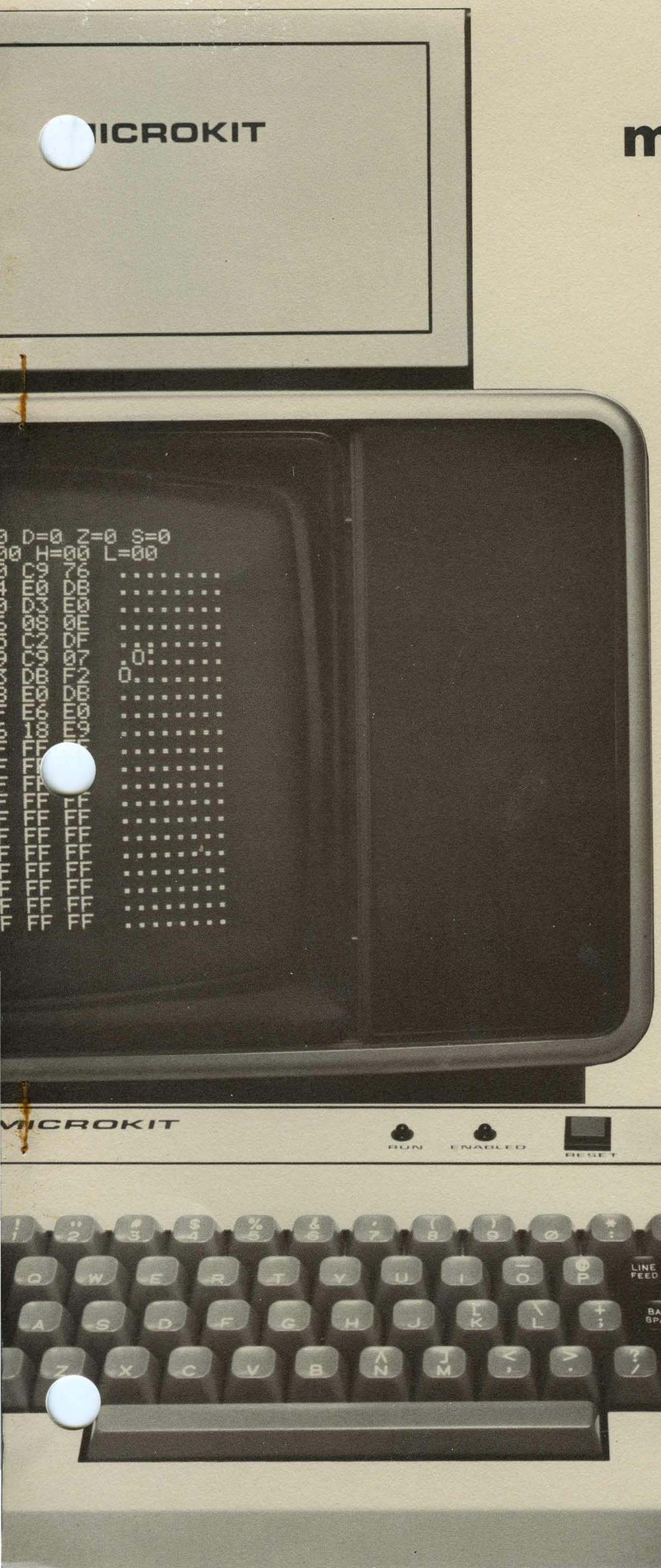


The one microcomputer that tackles every job

product development
data processing
dedicated control

MICROKIT Universal Systems are available for the most popular microprocessors—8080, 6800 and Z-80.

All of these systems are designed specifically to handle a wide variety of applications and are being used as powerful microcomputer development systems, general purpose data processing systems and dedicated controllers.

MICROKIT

MICROKIT MICROCOMPUTER SYSTEMS

MICROKIT INC.
11205 S. La Cienega Blvd.
Los Angeles, CA 90045
(213) 641-7700

May 1977

TABLE OF CONTENTS

Fig. 1 MICROKIT-8/16 System

1.0 MICROKIT-8/16 --- UNIVERSAL MICROCOMPUTER DEVELOPMENT SYSTEM FOR BOTH THE 8080 AND THE 6800

The MICROKIT-8/16 is a complete stand-alone development system for writing, debugging, and executing programs on the 8080 or the 6800 microprocessors. The system can be ordered as an 8080 based system or a 6800 based system. Conversion packages consisting of a plug-in processor module and software are available so that either system can be switched to handle the other processor.

1.1 IN-CIRCUIT MICROEMULATORS

Any microcomputer system used for product development must have an in-circuit emulator. The MICROKIT MICROEMULATOR provides for in-circuit emulation and EPROM programming all in one low-cost package. The MICROEMULATOR also provides a number of additional debugging aids including single-step execution, trace execution, hardware breakpoints, and software breakpoints.

1.2 TAPE AND DISK SYSTEMS COME COMPLETE WITH CRT DISPLAY

The standard MICROKIT system comes complete with an 8K memory, an alphanumeric CRT display, an ASCII keyboard, two cassette tape units, and software.

The MICROKIT CRT display which comes standard with every system holds 960 characters formatted as 24 lines of 40 characters. Since the display is refreshed directly from the microcomputer memory, the full 960 character screen can be written at 20,000 cps.

The cassette tape units are standard low-cost audio recorders. Because of MICROKIT's proprietary recording technique, data is transferred to the audio cassette units at the rate of 2000 bps with data reliability equivalent to digital cassette units.

Two types of floppy disk units are available as optional peripherals. The MICRODISK/2 is a standard dual drive floppy disk unit. This unit provides 500K bytes of high-speed data on-line. The MICRODISK/2M incorporates two minifloppy drives and provides 160K bytes of data.

1.3 TAPE OPERATING SYSTEM SOFTWARE

The MICROKIT Tape Operating System software includes a Debugger/Monitor, an Editor, an Assembler, and a Utility program. All of these programs are designed to take full advantage of the

high-speed CRT display and the tape I/O.

Using the fast CRT the interactive debugging program generates full screen hexadecimal memory dump displays instantaneously. This kind of quick system response speeds the debugging task.

The MICROKIT Editor also makes use of the fast CRT display. The screen based editor is quickly learned and easy to use, because changes are displayed instantly and in full context on the CRT.

1.4 QUICKRUN MEMORY RESIDENT OPERATING SYSTEM

The QUICKRUN system is an exclusive MICROKIT software system that provides the debugger, editor and assembler all co-resident in memory at once. Since the operating system software, the source program, and the object code are all in memory, the user is able to instantaneously switch from editing to assembling to debugging. Virtually all I/O is eliminated during the development processes except for initial loading and final dumping of data to tape. Because everything is in memory, QUICKRUN is even faster than disk for programs of up to 1000 statements that fit into the memory work area.

1.5 MICROKIT DISK OPERATING SYSTEM SOFTWARE

The MICROKIT Disk Operating System adds the speed and versatility of the MICRODISK/2 and MICRODISK/2M floppy disk storage units to the MICROKIT-8/16 system. The Disk Operating System is a sophisticated piece of software that provides full disk data management facilities.

1.6 MICROKIT-8/16 STANDARD FEATURES

The MICROKIT-8/16 incorporates a number of standard features, some of which are unique among microcomputer systems:

* 8K bytes of RAM memory -- expandable to 48K bytes.
* Memory write protect where each 1K page of memory can be write protected under software control.
* 960 character CRT display refreshed from system memory.
* 53 key ASCII keyboard with 2-key roll over.
* Two audio cassette tape units, fully supported by the software.

* A crystal controlled programmable real-time clock with 32 microsecond resolution.
* 8-level vectored interrupt.
* Bootstrap loader in PROM.
* Two EIA RS-232C serial interfaces - one for modem, one for teleprinter.
* 1 megabyte/second DMA capability.
* Six extra card slots for custom interfaces.

1.7 THE MICROKIT SYSTEM IS WELL SUITED FOR MANY APPLICATIONS

With all of the standard features, as well as the long list of optional enhancements, the MICROKIT system is an excellent choice as a microcomputer development system.

However, this is not its only application. It is ideal for many data processing applications especially those which can make effective use of the low-cost tape media or the minifloppy media.

In a barebones form, the MICROKIT system is also perfect for many micro-control applications. Any of the low-cost peripherals can be selected for addition to the barebones system to make a very cost-effective system in industrial/process control, data acquisition, or automatic test equipment applicatons.

2.0 MICROKIT TAPE OPERATING SYSTEM

The MICROKIT Tape Operating System (MTOS) comes with four software support programs: a Debugger/Monitor, an Editor, an Assembler, and a Utility program. The programs all run directly on the MICROKIT-8/16, are supplied on cassette tapes, and load in 30 to 50 seconds.

2.1 TAPE OPERATING SYSTEM DEBUGGER/MONITOR

* Loads and dumps programs on cassette tape.
* Debugging commands display and store memory, set and clear breakpoints, display and set contents of registers.
* Executes programs in memory.
* Handles breakpoint interrupts so programs can be interrupted and restarted.
* Sets and resets memory protect boundaries.
* Tape, EIA, CRT and Keyboard I/O drivers.

The MICROKIT Monitor is loaded from tape into the high order 5K bytes of memory by the PROM bootstrap loader. The memory is then write protected so that the Monitor cannot be accidentally clobbered by a user program.

When the Monitor is in control, a display is generated on the CRT which shows the contents of memory in hexadecimal form. Each display shows 160 bytes of memory. When program execution is interrupted by a stored breakpoint instruction or by the BREAK key, the Monitor stores and displays the CPU status (registers and flag bits). Status can be reloaded, and execution can be continued from the point of interruption. Listed below is a summary of Monitor commands:

D n	Display memory around location "n".
D +m	Advance the display by "m" bytes.
D -m	Back-up the display by "m" bytes.
D *	Display indirect.
CTRL A REPT	Scroll ahead through memory.
CTRL B REPT	Scroll back through memory.
S d,d,...	Store data bytes "d" (entered in hexadecimal) starting at the current display location.
E n	Execute a program starting at location "n".

```
* EXAMPLE ROUTINE, FINDS A PERIOD (.)
        ORG   X'100'
PERIOD  LXI   H,BUFFER    LOAD ADDR. POINT
LOOP    MOV   A,M         FETCH DATA
        CPI   '.'         COMP. WITH '.'
        JZ    FOUND       IF EQ. RETURN
        INX   H           INCR. ADDR. POINT
        MOV   A,L
        CPI   BUFFER+32   CHECK FOR END
--------+-----+----------+-----------------
        JNZ   LOOP
-----------------------------------------
        XRA   A           CLEAR ACC.
FOUND   RET               RETURN
        SPC
BUFFER  EQU   X'200'
        SPC
        END   PERIOD


OUTPUT READY?
```

MICROKIT
EDITOR
DISPLAY

```
 010A 7D                  MOV   A,L
 010B FE20                CPI   BUFFER+32 CHECK
 010D C20301              JNZ   LOOP
 0110 AF                  XRA   A           CLEAR
 0111 C9        FOUND     RET               RETUR

 0200           BUFFER    EQU   X'200'

 0112                     END   PERIOD
REWIND INPUT
-
-----------------------------------------
 PSW            0006
 PERIOD         0100
 LOOP           0103
 FOUND          0111
 BUFFER         0200
                     * EXAMPLE ROUTINE, FINDS A
 0000                     ORG   X'100'
 0100 210002    PERIOD    LXI   H,BUFFER    LOAD
 0103 7E        LOOP      MOV   A,M         FETCH
 0104 FE2E                CPI   '.'         COMP.
 0106 CA1101              JZ    FOUND       IF EQ
 0109 23                  INX   H           INCR.
```

MICROKIT
ASSEMBLER
DISPLAY

```
PC=0000 SP=1000 C=0 P=0 D=0 Z=0 S=0
A=00 B=00 C=00 D=00 E=00 H=00 L=00
00B0   3E CD E5 1B CD E2 1B CD   >.......
00B8   E5 1B FE 0D C2 C5 00 CD   ........
00C0   E1 0A C3 47 01 FE 18 CA   ...G....
00C8   7E 00 FE 31 C2 D4 00 3E   ...1...>
00D0   01 C3 3F 01 FE 32 CA B4   ..?..2..
00D8   00 FE 4C C2 E3 00 3E 02   ..L...>.
00E0   C3 3F 01 FE 4F C2 ED 00   .?..O...
00E8   3E 04 C3 3F 01 FE 54 C2   >..?..T.
00F0   F7 00 3E 08 C3 3F 01 FE   ..>..?..
00F8   45 C2 01 01 3E 10 C3 3F   E...>..?
0100  >21 00 02 7E FE 2E CA 11   !......
0108   01 23 7D FE 20 C2 03 01   .#..  ..
0110   AF C9 C3 3F 01 FE 4D CA   ...?..M.
0118   FD 1B FE 42 C2 37 01 CD   ...B.7..
0120   E2 1B CD E5 1B FE 30 DA   ......0.
0128   37 01 FE 3A D2 37 01 E6   7..:.7..
0130   0F 32 91 0C C3 B4 00 3E   .2.....>
0138   3F CD E5 1B C3 B4 00 21   ?......!
0140   A5 0C B6 77 C3 B4 00 21   ...w...!
0148   A5 0C 7E E6 01 CA 58 01   ..~...X.
0150   21 09 0F 36 80 C3 C1 01   !..6....
>_
```

MICROKIT
MONITOR
DISPLAY

L	Load a program from tape into memory.
W n,m	Write a program starting at location "n" and ending at "m" onto cassette tape.
Z r=d	Set register "r" to value "d". Registers are loaded prior to beginning execution by the "E" command.
X	Switch write protect status of the displayed memory page.
F d,d,...	Find data in memory.
B Sn	Set breakpoint. Four are available, specified by "n".
B Dn	Display breakpoint "n".
B R	Reset breakpoint.
B C	Clear all breakpoints.

A breakpoint can be set in a program by storing a hexadecimal byte "FF", or by using the "B" command which saves the data replaced by the "FF". When executed, the stored "FF" instruction returns to the Monitor for debugging and saves all the CPU registers for display.

The Monitor also provides several useful I/O service subroutines. These include drivers for cassette tape I/O, EIA I/O, and CRT-keyboard I/O.

2.2 TAPE OPERATING SYSTEM EDITOR

* Allows user to edit symbolic data entered from keyboard or cassette tape.
* CRT based, line oriented editor.
* String searching command.
* Character and line insert/replace/delete commands.
* Settable tab stops.

The MICROKIT Editor provides a means for creating and updating source data stored on cassette tapes. Under control of the MICROKIT user, data to be edited is read from tape or keyboard into a work area in the microcomputer memory. Data in this work area is displayed on the CRT, and can be modified by interactive commands. When editing is complete, the data is written from

memory onto a second tape. The work area holds about 200 lines at a time in the basic 8K system. Larger programs are edited by repeating the above process, working on one segment at a time. There is no inherent limit to program size.

The Editor will automatically configure the work area for systems with memories larger that 8K. Each additional 8K available provides space for an additional 550 lines of data.

The CRT display shows 21 data lines. The line in the center of the screen separated by horizontal bars is the line to be edited. Ten lines before and ten lines after are displayed for context. A summary of the Line Editor commands are as follows:

A n	Advance the display "n" lines.
CTRL A REPT	Scroll ahead.
B n	Back up the display "n" lines.
CTRL B REPT	Scroll back.
DEL n	Delete "n" lines.
X	Edit the center line.
I	Begin inserting lines.
F 'string'	Search for "string".
L	Load memory work area from tape.
W	Write memory work area to tape.
N	Write then load.
E	End-file output, and rewind.
REW	Rewind input.
CLR	Clear work area.
T n,n,...	Set tab stops.

During line edit:

SP (space)	Move cursor right.
TAB	Move cursor to next tab.

Back-space	Move cursor back.
CTRL X	Clear line for re-entry.
I	Insert characters.
D	Delete characters.
R	Replace characters.
RETURN	Exit from editing.

2.3 TAPE OPERATING SYSTEM ASSEMBLER

* Standard symbolic assemblers using Intel 8080 mnemonics or Motorola 6800 mnemonics.
* Input from cassette tape.
* Listing to CRT, or printer.
* Object to tape or directly to memory.
* Assembler can generate listing to CRT, listing to printer, object to tape, and object to memory simultaneously during pass 2.

The MICROKIT Assembler generates object programs (i.e., binary data which is loadable and executable by the microcomputer) from a source program written in symbolic assembly language. This program is prepared by the MICROKIT Editor. The Assembler processes the the source program in two passes. During the first pass the Assembler reads the source tape and generates a symbol table which is kept in memory. During pass 2, the Assembler uses the symbol table and converts the symbolic instructions into binary data.

During pass 2 a number of Assembler options are available as follows:

L	Generate a listing on the CRT display.
T	Generate a listing on the CRT display truncated to 40 characters.
P	Print a listing on the MICROKIT line printer.
Bn	Select Baud rate and output listing to a printing terminal connected via EIA port 1.
E	List only lines with errors.
O	Generate an object tape.

The assembly language itself includes a complete set of features. Constants may be symbolic, hexadecimal, decimal, or ASCII, and expressions are allowed using plus and minus. The Assembler allows free format statements, i.e. fields of the statements may be separated by one or more blanks.

In addition to all of the standard microcomputer instructions, the Assembler provides the following useful pseudo instructions:

ORG	Reorigin the location counter.
EQU	Equate a symbol to the value of an expression.
DC	Define constants - one byte, two byte, high byte, or low byte.
DS	Reserve storage space.
END	End of input.
SPC	Blank line in listing.
EJE	Skip to top of next page in listing.

2.4 MICROKIT TAPE OPERATING SYSTEM UTILITY

The Utility program provides and easy way to copy tapes generated by the Editor, and object tapes generated by the Assembler and Monitor.

The Utility program itself is available in source form. Since the Utility program is designed in a modular form it is possible to add drivers for I/O to devices other than tape. This allows transfers between cassette tape and other devices such as printers, paper-tape readers, paper tape punches, etc.

MICROKIT TAPE OPERATING SYSTEM (MTOS)

SAMPLE ASSEMBLY LISTING GENERATED BY MICROKIT ASSEMBLER:

```
MICROKIT ASSEMBLER -- VER 2.0

                ******************************************
                *                                        *
                *       BUBBLE SORT                      *
                *                                        *
                ******************************************

0000                    ORG   X'100'     ORIGIN AT 100
0100 310001 ENTRY       LXI   SP,X'100'  SET STACK PTR
0103 0EFF               MVI   C,255      INIT LOOP CTR
0105 210002             LXI   H,X'200'   AREA TO BE SORTED
0108 71     ILOOP       MOV   M,C
0109 23                 INX   H
010A 0D                 DCR   C
010B C20801             JNZ   ILOOP
010E 2600               MVI   H,X'200'
0110 0EFF               MVI   C,255

0112 7E     SORT        MOV   A,M        OUTER LOOP
0113 E5                 PUSH  H          SAVE ADDR
0114 41                 MOV   B,C        INIT INNER LOOP

0115 BE     SORTL       CMP   M          COMPARE
0116 DA1C01             JC    SORT1
0119 56                 MOV   D,M        XCHG DATA WITH MEM
011A 77                 MOV   M,A
011B 7A                 MOV   A,D
011C 23     SORT1       INX   H          INCR ADDRESS
011D 05                 DCR   B          DECR LOOP CTR
011E C21501             JNZ   SORTL

0121 E1                 POP   H          END OF INNER LOOP
0122 77                 MOV   M,A        STORE DATA
0123 23                 INX   H          INCR ADDRESS
0124 0D                 DCR   C          DECR LOOP CTR
0125 C21201             JNZ   SORT
0128 76                 HLT              DONE
0129                    END   ENTRY      ENTRY POINT
```

3.0 QUICKRUN --- MEMORY RESIDENT OPERATING SYSTEM

QUICKRUN is a complete "in memory" operating system for developing 8080 or 6800 microcomputer programs using the MICROKIT-8/16. QUICKRUN provides a program development tool unmatched by any other software development system. It consists of a monitor/debugger, editor and assembler all co-resident in memory along with a 14K byte source code workspace and 4K byte object code work space. This is sufficient to store a 1000 line source program.

3.1 INSTANTANEOUS PROGAM DEVELOPMENT

Since the system software, the source program and the object code are simultaneously in memory, the user is able to instantaneously switch from editing to assembling to debugging. This facilitates rapid programming development because changes are made directly in the source program. Then the program is re-assembled and is ready for testing in a matter of seconds.

Once the QUICKRUN software is loaded, and once the source program is loaded at the beginning of a development session, no further I/O is required. For the usual application where the source program will fit into the 1000 statement memory work area, program development is even faster than disk based systems.

3.2 MEMORY IS WRITE PROTECTED

QUICKRUN makes use of the MICROKIT system's unique 8K RAM memory boards with write protection registers. This allows each 1K page of memory to be individually protected and unprotected under program control. Without this feature, the user could inadvertently "clobber" his source program during execution and destroy hours of work.

Non-volatile storage of debugged programs (source and object) is provided by the MICROKIT-8/16 system cassette tape units. Loading and dumping to cassette tape takes less than 3 minutes and is usually done only once per session. Thus QUICKRUN provides the microcomputer programmer with tools that were previously available only on the largest of computer systems, and makes the MICROKIT-8/16 superior to systems costing more than twice as much. Yet the MICROKIT-8/16 with QUICKRUN is still the least expensive development system with peripherals.

3.3 QUICKRUN MEMORY REQUIREMENTS

The QUICKRUN Operating System requires 32K of microcomputer memory. User programs can occupy the first 4K of memory, from 0000 to 0FFF. The system software, editor work area, and assembler symbol table occupy the region from 1000 to 7FFF.

3.4 QUICKRUN DEBUGGER/MONITOR, EDITOR, AND ASSEMBLER

The QUICKRUN Memory Resident Operating System uses the standard MICROKIT editor, assembler and debugger/monitor programs with a series of enhancements.

Three system commands have been added to all programs. These allow the user to switch between the three system programs: Editor, Assembler and Monitor. They are:

JA - Jump to the Assembler
JM - Jump to the Monitor/Debugger
JE - Jump to Editor

The Assembler program differs from the standard tape assembler in that it ordinarily takes its input from the editor work area, and places the object data directly in memory. However, it can assemble from tape as well, and it can generate object tapes too.

The Debugger program has one significant enhancement over the standard Debugger/Monitor. Values of symbols in the assembler symbol table can be referenced by their symbolic name. For example, if a program just assembled has a symbol "LOOP" in it, the debugger command "D #LOOP" will move the display to the memory location assigned to the symbol by the assembler. Each time the program is reassembled, the symbol table value will be automatically updated to the new symbolic values.

3.5 PROGRAM DEVELOPMENT USING QUICKRUN

The program development process using QUICKRUN begins by loading the QUICKRUN system from tape. Control is transferred to the Editor using the "J E" command. If the program to be tested is already on tape, the source tape is loaded by the Editor. Otherwise the program is entered from the keyboard.

Once the program is in the Editor work area and has been modified, it is ready for assembly and testing. The Assembler is invoked by the "J A" command. While many assembly parameters are available, if no parameters are specified for the assembly, the QUICKRUN assembler will read the program in the Editor work area, and place

the object code directly in memory. An assembly will take from 1 to 15 seconds depending upon the length of the source program. A program of about 1000 statements will take about 15 seconds to assemble.

If any syntax errors are detected, a message will indicate this at the end of the assembly. When errors are expected it is a good idea to request the error only display on the CRT by the "E" option. Then error lines will appear and can be corrected in the Editor by doing a "J E".

Once the program is assembled, testing can begin. The Debugger is called using the "J M" command. The user program can then be executed using the "E" command. The full range of debugging commands are available to track down any bugs.

When changes are to be made, it is easiest to return to the Editor rather than patching in hexadecimal. When changes are made directly in the source program, it can be reassembled and ready for testing in a matter of seconds.

Only actual usage of the MICROKIT QUICKRUN system will communicate the real speed and power of this unique program development concept.

4.0 MICROKIT DISK OPERATING SYSTEM (MDOS)

The MICROKIT Disk Operating System (MDOS) adds the speed and versatility of the MICRODISK/2 or the MICRODISK/2M floppy disk storage units and a sophisticated disk operating system to the MICROKIT-8/16 Microcomputer Development System.

With MDOS the MICROKIT user is able to edit and assemble large programs 16 times faster that with the tape based system, and 160 times faster than TTY based systems. The floppy disk media combined with the unique screen based editor, interactive debugger, high-speed assembler, and optional in-circuit emulator makes the MICROKIT-8/16 the most powerful microcomputer development system available.

MDOS has a number of significant design features which make it one of the most flexible and powerful disk operating systems available on any microcomputer system. These features are:

* Variable size files.
* Allocation by tracks.
* Files can be write protected.
* Files can be permanent so they can't be accidentally deleted.
* Files can expand even after filling initial allocation.
* Unused space can be released from files.
* Loader in EPROM allows virtually all of memory to be loaded from diskette.
* Disk I/O can be overlapped with processor execution.
* Entire diskette can be copied in 3 minutes.
* Data can be loaded into memory at the rate of 3840 bytes per second.
* 32K bytes of memory can be loaded in less than 10 seconds.
* A 2000 line program can be read by the Editor in less than 10 seconds.
* A 2000 line program can be assembled in less than 45 seconds.
* Diskette files can be copied to and from cassette tape.
* Easy transfer from one system function to another.
* All data management functions are callable from user programs.
* User can easily add new functions to the system.
* Diskettes are formatted so that 5 sectors can be read per revolution, or 30 sectors per second.
* Automatic CRC test and error recovery.
* IBM compatible media.

4.1 DISKETTE FORMAT

Each diskette has 77 tracks, the first of which is reserved by MDOS to store the directory of files on the diskette. Thus a

maximum of 76 tracks is available for user files.

Each track has a capacity of 26 sectors, which are 128 bytes in length. This makes the capacity of a track 3328 bytes, and the capacity of the diskette 252,928 bytes. Since none of this space is required for system data, it is all available for user data.

Space for files under the MDOS system is reserved in multiples of tracks. That is, the smallest amount of disk space that can be allocated to a file is one track. Consequently there can be a maximum of 76 files per diskette. In the other extreme, the maximum amount of space that can be allocated to a single file is the full 76 tracks of available space.

4.2 USING THE MDOS SYSTEM FOR PROGRAM DEVELOPMENT

The MICROKIT Disk Operating System provides all of the software tools necessary for the rapid development of large microcomputer programs. The MDOS system is composed of four major functions: Editor, Assembler, Debugger, and Monitor.

The sofware development process typically proceeds in the following manner. First the EDITOR is used to enter a program in symbolic assembly language. The program is typed into the Editor which stores the data in files on the diskette media. The Editor is also used in later revision of the program to read it from diskette, revise it, and write it out again.

Once a program has been stored on diskette in symbolic assembly language form, the ASSEMBLER is used to translate it into binary or machine language form. The Assembler reads the file generated by the Editor, referred to as a "source" file, converts the symbolic program into the binary executable equivalent, and writes the executable form into another file called an "object" file. The Assembler is capable of detecting a number of simple syntax errors, and flags lines with errors in them when they are found.

The program can be tested when it is loaded into memory by the DEBUGGER. The Assembler only provides for the translation of the program from symbolic assembly language form into executable machine language form, but does not allow for program testing. The Debugger provides for the testing and "debugging" of the program. The program is loaded from the object file produced by the Assembler into the microcomputer memory. The memory then contains the machine readable and executable form of the symbolic program first generated in the Editor. The Debugger has a command which begins execution of the program. The program can make use of the MICROKIT CRT display for output, and the keyboard for input. If the program does not execute properly, as is usually

the case initially, a whole range of facilities are provided by the Debugger to diagnose the problem.

Execution of the program can be interrupted by breakpoints set by the Debugger. A breakpoint is a place in the program where control will be returned to the Debugger so that the contents of the CPU registers and memory can be examined. Memory data can be changed by the Debugger, and the program tested again. Through memory displays, register dumps, and breakpoints the program errors are soon found.

Then the program revision process begins again by entering the changes into the symbolic program through the Editor. The Assembler is again used to translate the source program to object, and the Debugger is used to load and test the program. Through repetition of this cycle, the program is quickly perfected.

The MDOS system facilitates the development cycle by considerably reducing the amount of time to carry out the editing, assembling, loading, and testing cycle. For large programs, the MDOS system quickly returns the initial investment by saving significant amounts of expensive program development time.

4.3 MDOS FILES

The MICROKIT Disk Operating System is actually a collection of files which contain the executable modules of the system:

MONITOR (File name: MON; Size: 2 tracks)

> Provides disk data managment functions such as create files, delete (scratch) files, rename files, display file names, change file attributes, and copy files.

EDITOR (File name: EDIT; Size: 2 tracks)

> Allows for creating and updating source (text) files stored on diskette.

ASSEMBLER (File name: ASM; Size: 3 tracks)

> Converts.source files created by the editor into object files which are loadable and executable.

DEBUGGER (File name: DEBUG; Size: 2 tracks)

> Displays the microcomputer memory in hexadecimal and ASCII, and allows for the editing of data in memory. The debugger also allows program breakpoints to be set and reset. When a

breakpoint is encountered during execution, the debugger generates a full register and flag display.

The Debugger provides for object files to be loaded into memory for execution or dumped from memory after being modified by the debugger.

UTILITY (File name: UTIL; Size: 3 tracks)

The utility program provides the capability of copying data between the various peripheral devices (e.g. tapes, disk, paper tape, etc.) available for the MICROKIT-8/16.

OBJECT FILES (File name: TAPE-IO, Size: 1 track)

Contains the tape, EIA, CRT and Keyboard routines available in the Tape Operating System Monitor. This is an object file which means that it contains the routines in executable form.

SOURCE FILES (File names: IO-SUBR and UTIL-SRC)

The first file contains source code for the EIA, CRT, and Keyboard routines, and the second file contains source for the Utility program. The I/O subroutines are provided for inclusion with user generated programs. The Utility program source code is provided so that additional customized functions can be easily added.

4.4 TRANSFERING BETWEEN SYSTEM FUNCTIONS

All functions of MDOS recognize the jump "J" command to pass control to another system function. The form of the "J" command is the character "J" followed by the first letter of the system file name containing the desired function.

The following table lists the jump commands:

Command	Transfers to
JM	Monitor
JE	Editor
JA	Assembler
JD	Debugger
JU	Utility
JI	Emulator Debugger (optional)
JW	Word Processor (optional)

4.5 MDOS MONITOR

When the MICROKIT disk operating system is initially loaded, the Monitor program receives control. The Monitor provides all the necessary disk data management functions including create, delete (scratch), rename, free space, exchange names, change attributes, display file names, copy file, and copy all files.

The monitor commands are as follows:

COMMAND	FUNCTION
C filename	Creates a file with name "filename".
C filename,init,ext	Creates a file with the name "filename", having an initial size of "init", and an extension size of "ext". Both "init" and "ext" are specified as decimal numbers between 1 and 76.
S filename	Delete (Scratch) "filename".
A filename,attributes	Changes the attributes of "filename". The list of attributes follows as a sequence of the letters O, S, W, P, or Z. Attributes are: S -- source file O -- object file W -- write protect file P -- permanent file Z -- system file
D	Display information about the files on unit 0. Along with each file name, is a list of the attributes and the size in tracks.
D unit	Display the files on "unit", which may be "0" or "1".
I	Initialize the diskette in unit 1. All files are deleted and an empty directory is formatted.
J function	Transfer control (Jump) to the system function specified by "function".

M from-name,to-name	Copies (Moves) the contents of "from-name" to the file specified by "to-name".
R old-name,new-name	Renames "old-name" to "new-name".
E filename1,filename2	Exchanges the names of the two files "filename1" and "filename2". This can be used to exchange "old" and "new" Editor source files.
F filename	Frees any unused tracks in "filename" and returns them to the free track list.
X from-unit,to-unit	Copies (Xfrs) all non-system files (i.e. without "Z" attribute) from "from-unit" (0 or 1) to "to-unit" (0 or 1).
X from-unit,to-unit,S	Copies only system files. This allows a convenient way of copying MDOS to a new diskette.
X from-unit,to-unit,A	Copies all files, system and non-system. This allows a convenient way of making a backup copy of a diskette.

4.6 MDOS EDITOR

The MICROKIT disk operating system editor is entered from any of the system functions using the "JE" command. The editor provides the capability for creating and updating source (text) files on diskette.

The commands of the DOS editor are for the most part identical to those of the Tape Operating System Editor.

The commands "L" (Load), "REW" (Rewind), "W" (Write), "N" (Next), and "E" (End-file) are either not needed, or have been modified to provide for the specification of input and output files necessary for the operation of the disk system.

4.7 MDOS ASSEMBLER

The MICROKIT Disk Operating System Assembler is capable of reading assembly language source files and performing the translation into machine executable object files.

The assembler functions are identical to those of the Tape Operating System.

4.8 MDOS DEBUGGER and MICROEMULATOR DEBUGGER

The Disk Operating System Debugger provides the same basic debugging features as the Tape Operating System Monitor. The commands provided by the debugger are "D" (Display), "S" (Store), "E" (Execute), "X" (Change protection), "F" (Find), "B" (Breakpoint set-reset), "L" (Load), "W" (Write), "Z" (Change register values), and "J" (Exit). The breakpoint function of the BRK key and the breakpoint instruction RST 7 (X'FF') are also implemented by the disk system debugger.

The Microemulator Debugger is made available when the optional MICROEMULATOR in-circuit emulator and EPROM programmer is ordered. In addition to the commands provided by the standard debugger, the Microemulator Debugger has the commands for controlling emulation, single-step execution, trace execution, hardware breakpoint, and EPROM programming. The additional commands provided by the Microemulator Debugger are "N" (Execute with interrupts disabled), "T" (Trace execution), "M" (Set emulation mode), "C" (Verify EPROM empty), "P" (Program EPROM), "V" (Verify EPROM), and "I" (Input data from EPROM).

The primary difference between the MDOS debuggers and the Tape Operating System Debuggers is the memory loading and dumping commands. The "L" (Load) and "W" (Write) commands have been extended to allow for loading memory from and dumping memory to disk files.

4.9 MDOS UTILITY

The Disk System Utility function provides the capability of transferring data between the various media supported by the MICROKIT system.

In the first release of the Disk Operating System, only the transfer of data between disk and cassette tape is implemented. New releases of the utility will be made available when additional functions are available. However, the complete source code for the utility module is supplied with MDOS so that the user can implement drivers for additional or custom peripheral devices.

5.0 MICROEMULATOR IN-CIRCUIT EMULATOR

The MICROKIT MICROEMULATOR provides the design engineer and programmer with a valuable tool that extends all of the MICROKIT-8/16's powerful editing, assembling and debugging facilities directly into a hardware system that is being developed. The designer/programmer is able to exercise all of the user system I/O devices, memory, interrupt structure and direct memory access facilities through the MICROEMULATOR plug. With the MICROEMULATOR, the MICROKIT-8/16 will respond to user system interrupts, ready line, reset line, hold line and clocks. It will generate addresses, status signals and strobes and it will send to and receive data from the user system.

When the MICROEMULATOR plug is installed in the CPU socket of the user system, the MICROKIT system can access the memory and I/O devices of the user system. Programs residing in the MICROKIT RAM memory can execute and access memory of the user system as if they were running in the user system. Thus the MICROKIT RAM can be used as PROM or RAM simulator in place of memory in the user system. Of course since the MICROKIT system can access memory in the user system, it can directly execute programs residing in the PROM or RAM memory of the user system.

The MICROEMULATOR consists of four modules:

1. The MICROEMULATOR logic board, which plugs into the MICROKIT system card cage, and controls the connection between the microprocessor within the MICROKIT and the user system.

2. The MICROEMULATOR plug assembly, which provides the processor signals at the end of a remote cable that can be plugged directly into the system microprocessor socket.

3. The hardware breakpoint, single-step, and EPROM programmer interface module, which also plugs into the MICROKIT card cage.

4. The EPROM programming assembly with a zero-insertion force socket, which is connected via 26 pin flat cable to the EPROM programmer interface module.

The MICROEMULATOR package provides three separate functions. These are:

1. The extension of the MICROKIT-8/16's microprocessor into the user system.
2. The analysis features of hardware breakpoint and single step.
3. The ability to program the 2708 (1024 x 8) and the 2704 (512 x 8) EPROMs.

This combination of tools is especially useful in the task of combined hardware and software development. It should be particularly valuable during product development, hardware software integration, production test, and depot maintenance of microprocessor systems.

5.1 BASIC OPERATION OF THE MICROEMULATOR

The MICROEMULATOR uses the same microprocessor for both the host functions of program development, and the user functions of emulating the microprocessor in the system under test. Commands are included in the MICROEMULATOR Monitor that allow the designer to switch over various of the microprocessor functions into the user system. Thus, there is separate software control of 1) the source of clocks; 2) the source and response to interrupts, reset line and ready line; 3) response to user DMA requests; and 4) memory mapping functions.

The MICROEMULATOR software package is extremely simple to use and easy to learn. In addition to the basic MICROKIT Monitor commands, the MICROEMULATOR Monitor provides for enabling emulation mode, single-step execution, trace execution, hardware breakpoints, and EPROM programming.

During the initial phases of a design program, before any prototype hardware is available, the MICROKIT is used in its normal mode to develop software. Gradually, as the hardware system is brought up, various functions can be switched into the user system for testing.

First the clocks are switched over to see if the clock circuits, sync., etc., are working properly. Memory of the user system can be displayed by the MICROEMULATOR Monitor.

Second, test programs are executed so that response to interrupts, reset and ready line can be tested. User DMA mode is connected and enabled as required.

Third, diagnostic programs are developed and run using the MICROEMULATOR to test the operation of I/O devices and verify the hardware design.

Fourth, the user system is exercised with all program and data memory in the MICROKIT memory, thereby simulating PROM and RAM in the user system.

Fifth, the memory functions are switched over to the user system with user RAM and user PROM taking the place of

MICROKIT PROM and RAM simulator memory.

Finally, sixth, the MICROKIT is completely disconnected and the user microprocessor is installed in the system.

This stepwise approach to debugging hardware and software is extremely effective since each step is a relatively small extension of the preceding step.

5.2 MICROEMULATOR DEBUG COMMANDS

Only a few commands have been added to the standard Monitor/Debugger but provide access to all the power of the MICROEMULATOR. Because the MICROEMULATOR has the effect of making all of the user system memory appear to be within the MICROKIT system, the user PROM and RAM memory can be displayed exactly like MICROKIT system memory. In fact, when the MICROEMULATOR is plugged into the user system, all of the normal Debugger/Monitor commands function for the user system exactly the same way that they do for the MICROKIT system itself. The "D" display command is used to display 160 bytes at a time, and the "S" store command is used to make changes in the memory data. Also, the "E" execute command is used to begin execution of programs in the user system memory.

The commands provided to handle the special features of the MICROEMULATOR are as follows:

M P,P,P	Set operating mode. Four parameters can be specified. C -- switch to user clocks. U -- enable to user control lines. D -- enable user DMA request. E -- does all of above.
T	Begin trace execution at present display location.
T addr	Begin trace execution at hexadecimal address "addr".
T#	Recall last trace display.
CTRL E	Single-step the program.
CTRL E REPT	Slow-step the program at about 10 instructions per second.
B 50	Set hardware breakpoint to current display location.

B D0	Display location of hardware breakpoint register.
N	Begin execution at the current display location with interrupts disabled.
N addr	Begin execution at location "addr" with interrupts disabled.
C	Check that EPROM is completely erased.
I addr	Input data from EPROM into memory starting at location "addr.
P addr	Program EPROM from memory starting at location "addr".
P addr,len	When length is specified, this provides for programming smaller 2704 EPROMs.
V addr	Verify that the EPROM data matches memory data.

When single-step execution is used, the MICROEMULATOR Monitor updates the memory dump display and the register display after execution of each instruction. This lets the programmer watch the execution of a program in intimate detail.

The trace execution mode stores the instruction executed and the full register contents after the execution of each instruction. The trace execution save area hold data for the last 20 instructions. When a program is interrupted by a hardware breakpoint, software breakpoint, or manual breakpoint (BRK key), the MICROEMULATOR Monitor generates a complete trace display. This display has one line for each instruction in the trace table. The address, the instruction code, the registers, and the stack pointer are all saved and displayed.

MICROKIT SYSTEM SPECIFICATIONS

6.1 STANDARD MICROKIT-8/16 HARDWARE CONFIGURATION

The MICROKIT-8/16 system consists of the following elements: mainframe, keyboard, CRT display, and two cassette tape units. The mainframe is packaged in a compact desk-top enclosure and contains power supplies, a card cage, and the four system modules: CPU, RAM, CRT-Keyboard I/O, Tape-EIA I/O.

The MICROKIT system is built around a universal system buss through which all the system modules communicate. The CPU module interfaces to this buss and controls the operation of the system. The 8080 and 6800 are only the first of a series of processors to be supported by the MICROKIT-8/16 system. The modular nature of the system makes it adaptable to other popular 8 and 16 bit processors.

The 8K byte memory of the standard system is contained on one memory module. The system memory is easily expanded by plugging in additional 8K modules. Each 1K page of memory is separately write protectable under software control.

Input/output from the MICROKIT-8/16 is provided by the two I/O modules. The CRT-Keyboard I/O module contains the interface logic for the keyboard and the CRT display. The keyboard interface supports the mini-operator console located on the keyboard. The console functions include two switches: "system reset" and "initial program load" (from cassette tape), and two status displays: "run", and "interrupt enabled". All other front panel functions and debugging aids are provided by an interactive debugger through the CRT display.

The second I/O module, which is the Tape-EIA module, contains a number of interfaces as follows:

Tape Interface - The tape interface connects the microcomputer to the two audio cassette tape units.

Dual RS-232C Interface - This allows bit serial communication with both a terminal and a modem. Speeds up to 1200 Baud are possible, under program control.

Crystal Controlled Real Time Clock - The real time clock is used by the RS-232C interfaces for accurate bit timing. The clock is also available for other timing applications.

8-Bit Parallel I/O Port - This TTL port is unassigned and allows for interfacing a user designed device. It provides a bidirectional test port that can interface with user equipment.

Bootstrap PROM - The 256 byte PROM contains a bootstrap loader which may be invoked by the initial load button on the keyboard-console. It loads the system software from cassette tapes.

6.2 MICRODISK FLOPPY DISK UNITS

Two floppy disk units are available for the MICROKIT system. The MICRODISK/2 is a dual standard floppy disk unit, while the MICRODISK/2M is a dual minifloppy disk unit.

Both systems consists of two elements. The dual disk drive and an interface card which plugs into the MICROKIT system buss. The interface card contains the disk bootstrap PROM and the parallel I/O interfaces required by the disk controllers.

The disk drive controllers are contained in the disk units themselves. Both disk units contain the following: two disk drives, a controller, and power supplies. The disk controllers provide a full compliment of disk commands, as well as sector size read and write buffers. The disk commands include seek, seek track 0, read, write, and reset.

6.3 MICROPRINT/65 LINE PRINTER

The MICROPRINT/65 is a 65 lpm line printer for use with MICROKIT-8/16,Microcomputer Development System. The MICROPRINT/65 prints bidirectionally at 65 lines-per-minute on standard TTY roll paper. The printer utilizes a 5 x 7 dot matrix to produce highly legible characters. Standard TTY ribbons are also used, thus the user is free from the bother and cost of special paper or special ribbons.

The MICROPRINT/65 is a simple, proven and reliable line printer capable of heavy duty operation. It can print single and multipart forms. Options are available for form feed fan-fold paper, and upper and lower case printing.

The MICROPRINT/65 is delivered complete with interface and software to run with the MICROKIT-8/16 microcomputer system. Optional software includes a Word Processor package for use with the upper/lower case printer.

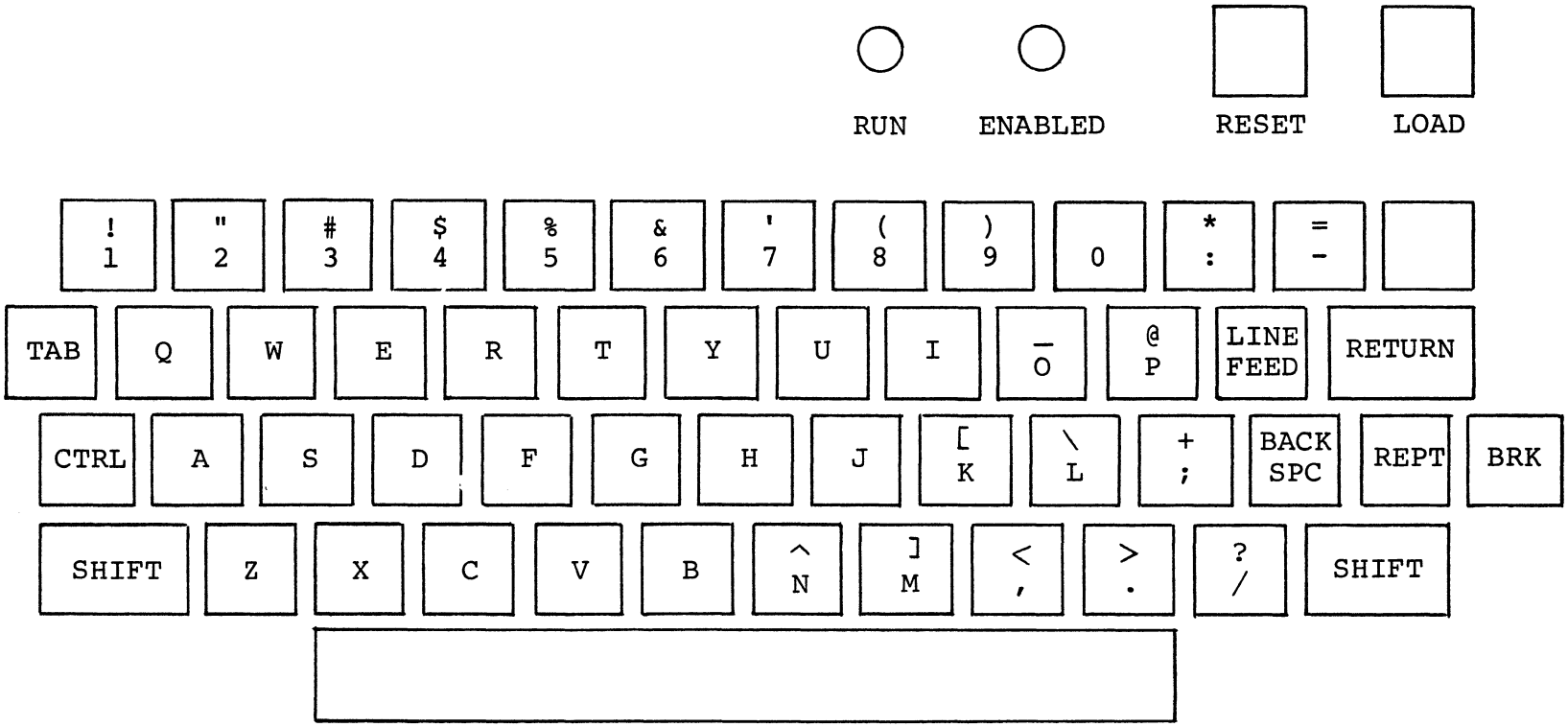


Fig. 2 MICROKIT KEYBOARD

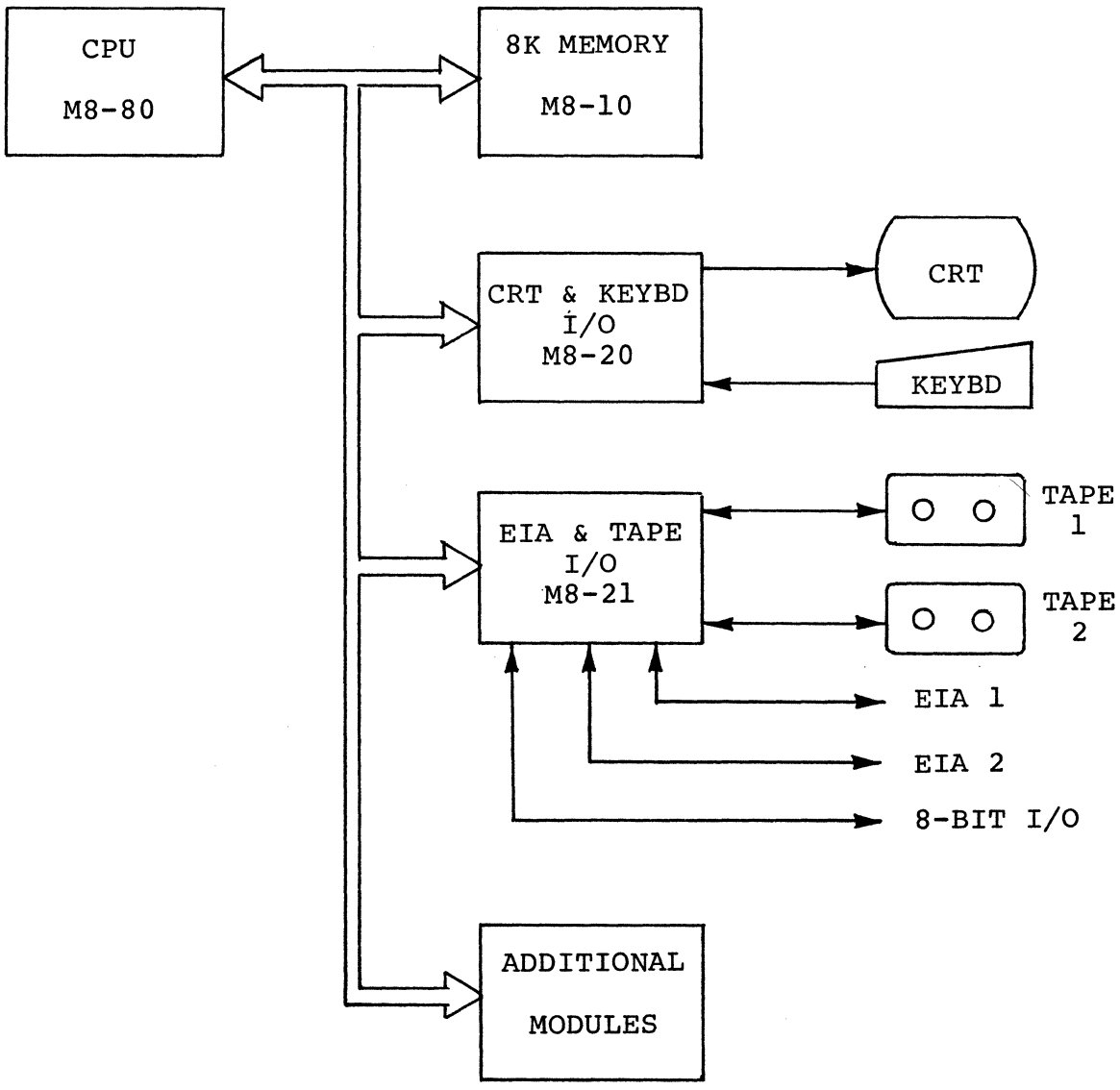
6.4 TABULAR MICROKIT-8/16 SYSTEM SPECIFICATIONS

CPU	8080 or 6800
Word Size	8 bits
Memory Size	8K bytes, expandable using 8K modules to 32K bytes within the basic enclosure
Memory Protect	Write protect for each 1K bytes of memory
Memory Cycle	460 ns read, 970 ns write
Interrupts	All I/O devices have maskable interrupts
System Clock	Crystal controlled, 2 MHz +/- 0.1% for 8080, 1 MHz +/- 0.1 % for 6800
I/O Channels	CRT interface Keyboard with mini-console interface Dual tape interface Dual EIA RS-232C interface 8-bit parallel input/output port
DMA	Standard
Real Time Clock	Crystal controlled, with 32 us +/- 0.1% resolution
System Buss	Universal, no assigned card positions
Keyboard	53 key ASCII, 2 key rollover
Mini-console	System "restart", bootstrap "load", "run" display, "interrupt enabled" display
Cassettes	(2) single track, Philips type audio cassette units. Read and write at 100 to 200 characters per second.
CRT	Modified Panasonic 12" black and white television
Standard System Modules	CPU 8K RAM CRT-Keyboard I/O Tape-EIA I/O

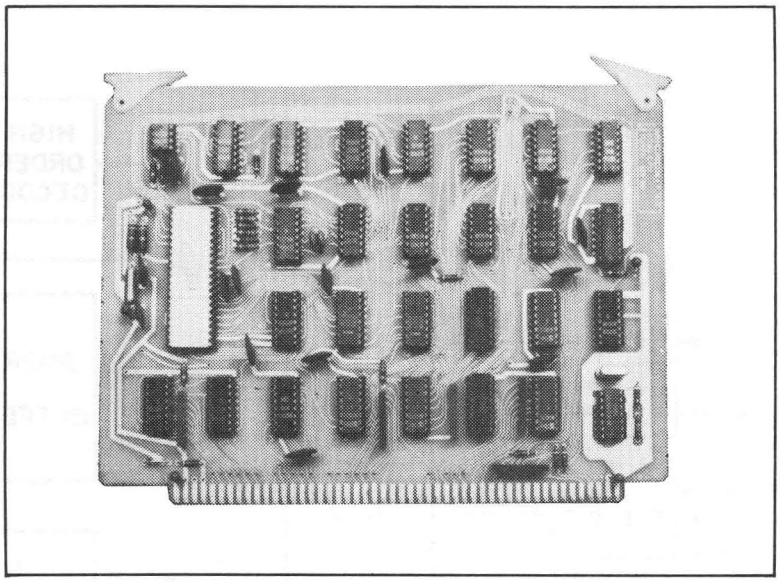
MICROKIT SYSTEM SPECIFICATIONS

Standard Software	Monitor Editor Assembler Utility PROM bootstrap loader	
Standard Documentation	Assembler Reference Manual	(74 pgs.)
	Editor Reference Manual	(28 pgs.)
	Hardware Reference Manual	(46 pgs.)
	Operating and Programming Manual	(72 pgs.)
No. of Card Slots	10	
Operating Temp.	10 C to 40 C	
DC Power Supplies	+5V @ 8A +12V @ 1.5A -12V @ 1.5A (Larger power supplies may be required for expanded systems)	
AC Power	50/60 Hz, 115/230 VAC, 150 Watts	
AC Receptacles	(3) switched outlets for tape units and CRT	
Physical Size	7" x 17" x 12"	
Weight	25 lbs.	
Optional Modules	M8-30 Universal prototype board M8-31 Module extender M8-10 8K RAM memory module M8-11 8K PROM and 2K static RAM module	

MICROKIT SYSTEM BLOCK DIAGRAM

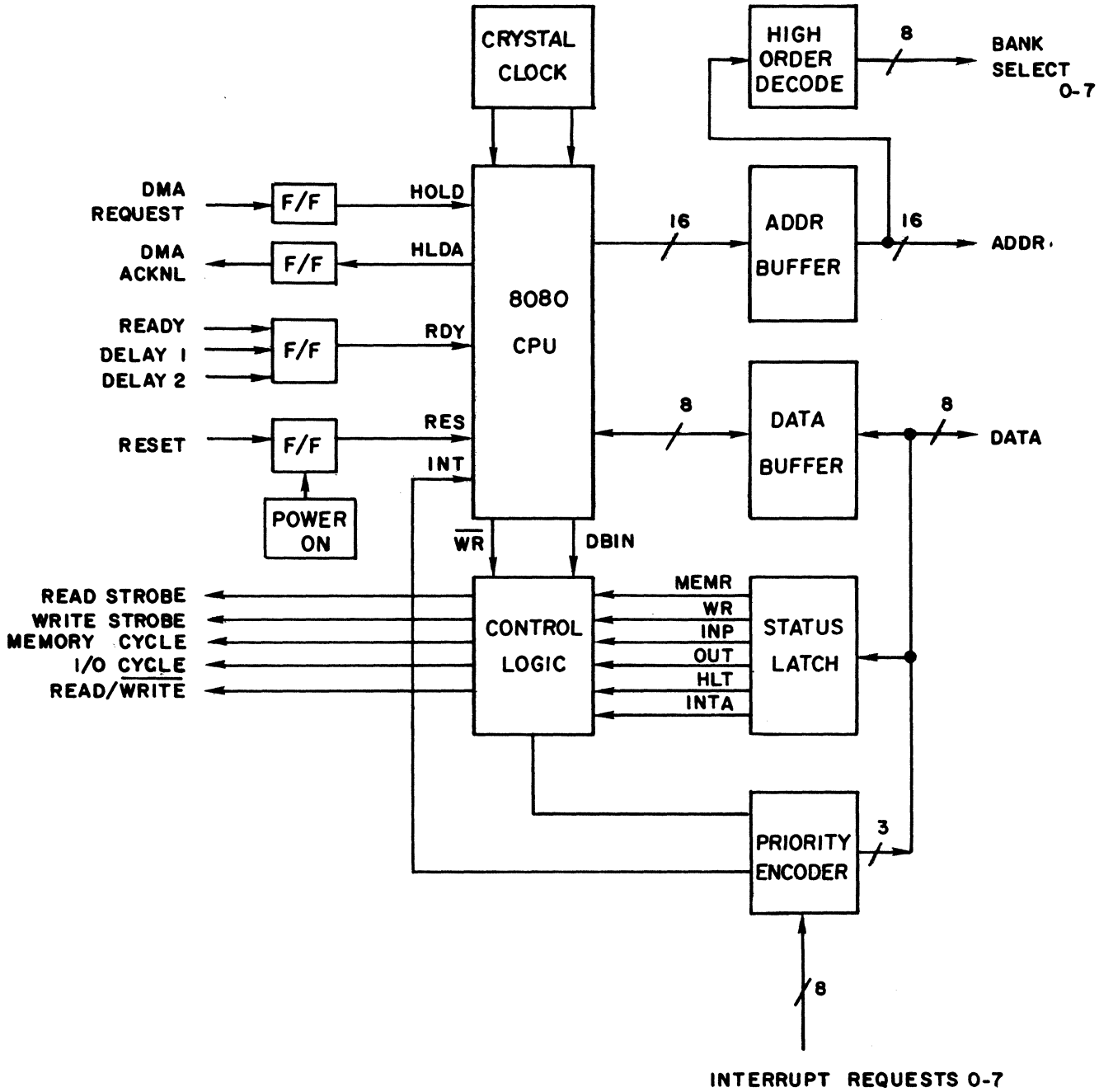


7.1 8080 CPU MODULE SPECIFICATIONS M8-80

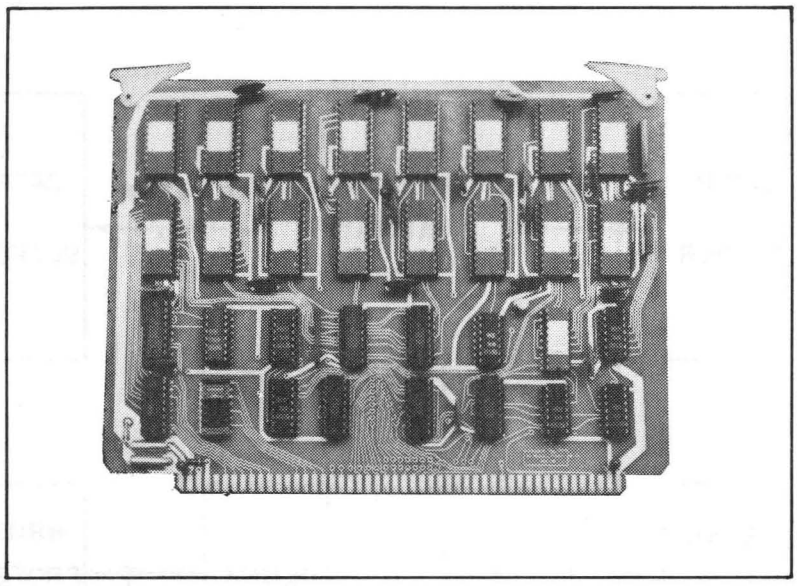


Processor	8080
Instrucion Set	78 including conditional branching, binary and BCD arithmetic, logical operations, subroutine stack, 7 registers
Memory Addressing	16 bits, up to 64K bytes
I/O Addressing	8 bits, up to 256 input and 256 output addresses
System Clock	Crystal Controlled, 2.0 MHz +/- 0.1%
Interrupts	8 level vectored priority interrupts
Power-on Restart	Automatic bootstrap load
Connector	Dual 50-pin on 0.125 centers Wirewrap P/N CPH8100-100 (SAE)
Board Dimensions	6.00" x 8.00" x 0.062"
Operating Temp.	10 C to 40 C
DC Power Requirements	+5 +/- 5% @ 0.65A +12 +/- 5% @ 0.07A -12 +/- 5% @ 0.05A

8080 CPU BLOCK DIAGRAM (M8-80)

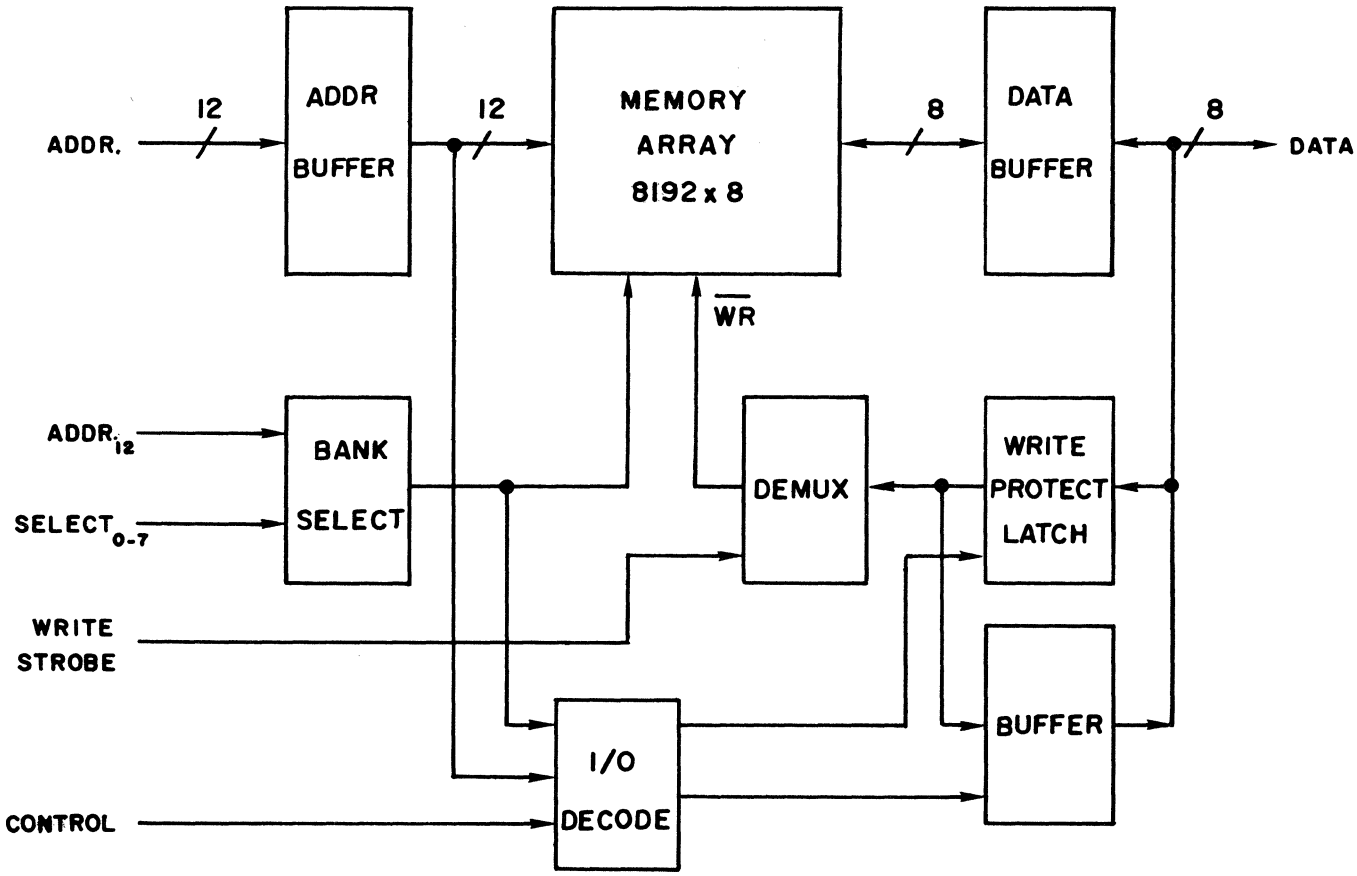


7.2 RAM MEMORY MODULE SPECIFICATIONS M8-10

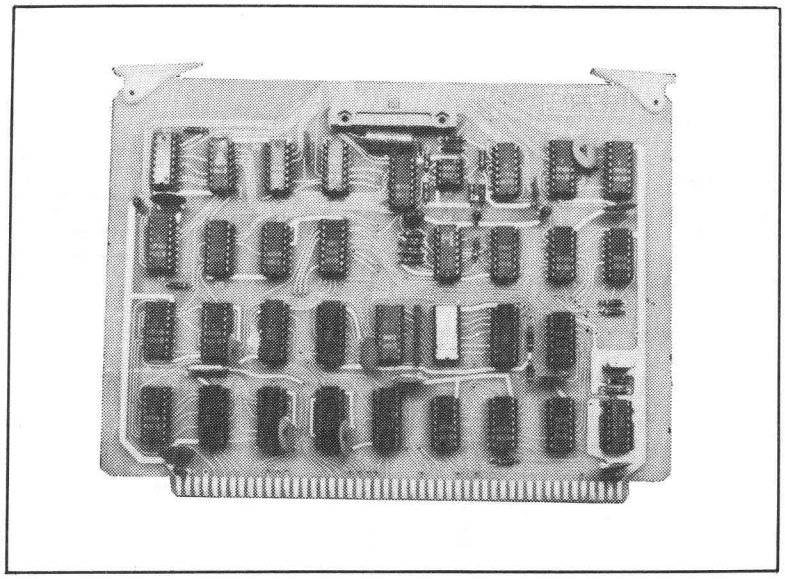


Memory Size	8K bytes
Word Size	8 or 16 bits (16 bits requires two modules)
Cycle	460 ns read; 970 ns write using 4K dynamic RAM chips
Refresh	64 us every 512 us accomplished by CRT access
Connector	Dual 50-pin on 0.125 centers Wirewrap P/N CPH8100-100 (SAE)
Board Dimensions	6.00" x 8.00" x 0.062"
Operating Temp.	10 C to 40 C
DC Power Requirements	+5 +/- 5% @ 0.5A +12 +/- 5% @ 0.20A -12 +/- 5% @ 0.05A

MEMORY BLOCK DIAGRAM (M8-10)

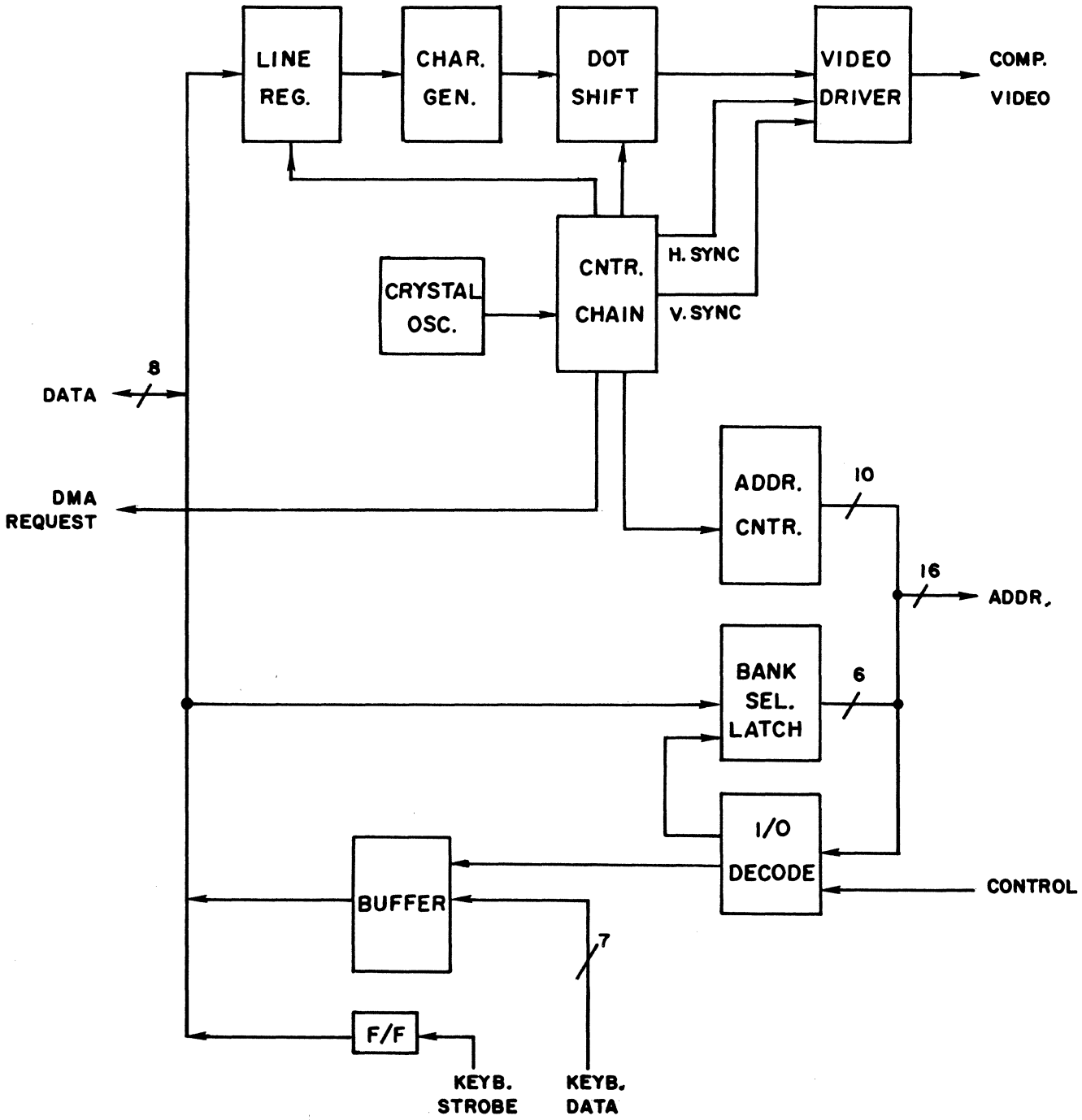


7.3 CRT - KEYBOARD I/O MODULE SPECIFICATIONS M8-20



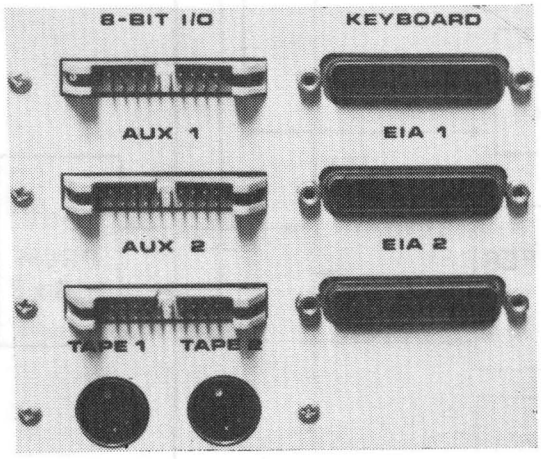
CRT Display Size	24 lines of 40 characters
Character Set	Standard 64 character ASCII set
Character Generation	5 x 7 dot matrix
Display Memory	Accesses main memory using DMA
I/O Transfer Rate	20,000 cps
Refresh Rate	60 Hz
Video Output	Composite video without interlace
Interrupts	Maskable interrupts for keyboard strobe and break interrupt (BRK) key
Keyboard Interface	7 bit ASCII with character strobe
Mini-Console Interface	Switch closures for system "reset" and bootstrap "load"; TTL outputs for driving "run" and "interrupt enabled" LED's.
Connector	Dual 50-pin on 0.125 centers Wirewrap P/N CPH8100-100 (SAE) 20-pin Scotchflex header Connector P/N 3421 (3M)

CRT & KEYBOARD I/O BLOCK DIAGRAM (M8-20)



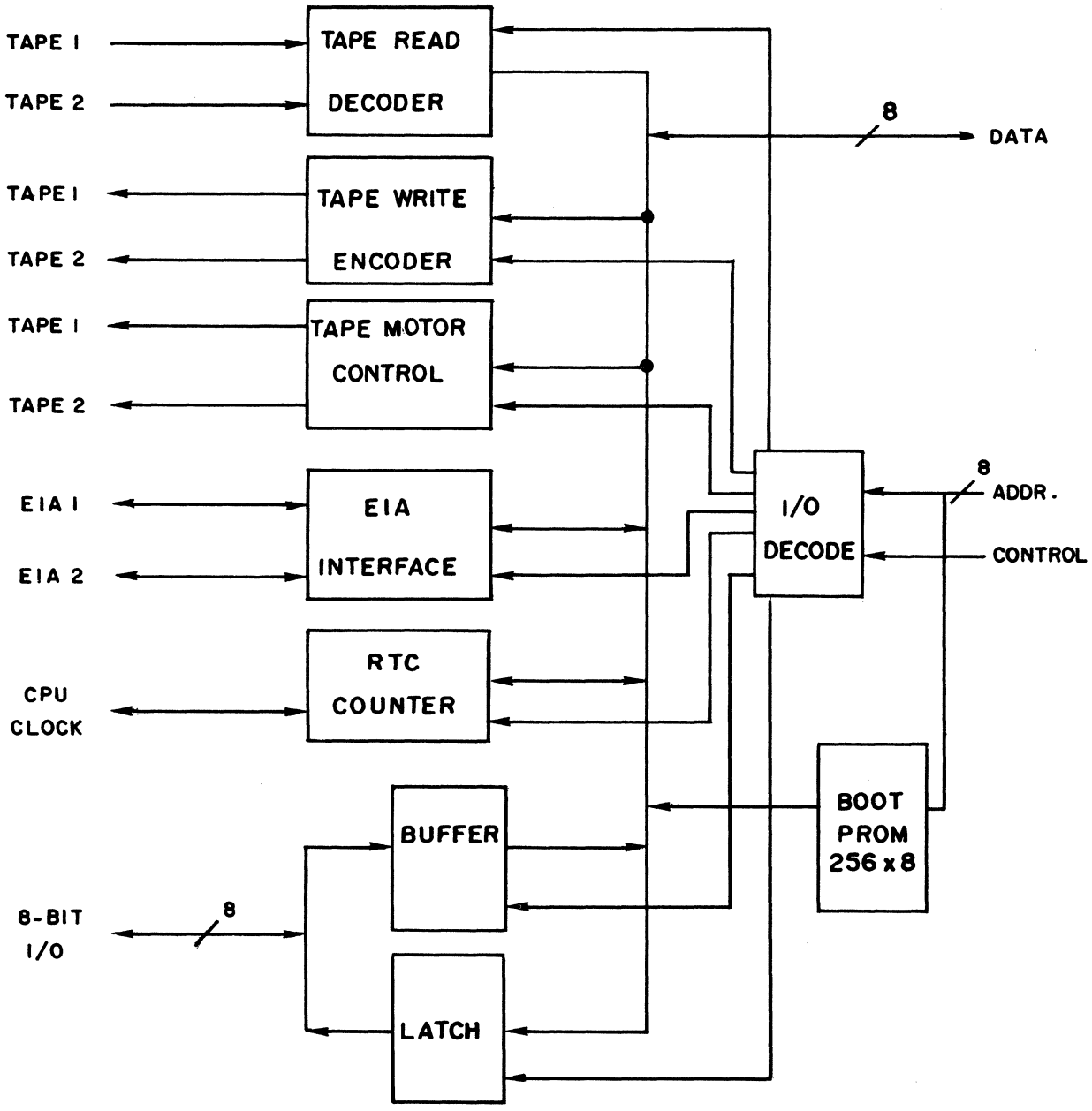
MICROKIT MODULE SPECIFICATIONS

Board Dimensions	6.00" x 8.00" x 0.062"
Operating Temp.	10 C to 40 C
DC Power Requirements	+5 +/- 5% @ 0.7A -12 +/- 5% @ 0.05A

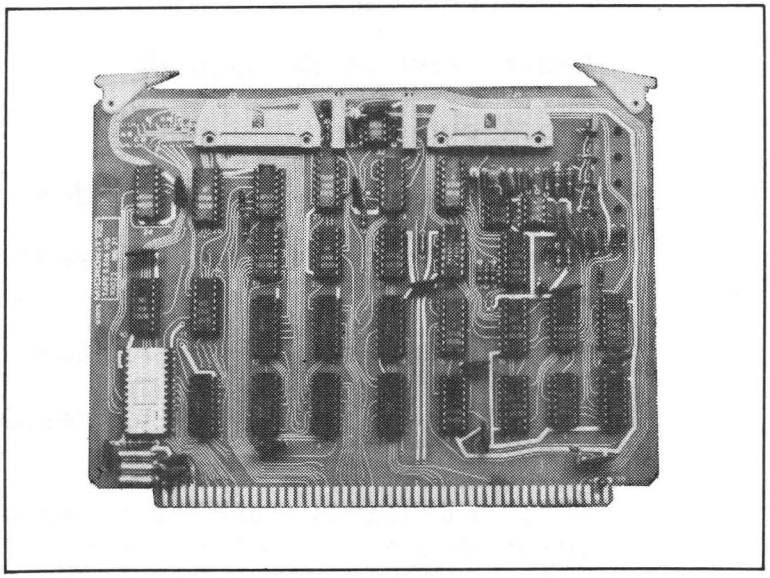


MICROKIT-8/16
I/O Connectors

TAPE & EIA I/O BLOCK DIAGRAM (M8-21)



7.4 TAPE - EIA I/O MODULE SPECIFICATIONS M8-21



Cassette Type	Philips type audio cassettes
Tape Units	Panasonic RQ-309AS
No. of Heads	One
No. of Gaps	One
No. of Tracks	One
Recording Technique	Phase encoded (Harvard)
Bit Transfer Rate	2000 bps
Recording Density	1067 +/- 2% bpi
Character Transfer Rate	225 cps max. 100 cps average including overhead
Error Detection	16-bit checksum
Error Correction	Redundant recorded data
Error Rate	Recoverable: 1 in 1,000,000 bits Unrecoverable: 1 in 100,000,000 bits
Record Size	Variable (1 to 32K bytes)

MICROKIT MODULE SPECIFICATIONS

Inter Record Gap	1.875 inch (1 second)
BOT, EOT	Recorded codes
Manual Rewind	End-to-end in 60 seconds
Write Protect	Knock out tab
No. of EIA lines	One to terminal, one to modem
Modem Control	Request to send, clear to send, carrier detect
Baud Rates	50 to 1200 software generated
Real Time Clock	Crystal controlled, 32 us resolution; overflows at 2.048 ms
I/O Port	8 bit parallel, general purpose TTL, input and/or output for party line I/O or interprocessor communication.
PROM Memory	256 bytes for loader
Interrupts	Clock overflow and I/O port request; all maskable
Board Dimensions	6.00" x 8.00" x 0.062"
Connector	Dual 50-pin on 0.125 centers Wirewrap P/N CPH8100-100 (SAE) 20-pin Scotchflex header Connector P/N 3421 (3M)
Operating Temp.	10 C to 40 C
Power Requirements	+5 +/- 5% @ 0.65A +12 +/- 5% @ 0.03A -12 +/- 5% @ 0.05A

Note on the preparation of this document:

8.0 MICROKIT WORD PROCESSOR

This report was prepared using the MICROKIT Word Processing system. The MICROKIT Word Processor allows the user to print neatly formatted reports, manuals, documents etc. It is used in conjunction with the MICROKIT Editor program. The text to be printed is first entered, along with format commands, using the Editor. The Word Processor program is then used to read the text from the Editor tape, format the data according to the imbedded commands, and print it.

The Word Processor recognizes a series of format commands. These format commands are of the form ↑AA (where AA represents two alphabetic characters). The Word Processor also recognizes /A (where A is any alphabetic character) as a case shift. Thus if the user is in the normal lower case mode then "/THE" will print "The."

The basic unit of text is the paragraph. New paragraphs are signalled by ↑PP at the begining. Each time the Word Processor encounters ↑PP it will skip two lines and begin a new paragraph. Text within a paragraph is entirely free form. The Word Processor will fit the maximum number of words on each line that it can without violating the line width limits. Thus all extra spaces will be suppressed on print out. Other commands such as ↑AI (As-Is Mode) are used to suppress this blank suppression mode of operation.

The Word Processor has software interfaces for the MICROPRINT/65-LC line printer and the BCD version of the IBM 2741 "Selectric" terminal.

The format commands recognized by the Word Processor are:

COMMAND	FUNCTION
↑NL	Execute a carriage-return and line feed to position the print head at the beginning of a new line. Overides paragraph rules.
↑PG	Execute a form feed (line printer) or request a new page (2741 terminal).
↑UC	Set upper case mode (shift-lock).
↑LC	Set lower case mode (shift-unlock).

↑PP	Skip two lines. Position print head at the first column and start a new paragraph.
↑BS	Backspace on the 2741 terminal only. Used for underlining or overstriking.
↑SKnn	Skip nn lines.
↑IFnn	Conditional new page. Avoids printing of headings as the last lines on a page. Skips if line is within the last nn lines.
↑AI	Enter the As-Is mode. This will defeat the extra space suppression feature of the normal paragraph mode of operation.
↑EA	End the As-Is mode.
↑TI	Flags the following text as a title to be printed on each page. (Note: title must be followed by a ↑PG command to ensure printing title on the first page).
↑ET	Ends the title.
↑NT	Suppresses the title on all subsequent pages.
↑IM	Indents all subsequent text. The indent will be 5 spaces unless reset by the ↑IS command.
↑EI	Ends the indent mode.
↑ISnn	Set the indent to value nn.
↑LWnn	Set the line width to nn. The default line width is 66 characters.
↑PSnn	Set the number of printed lines per page to nn. The default number of printed lines is 53.
↑IL	Start a label at the left margin during indent mode.
↑EL	Flags the end of a label.
↑TBnn	Position the print head to column nn. Overides the paragraph mode.
↑RS	Reset to default Page Width, Page Length and Indent.

EXAMPLE OF TEXT AS SEEN ON THE CRT SCREEN

```
↑PP/THIS IS AN EXAMPLE
OF THE USE OF THE
↑UCMICROKIT↑LC /WORD
/PROCESSOR /PROGRAM.
/IT USES MANY OF THE
COMMANDS IN THE
REPORTOIRE.
/FOR EXAMPLE:
↑NL↑IM↑IS20
↑PP↑IL/LABEL↑EL
/LABELING AN INDENTED
PARAGRAPH.
/THIS USES THE /I/L
AND /E/L COMMANDS.
↑PP↑IL/TAB /COMMAND↑EL
↑TB40 /MOVES THE PRINT HEAD
↑NL↑TB40 TO POSITION 40.↑EI
↑PP/THESE ARE A FEW
EXAMPLES OF THE USE OF
THE ↑UCMICROKIT↑LC
/WORD /PROCESSOR /PROGRAM.
/WITH A LITTLE
/PRACTICE ITS USE
SHOULD BECOME SECOND
NATURE.
```

EXAMPLE OF THE WORD PROCESSOR OUTPUT AS PRINTED OUT IN HARD COPY

This is an example of the use of the MICROKIT Word Processor Program. It uses many of the commands in the reportoire. For example:

Label — Labeling an indented paragraph. This uses the IL and EL commands.

Tab Command — Moves the print head to position 40.

These are a few examples of the use of the MICROKIT Word Processor Program. With a little practice its use should become second nature.

MICROKIT, INC. • 11205 S. La Cienega Blvd. • Los Angeles, CA 90045 • (213) 641-7700

MICROKIT