DEMO"
20 PRINT"OK"

>RUN
DEMO
OK
```

(2) COPY MEMORY

The copy memory routine allows you to move any type of data around in the 64K address space of the microprocessor. It is up to you to know that the addresses you give it are valid. This routine can move BASIC progarms as well, but it is not as speedy or convenient as the routine designed to do that.

(3) HEX DUMP

The hexadecimal screen dump displays the contents of 256 memory locations on the screen as 16 rows of 16 locations each. Each row is identified with the address of the first location. Additionally, the ASCII equivalent characters are printed at the right edge of the screen. Those characters that cannot be printed are marked with a ".".

(4) DEC/HEX CONVERT

The decimal to hexadecimal conversion routine is useful to those getting started in microprocessor applications. Enter a decimal number and the hex equivalent is printed. As always, entry of a "0" returns the menu.

(5) LLIST SERIAL

The serial line listing presents the program a line at a time to a serial printer connected in parallel with the CRT. Because the printer is operating "open loop" or without handshaking, a 700 millisecond delay is inserted in line 51. Increasing or decreasing this delay will optimize the routine for an individual printer. If this is too slow, the current 7 can be reprogrammed to a lower figure. Remember - a 2716 is erased to all "1"s. Copy the SYS-1 utility library into RAM and determine the optimum delay (100, 200, 300, etc) for your printer. Place your EPROM into the programming socket and locate the byte containing the "7" in the EPROM. Add 1000H to that location and reprogram the single byte after energizing the 25 VDC supply by executing an @ statement.

```
@(#9000+#54E)=#32
```

The 7 is the 54Eth (HEX) byte. In this example it is changed to a "2". The ASCII code for any digit is 30H

```
ØPR"LIB-1A *OCTAGON SYSTEMS CORP* (C) 4/82"
1S=@#FFD9*256+#E8:T=S+2Ø:U=@#FFD5*256:$S="--------":PR"":PR"SELECT"
2PR$S:PR"<1> MOVE BASIC PROGRAM"
3PR"<2> COPY MEMORY"
4PR"<3> HEX DUMP"
5PR"<4> DEC/HEX CONVERT"
6PR"<5> LLIST SERIAL"
7PR"":INPUTI:PR"":GOTOI*1Ø*(I>Ø)*(I<6)
Y=@(J+I):IF(Y=127)OR(Y>57)OR(Y<49)X=Ø
9RETURN
1ØPR$(U+#77):PR"SOURCE":GOSUB8Ø:GOSUB81:INPUTJ:IFJ=ØGO1
11J=J-#1ØFF*(J=1)-#7FFE*(J=2):V=#11ØØ:W=#8ØØØ:X=#9ØØØ:Z=#98ØØ
12IF(J<V)OR((J>#1FFF)AND(J<W))OR(J>=X)PR"NO SOURCE":GO1Ø
13PR"DESTINATION":GOSUB81:INPUTK:K=K-#1ØFF*(K=1)-#8FFE*(K=2)
14IF((K<V)AND(K>=Ø))OR((K>S)AND(K<X))OR((K>=Z)AND(K<Ø))PR"*NO DEST":GO13
15L=Ø:IF(J=V)AND(@(TOP-1)=127)L=TOP-V-1
16I=Ø:IFK<ØGOSUB82:GOTO19
17$(K+I)=$(J+I):PR$(K+I):DO:I=I+1:UNTIL@(J+I)=13:I=I+1:GOSUB8:IFX<>ØGO17
18@(K+I)=127:IFK<SGOSUB9Ø:GOTO19
19PR"":GOTO1
2ØPR$(U+#93):GOSUB8Ø:PR"DESTINATION";:INPUTK:IFK=ØGOTO19
21PR"SOURCE START";:INPUTJ:PR"SOURCE END";:INPUTL
5DO:@K=@J:K=K+1:J=J+1:UNTILJ>L:GOTO19
3ØM=#8FBF:PR$(U+#A8):GOSUB8Ø:PR"START AT";:INPUTJ:IFJ=ØGOTO1
33PR"":FORI=JTOJ+255STEP16:N=I:LINKM:PR"  ";
35$S="................":FORK=ØTO15:L=I+K:N=@L:LINKM-15
36IF(N>31)AND(N<127)@(S+K)=N
37NEXTK:PR"    ",$S:NEXTI:PR"":GOTO3Ø
4ØM=#8FC2:PR$(U+#BA):GOSUB8Ø
41PR"":PR"DEC=";:INPUTN:LINKM:IFN=ØGOTO19
42GOTO41
5ØPR$(U+#D2),"SOURCE";:INPUTJ:PR"TURN PRINTER ON";:INPUT$T
51PR$J:DO:J=J+1:UNTIL@J=13:J=J+1:DELAY7ØØ:IF@J<>127GO51
52PR"":PR"TURN PRINTER OFF":INPUT$T:GO19
8ØPR"<Ø>        EXIT TO MENU":GO9
81PR"<1>        RAM (#11ØØ)":PR"<2>          EPROM":PR"<ADDRESS> OTHER":GO9
82PR"":PR"LENGTH = ..";:IFL=ØL=J:DO:L=L+1:UNTIL@L=127:L=L-J
83PRL+1,"BYTES":IFL>2Ø47PR"*NO FIT <CONT>":STOP
84PR"TURN ON 25V";:INPUT$T:PR""
85$(K+I)=$(J+I):PR$(K+I-4Ø96):IF@(K+I-4Ø96)<>@(J+I)PR"*BAD BIT":STOP
86DO:I=I+1:UNTIL@(J+I)=13:I=I+1:GOSUB8:IFX<>ØGO85
87@(K+I)=@(J+I):PR"TURN OFF 25V":INPUT$T:GOTO9
9ØPR"":PR$(U+#65):PR$S:GOSUB8Ø:PR"<1> GO TO DEST":INPUTH:IFH<>1GO9
91@#FFD4=@#1Ø14:@#FFD5=@#1Ø15
92I=I+K:IFK>Ø@#FFD6=@#1Ø1Ø:@#FFD7=@#1Ø11


8FBØ    84 F6 FF ØØ ØØ B4 Ø3 ØØ 46 C4 Ø1 CB Ø9 74 ØD 84    ........F.......
8FCØ    F6 FF ØØ ØØ B4 Ø5 ØØ 46 C4 Ø2 CB Ø9 C4 AØ CE FF    .......F........
8FDØ    22 ØØ 1Ø 82 1A Ø9 5E 2Ø E3 8F 9B Ø9 6C Ø4 ØB 4Ø    "....... .......@
8FEØ    74 F5 1E 5C 48 2Ø FØ 8F 4Ø 3C 3C 3C 3C 2Ø FØ 8F    ....H ..@<<<< ..
8FFØ    5C D4 ØF FC ØA 64 Ø4 F4 3A 74 Ø2 F4 41 CE FF 5C    ..........:...A...
```

INPUT / OUTPUT

The 24 programmable I/O lines are divided into 3 addressable ports of 8 bits each. A fourth addressable location contains a control word that determines the configuration of the 24 lines.

LOCATION	FUNCTION
0A00	Port A
0A01	Port B
0A02	Port C
0A03	Control register

A complete description of the capabilities of the 8255A Programmable Peripheral Interface can be found in a data sheet. The following brief description will illustrate some of its capacity.

Reset on power up sets all of the lines in the input state. The function can be changed to all output by executing:

```
>@# 0A03 = #80
```

Port A can now output all "1"'s if:

```
>@# 0A00 = #FF
```

or all zero's:

```
>@# 0A00 = 0
```

All 24 lines can be returned to inputs by storing the control word 9B in location 0A03.

CONTROL	PORT A	PORT B	PORT C
80	O	O	O
82	O	I	I
89	O	O	I
8B	O	I	I
90	I	O	O
92	I	I	O
99	I	O	I
9B	I	I	I

The statement @#A03=#99 will turn A and C into 8 bit input ports and port B into an 8 bit ouput port.

Additionally, port C can be split into two 4 bit ports.

All of the above describes only one of three possible modes of operation (Mode 1).

OUTPUT DRIVE

The output drive capacity of the flag outputs of the 8073 and the 8255 is one TTL load. The 8255 will supply at least 1 ma at 1.5 volts out. This specification is designed to meet darlington transistor output drivers input requirements.

BIT MASKING

The principle of bit masking is useful for controlling a single bit of any of the 8 bit ports. If we assume port A is an output and we wish to turn on only the eighth or PA7 I/O line (on meaning turn it into a "1"). A simple way to do this without affecting the other lines is:

```
@#A00=@#A00 OR #80
```

If we wish to turn it off;

```
@#A00=@#A00 AND #7F
```

BAUD RATE PROGRAMMING

SYS-1 can be programmed for baud rates other than the 4800 baud that is standard. Programming other baud rates is accomplished by breaking a topside foil jumper and installing a corresponding jumper. Locate jumper positions E1 and E2 near pin 9 of U8. Use the following chart to determine which topside connection to break (indicated with an X) and install a jumper as indicated.

```
BAUD RATE    JUMPER
---------    ------
4800         O    O------ E2
             O    O------ E1

1200         O    O------ E2
             O---O--X--  E1

300          O---O--X--  E2
             O    O------ E1

110          O---O--X--  E2
             O---O--X--  E1
```

APPLICATIONS

The SYS-1 bus structure is the most convenient type to add peripherals onto. Three control signals control the operation; Read Data Strobe (NRDS, active low), Write Data Strobe (NWDS) and HOLD (NHOLD). A read memory statement will cause that memory location address to appear on the address bus and the data bus to assume a high impedence input state. When NRDS pulses low the data on the data bus is acquired. Writing to a memory location produces much the same sequence of events. The address appears on the address bus, the data is output on the data bus and NWDS pulses low.

Peripheral inputs and outputs are memory mapped in SYS-1. This means that a memory location is assigned to each peripheral and that memory location's contents are read from (input device) or written to (output device). SYS-1 provides a User Peripheral Select strobe (NUPS) for any memory location between 9800 and 9FFF.

The memory map for the entire system is:

LOCATION	FUNCTION
0000 - 09FF	Internal ROM
0A00 - 0FFF	8255 24 line I/O port
1000 - 17FF	RAM U1
1800 - 1FFF	RAM U2
2000 - 3FFF	Unused
4000 - 7FFF	Baud rate select
8000 - 87FF	EPROM U4
8800 - 8FFF	EPROM U3 (System utilities)
9000 - 97FF	EPROM programmer
9800 - 9FFF	User Peripheral Select (NUPS)
A000 - BFFF	Unused
C000 - FFBF	Baud rate select
FFC0 - FFFF	Internal RAM

An extensive prototype area is provided. All power busses have holes provided for wire wrap "J" pins. Most custom requirements will fit in the prototype area. Those that do not can be built on additional cards and the J1 connector bus utilized. See the schematic for these connections. In general, the loading capability of the microprocessor is 1 TTL load and 10 MOS inputs.

The -5 volt supply should not be loaded with more than 1 ma. If an external negative supply is used the requirements are -3 to -12 VDC at 10 ma.

A hardware reset switch can easily be added by connecting a normally open pushbutton switch across capacitor C7 in series with a 47 ohm resistor. This line is available at pin 18 of the edge connector

TESTING

The following tests are performed before shipment:

RAM test

The INS8073 is programmed to perform a non destructive RAM search and test during initialization according to the following algorithm: for each contiguous byte of RAM the contents are read, complimented, restored as the compliment, read again, checked and the original value restored. This test proceeds through RAM until a failure is noted. This is the last byte of recognized RAM. The last byte of RAM passing this test is available at locations FFD4 and FFD5.

SYS-1 Utility Library

The EPROM supplied to you has been programmed in accordance with the manufacturer's specifications. During programming each byte is verified. Each EPROM is verified before shipment with a checksum. The checksum used is the sum of all the bytes in the EPROM expressed as a two byte number. The checksum for your EPROM is written on the label. You can confirm this at any time with the following statement:

```
J=0 : FOR I=#8800 TO #8FFF : J=@I+J : NEXT I : PRINT J
```

The above statement gives the checksum in decimal numbers. To convert the hex checksum listed on the program label into decimal numbers use the following statement:

```
PRINT #nnnn (inserting the hex number listed on the EPROM)
```

Microprocessor test

The 8073 is tested in circuit by performing the following tasks:

1) A RAM search successfully identifies the first byte of available RAM, 100H is added. The results of this test are available at FFD4 and FFD5.

2) The comprehensive RAM test described above is performed.

3) Baud rate information is properly read.

4) A program stored in EPROM is automatically booted up.

5) A program is written into RAM and run.

I/O port testing

The three 8 bit ports are initialized as inputs. They are then turned into outputs. Data is written into each port driving each port high then low.

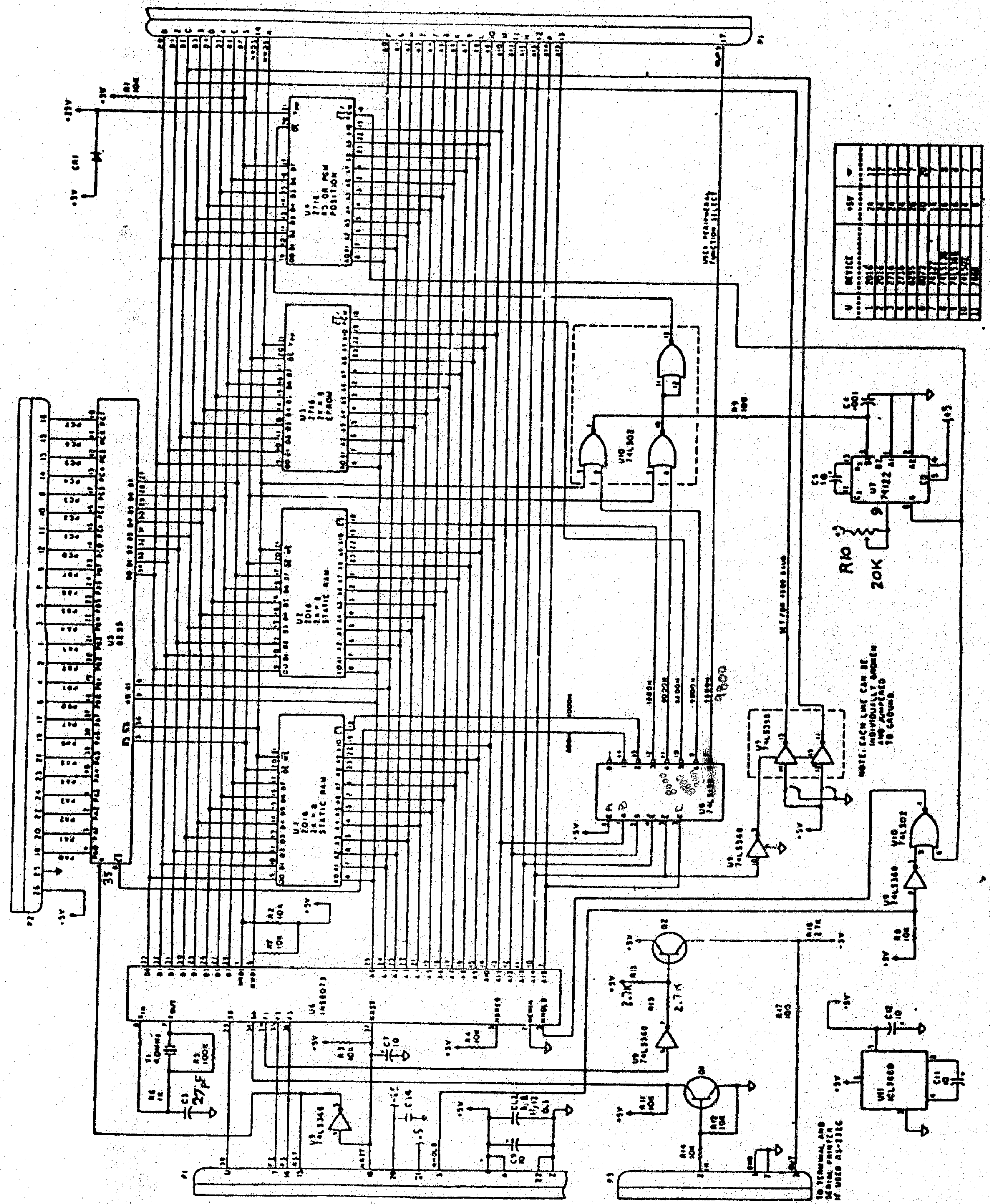
BURN IN

All boards are burned in for 96 hours at 50 degrees centigrade.

FCC CERTIFICATION

This equipment has not been tested for compliance with the new FCC Rules (47 CFR Part 15) designed to limit the interference to radio and TV reception. Operation of this equipment in a residential area is likely to cause unacceptable interference to radio communication requiring the operator to take whatever steps are necessary to correct this interference.

SCHEMATIC SYS-1 REVISION 2

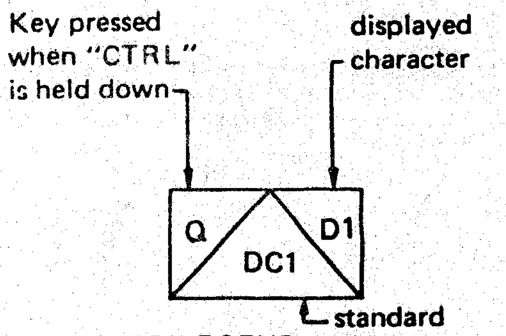

U1 2016 2K x 8 STATIC RAM
U2 2016 2K x 8 STATIC RAM
U3 2716 2K x 8 EPROM
U4 2716 R3 OR PGM POSITION
U5 8255
U6 INS8073
U7 74122
U8 74LS138
U9 74LS368
U10 74LS02
U11 ICL7660
R10 20K
27pF
2.7K
USED PERIPHERAL FUNCTION SELECT
NOTE: EACH LINE CAN BE INDIVIDUALLY BROKEN AND JUMPERED TO GROUND.
TO TERMINAL AND SERIAL PRINTER IF USED RS-232C
U
DEVICE
+5V

Appendix A

ASCII CODED CHARACTER SET

	CONTROL CHARACTERS		DISPLAYABLE CHARACTERS					
BIT 817 → 7 6 5 / BIT 4 3 2 1	000	001	010	011	100	101	110	111
0000	@ NUL NU	P DLE DL	SP	0	@	P	'	p
0001	A SOH SH	Q DCI DI	!	1	A	Q	a	q
0010	B STX SX	R DC2 D2	"	2	B	R	b	r
0011	C ETX EX	S DC3 D3	#	3	C	S	c	s
0100	D EOT ET	T DC4 D4	$	4	D	T	d	t
0101	E ENQ EQ	U NAK NK	%	5	E	U	e	u
0110	F ACK AK	V SYN SY	&	6	F	V	f	v
0111	G BEL BL	W ETB EB	'	7	G	W	g	w
1000	H BS BS	X CAN CN	(	8	H	X	h	x
1001	I HT HT	Y EM EM	)	9	I	Y	i	y
1010	J LF LF	Z SUB SB	*	:	J	Z	j	z
1011	K VT VT	[ESC EC	+	;	K	[	k	{
1100	L FF FF	\ FS FS	,	<	L	\	l	¦
1101	M CR CR	] GS GS	–	=	M	]	m	}
1110	N SO SO	^ RS RS	.	>	N	^	n	~
1111	O SI SI	– US US	/	?	O	_	o	DEL

AK	–ACKNOWLEDGE
BL	–BELL
BS	–BACKSPACE
CN	–CANCEL LINE
CR	–CARRIAGE RETURN
DL	–DATA LINK ESCAPE
D1	–DEVICE CONTROL 1
D2	–DEVICE CONTROL 2
D3	–DEVICE CONTROL 3
D4	–DEVICE CONTROL 4
EM	–END OF MEDIUM
EQ	–ENQUIRY
EX	–END OF TRANSMISSION
EC	–ESCAPE
EB	–END OF BLOCK
ET	–END OF TEXT
FF	–FORM FEED
FS	–FILE SEPARATOR
GS	–GROUP SEPARATOR
HT	–HORIZONTAL TAB
LF	–LINE FEED
NK	–NEGATIVE ACKNOWLEDGE
NU	–NULL
RS	–RECORD SEPARATOR
SI	–SHIFT IN
SO	–SHIFT OUT
SH	–START OF HEADING
SX	–START OF TEXT
SB	–SUBSTITUTE
SY	–SYNCHRONOUS IDLE
US	–UNIT SEPARATOR
VT	–VERTICAL TAB



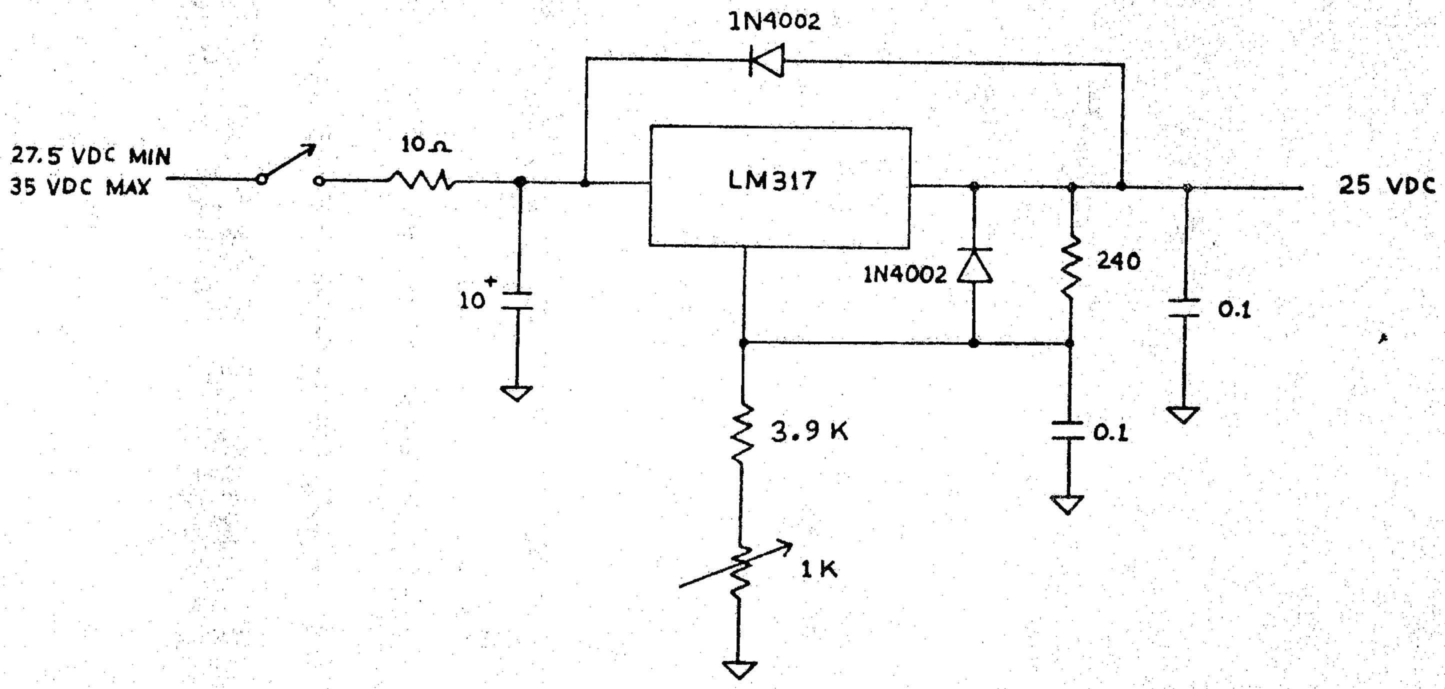
CONTROL CHARACTER LEGEND

APPLICATION NOTE

The 2716 EPROM can be damaged if the programming supply exceeds 26V even for a few microseconds. The symptoms of a damaged EPROM usually are a failure to program or a sudden increase in programming current (several hundred milliamps) followed by a dramatic decrease in the current.

Power supplies should be checked with an oscilloscope for overshoot on power up. Investigation has shown that placing a switch between the EPROM and the programming supply can actually aggravate the overshoot problem. When the switch is closed, the rate of rise of current is limited only by the EPROM impedence and the connecting leads. These leads act as improperly terminated transmission lines and very fast ringing exceeding 33V has been observed (on a 25V supply). The simplest solution is to place a 68-100 ohm resistor in series with the programming supply. THIS SOLUTION WILL NOT SOLVE POWER SUPPLY OVERSHOOT PROBLEMS.

If the power supply overshoots, the circuit below will provide the proper output voltage. Not only will it absorb the overshoot, but its output rate of rise is slow enough that transmission line effects are negligible.



CHANE LOCATIONS

818D

8F2C 8E → 97

8FØD ✓	8F	to	9F
8F11 ✓	8F	to	9F
8F2Ø ✓	8F	TO	9F
8EEE ✓	8F	TO	9F
8EF5 ✓	8F	TO	9F
8EF8 ✓	8F	TO	9F
8FC2 ✓	8F	to	9F
8F4C ✓	8F	TO	9F
8F5Ø ✓	8F	TO	9F
8F63 ✓	8F	TO	9F
8F6C ✓	8F	TO	9F
8F86 ✓	8F	TO	9F
8F8D ✓	8F	TO	9F
8DA2 ✓	8F	TO	9F
8DA8 ✓	8F	TO	9F
8DAD ✓	8D*	TO	9D
8DC8 ✓	8D*	to	9D
8DD2 ✓	8F	TO	9F
8DDB ✓	8F	TO	9F
8DDE ✓	8F	TO	9F
8DEE ✓	8D*	to	9D
8EØ3 ✓	8F	TO	9F
8E2Ø ✓	8F	TO	9F
8E26 ✓	8E	To	9E
8E2B ✓	8E	to	9E
8E33 ✓	8E	to	9E
8E6C ✓	8F	TO	9F
8E72 ✓	8F	to	9F
8E82 ✓	8F	TO	9F
8E92 ✓	8F	TO	9F
8E9B ✓	8F	TO	9F
8E9F ✓	8F	TO	9F
8EAF ✓	8F	to	9F
8EB4 ✓	8F	TO	9F
8EBA ✓	8F	TO	9F
8EBD ✓	8F	TO	9F
8ED2 ✓	8F	TO	9F
8ED5 ✓	8F	TO	9F

8FA6 ✓	8F	To	9F
~~8FC2~~	~~8F~~	~~to~~	~~9F~~
8FCC ✓	8F	TO	9F
8FD2 ✓	8F	TO	9F
8FD9 ✓	8F	TO	9F

8982 ✓	38	to	39
8A52 ✓	38	to	39
8A6C ✓	38	to	39
8A98 ✓	38	TO	39
8AFD ✓	38	to	39
8B6D ✓	38	to	39
8BBA ✓	38	to	39
8BDC ✓	38	to	39
8C2Ø ✓	38	TO	39
8C4Ø	38	to	39
8C72 ✓	38	TO	39
8CB6 ✓	38	to	39
8CF2 ✓	38	to	39
8C48	#38	to	#39

0	8
1	9
2	A
3	B
4	C
5	D
6	E
7	F