SYSTEM ONE SERIES - PROFESSIONAL UTILITY - 1.0

(c) Octagon Systems Corporation 12/81

Programming tasks can be greatly simplified by using a fast, efficient utility library. The Octagon Professional Utility 1.0 is a powerful debugging tool yet can be used by those who have just begun to program. It is equally useful for the Basic language as well as the object code programmer. A combination of Basic and object code is used with all actual work being done in fast machine code. This 2K utility will work only if placed in the socket mapped at 8800H. In the following descriptions, all addresses and numbers are hexadecimal, except where they would obviously be commonly expressed as decimal (e.g., a 2K EPROM has 2048 bytes). To run the utility, type:

```
>NEW#8800
```

The menu will appear on the screen as shown below:

```
  (C) 1981 OCTAGON SYSTEMS CORPORATION - SERIES ONE UTILITY 1.0

 <0> SCAN MEM    <4> PROG EPROM     <8> LP HEX DUMP    <12> GOTO #8000
 <1> FILL MEM    <5> DEC/HEX        <9> INITIALIZE
 <2> COPY MEM    <6> HEX DUMP       <10> ERASE CHECK
 <3> POKE MEM    <7> LP PROGRAM     <11> CHECKSUM

COMMAND ?
```

Below is an explanation of all the menu utilities:

<0> SCAN MEMORY

It is often useful for object code programmers to scan large amounts of memory to find a key section. This can be done with a normal hex dump but that displays only 256 bytes at a time. This utility will scan 4096 bytes per starting address. The scan rate is 5-6 lines of 16 bytes per second. The command sequence is as follows:

```
 COMMAND ? 0                       ;Select Scan Mem

 ADDRESS ? #8800                   ;Scan this utility

8800 31 50 52 22 22 3A 50 52 22 22 0D 32 20 50 52 22 1PR"":PR"".2 PR"
8810 20 20 28 43 29 20 31 39 38 31 20 4F 43 54 41 47   (C) 1981 OCTAG
  "                                                          "
  "                                                          "
  "                                                          "
  "                                                          "
8FE0 09 58 5C 06 D4 10 6C 01 5C 24 D9 01 26 10 82 .X.......$..&...
8FF0 0E 8D F6 82 10 8D F8 A2 16 82 14 47 82 12 46 5C ...........G..F.
```

On the left margin is the address in hex of each line. It is followed by a hex representation of each byte. On the far right are those same 16 bytes but in ASCII format. All bytes from 20H to 5FH displayed as their ASCII equivalents while all other codes are converted as a period ".". No lower case alpha characters are shown as these are not legal with the INS8073 interpreter. Note that this utility begins with Basic text (or program) and ends with object code.

<1> FILL MEMORY

This utility has two major uses. After revising a Basic program several times, a hex dump of the end of the program might look like the following:

```
1890 45 41 44 59 22 0D 7F 50 52 22 22 3A 50 52 22 52 EADY"..PR"":PR"R
18A0 45 41 44 59 22 0D 7F 22 52 45 41 44 59 22 0D 7F EADY".."READY"..
18B0 52 22 52 45 41 44 59 22 0D 7F 50 52 22 52 45 41 R"READY"..PR"REA
18C0 44 59 22 0D 7F 03 15 20 03 03 03 CA CD CF A8 F0 DY".............
```

Your program is now finished and you want to commit it to EPROM. Since you are a perfectionist, you want to eliminate the garbage at the end of the program. The end of program marker is 7F immediately after an OD (carriage return). There are four 7F's and you think that the first one at 1896 is the correct one. To confirm this you type:

```
 >PRINT TOP
 6295                              ;It answers
 >PRINT #1896                      ;You cleverly type
 6294                              ;it answers again
```

Since TOP is the next available byte for program storage, the guess was correct. Now to clean up the program:

```
COMMAND ? <1>

 SOURCE START ADDR ?  6295      ;or #1897
 SOURCE END ADDR   ?  #18FF     ;18FF-1100 = 2048 Bytes
 FILL VALUE        ?  #FF       ;Value of unprogrammed byte
```

Now when you program the EPROM, all those FF's will be left unprogrammed for future use. The second major application is "zeroing" memory. This is done to prepare a "virgin" space in which to copy programs so that the endings and beginnings are immediately apparent.

<2> COPY MEMORY

This utility will copy a block of RAM or EPROM to any part of RAM. The destination will then equal the source. The souce will remain undisturbed. CAUTION: Do not copy anything into the memory location reserved for the interpreter (1000-10FF and FFC0-FFFF) or the system will crash. The interpreter occupies 0000-09FF. Trying to copy into this location will have no effect on the system. The command sequence is as follows:

```
COMMAND ? 2

 DESTINATION ADDR   ?  #2000
 SOURCE START ADDR ?  #1000
 SOURCE END ADDR    ?  #10FF
```

The code is moved instantly to the new location.

<3> POKE MEMORY

This utility is used almost exclusively for object code programming. You enter a beginning address and it automatically indexes the address after each input.

```
COMMAND ? 3

ADDRESS ? #2000

2000 ?  56
2001 ?  23
2002 ?  12
2003 ?  98
2004 ?  76
```

This will go on forever. To get out of this routine, you must enter Control C. Then type RUN, which brings you back to the menu.

<4> PROGRAM EPROM

The routine is about as foolproof as they come. It checks every byte BEFORE it programs to insure that it doesn't write over an existing program. If it finds a preprogrammed byte, an "OVERLAP AT nnnn" message occurs (nnnn is the address of the offending byte). After programming each programmed byte is checked for accuracy against the source. If there is a difference, a "MISMATCH AT nnnn" message is printed on the screen. Note that this utility operates differently than the utility supplied with your system. The EPROM programmer is mapped at 9000 but you address it at 8000 in this routine. The offset is handled automatically. You can program any portion of the EPROM and add sections later. CAUTION: Typing NEW #mmmm where mmmm is any number from 9000 to 97FF will write a byte into the EPROM. This is harware, not software dependent. The command sequence is below:

```
COMMAND ? 4

 DESTINATION ADDR  ?  #8000
 SOURCE START ADDR ?  #1100
 SOURCE END ADDR   ?  #18FF
 PROGRAM SUPPLY ON ?                  ;<ENTER> when accomplished
 STANDBY 102 SECONDS                  ;Time to program 2048 bytes
 PRGRM SUPPLY OFF  ?                  ;<ENTER> when accomplished
```

The menu now reappears. According to EPROM manufacturers the fastest that a 2K EPROM can be programmed is 2048 x 50 milliseconds = 102.4 seconds. This program is only slightly slower. If an error message is appropriate, it will appear after the "STANDBY ...." line and will be offset one space to the left. If you forget to put an EPROM in the socket, an "OVERLAP....." will occur since the data lines will be floating and no FF’s will be read from that socket. To break out of this routine, either hit the CPU reset button or turn the programming supply off. CAUTION: It is very important that the programming supply be 24V - 26.5V and that it not overshoot even momentarily above 26.5V. Permanent damage to the EPROM will result. The best method is to put a switch in series with the output of the supply and turn the supply off and on only when the programming switch is open.

<5> DECIMAL TO HEX CONVERSION

Convert an decimal number from -32767 to +32767 to a four digit hex number. Entering a zero will recall the menu.

```
 COMMAND ? 5

 <0> = EXIT

DEC ?  32000
HEX    7D00

DEC ?  14
HEX    000E

DEC    -1
HEX    FFFF
```

<6> HEX MEMORY DUMP

The format is exactly the same as Scan Memory except that only 16 lines are displayed at a time. This is followed by a prompt for another address. Entering "1" will recall the menu.

```
COMMAND ?  6

ADDRESS <1> = EXIT ?
```

<7> LINE PRINTER PROGRAM LIST

This routine works just like the LIST command of the interpreter excepts that it waits at the end of each line for the printer to do a carriage return. The delay is computed as follows: 1.0 seconds are allowed for the printing of the line (printer must print at least 72 characters per second) and 0.5 seconds for a carriage return and line feed. Since the Basic program lines will rarely be the full 72 characters long, the wait will seem long for short lines. The process could be speeded up by using one of the "handshake" lines from the printer. However, this would require modification of the system circuitry. The program assumes a valid Basic program with a 7F end of text marker. If you enter an address which is not ASCII code, these will be sent to the line printer with unpredictable results (harmless, however). Hitting any key will exit to the menu.

```
COMMAND ? 7

ADDRESS ? #1100
```

<8> LINE PRINT HEX DUMP

This is the routine you want to use to dump nonASCII code. The format is the same as the Hex Memory Scan except that it will print between the two address you specify. The timing requirements are the same as in commmand 7.

```
COMMAND ? 8

SOURCE START ADDRESS ?  #2000
SOURCE END ADDRESS   ?  #21FF
```

<9> INITIALIZE

This program will set the Basic Interpreter pointers so that you can modify, list, and run any number of programs in EPROM or RAM. It prompts you for the beginning address of the Basic text. It then analyzes the text to insure that (1) the address you entered is really the beginning of a Basic program and (2) that a valid program exits.

If it fails either of the tests, a "NOT VALID TEXT" error message appears.

```
 COMMAND ? 9                               ;You are now at 8800

 ADDRESS ? #1100                           ;Your program location

READY

 >                                         ;Now at 1100
```

<10> ERASE CHECK

All 2048 bytes of an EPROM at 8000 are checked for FF's. If none have been programmed, an "EPROM OK" message will appear followed by a return of the menu. A byte that is not FF will envoke a "BAD BYTE AT nnnn" display.

```
COMMAND ? 10

EPROM OK
```

<11> FIND CHECKSUM

This utility has numerous important uses. You will notice that a four digit number appears on the label of this utility EPROM. That is a two byte checksum of all the bytes on the chip. To assure yourself that none of the bits have "flipped" back, do the following:

```
COMMAND ? 11

 SOURCE START ADDR ? #8800
 SOURCE END ADDR   ? #8FFF
 CHECKSUM =          kkkk H
```

kkkk should exactly equal the checksum on the label of this utility. If you have a power supply that does not adequately reject severe bursts of line noise, you may occassionally get your RAM "glitched" while you are on coffee break. Running a checksum before you leave and comparing it to the checksum when you return can save a measure of grief. You can also compare blocks of memory with each other because they must have identical checksums. Unmarked EPROMs can be identified by their checksums (if you remembered to write it down on the printer listing, etc.). NOTE: CHECKSUM ON ORIGINAL UTILITY IN DECIMAL

<12> GOTO #8000

This routine can only be used to link to object code programs at 8000. It will NOT run a Basic program. It was added to allow expansion of this utility to other useful programs that Octagon will introduce. You will be able to jump back and forth between the two utility program as if they were one 4K EPROM.

We are always interested in offering new support products, both software and hardware, for the SYSTEM ONE series. Please give us your comments.

PLEASE REMEMBER THAT THE SOFTWARE AND SUPPORT MATERIAL FOR THIS UTILITY ARE COPYRIGHTED WITH ALL RIGHTS RESERVED.

END USER — SUBLICENSE AGREEMENT

SYSTEM ONE SERIES -- PROFESSIONAL UTILITY

In consideration of payment by Customer for Professional Utility and related materials, Octagon Systems Corporation ("Octagon") grants to the customer a non-exclusive and nontransferable sublicense to use Professional Utility and any and all related documentation and information in conjunction with a standard Series One computer system manufactured by Octagon. This sublicense is for a single computer system as described in the previous sentence.

Customer agrees with Octagon that all or any part of Professional Utility and related documentation and information are trade secrets of Octagon, are protected by civil and criminal law and by law of copyright, and are very valuable to Octagon. Customer shall not provide or otherwise make available all or any part of Professional Utility and related materials in any form to any third party. Customer acquires no title or ownership rights to Professional Utility by this transaction.

Customer understands and agrees that she/he will not copy, duplicate, reprint or modify all or any part of Professional Utility and related documentation and information. Customer further understands and agrees that she/he will not create or attempt to create, by reverse engineering, modification or otherwise, the source programs or any part thereof from the Professional Utility object program or from other information made available under this agreement or otherwise.

OCTAGON AND SELLER MAKE NO EXPRESS WARRANTIES AND DISCLAIM ALL IMPLIED WARRANTIES WITH REGARD TO PROFESSIONAL UTILITY, INCLUDING ALL WARRANTIES OF MERCHANTABILITY AND FITNESS AND CUSTOMER WAIVES ALL CLAIMS AGAINST OCTAGON, SELLER AND ANYONE ELSE IN OCTAGON DISTRIBUTION NETWORK FOR SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF PROFESSIONAL UTILITY.

By ____________________
For Octagon Systems Corporation

===

STOP — READ THIS PAGE BEFORE OPENING

Professional Utility is sold on the condition that the customer agrees to the above license. READ THIS LICENSE CAREFULLY. By opening this package, you signify that you agree to the terms of the license.