IMDOS SYSTEM ALTERATION GUIDE

8/25/78

81-0000056 Rev. G

August, 1978

TABLE OF CONTENTS

IMDOS ALTERATION GUIDE

1. INTRODUCTION

This chapter describes a number of modifications which can be made to IMSAI IMDOS by the user. Extensive technical background is given to facilitate making these changes and other changes not anticipated by the authors. Some of the changes relate to the layout of a specific diskette, such as changing the number of entries in the file directory; others relate to the operating system's capabilities, such as adding a driver for a new I/O device.

Some changes are accomplished with console commands, particularly GENESYS and FORMAT. Others are done by modifications of the code of part of the IMDOS system, which can be done by "patching" for minor changes or by editing and reassembling the source file for more extensive changes. The source file for one module of IMDOS, BIOS.MAC, is supplied on the distribution diskette. The assembly listing for this module is given in Appendix A of this chapter, and a detailed program description is given in Section 4.4. BIOS is the portion of IMDOS whih contains all non-disk I/O drivers.

To get an idea of the types of changes described, the reader may wish to read Section 4.5, "Some Common Alterations", first, then come back to preceding sections.

This chapter assumes the reader is familiar with the "IMDOS System User's Guide" and the "IMDOS Programming Guide"; parts of it assume a familiarity with DDT and with 8085 assembly language programming.

2. IMDOS DISKETTE LAYOUT

This section describes the organization of an IMSAI IMDOS diskette. The physical diskette format is discussed as well as the logical use of the diskette space by IMDOS. The conventions for hardware configuration of the disk interfaces are described.

2.1 Areas on the IMDOS Diskette

The IMDOS diskette contains two major areas, the system area and the file area. Two more significant features are the Volume Control Block, which is within the system area, and the directory, which is within the file area.

The SYSTEM AREA normally occupies the first 1 to 3 tracks of a diskette. When the system is cold- or warm-started from a diskette the code loaded into RAM comes from the system area. A system image is written into the system area with SYSMOV; a diskette may hold files even if no system image has been written into its system area.

The FILE AREA is the rest of the disk; its space is allocated by BDOS in 1K or 2K byte units to files as they are written.

The DIRECTORY is a portion of the file area which is reserved when the diskette is initialized with FORMAT. The system maintains a 32-byte entry for each extent of each file in the directory.

The VOLUME CONTROL BLOCK (VCB) is a portion of sector 1, track 0 which is in the system area. It contains variables specifying the size and location of the different areas on the disk, as well as the number of tracks, sectors, etc. on the disk and the user-specified volume name.

The VCB is initially written on the disk when it is FORMATted; it is preserved by being copied into the new system image when SYSMOV or GENESYS writes into the system area. The VCB is the center of the mechanisms which permit IMSAI IMDOS to support a variety of floppy disk capacities and formats simultaneously, as well as allowing the user a number of options in diskette layout.

2.2 Diskette Sector Formats

IMDOS Ver 2.05 supports a number of different diskette formats, both single and double density. In order to use any of the formats, the computer system must have a floppy disk controller capable of the format of interest. It is possible to have several interfaces each of which has different capabilities; IMDOS will give you access to the formats of which the interfaces are capable, and give you an error if you try to set a format which the interface cannot support. At run time, IMDOS receives all needed information about the diskette's format from the VCB. Interfaces are automatically switched to the diskette format required except for format V. All IMSAI formats are soft-sectored.

There are a total of five formats currently supported by IMSAI IMDOS and disk interfaces:

FORMAT	DENSITY	SECTOR LENGTH
I	Single	128
II	Single	256
III	Double	256
IV	Double	1024
V	Double	128

STANDARD DISKETTES

On standard size diskettes, format I is the single density IBM 3740 physical format, which is the default format for standard drives. IMSAI software is distributed using this format. Virtually all microcomputer disk systems using standard diskettes can use this IBM 3740 physical format.

The next three formats (II, III, IV) on standard size diskettes are the more recently announced IBM physical formats. Format II is an additional single density format, and formats III and IV are the IBM dual density formats. Format III is the IMSAI default format for dual density.

Format V is the initial double density format used by IMSAI (DIO rev 0 firmware). We recommend updating the DIO firmware so that Format V is not used, and that you use instead either format III or IV. IMDOS still supports the original double density format, so no change is required if you have only this double density capability. It is, however, possible to update older machines to gain the capability of formats III and IV. The advantages of updating are:

1) The large safety margins you'd expect of an IBM format (regarding gap sizes, speed variations, and similar physical considerations).
2) Greater interchangeability with other computer systems as they adopt this new standard.
3) Slightly greater storage capacity using format IV.
4) Automatic density/sector size sensing and accomodation.

TRACK 0

A significant feature of the IBM formats is that regardless of the density or sector size, track 0 is always formatted with 26 sectors of 128 bytes, single density. This is the same as throughout diskettes formatted in format I.

Because of this feature, it is possible for IMDOS to determine the diskette format by reading the VCB from the first track. Using the information in the VCB, IMDOS sets the interface to the appropriate density for further operations. This occurs anytime IMDOS "mounts" a disk, so in normal operation the operator need not be concerned with diskette format. The only times input regarding format is required are in running the FORMAT program and setting the DIO option switch for cold-booting. All other times the format is changed automatically. This is not true for format V.

Under the IMDOS system, information about the format of the diskette is contained in the VOLUME CONTROL BLOCK (VCB).

MINI DISKETTES

IMDOS supports formats I, II, III, and IV on mini diskettes. The only format changes from the standard size diskettes are in the number of tracks and the number of sectors on each track. All formats on mini diskettes have the same format on track 0, which is format I (single density, 18 sectors of 128 bytes each). IMDOS currently supports minis with 35, 40, and 77 tracks.

SUMMARY OF FORMAT DATA

FORMAT - - -	I	II	III	IV	V
Density	Single	Single	Double	Double	Double
Record Format	FM	FM	MFM	MFM	MFM
Bytes/Sector	128	256	256	1024	128
Sec/Tk Std	26	15	26	8	58
Sec/Tk Mini	18	9	17	5	
Tk 0 density	Single	Single	Single	Single	Double
Tk 0 Bytes/Sec	128	128	128	128	128
Sec/Tk 0 Std	26	26	26	26	58
Sec/Tk 0 Mini	18	18	18	18	
Diskette Capacity in 80H blocks (128)					
Standard	2002	2306	3978	4890	4466
35 tk Mini	630	630	1174	1378	
40 tk Mini	720	720	1344	1578	
77 tk Mini	1386	1386	2602	3058	
Diskette Capacity in 400H blocks (1024)					
Standard	250.25	288.25	497.25	611.25	558.25
35 tk Mini	78.75	78.75	146.75	172.25	
40 tk Mini	90	90	168	197.25	
77 tk Mini	173.25	173.25	325.25	382.25	
Diskette Capacity in Bytes					
Standard	256256	295168	509184	625920	571648
35 tk Mini	80640	80640	150272	176384	
40 tk Mini	92160	92160	172032	201984	
77 tk Mini	177408	177408	333056	391424	

2.3 DISK INTERFACE ADDRESSING

IMDOS 2.05 has extended disk capabilities from versions 2.01 and 2.02. The number of DIO type interfaces is no longer limited to two, though the number of FIF type interfaces is still limited to one. The physical drive numbers assigned to drives on a FIF interface remains the same (1:, 2:, 3:, 4:). The physical drive numbers assigned to drives on DIO type interfaces changes, as well as being extended. The change is due to allowing for a maximum of 8 drives per interface rather than 7 (although no version of

the interface actually supports 8 drives at once).

The I/O port number addressing of the DIO type interfaces has also been modified, to allow for a logical extension up to 11 interfaces. (Actual number of interfaces possible in a system will be limited by the number of available backplane slots.)

!!!!! NOTE !!!!!

In each system, one interface is designated as primary. It has one jumper set differently from the others so that a hardware reset enables it for operation. For IMDOS 2.01 or 2.02, this board is addressed in the D block (or, for single interface systems, NOT in D or E blocks). For IMDOS 2.05 this board must be switched to block E on the board address DIP switch.

Other interfaces must be located in consecutive blocks down from the first at E (i.e., second at D, third at C, etc.).

!!!!!!!!!!

IMDOS PHYSICAL DRIVE NUMBERS

Interface Port Address	Block Address	Interface	Physical Drive Numbers
Version 2.01, 2.02			
DE,DF	D	Primary	5: to 11:
EE,EF	E	Secondary	12: to 19: *
Version 2.05			
EE,EF	E	Primary	5: to 12:
DE,DF	D	Secondary	13: to 20: *
CE,CF	C	Tertiary	21: to 28: *
BE,BF	B	.	. *
	.	.	. *
		.	. *
		.	. *

* Can't cold boot

3. THE SYSTEM IMAGE

This section gives background useful for some of the descriptions of system modifications given in following sections. The "System Image", which is the system as it exists on the system area of a diskette, and as it is moved to and from the Transient Program Area of RAM (TPA) with SYSMOV, is described. The source code for the BIOS module of the system (portion of the "System Image") is supplied on the distribution diskette and extensive information is given later in this chapter on modifying this module and on incorporating the modified code into the system.

3.1 IMDOS Memory Organization

The sections of the system in memory when the system is running are:

```
System Parameters Area  (locations 0-FF hex)
Transient Program Area  (TPA)
System area containing:
        Console Command Processor (CCP)
        Basic Disk Oper. System (BDOS)
        Initialization module (ENTRY)
          Single Density module (SDENS)
                or
          Double Density module (DDENS)
        Basic I/O System (BIOS)
Space for Disk Mount information
Space for File Buffers
```

Refer to "IMDOS Programming Guide", particularly Section 2, for some more information about these sections.

3.2 Introducing BOOT

BOOT is an important module of the IMDOS system which has not yet been introduced. BOOT is a 100 hex byte program which loads the rest of IMDOS into memory. BOOT occupies two sectors on disk. The first sector is read by BDOS, in the case of a "warm" restart, or by external firmware when the system is cold-started. Code in the first sector reads the second sector to locations 80-FF hex, after which BOOT reads in the rest of the system and performs some initialization functions. BOOT is only used while the system is being read in; the system parameters area overwrites BOOT after the system is loaded.

3.3 The System Image

The SYSTEM IMAGE is the system as it exists on the system area of the diskette and as transferred to and from the TPA with SYSMOV or GENESYS. The System Image is loaded (booted) into RAM for execution by BOOT. The system modules as described above are loaded at high addresses, leaving space for file buffers and disk mount information between the system and the top of memory.

In IMDOS Ver 2.01 and 2.02 changes were often made by patching the System Image in the TPA (via running SYSMOV under DDT). In IMDOS Ver 2.05 this is at once more difficult and less necessary. Since the system modules are relocatable, it is difficult to say exactly where each module is in the System Image. Since the modules are relocatable, though, it is a much simpler task to incorporate a module assembled from a modified source. Even large expansions in the system size created by adding new drivers are easily incorporated simply by correctly answering the prompts in the GENESYS program, which is used to create the System Image from the relocatable system modules. The new module must simply first be assembled using our standard relocating assembler (M80, included on the FORTRAN system diskette).

3.4 The System Image in the TPA

Since the system area on diskette is outside of the file area the special program SYSMOV is used to write the system image onto the diskette ("PUT" the system) and to read the system from the diskettes ("GET" the system) in preparation for writing it onto a different diskette or patching the system image.

Basic SYSMOV useage is described in the "IMDOS System User's Guide"; details are added in the following section and Section 5.

SYSMOV transfers the System Image from and to the TPA area of RAM, starting at location 600 hex. SYSMOV itself occupies locations 100-5FF hex.

3.5 Patching the System Image

Minor changes can be accomplished by "patching", or altering the binary System Image instead of the source. We recommend that you not use this method for changes, as you maintain much more control over the situation by modifying and properly commenting the source. However, for those who may insist on the route of patching, we provide some information here to maximize your chances of success.

To patch the System Image, you need to get it into the TPA. The procedure is to run SYSMOV to put the System Image in the TPA; SAVE the image on disk; call in DDT and the image to be modified; modify using DDT; return to system; save patched image if desired; run SYSMOV again to put System Image on new diskette system area. For example:

```
SYSMOV N-Y              ;put drive A: system image in
                        ;TPA

SAVE 48 NEWSYS.COM      ;save in COM file

DDT NEWSYS.COM          ;bring back into memory with
                        ;DDT
                        ;make your changes

GO                      ;return to system

SAVE 48 PATSYS.COM      ;save patch version if desired

SYSMOV Y=N              ;write new system to B:
```

Your patches will invariably be in BIOS, the I/O driver module. The source listing for BIOS is provided. Note the conditional assembly control (to create BIOS, BIOS4, and BIOS22). If you need a source listing for one of the versions other than that printed here, you can use M80 to assemble the source with the appropriate conditions set, and list the resultant print file. To find BIOS code in the TPA use these base addresses. They differ depending on whether extended length sector capability was GENESYSed in. For 128 byte systems, BIOS starts at 1D1FH in 2.05. For extended length sector systems, BIOS starts at 1F1EH in 2.05.

From here on it's up to you. If all else fails, edit your changes into the source, reassemble, and run GENESYS.

3.6 Relocating the System Image

The unmodified system image can be relocated for larger or smaller amounts of RAM with the GENESYS console command with parameters. The reader should refer to the description of the use of this command in the "IMDOS System User's Guide" (look for GENESYS in the contents). We will reiterate here that GENESYS permits placing the relocated system at its execution address, to run until the next warm restart, or to write it onto the system area of a diskette, to be run after the system is cold-started or warm-started from that diskette. The use of the GENESYS command without parameters invokes the selection of many variables besides the memory size, and is described in further detail in this chapter Section 4.2.

4. MODIFYING THE SYSTEM IMAGE

This section gives techniques for incorporating changes in the IMDOS system. Changes may be simply relocating the system for use in a system with larger or smaller memory, or they may involve a re-written set of I/O drivers for your own special hardware configuration.

Most common changes can be accomplished using the built-in options in the FORMAT and GENESYS programs. These permit you to configure a system for various size memories, for a number of different diskette formats, and for several common variations of line printer interface.

For extensive changes of the I/O interfacing, a source file of the I/O drivers is provided (BIOS), along with relocatable object files of other necessary parts of the system. To compile your modified drivers, you will need the M80 assembler (which produces relocatable object code). M80 is included as part of the FORTRAN diskette, along with linking loader and library utilities for advanced program development. Once your new drivers are compiled, the GENESYS and FORMAT utilities will automatically create a new system incorporating your new drivers using only normal parameter selection options.

4.1 ALTERATION WITH FORMAT (VCB options)

There are several parameters in the VCB which the user may optionally specify during execution of the FORMAT program. These often must be changed to accomodate a modified SYSTEM

IMAGE, and a number of useful changes are accomplished by changing these parameters only. This section describes the parameters, acceptable changes to them, and considerations in changing them. FORMAT is normally run only when initializing a diskette, as the value of many VCB variables MUST correspond with the physical diskette format. Options in FORMAT permit changing the VOLUME NAME without changing any other VCB variables. The format options can be totally bypassed by entering the FORMAT command with a drive logical name on the command line (e.g. FORMAT B:). For further information about this mode and other general information, refer to the chapter "IMDOS System User's Guide" for a description of the use of FORMAT. (Look up FORMAT in that chapter's table of contents.) This section describes in more detail the options available for each of the VCB variables.

In order to display the current VCB variables on the formatted diskette d:, use the command STAT d: VCB. Note the space between the d: and VCB. Following is a typical listing from this command:

```
A>STAT B: VCB

VOLUME CONTROL BLOCK DRIVE B:
VOLUME NAME: TEMP 01
SECTOR SPACING                          6
NUMBER OF DIRECTORY ENTRIES            64
# OF SECTORS FOR SYSTEM AREA            52
DIRECTORY OFFSET IN  1K UNITS           0
TOTAL UNITS INCLUDING DIRECTORY       243
26 SECTORS    77 TRACKS
128 BYTES PER SECTOR

A>
```

If you desire to select other than the default values for the VCB, invoke FORMAT with no parameters (i.e. after the prompt (d>), type FORMAT<cr>), enter the drive number and volume name as requested, then type "*"<cr> in response to the question:

```
OK TO GO (Y, N, V, OR *)
```

FORMAT will then ask for a series of values which define the diskette layout. These may be changed from the default values according to the diskette usage requirements. If * is not answered, then default values appropriate to the drive type and format I (single density) or fomat III (double density) are used. The options will be:

A USE OLD VCB FOR DEFAULT VALUES
B NUMBER OF 80H BLOCKS PER SECTOR (sector size)
This is asked only if your interface will support long sectors.
C SECTOR SPACING (sector skew for timing)
D NUMBER OF DIRECTORY ENTRIES
E NUMBER OF 80H BLOCKS FOR SYSTEM AREA
F DIRECTORY OFFSET

A. USE OLD VCB FOR DEFAULT VALUES ? (Y OR N)

A yes answer is useful for changing a VOLUME NAME only on diskettes with existing files. Otherwise you must know and correctly enter all VCB variables the same as before in order to continue to use the files on the diskette. This may also be used if you wish to re-initialize the diskette (destroying any files on it) and reformat with the same configuration as before. If you wish to write a VCB only and maintain your old files, type "V" in response to:

OK TO GO (Y, N, V, OR *)

B. NUMBER OF 80H BLOCKS PER SECTOR

If you have configured (or intend to configure) your SYSTEM IMAGE to access diskettes with extended sector lengths, you may format a diskette with long sectors. This option is not offered to you if the drive interface or firmware cannot support longer sectors or if you are using the old VCB for default values. Your choices are:

SINGLE DENSITY

1 block per sector (128 bytes per sector) (default)
2 blocks per sector (256 bytes per sector)

With 2 blocks per sector on standard diskettes you gain only 15% capacity, and require more SYSTEM memory to accomodate file buffers. We recommend using 1 block per sector (IBM 3740 physical format) unless you require physical compatibility with the 256 byte/sector single density IBM format. The most common single density format is the 128 byte sector format.

On mini diskettes, 2 blocks per sector gives you no added capacity, but provides greater tolerance for spindle speed being out of adjustment. The same requirements for more SYSTEM memory apply.

DOUBLE DENSITY

2 blocks per sector (256 bytes per sector) (default)
8 blocks per sector (1024 bytes per sector)

A sector length of 1024 bytes gives significantly more storage capacity than a sector length of 256 bytes, at the expense of larger SYSTEM memory requirements to handle the larger file buffers.

See GENESYS section for discussion of exact memory requirements for SYSTEM file buffers.

NOTE: The Disk Drive density must be set (using STAT) to single density or double density as appropriate to your desired format BEFORE FORMAT is run (unless it is already set to the desired density or a disk with a good VCB of the correct density is in the drive when you answer yes to "use old VCB for default values?"). Use STAT DSK: to see which logical drives are set to double density.

C. SECTOR SPACING

This is the number of physical sectors between logically adjacent file sectors. Reducing this to the minimum the particular interface can handle without dropping revolutions will maximize speed of program loading and other operations which do little processing between sector reads; increasing it may improve performance of programs that do a known, finite amount of processing for each sector. Since this parameter (as all VCB parameters) applies to the entire diskette, files requiring different sector spacings should be on different diskettes.

The following table lists default sector spacing and additional time required for each additional sector to pass.

FORMAT - - -	I	II	III	IV	V
Default Sector Spacing					
Standard	6	4	5	3	5
Mini	3	4	5	3	
Milliseconds per Sector					
Standard	6.4	11.1	6.4	20.8	2.9
Mini	11.1	22.2	11.7	40	

Improvement over the default values is available on standard diskettes that will be restricted to DIC type interfaces by reducing the sector spacing (try 3 for format I). This will severely slow operation if the diskette is used on a FIF type interface.

D. NUMBER OF DIRECTORY ENTRIES

This is the number of file extents which may exist on the disk. This may be reduced for faster system response in directory operations and mounting disks, or increased to allow more (small) files on a diskette. In particular, the backwards-compatible (with CP/M) default length for single density standard diskettes is not large enough to allow use of the full capacity of the diskette if your files are small. The number of directory entries must be a multiple of 32 in some cases and a multiple of 64 in others; refer to the table below. The maximum must be less than 256 and follow the previous rule.

FORMAT - - -	I	II	III	IV	V
Minimum Number of Directory Entries					
Standard	32	64	64	64	64
35, 40 tk Mini	32	32	32	32	
77 tk Mini	32	32	64	64	
Default Number of Directory Entries					
Standard	64	64	128	128	128
Mini	64	64	64	64	

	Maximum Number of Directory Entries				
Standard	224	192	192	192	192
35, 40 tk Mini	224	224	224	224	
77 tk Mini	224	224	192	192	

E. NUMBER OF 80H BLOCKS FOR SYSTEM AREA

This is the number of 128 byte blocks reserved on the disk for a system image. It may be increased to allow for larger file buffer areas or additions to BIOS. If it is increased from the default value, it must be increased by steps of entire tracks. Refer to the table below for blocks per track.

This may also be decreased from the default value to allow more room on a files-only diskette. It must also be decreased only by entire track steps, except that on diskettes where track 0 is the same format as the remaining tracks it may be decreased to 1 sector. Minimum acceptable sizes are listed in the table below.

FORMAT - - -	I	II	III	IV	V
Minimum System Size					
Standard	1	26	26	26	1
Mini	1	18	18	18	
Blocks per Track (80H)					
Standard	26	30	52	64	58
Mini	18	18	34	40	
Default System Size					
Standard	52	56	78	90	58
Mini	54	54	86	58	
Default plus 1 track					
Standard	78	86	130	154	116
Mini	72	72	120	98	

F. DIRECTORY OFFSET (IN mK BYTE UNITS)

This is the number of allocation units from the beginning of the file area to the beginning of the directory. Making this small provides fastest response to console commands, provided the COM files for transient commands are the first files written on the diskette; placing it near the center of the file area provides faster file access by programs, which is particularly important when random-accessing large files with many extents.

FORMAT - - -	I	II	III	IV	V
Default Directory Offsets					
Standard	0	30	52	64	64
35 tk Mini	18	18	34	40	
40 tk Mini	18	18	34	40	
77 tk Mini	36	36	34	40	

These offsets are chosen to be about 1/4 of the disk (except the 0 which provides CP/M compatibility). They are about 16 tracks for the standard drives and the 77 track Minis; and about 8 tracks for the other Minis.

4.2 Alteration with GENESYS

GENESYS is a new utility with IMDOS Ver 2.05 that replaces RELOC in Vers 2.01 and 2.02. It will create a System Image using the files:

```
BOOT.SYS
CCP.SYS
BDOS.SYS
ENTRY.SYS
SDENS.SYS
DDENS.SYS
BIOS.SYS
```

GENESYS will accept parameters on the command line in a mode which relocates the System Image for different size memories, leaving all other variables constant. This will be a common use for GENESYS and details of the syntax and functions of parameters on the command line are discussed in the "IMDOS System User's Guide".

Many desired changes can be made to the system using this command line mode of GENESYS through the use of the "Cold Boot Command". Any IMDOS command line can be entered to be included in the System Image and automatically be run each time the system is cold-booted. Type in the command when GENESYS prompts for "cold boot command". This can range from a ";comment line", through a "STAT LST:=LPT:" to enable a line printer if available, to "SUBMIT INIT" to"run a complete submit file (INIT.SUB) automatically upon cold boot.

If GENESYS is invoked without any parameters on the command line, then it proceeds to request selection of up to 5 variables in the construction of a System Image:

```
A  ACCESS / NOT ACCESS EXTENDED SECTOR LENGTHS
B  SIZE OF LARGE SECTOR FILE BUFERS
C  MEMORY SIZE
D  SIZE OF DISK VOLUME MOUNT SPACE
E  COLD BOOT COMMAND
```

GENESYS contains a special linking loader that when used in conjunction with M80 (the IMSAI relocating assembler) allows an extremely simplified procedure for modifying BIOS (the system I/O driver module). Any driver changes desired can be made to the BIOS source

(included on the distribution diskette) and assembled. GENESYS will incorporate the modified BIOS and automatically accomodate any change in length.

Following is a description of each of the options available under GENESYS, along with information you will need to choose the value appropriate to your desired system use.

A. How Many Multiple Density System Buffers do you Want?

If you answer 0, then the resulting system will only support sectors of length 128 bytes. The distribution diskette system is GENESYSed for 128 byte sectors only. An answer to this question of 1 or greater allows access to longer sectors (as long as the disk controller is capable of the long sector format).

The size of the final system is strongly affected by your choice of how many buffers. Each system buffer takes up RAM space which is no longer availabe to the TPA, even if the buffer is not used.

IMDOS and user programs work with logical records of 128 bytes. The system will maintain buffers for larger physical sectors, "reading" and "writing" to these buffers only, unless a physical read or write to the disk is required (buffer full, etc.). In order not to force the system to do more physical reads and writes than necessary, sufficient buffers should be defined in the system to avoid contentions:

First: For maximum efficiency you should specify 1 buffer for every extended sector length file expected to be OPEN and accessed simultaneously.

Second: Specify enough additional buffers to contain the entire directory of each disk on which extensive directory use is made (frequent OPENs and CLOSEs, or random access files). A compromise is to specify one buffer additional for each directory involved in an open file of extended sector length.

The disk file access from utilities is weakly affected by the number of buffers. Two buffers

shows about 15% increase in speed (over 1 buffer) in file access for the 1024 byte buffer case for system utilities.

Any one or two digit positive number of buffers may be specified, giving a maximum of 99 if you have enough RAM space. A good starting place for general use is 1 or 2 buffers. You do not need any buffers unless you are using extended length sectors (greater than 128). No additional buffers are required for any open files with 128 byte sectors. See following for amount of space needed.

Note that if you GENESYS a system to have buffers, it will no longer fit in the system area provided by the default format for a standard diskette. Refer to the FORMAT description in this chapter to see how to provide more system space.

B. Buffer Size (256 or 1024)?

Either one of two buffer sizes may be selected, as required by your choice of diskette formats. This is not necessarily related to the system disk format, but rather the format on the diskettes holding the files to be opened, which would normally include the system disk. A 256 byte buffer takes up 265 bytes of RAM (109H), and a 1024 byte buffer uses up 1033 bytes (409H).

Since different drivers need to be linked into the system to handle large sectors, there is a one-time space penalty for using large sectors. For a rule of thumb, 256 byte sectors cost about (N+3)/4 K (N = number of buffers) over 128 byte sectors. 1024 byte sectors cost about (N+1.5) K over 128 byte sectors. For overall space required see table following mount space ("free space") discussion.

C. Top of System (in K)?

Specify here the top memory location you wish to be used by the system. If you are GENESYSing a system on the same machine it will be run on (or a machine with the same memory size), and you wish to use all the RAM, just hit return. GENESYS will then find the top of memory and proceed to create a system that uses all of memory.

A two digit memory size may be entered if you are GENESYSing a system for a machine with different size memory, or if you want the system to leave the top portion of memory alone. Leaving some space at the top of memory may be useful for some safe area for BASIC to PEEK and POKE into, or a user machine language program desired to co-exist with IMDOS. (Don't forget the possibility of including any routines desired to always be present in a modification of BIOS, thus providing automatic loading along with IMDOS.)

D. Size of Disk Volume Mount Space

After specifying the top of memory, GENESYS calculates the size of the System Image with the specified buffers, subtracts this from the given total memory size, and prints the amount of room left (in hex). Then it asks how much of this remainder to allocate to free space for the system to maintain disk mount information in. A minimum is given, and you must make sure you have sufficient mount space for each disk drive to be accessed.

If space is not at a premium, you may avoid some arithmetic by adding up 60H for each drive to be accessed. To obtain the exact size needed for mount space add the exact amounts shown in the table and then add 4. Check that the result is at least as large as the minimum printed by GENESYS. If it is not as large as the printed minimum use the printed minimum; otherwise, use your sum. Remember to work in hex.

DISK MOUNT SPACE MEMORY REQUIREMENTS (hex)

FORMAT - - -	I	II	III	IV	V
Standard	40	40	40	54	50
35 tk Mini	38	38	40	44	
40 tk Mini	38	38	44	44	
77 tk Mini	44	44	44	48	

OVERALL SYSTEM SIZE

Buffer Size dec	System Image Size on Disk dec	System Memory Requirements IMDOS 2.05 hex	Disk Formats Available
(128)	52 blocks	1760 + M	I,V
256	54 blocks	1AD0 + 109N + M	I,II,III,V
1024	54 blocks	1DD0 + 409N + M	I,II,III, IV,V

N = number of buffers
M = mount space

E. Enter cold boot command:

After completing linking the modules into a System Image, GENESYS asks for a command line to be executed on cold-boot. (Cold boot is when the system is booted by ROM firmware - for example power on, RESET button - as opposed to the warm-boot which is a system load initiated by a normal transient program system return, INTERRUPT or ^C.) This may be any valid IMDOS command line including the null command (just hit return). Some useful commands are:

Comment (following a ";")
"STAT LST:=LPT:" to assign the line printer
"<Application Program Name>" for running a .COM file
"CRUN <App Program Name>" for running a .INT file
"SUBMIT <filename.SUB>" in case you want to do more

Whatever command line you enter will be executed, once, each time this System Image is cold booted from the disk.

Then GENESYS will ask for the disk drive on which to write the newly GENESYSed System Image. The diskette, of course, must previously have been appropriately formatted to provide a VCB and sufficient system area using FORMAT 2.05.

4.3 Installing alternate BIOS (Line Printer options)

Three assembled versions of BIOS are provided on the IMDOS distribution diskette. The only difference between them is in the interface to the list device. They all support the IMSAI PTR-300 (Teletype model 40 300 line per minute printer), but on different interfaces. The BIOS source provided is the same for all three versions, using conditional assembly to effect the differences. Your choice of which version to use will depend on which interface your system uses to run the line printer.

BIO4.SYS

This interfaces the line printer through RS-232 on ports 4 and 5 (as on the MPU-B).

BIOS22.SYS

This interfaces the line printer through RS-232 on ports 22 and 23 (as on an SIO board).

BIOS.SYS

Installed in the distribution diskette System Image. This interfaces the line printer using the SSI interface with the IMSAI LIF interface (IFM and LIB cards).

In order to GENESYS a system with other than BIOS in it, you simply have to REName the appropriate version of BIOS as BIOS.SYS. Since you already have one file with that name, you will first have to rename it to something else (e.g. BIOS0.SYS).

```
REN BIOS0.SYS = BIOS.SYS
REN BIOS.SYS = BIOS4.SYS
```

After this is done, simply run GENESYS to create a System Image on your diskette. It will automatically use the new module you have just named BIOS.SYS.

4.4 Modifications to BIOS

Changing device drivers is easy to accomplish. The problems of incorporating changed code into a System Image are all solved for you, and the only significant task left is the creation of the actual device driver code. To add or change an I/O driver in IMDOS takes four steps:

1. Use the editor to insert your code into BIOS.MAC (Listing provided in Appendix A).
2. Use M80 to assemble BIOS.MAC into a system module
3. Make sure you have a diskette formatted with adequate system space by using FORMAT.
4. Run GENESYS to create new System Image.

Your next step should be to study the BIOS source listing in Appendix A. The listing is heavily commented, and any programmer up to writing a driver in assembly code should be able to have it interact with the system properly by observing the many comments in BIOS. A brief overview follows to assist in mastering BIOS. Note that the page references below refer to the page number in Appendix A.

INITIOBYTE Equate -- page 1

This may be changed if you wish your system to come up with different logical device assignments at cold start, before any STAT device assignment commands are given. Read the comments above it. The code that uses it is on page 19. Logical device selection based on it takes place in the code on pages 6 and 7.

Other Equates -- pages 2-3

Symbols are defined for numerous other address, and all I/O ports accessed in BIOS. The I/O ports for non-disk devices may be changed by the user.

BIOS Entry Points -- page 4

This is a series of JMPs at fixed locations relative to the beginning of BIOS to routines for various functions. Several of the JMPs actually go to external routines located in BDOS.

The entry at BIOS jumps to a routine in BDOS to do a "warm restart". A warm restart, used at termination of many programs, consists of reloading the CCP module from the same drive it was cold-started from, repeating some initialization, and transferring control to the CCP. The I/O assignments (IOBYTE), the CCP's current disk (LOGDISK), and BDOS' "preserve table" (which contains, among other things, the logical-to-physical disk assignments), and the contents of the TPA are preserved through a warm restart.

The entry at BIOS is normally accessed via the JMP to it at 0; the address of the JMP at 0 is also used by programs to locate other BIOS entries regardless of the system location. Since it is possible these calls will change location is future systems, the programmer should normally use the BDOS call procedure described in the "IMDOS Programming Guide". IMDOS 2.05 has physical (track and sector) disk I/O calls added so it is no longer necessary to depend on BIOS call addresses.

The programmer using any of the BIOS entries should assume that they use considerable stack -- maybe forty bytes -- and destroy the contents of all registers.

Logical Device Routines -- pages 6 and 7

Each of these routines accesses a specific logical device by testing IOBYTE and branching to the appropriate physical device routine. The function of each routine, and the physical devices it goes to as a function of what IOBYTE bits, is given in comments. IOBYTE, the I/O Assignments Byte, is documented in the "IMDOS Programming Guide".

The testing of IOBYTE and branching to one of four addresses is done by the subroutine DISPATCH at the bottom of page 7. Each call to dispatch is followed by a byte specifying the bits to test and four two-byte addresses to be branched to if the IOBYTE field contains 0, 1, 2, and 3 respectively.

Changes in the physical device accessed by a given logical device for a given IOBYTE value can be accomplished by substituting the address of the desired driver in the appropriate DW statement

after the CALL DISPATCH in the appropriate logical device routine. This procedure is suggested for allowing access to custom drivers in place of standard drivers or unimplemented drivers.

Rearranging the correspondence between IOBYTE values and standard physical devices is discouraged, as the device names in PIP and STAT would then no longer correspond correctly to the devices.

If you make any changes in the possible physical devices for CON:, review the "console finder" on later pages, to see if it needs corresponding changes to insure a valid console assignment at cold start.

Physical Device Routines -- pages 9-12

These are the actual hardware drivers for the physical devices supported by IMDOS. They are accessed via the dispatch tables in the preceding logical device routines.

Any added drivers, etc., should be inserted in the source file before the comment "end of code" on page 13, as the part of BIOS following this comment is overwritten by BDOS storage once the system has been initialized.

Initialization

Any device initialization should be inserted following the ENSYS label on page 15.

Backup

Remember to keep several copies of your original system as backup, and try out your new system on diskettes without any valuable files.

4.5 Some Common Alterations

This section reviews a number of the more common alterations, and makes reference to the other chapter sections that relate to making them. Some can be done with console commands only; others require program changes in the BIOS module of IMDOS.

Relocating the system for a different amount of RAM

This is done with the GENESYS console command, as introduced in the "IMDOS System User's Guide" and further developed in Section 4.2 of this alteration guide.

The initial reason for relocating the system is usually that you have more memory than the distribution diskette system size uses, and wish to relocate the system to make use of it; i.e., to move the system to higher numbered addresses so additional memory is in the TPA.

Another reason to relocate the system might be to move it down from the top of memory, to leave space not in the TPA for special routines (possibly routines to be CALLed from one or another dialect of BASIC). In this case the system can be moved "down" (to lower-numbered addresses) in 1024-byte increments by giving a memory size smaller than what is actually present (as in "GENSYS 31 *" in a 32K system).

Increasing "Mount Space"

If you use a number of drives simultaneously you may encounter the error message "MOUNT SPACE FULL" as described in the "IMDOS Programming Guide". The cure is to run GENESYS and specify a sufficiently large mount space to the "how much free space" prompt. Refer to Section 4.2 of this chapter.

Changing to Double Density Diskettes

1. Use STAT to switch a NON-cold-boot drive to double density (so that you can continue reading programs off the system diskette for now.)
2. Use FORMAT to format a diskette in double density, answering the FORMAT prompts according to your desired disk configuration.
3. Run GENESYS to create a system image that will run double-density, with sufficient buffers and mount space to meet your requirements. Have your double-density formatted diskette in a drive for GENESYS to write the system on.
4. While you still have one double-density drive and one single density (system diskette), pip

any desired files onto your double-density diskette.

5. If you now wish to boot it, use stat to set density of boot drive or change the option switch on the DIO for cold boot.

Changing Sector Sizes

Changing sector sizes is exactly analogous to changing density except that you don't need to run STAT if the drive is set to the correct density. Follow the steps in the previous section.

Changing Initial I/O byte assignment

There are two methods:

First - Run GENESYS n * to create a System Image and put it on another diskette. When GENESYS prompts for a cold boot command, enter the STAT command to effect your desired change. Also, you can add this STAT command to a submit file if your cold boot command is to run a submit file.

Second - Alter the EQU statement for INITIOBYTE in BIOS. Re-assemble BIOS. Then run GENESYS to create a new System Image incorporating BIOS with a modified INITIOBYTE. Check the Modifications to BIOS section of this chapter, and look at the BIOS source listing.

Changing Printer Interface driver

To change between different IMSAI printer drivers, rename BIOS files appropriately and run GENESYS. See this chapter Section 4.3.

Changing Non-disk Drivers

All non-disk drivers are in BIOS, and may be changed as desired by the user. To make these changes, you must be able to get the system up on some machine, but it need not be the machine for which the changes are being made.

Once the desired changes have been made to the BIOS source, reassemble the source and run GENESYS to create a System Image incorporating the changes.

Adding a Different Driver

Put the desired routine into BIOS. We suggest putting it after the existing group of "physical device routines". Decide what logical device name and which IOBYTE value for that logical device will access it, and change the appropriate dispatch address for that logical device, as described in source listing comments. If the device to be driven requires initialization when the system boots, add the necessary code to the "system initialization" section of BIOS.

After the added code is edited into BIOS.MAC, reassemble BIOS with M80, and run GENESYS to create a System Image incorporating the new code.

Diskette Layout Changes

The FORMAT program has prompts, the answers to which direct it to format diskettes with different areas reserved for the system, different size directories, and different locations for the directory. It also allows for several different standard sector/density formats. See this chapter Section 4.1.

5. "SYSMOV" TECHNICAL DESCRIPTION

The SYSMOV command was introduced in the "IMDOS System User's Guide" and has already been mentioned several places in this chapter. We will now review SYSMOV for reference purposes and add a few technical details. The reader may wish to skip reading this section initially, then refer back to it when a specific question or problem about SYSMOV occurs.

SYSMOV runs in memory locations 100-5FF hex; it thus fits in the TPA below the system image (which starts at 600 hex).

When SYSMOV is first started, it verifies that it is running under IMDOS Version 2.05 or newer as older versions do not support certain system calls used in SYSMOV.

Then, SYSMOV asks

GET SYSTEM? (Y/N/DISK:)

The user response should be terminated with a carriage return. An "N" causes SYSMOV to skip to the PUT SYSTEM question below without getting the system. A drive name (as documented in the "IMDOS Programming Guide") may be entered to cause the system to be read from that disk. The colon must be included in the type-in and either a logical or physical name is allowed. A "Y" may be typed to cause the system to be gotten from logical drive A:.

Unless the response to the above prompt was "N", SYSMOV then types

SOURCE ON d:, TYPE RETURN

This gives you a chance to insert a different diskette and to verify that the drive name was entered correctly. A control-C input at this point will abort SYSMOV.

When a carriage return is input, SYSMOV will then "get" the system by reading the entire system image area (length as specified in the VCB) from the indicated disk into the TPA, at 600 hex and sequentially higher addresses. The I/O for this operation uses the new BDOS DOIO call, as described in the programming guide.

If the source disk is not correctly formatted, or does not have an IMDOS system image on it, an error message such as

BAD VOLUME CONTROL BLOCK

will be typed and the system will not be gotten. SYSMOV version 2.05 cannot "get" the IMSAI CP/M Version 1.33 and older system images. Upon successful completion of the "get" SYSMOV types

FUNCTION COMPLETE

SYSMOV then types

PUT SYSTEM? (Y/N/DISK:)

An "N" response terminates SYSMOV; a "Y" causes disk B: to be assumed; any desired disk name may also be typed.

SYSMOV then types

DESTINATION ON d:, TYPE RETURN.

When the return is entered SYSMOV makes several checks before putting the system. First, the diskette is checked to make sure it has been formatted with FORMAT Version 2.05 and has a valid volume control block. An error message is typed if the check fails. Then, the TPA is checked; if it is found to contain a CP/M version 1.33 or older system (as indicated by 16 hex in the first byte), an error message is typed. Next, the system image length (in location 6FF hex, the last byte of BOOT) is checked to make sure it is not greater than the size of the system image area on the diskette (as indicated in the diskette's VCB). If the system image in the TPA is too large, the message

WON'T FIT

is typed. If this occurs you must use FORMAT to set up a diskette with a larger system image area before you can "put" your new system.

After all of the above checks have succeeded, SYSMOV then inserts the diskette's VCB into the correct place in the system image in the TPA, so that the system image on the disk contains the VCB for that disk rather than the disk the system came from. Finally, SYSMOV "puts" the system by writing the contents of the TPA, from 600 hex up, into the diskette's system image area. The entire length of the diskette's system image area is written, even if greater than

the length of the system image as specified by byte FF hex of BOOT.

Lastly, "FUNCTION COMPLETE" is typed.

If you have invested a lot of work altering a system image in DDT then encounter s SYSMOV error, you can use the SAVE command to preserve it on a COM file.

```
                ;BIOS.MAC  VERSION 2.05  REV 0    8/1/78
                ;
                ;       IMDOS BASIC INPUT-OUTPUT SYSTEM
                ;
                ; THIS MODULE CONTAINS DRIVERS FOR NON-DISK DEVICES,
                ; LOGICAL DEVICE ROUTINES, AND SOME SYSTEM INITIALIZATION
                ; CODE.
                ;
                ;CHANGES IN LATEST VERSION:
                ;       MODIFY TO USE RELOCATIBLE ASSEMBLER
                ;
                ;
                ; COPYRIGHT (C) 1978
                ;       IMSAI MANUFACTURING CORP
                ;          14860 WICKS BLVD, SAN LEANDO, CA 94577 USA
                ;
                ;
0000'           ENTRY   ENSYS,BIOS
0000'           ENTRY   CONIF,CONRDY,CONOUT,PUNOUT,LSTOUT,READIN
0000'           EXT     ENCCP,BDOSRE,WBOOT,VIOF,IOBYTE,OPTIONS,COPYHL
00              EXT     HOME,SELDISK,SELTRK,SELSEC,BCMD,DEX
                ;
                ;INITIAL I/O ASSIGNMENTS EQUATE:
                ;
                        ; MAY BE CHANGED BY USER TO PUT VALUE OTHER THAN
                        ; 0 INTO LST:, RDR:, AND PUN: IOBYTE FIELDS AT
                        ; COLD START.  CON: BITS (0 AND 1) SHOULD BE LEFT
                        ; 0 FOR CORRECT OPERATION OF "CONSOLE FINDER" IN
                        ; INITIALIZATION CODE NEAR END OF BIOS.
                        ; CHANGING INITIOBYTE IS NEVER ESSENTIAL, AS
                        ; DEVICES CAN BE ASSIGNED WITH THE "STAT"
                        ; COMMAND; THIS PROVIOSION FOR CHANGE IS
                        ; INTENDED AS A WAY TO MAKE A MORE CONVENIENT
                        ; SYSTEM ONLY.
                        ;EXAMPLE: TO MAKE SYSTEM COME UP WITH LST:=LPT:,
                        ; CHANGE INITIOBYTE TO 1000S0000B.
                ;
0000            INITIOBYTE  EQU  0      ;INITIAL IOBYTE (LEAVE 0 FOR CON:)
```

```
0000            FALSE   EQU     0
FFFF            TRUE    EQU     NOT FALSE
                ;
                ;
                ;
                ; I/O PORT EQUATES
                ;
0000            SER40A  EQU     FALSE   ;TRUE FOR PRINTER ON SIO PORTS 4 & 5
0000            SER40B  EQU     FALSE   ;TRUE FOR PRINTER ON SIO PORTS 22 & 23
0000            SER40   EQU     SER40A OR SER40B
                ;
0002            TTY     EQU  02H
0003            TTYS    EQU  03H
0004            CRT     EQU  04H
0005            CRTS    EQU  05H
0008            SIOC    EQU  08H
                ;
                ;
                ; IMSAI PTR-300 LINE PRINTER
                ;
0000'                   IF NOT SER40    ;IF PRINTER USES LIF
00F6            PRINTER EQU  0F6H       ;OUTPUT PORT, AND STATUS IN
0080            PINIT   EQU  80H        ;INITIALIZE COMMAND
                ;ANY DATA WITH B7=0 IS TAKEN AS ASCII CHAR TO PRINT.
0000'                   ENDIF
                ;
                ;
0000'                   IF SER40A
                PTR     EQU     4
                PTRS    EQU     5
0000'                   ENDIF
                ;
                ;
0000'                   IF SER40B
                PTR     EQU     22H
                PTRS    EQU     23H
0000'                   ENDIF
                ;
                ;
                ;
                ; IMSAI VIO VIDEO DISPLAY BOARD
F800            VIOINIT EQU     0F800H  ;CALL THIS ADDR TO INIT
F803            VIOADR  EQU     0F803H  ;CALL TO OUT CHAR (A)
```

```
                        ;
                        ; EQUATES FOR MPU-B TIMER INITAILIZATION
                        ;
0000'                           IF SER40A       ;ONLY SET TIMER ON MPU-B
                                                ;FOR PTR ON SIO PORT 4 & 5
                        MPUIN   EQU     40H     ;INSTALL MPU-B
                        MPUOUT  EQU     0C0H    ;REMOVE FROM MEMORY SPACE
                        MEMCTL  EQU     0F3H    ;PORT TO INSTALL/REMOVE MPU-B
                        MPUAT   EQU     0D000H  ;ADDRESS OF MPU-B
                        MPUTSA  EQU MPUAT+800H  ;ADDRESS OF SIGNATURE
                        MPUTST  EQU     3EH     ;VALUE OF SIGNATURE
                        MPUTMC  EQU MPUAT+103H  ;ADDRESS OF TIMER CONTROL PORT
                        MPUTV1  EQU     36H     ;WHAT TO PUT THERE
                        MPUTIM  EQU MPUAT+100H  ;ADDRESS OF TIMER PORT
                        MPUTV2  EQU     13      ;WHAT TO PUT THERE
0000'                           ENDIF
                        ;
                        ;
                        ; EQUATES FOR ASCII CHARACTERS
                        ;
00                      CTRLC   EQU  3
0014                    CTRLT   EQU     14H
0009                    TAB     EQU  9
000A                    LF      EQU  0AH
000C                    FF      EQU  0CH
000D                    CR      EQU  0DH
001A                    CTRLZ   EQU  1AH
005F                    UNDERLINE EQU 5FH
007F                    RUBOUT  EQU  7FH
                        ;
```

```
                        ;
                        ;CODE BEGINS HERE
                        ;
                        ;
                        ; ENTRY POINT TABLE
                        ;
                        ;DO NOT REARRANGE THE FOLLOWING, AS PROGRAMS HAVE
                        ;THESE ADDRESSES BUILT IN TO THEM
                        ;
                                                ;BOOT RETURNS HERE
0000' C3 0000 * BIOS:   JMP  WBOOT              ;COME HERE FOR REBOOT (VIA 0)
                    ;LOGICAL DEVICE ENTRIES: THESE ROUTINES GO TO THE
                    ;CURRENTLY ASSIGNED PHYSICAL DEVICE PER IOBYTE.
0003' C3 0039 '         JMP  CONRDY             ;CON: STATUS. NON-0 FOR CHR RDY
0006' C3 004D '         JMP  CONIF              ;CONSOLE IN TO A
0009' C3 0059 '         JMP  CONOUT             ;CONSOLE OUT FROM C
000C' C3 0065 '         JMP  LSTOUT             ;LIST OUT FROM C
000F' C3 0071 '         JMP  PUNOUT             ;PUNCH OUT FROM C
0012' C3 007D '         JMP  READIN             ;READER IN TO A
                    ;DISK FUNCTION ENTRIES: THESE NOW ARE PERFORMED IN
                    ;BDOS; ENTRIES ARE MAINTAINED HERE FOR PROGRAMS THAT
                    ;NEED TO ACCESS DISK VIA TRACK AND SECTOR ADDRESSES
                    ;RATHER THAN VIA FILE STRUCTURE (E.G. SYSMOV, FORMAT)
0015' C3 0000 *         JMP  HOME               ;RESTORE DRIVE
0018' C3 0000 *         JMP  SELDISK            ;SELECT DRIVE (C)+1
001B' C3 0000 *         JMP  SELTRK             ;SELECT TRACK (C)
001E' C3 0000 *         JMP  SELSEC             ;SELECT SECTOR (C)
0021' C3 0032 '         JMP  SELDMA             ;SET BUFFER ADDRESS (BC)
                    ;READ DISK SECTOR
0024' 3E 20             MVI A,20H               ;ENTRY FOR DISK READ
0026' 01                DB 01                   ;CODE FOR LXI B,
                    ;WRITE DISK SECTOR
0027' 3E 10             MVI A,10H               ;ENTRY FOR DISK WRITE
0029' 32 0000 *         STA OPTIONS             ;CLEAR ERROR CONTROL OPTIONS
002C' C3 0000 *         JMP DEX                 ;USE BDOS DISK I/O
                    ;SIGNATURE
002F' 00 00 FF          DB 0,0,0FFH             ;MAKE 2.02 UTILITIES INCOMPATIBLE
                        ;
                        ; SET DISK BUFFER ADDRESS IN BDOS
                        ;
0032' C5            SELDMA: PUSH B
0033' E3                    XTHL
0034' 22 0000 *             SHLD BCMD
0037' E1                    POP H
0038' C9                    RET
```

```
                ;***********************************************
                ;
                ;       LOGICAL DEVICE ROUTINES
                ;
                ;   THESE ROUTINES USE VARIOUS PHYSICAL DEVICES
                ;   DEPENDING ON CONTENTS OF IOBYTE.
                ;
                ;   AN APPLICATION PROGRAM USUALLY CALLS THESE
                ;   ROUTINES VIA BDOS SYSTEM CALLS, BUT MAY CALL THE
                ;   BIOS ENTRY POINTS DIRECTLY.
                ;
                ;   ALL CLOBBER DE, SOME CLOBBER OTHER REGS.
                ;
                ;
                ; CONSOLE STATUS
                ;
0039' CD 0041 ' CONRDY: CALL CONS       ;GETS STAT OF SPECIFIC DEVICE
003C' B7                ORA  A
003D' C8                RZ              ;IF NOT READY RETURN 0 IN A
003E' 3E FF             MVI  A,0FFH     ;ELSE RETURN FF
00    C9                RET             ;Z FLAG SET CORRESPONDINGLY
                ;
0041' CD 0089 ' CONS:   CALL DISPATCH   ;GO TO ONE OF 4 ADDRESSES PER IOBYTE
0044' 01                 DB     1       ;..BITS 1-0
0045' 00A9 '             DW  TTYSTAT     ;00 - TTY:
0047' 00CD '             DW  CRTSTAT     ;01 - CRT:
0049' 00CD '             DW  CRTSTAT     ;10 - VK2:
004B' 00A9 '             DW  TTYSTAT     ;11 - VK1:
                ;
                ; CONSOLE IN
                ;
004D' CD 0089 ' CONIF:  CALL DISPATCH
0050' 01                 DB     1       ;USE IOBYTE BITS 1-0
0051' 009E '             DW  TTYIN       ;00 - TTY:
0053' 00C2 '             DW  CRTIN       ;01 - CRT:
0055' 00C2 '             DW  CRTIN       ;10 - VK2:
0057' 009E '             DW  TTYIN       ;11 - VK1:
                ;
                ; CONSOLE OUT
                ;
0059' CD 0089 ' CONOUT: CALL DISPATCH  ;GO TO ONE OF FOLLOWING ADDRS
005C' 01                 DB     1       ;USE BITS 1-0 OF IOBYTE
005D' 00AE '             DW  TTYOUT      ;BITS=00 - USE TTY AS CONSOLE
005F' 00D2 '             DW  CRTOUT      ;01 - CRT:
0061' 00E2 '             DW  VIO$OUTPUT  ;10 - VK2:
00[illegible]' 00E2 '    DW  VIO$OUTPUT  ;11 - VK1:
                ;
                ; CAUTION:  LOGICAL DEVICE ROUTINES FOR CONSOLE (ABOVE)
                ;           AND CONSOLE FINDER (SEE LABEL 'FINCON') MUST
                ;           CORRESPOND.
                ;
```

```
                        ;
                        ; LIST OUT
                        ;
0065' CD 0089 ' LSTOUT: CALL DISPATCH   ;GO TO 1 OF 4 ADDRESSES PER
0068' 03                DB     3        ;IOBYTE BITS 7-6
0069' 00AE '            DW   TTYOUT     ;00 - TTY:
006B' 00D2 '            DW   CRTOUT     ;01 - CRT:
006D' 0102 '            DW   LPTOUT     ;10 - LINE PRINTER
006F' 00E2 '            DW   VIO$OUTPUT ;11 - VK2: (VIDEO OUTPUT)
                        ;
                        ; PUNCH OUT
                        ;
0071' CD 0089 ' PUNOUT: CALL DISPATCH
0074' 05                DB     5        ;USE IOBYTE BITS 5-4
0075' 00AE '            DW   TTYOUT     ;00 - TTY:
0077' 00D2 '            DW   PUNO       ;01 - HIGH SPEED PUNCH
0079' 00D2 '            DW   CRTOUT     ;10 - UP1: USES CRT:
007B' 00E2 '            DW   VIO$OUTPUT ;11 - VK1: (VIDEO OUTPUT)
                        ;
                        ; READER IN
                        ;
007D' CD 0089 ' READIN: CALL DISPATCH
0080' 07                DB     7        ;USE IOBYTE BITS 3-2
0081' 009E '            DW   TTYIN      ;00 - TTY:
0083' 0127 '            DW   RDRIN      ;01 - HIGH SPEED READER
0085' 00C2 '            DW   CRTIN      ;10 - UR1:  USES CRT: IN
0087' 009E '            DW   TTYIN      ;11 - VK1:
                        ;
                        ;SUBROUTINE TO DISPATCH TO ONE OF 4 FOLLOWING ADDRESSES
                        ;DEPENDING ON CONTENTS OF TWO BITS OF IOBYTE.  SPECIFIC
                        ;BITS OF IOBYTE ARE SPECIFIED BY SHIFT COUNT FOLLOWING CALL.
                        ;RETURNS TO SUBROUTINE CALL PRIOR TO CALL TO DISPATCH.
                        ;
0089'                   DISPATCH:
0089' E3                        XTHL            ;SAVE CALLER'S H, GET TABLE ADDR
008A' 56                        MOV D,M         ;SHIFT COUNT
008B' 23                        INX H           ;POINT TABLE
008C' 3A 0000 *                 LDA IOBYTE      ;GET IO ASSIGNMENTS BYTE
008F' 07                DSHFT:  RLC
0090' 15                        DCR D
0091' C2 008F '                 JNZ DSHFT       ;SHIFT TO POSITION BITS
0094' E6 06                     ANI 06H         ;MASK BITS
0096' 5F                        MOV E,A         ;D ALREADY CLEAR
0097' 19                        DAD  D          ;INDEX INTO TABLE
0098' 7E                        MOV  A,M
0099' 23                        INX H           ;TABLE WORD TO HL
009A' 66                        MOV  H,M
009B' 6F                        MOV L,A         ;..
009C' E3                        XTHL            ;PUT ADDR OF ROUTINE, GET CALLER'S H
009D' C9                        RET             ;GO TO ROUTINE
```

```
                        ;
                        ;*****************************************************
                        ;
                        ;       PHYSICAL DEVICE ROUTINES
                        ;
                        ; ACCESSED VIA LOGICAL DEVICE ROUTINES ABOVE
                        ;
                        ;
                        ; TELETYPE INPUT
                        ;
                        ; USED FOR TTY: AND VK1: (KEYBOARD ON PORT 2)
                        ; MUST PRESERVE B, D, E, H, L FOR CONSOLE FINDER
009E' CD 00A9 '         TTYIN:  CALL TTYSTAT
00A1' CA 009E '                 JZ   TTYIN      ;WAIT FOR A CHAR TO BE AVAILABLE
00A4' DB 02                     IN   TTY        ;INPUT IT
00A6' E6 7F                     ANI  7FH        ;REMOVE PARITY
00A8' C9                        RET
                        ;
00A9'                   TTYSTAT:        ;USED HERE AND IN CONSTAT ABOVE
00A9' DB 03                     IN   TTYS       ;GET STATUS
00AB' E6 02                     ANI  02H        ;MASK BIT
00.   C9                        RET             ;A IS NON-0 IF CHAR AVAILABLE
                        ;
                        ; TELETYPE OUTPUT
                        ;
                        ;CLOBBERS DE. BOOT DEPENDS ON PRESERVING HL
                        ;MUST PRESERVE HL FOR NXM, BOOTSTRAP
00AE' DB 03             TTYOUT: IN   TTYS       ;STATUS
00B0' 0F                        RRC             ;TEST BIT 0
00B1' D2 00AE '                 JNC  TTYOUT     ;WAIT TILL READY TO ACCEPT CHAR
00B4' 79                        MOV  A,C
00B5' D3 02                     OUT TTY         ;OUTPUT THE CHARACTER
                        ;
                        ; IF YOUR TERMINAL NEEDS A DELAY FOR CARRIAGE TO RETURN,
                        ; ALTER THE FOLLOWING INSTRUCTION TO MVI D,28H FOR A
                        ; DELAY OF ABOUT 100 MSEC.
00B7' 16 00                     MVI D,0
00B9' FE 0D                     CPI CR
00BB' C0                TTYWT0: RNZ
00BC' 1B                TTYWT1: DCX  D
00BD' B2                        ORA  D          ;DEPENDS ON A7=0 AT ENTRY
00BE' F2 00BC '                 JP   TTYWT1     ;LOOP TAKES 9.5 USEC PER COUNT
00C1' C9                        RET
```

```
                        ;
                        ;  CRT INPUT
                        ;
                        ; USED FOR CRT: AND VK2: (KKEYBOARD ON PORT 4).
                        ; MUST PRESERVE B, D, E, H, L FOR CONSOLE FINDER
00C2' CD 00CD ' CRTIN:  CALL CRTSTAT
00C5' CA 00C2 '         JZ   CRTIN
00C8' DB 04             IN   CRT
00CA' E6 7F             ANI  7FH
00CC' C9                RET
                        ;
00CD' DB 05     CRTSTAT: IN  CRTS
00CF' E6 02             ANI  02H
00D1' C9                RET
                        ;
                        ; CRT OUTPUT
                        ;
                        ;MUST PRESERVE HL FOR BOOT, NXM
                        ;CLOBBERS DE
00D2' DB 05     CRTOUT: IN   CRTS
00D4' 0F                RRC
00D5' D2 00D2 '         JNC  CRTOUT
00D8' 79                MOV  A,C
00D9' D3 04             OUT  CRT
                        ;DELAY AFTER LINE FEED. THIS IS SO
                        ;(1) IF THERE ARE SEVERAL BLANK LINES, PRECEDING
                        ;OUTPUT DOESN'T DISSAPPEAR OFF SCREEN BEFORE YOU
                        ;HAVE TIME TO HIT CTL-S, AND (2) TO GIVE CERTAIN
                        ;"INTELLIGENT" TERMINALS TIME TO "THINK"
00DB' 16 14             MVI D,14H       ;DELAY: APPROX 50 MSEC
00DD' FE 0A             CPI LF          ;IF WAIT AFTER CR IS REQUIRED
                                        ;CHANGE INSTRUCTION TO "CPI CR"
00DF' C3 00BB '         JMP TTYWT0      ;USE TTY: WAIT CODE
                        ;
                        ;
                        ;
                        ; VIDEO OUTPUT (IMSAI VIO BOARD)
                        ;
00E2' 3A 0000 * VIO$OUTPUT: LDA VIOF    ;TEST FLAG THAT SAYS
00E5' B7                ORA A           ;VIO IS PRESENT
00E6' C2 00AE '         JNZ  TTYOUT     ;IF NOT THERE, USE TTY INSTEAD.
00E9' 79                MOV A,C         ;CHARACTER
00EA' C3 F803           JMP VIOADR      ;DO IT
```

```
                        ; LINE PRINTER OUT
                        ;
00ED' C5                LPTOU2: PUSH B          ; ERROR RETRY FOR LPTOUT
00EE' 11 0118 '                 LXI D,PRINTMSG  ;"LPT NOT READY"
00F1' 01 0009                   LXI B,9         ;"PRINT" FUNCTION NUMBER
00F4' CD 0000 *                 CALL BDOSRE     ;BDOS PRINT
00F7' 0E 01                     MVI C,1         ;FCN NUMBER TO INPUT CHAR & ECHO
00F9' CD 00F5 *                 CALL BDOSRE     ;CALL BDOS
00FC' FE 03                     CPI  CTRLC
00FE' CA 0001 *                 JZ   WBOOT      ;IF ^C TYPED, REBOOT
0101' C1                        POP  B          ;GET CHAR TO PRINT BACK
                        ;
0102'                   LPTOUT:         ;ENTRY POINT FOR PRINTER OUTPUT
                        ;
0102'                           IF NOT SER40
0102' 06 00                     MVI  B,0        ;BUSY TIME-OUT COUNTER
0104' DB F6             LPTOU0: IN   PRINTER    ;THIS GETS: FF, NO LPT INTERFACE,
                                 ;F4: READY, F0: BUSY, NO PAPER, COVER OPEN, &C.
0106' FE FF                     CPI  0FFH
0108' CA 00ED '                 JZ   LPTOU2     ;NO PRINTER, USE "HUNG" MSG
010B' 05                        DCR  B          ;ON TIME-OUT,
010C' CA 00ED '                 JZ   LPTOU2     ;PRINT MESSAGE AND AWAIT INPUT
010F' E6 04                     ANI  04H        ;CHECK "READY" BIT
0111' CA 0104 '                 JZ   LPTOU0     ;IF NOT READY, TRY TILL TIME-OUT
0114' 79                        MOV  A,C        ;THE CHARACTER
0115' D3 F6                     OUT  PRINTER    ;OUTPUT IT
0117' C9                        RET
0118'                           ENDIF
                        ;
                        ;
                        ;
0118'                           IF SER40
                                LXI D,0
                        LPTOU0: DCX D           ;CHECK TIMEOUT
                                MOV A,D
                                ORA E
                                JZ LPTOU2       ;JIF TIMEOUT
                                IN PTRS         ;CHECK STATUS PORT
                                CPI 0FFH
                                JZ LPTOU2       ;JIF NOTHING THERE
                                ANI 85H         ;CHECK BOTH READY BITS
                                CPI 85H
                                JNZ LPTOU0      ;LOOP IF NOT READY
                                MOV A,C
                                OUT PTR         ;OUTPUT THE BYTE
                                RET
0   '                           ENDIF
                        ;
                        ;
0118' 0D 0A 4C          PRINTMSG: DB CR,LF,'LPT NOT RDY ',0
011D' 20 4E 4F
0122' 52 44 59
```

```
                        ;
                        ;
                        ; NULL DEVICE, FOR UNDEFINED DEVICES.
                        ;
                        ;FOR UNASSIGNED AND AND UNIMPLEMENTED INPUT DEVICES,
                        ;HERE IS AN INFINITE SOURCE OF EOF'S:
0127' 3E 1A             NULLI:  MVI  A,CTRLZ
0129' B7                        ORA A
012A' C9                        RET
                        ;
                        ;DON'T USE CRT FOR UNASS INPUT DEVICES CAUSE IF THERE
                        ; IS NO CRT ON SYSTEM BUT INTERFACE BOARD IS PRESENT,
                        ; SYSTEM WILL HANG.
                        ;
                        ;FOR UNUASS AND UNIMP OUTPUT DEVICES, USE CRT.
                        ;  IF NO CRT IS PRESENT, THIS IS AN INFINITE DATA SINK.
00D2                    NULLO   EQU  CRTOUT
                        ;
                        ; HERE IS A PLACE WHERE USER MAY ADD HIGH SPEED
                        ; READER DRIVER
                        ;
0127                    RDRIN   EQU  NULLI      ;MEANWHILE, USE NULL DEVICE
                        ;
                        ; HERE IS A PLACE WHERE USER MAY ADD HIGH SPEED
                        ; PUNCH DRIVER
                        ;
00D2                    PUNO    EQU  NULLO      ;MEANWHILE, USE NULL DEVICE
                        ;
                        ;
                        ;
                        ;
                        ; ***************************************************
                        ;
                        ; END OF CODE USED WHILE SYSTEM IS RUNNING
                        ;
                        ;       ALL CODE USED DURING SYSTEM EXECUTION
                        ;       MUST PRECEDE THIS POINT -- FOLLOWING
                        ;       CODE IS USED DURING SYSTEM INITIALIZATION
                        ;       ONLY AND IS OVERLAID WITH STORAGE THEREAFTER.
                        ;       MORE COMMENTS ON WHAT OVERLAYS INITIALIZATION
                        ;       CODE ARE ON THE LAST TWO PAGES OF THIS LISTING.
                        ;
```

```
                        ;
                        ;*************************************************
                        ;
                        ;       SYSTEM INITIALIZATION, BIOS PORTION
                        ;
                        ; INITIALIZATION STARTS IN BOOT, CONTINUES HERE.
                        ; CODE HERE SETS SYSTEM VIO AND DIO FLAGS,
                        ; INTITIALIZES VIO VIDEO INTERFACE IF PRESENT,
                        ; INITIALIZES LINE PRINTER IF PRESENT,
                        ; AND, IF COLD START, FINDS CONSOLE AND INTIALIZES
                        ;   IOBYTE.
                        ; CONTROL IS THEN TRANSFERRED TO THE CCP, WHICH
                        ; DOES ADDITIONAL INITIALIZATION INCLUDING TYPING
                        ;  THE SYSTEM SIGN-ON.
                        ;
                        ;
012B'                   ENSYS:
                                ;ENTRY FROM BOOT
                                ; ON ARRIVAL HERE
                                ;   IOBYTE = 0FFH IF IT NEEDS SETTING
                        ;
01    31 0068                   LXI SP,68H      ;SET STACK
                        ;
                        ;
                        ; INIT LINE PRINTER
                        ;
                        ;
                        ;
012E'                           IF NOT SER40
012E' DB F6                     IN   PRINTER    ;GET STATUS: FF NOT THERE,
                                                ;F4 OK, F0 NOT READY (MIGHT
                                                ;HANG US IF WE OUTPUT TO IT
0130' E6 04                     ANI  04H        ;MASK THE TELLING BIT
0132' CA 0139                   JZ   NOPINIT    ;BIT OFF, SKIP INIT
0135' 3E 80                     MVI  A,PINIT
0137' D3 F6                     OUT  PRINTER
0139'                   NOPINIT:
0139'                           ENDIF
```

```
                        ;
                        ; IF SIO ON PORT 4 & 5, INIT MPU-B TIMER
                        ;
0139'                           IF SER40A
                                LXI H,TEST      ;FIRST MOVE CODE DOWN TO 80H
                                LXI D,80H       ; SINCE BIOS MIGHT BE IN MPU LAND
                                MOV B,E         ;NUMBER OF BYTES TO MOVE (80H IS ENOUGH)
                                CALL COPYHL     ;USE BDOS COPY ROUTINE
                                CALL 80H        ;INIT TIMER IF THERE
0139'                           ENDIF
                        ;
                        ; INIT BOTH CHANNELS OF SIO
                        ;   AND PRINTER IF ON SIO
                        ;
0139' 21 0192 '                 LXI H,INITST    ;POINT INIT STRING
013C' 7E                SIOLUP: MOV A,M         ;GET BYTE
013D' D3 03                     OUT TTYS        ;CHANNEL 1
013F' 23                        INX H           ;MUST LEAVE TIME BETWEEN TWO
0140' 00                        NOP             ;NEEDS MORE TIME BETWEEN OUT'S TO SIO
0141'                           IF NOT SER40A   ;SER4A USES THIS PORT FOR PTR
0141' D3 05                     OUT CRTS        ;CHANNEL 2
0143'                           ENDIF
0143'                           IF SER40
                                MOV B,A         ;CHANGE A 37H TO 33H
                                ANI 1
                                ADD A
                                ADD A
                                XRA B
                                OUT PTRS        ;OUTPUT TO THE PTR CONTROL PORT
0143'                           ENDIF
0143' 1F                        RAR             ;ONLY LAST IS ODD
0144' D2 013C '                 JNC SIOLUP
0147' 7E                        MOV A,M
0148' D3 08                     OUT SIOC
                        ;
                        ; ENTER CCP IF WARM START
                        ;
014A' 3A 008D *                 LDA IOBYTE      ;TEST IOBYTE
014D' 3C                        INR A
014E' C2 0000 *                 JNZ ENCCP       ;DONE IF WARM START:
                                                ; GO TO CCP
```

```
                        ;SYSTEM INITIALIZATION, BIOS PORTION, COLD START
                        ;
                        ; INITIALIZE IOBYTE: LST:, RDR:, PUN: FIELDS TO
                        ; 0 OR VALUE OF "INITIOBYTE" ASSEMBLY PARAMETER
                        ; EQUATED ON PAGE 1; CONSOLE TO DEVICE "FOUND" BY
                        ; FOLLOWING CODE
                        ;
                        ; CONSOLE FINDER:
                        ;       CONSOLE INPUT IS FROM PORT 2 OR PORT 4,
                        ;          WHICHEVER INPUTS A SPACE OR CONTROL-T
                        ;          CHARACTER FIRST, OR PORT 2 ON TIME-OUT.
                        ;       CONSOLE OUTPUT IS TO VIO IF PRESENT AND
                        ;          SPACE TYPED OR TIME-OUT; TO SAME PORT
                        ;          AS INPUT IF FORCED BY TYPING CONTROL-T
                        ;          OR IF VIO NOT PRESENT.
                        ;       THE ABOVE RULES DEFINE THE SELECTION OF
                        ;          ONE OF THE PHYSICAL DEVICES TTY:, CRT:,
                        ;          VK1:, OR VK2: FOR THE INITIAL CON:.
                        ;       PORT NUMBERS AND DEVICES ACCESSED WILL
                        ;          CHANGE APPROPRIATELY IF USER CHANGES
                        ;          PHYSICAL DEVICE ROUTINES ABOVE, SINCE
                        ;          CONSOLE FINDER DOES ITS IO THRU PHYSICAL
                        ;          DEVICE ROUTINES.
                        ;
                        ; CAUTION:  IF YOU CHANGE THE CORRESPONDENCE BETWEEN
                        ;   IOBYTE VALUES AND PHYSICAL DEVICES, OR ACCESS A
                        ;   COMPLETELY DIFFERENT DDRIVER IN CON: LOGICAL
                        ;   DEVICE ROUTINES, FOLLOWING CODE MAY REQUIRE CHANGES.
                        ;
                        ;
0151' 11 6000           LXI D,6000H     ;TIME-OUT COUNTER
                                        ;6000=ABOUT 3 SECS
0154'                   FINCON: ;LOOP BACK HERE TIL INPUT OR TIL TIMEOUT
                            ;TRY CRT PORT
0154' CD 00CD '             CALL CRTSTAT
0157' C4 00C2 '             CNZ CRTIN       ;GET BYTE IF READY
015A' 01 0201               LXI B,201H      ;2 FOR VIO, 1 FOR NO VIO
015D' CD 0174 '             CALL TESTIN     ;RETURN IF NOT SPACE OR ^T
                            ;TRY TTY/KEYBOARD PORT LIKEWISE
0160' CD 00A9 '             CALL TTYSTAT
0163' C4 009E '             CNZ TTYIN       ;GET BYTE IF READY
0166' 01 0300               LXI B,300H      ;3 FOR VIO, 0 FOR NO VIO
0169' CD 0174 '             CALL TESTIN     ;RETURN IF NOT SPACE OR ^T
016C' 1B                    DCX D
016D' 7A                    MOV A,D         ;TEST TIME-OUT
01[illegible]' B3           ORA E
0[illegible]' C2 0154 '     JNZ FINCON
```

```
                        ;
                        ; OPERATOR HAS NOT RESPONDED WITH A SPACE OR CONTROL-T
                        ; ON EITHER PORT 2 OR PORT 4.  SUPPLY ONE FOR HIM.
                        ;
0172' 3E 20                     MVI A,' '       ;USE PORT 2
                        ;
                        ;
                        ;
                        ; ROUTINE TO TEST FOR SPACE BAR INPUT (USE VIO IF THERE)
                        ; OR CONTROL-T (DON'T USE VIO) IN ACC.
                        ; RETURNS ONLY IF ACC IS NOT SPACE OR CONTROL-T.
                        ; OTHERWISE, FALLS THROUGH TO SET IOBYTE.
                        ; ON ENTRY:
                        ;       B = IOBYTE VALUE IF WANT TO USE VIO
                        ;       C = IOBYTE VALUE TO SUPPRESS VIO
                        ;
0174' FE 20             TESTIN: CPI ' '
0176' CA 017D '                 JZ TSTIN1       ;SPACE BAR GOOD
0179' FE 14                     CPI CTRLT
017B' C0                        RNZ             ;NO CONTROL-T BAD
017C' 41                        MOV B,C         ;SUPPRESS VIO
017D' 2A 00E3 *         TSTIN1: LHLD VIOF
0180' 79                        MOV A,C         ;ASSUME NO VIO
0181' 2C                        INR L
0182' 2D                        DCR L           ;TEST
0183' C2 018A '                 JNZ SETIOBYTE   ;BRIF VIO NOT THERE
0186' CD F800                   CALL VIOINIT    ;INIT IT
0189' 78                        MOV A,B         ;MOVE MAGIC VALUE TO ACC
                        ;
                        ; AT THIS POINT THE IOBYTE FIELD FOR CONSOLE IS IN A:
                        ;  0 - TTY: (PORT 2 INPUT AND OUTPUT)
                        ;  3 - VK1: (PORT 2 INPUT, VIO OUTPUT)
                        ;  1 - CRT: (PORT 4 INPUT AND OUTPUT)
                        ;  2 - VK2: (PORT 4 INPUT, VIO OUTPUT)
                        ;
018A'                   SETIOBYTE:
                        ; INITIALIZE IOBYTE. CONSOLE FIELD IS IN A.
018A' F6 00                     ORI INITIOBYTE  ;OTHER FIELDS PER INITIOBYTE
                                                ;EQUATE ON PAGE 1 (0 IF NOT
                                                ;MODIFIED BY USER)
018C' 32 014B *                 STA IOBYTE      ;SET IOBYTE

                        ;
                        ;EXIT FROM BIOS TO CCP COLD START ENTRY POINT
018F' C3 014F *                 JMP ENCCP       ;CONTINUE IN CCP
                        ;
                        ;
                        ;
```

```
0192' 00 00 00  INITST: DB 0,0,0,40H,0AEH,37H,22H
0197' 37 22
                ;
                ; TEST FOR MPU-B AND SET TIMER IF THERE
                ;  THIS CODE IS MEANT TO BE EXECUTED AT 80H, NOT HERE
                ;
0199'                   IF SER40A       ;ONLY IF PTR ON SIO PORT 4 & 5
                TEST:   MVI A,MPUIN
                        OUT MEMCTL      ;INSTALL MPU-B AT D800H
                        LXI H,MPUTSA    ;ADDRESS OF MPU-B SIGNITURE
                        MOV B,M         ;SAVE WHAT'S THERE
                        MVI A,0FFH-MPUTST ;COMPLEMENT OF WHAT SHOULD BE THERE
                        MOV M,A         ;TRY TO COMPLEMENT MEMORY
                        XRA M           ;0 IF RAM, 0FFH IF ROM
                        INR A           ;TEST FOR TRUE
                        MOV M,B         ;RESTORE IN CASE RAM
                        JNZ TEST1-TEST+80H ;JIF NOT MPU-B
                        LXI H,MPUTMC    ;POINT TIMER CONTROL PORT
                        MVI M,MPUTV1
                        LXI H,MPUTIM    ;TIMER PORT
                        LXI D,MPUTV2    ;VALUE FOR 9600 BAUD
                        MOV M,E         ;OUTPUT TO PORT
                        MOV M,D
                TEST1:  MVI A,MPUOUT    ;REMOVE MPU-B FROM ADDRESS SPACE
                        OUT MEMCTL
                        RET
0199'                   ENDIF
                ;
0199' 20 28 43          DB ' (C) 1976,1977'
019E' 31 39 37
01A3' 31 39 37
                ;
                ;
                ;
                ;CAUTION: DO NOT ADD ROUTINES HERE UNLESS THEY ARE USED
                ;ONLY DURING SYSTEM INITIALIZATION.  PUT ADDED CODE USED
                ;DURING SYSTEM INITIALIZATION ABOVE "ENSYS".  READ COMMENTS
                ;ON PAGE 15.
                ;
01A7'                   END
```

```
ENSYS   012B'   BIOS    0000'   CONIF   004D'   CONRDY  0039'
CONOUT  0059'   PUNOUT  0071'   LSTOUT  0065'   READIN  007D'
ENCCP   0190*   BDOSRE  00FA*   WBOOT   00FF*   VIOF    017E*
IOBYTE  018D*   OPTION  002A*   COPYHL  0000*   HOME    0016*
SELDIS  0019*   SELTRK  001C*   SELSEC  001F*   BCMD    0035*
DEX     002D*   INITIO  0000    FALSE   0000    TRUE    FFFF
SER40A  0000    SER40B  0000    SER40   0000    TTY     0002
TTYS    0003    CRT     0004    CRTS    0005    SIOC    0008
PRINTE  00F6    PINIT   0080    VIOINI  F800    VIOADR  F803
CTRLC   0003    CTRLT   0014    TAB     0009    LF      000A
FF      000C    CR      000D    CTRLZ   001A    UNDERL  005F
RUBOUT  007F    SELDMA  0032'   CONS    0041'   DISPAT  0089'
TTYSTA  00A9'   CRTSTA  00CD'   TTYIN   009E'   CRTIN   00C2'
TTYOUT  00AE'   CRTOUT  00D2'   VIO$OU  00E2'   LPTOUT  0102'
PUNO    00D2'   RDRIN   0127'   DSHFT   008F'   TTYWT0  00BB'
TTYWT1  00BC'   LPTOU2  00ED'   PRINTM  0118'   LPTOU0  0104'
NULLI   0127'   NULLO   00D2'   NOPINI  0139'   INITST  0192'
SIOLUP  013C'   FINCON  0154'   TESTIN  0174'   TSTIN1  017D'
SETIOB  018A'

ASSEMBLY COMPLETE, NO ERRORS
```