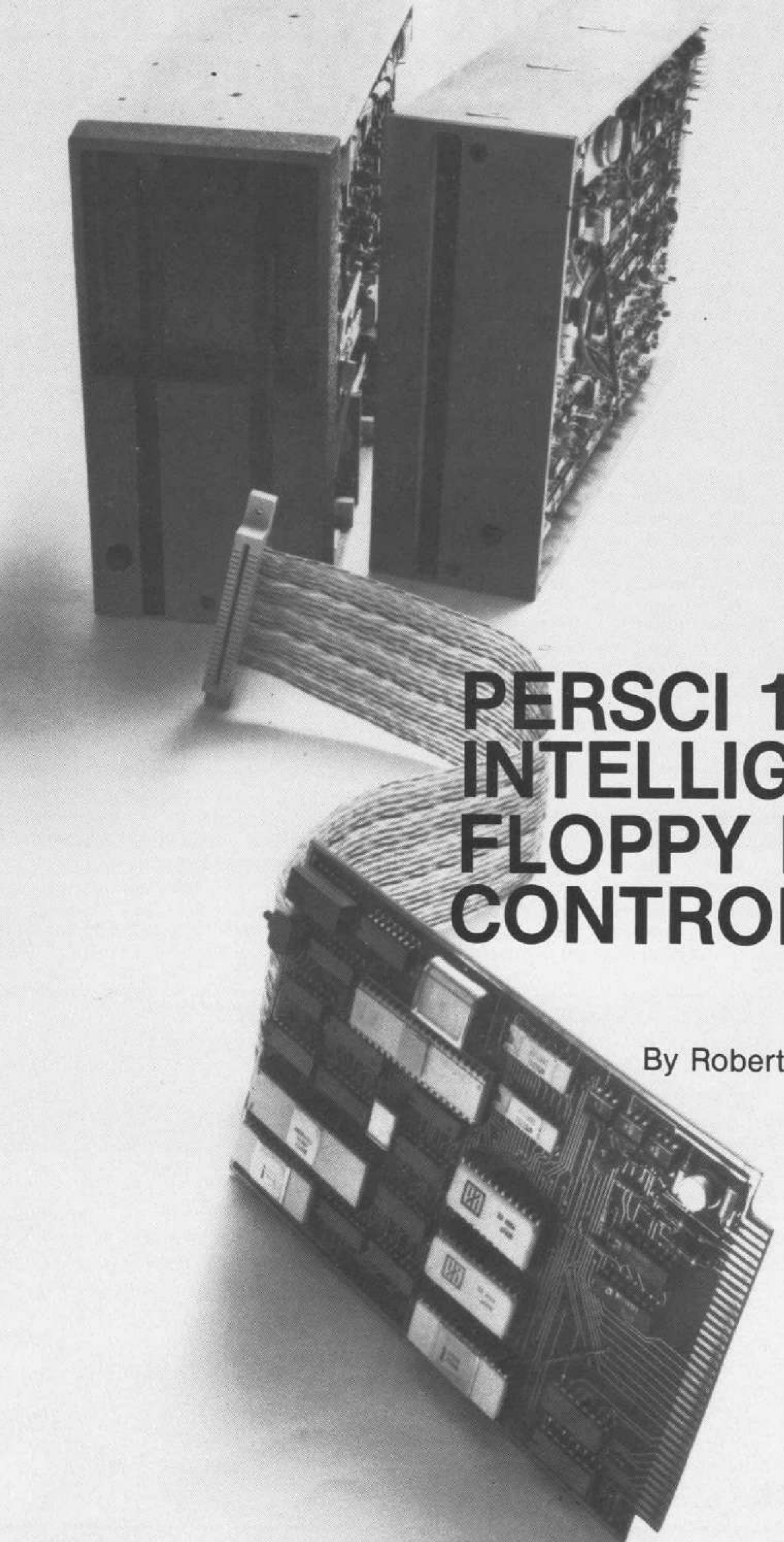


PERSCI 1070 INTELLIGENT FLOPPY DISK CONTROLLER

By Robert A. Stevens

*Reprinted from **INTERFACE AGE MAGAZINE** September 1977*

PERSCI 1070 INTELLIGENT FLOPPY DISK CONTROLLER

By Robert A. Stevens

FOREWORD

This article is the first article of a two-part series covering PerSci's Model 1070 Intelligent Floppy Disk Controller and Model 70 and 277 Floppy Disk drives. The first article covers the 1070 Intelligent Floppy Disk Controller functional architecture in detail and includes a logical interface design for integrating PerSci's Controller and Floppy Disk Drives into your S-100 bus 8080 microcomputer system.

INTRODUCTION

The Model 1070 Controller is the first and still the only truly intelligent diskette controller with its own 8080 microcomputer, two memory ports and a file type disk operating system capable of being interfaced to any microcomputer. The 1070 controller provides on-board capabilities to communicate by file name with most microcomputers and at the same time, takes care of all housekeeping functions. One controller board controls up to four PerSci Model 70 single diskette drives or up to two PerSci Model 277 dual diskette drives, providing a high-performance mass storage subsystem with an on-line date storage capacity of more than one million bytes. For reference, see Figure 1.

MAJOR INTELLIGENT CONTROLLER FEATURES

The common denominator 1070 Controller PCB provides the following on-board major features when fully populated with all options:

- Controller configured as a dedicated general purpose 8080 microcomputer.
- Controller mechanized with Western Digital's single chip floppy disk controller LSI IC.
- Co-ordinated handshaking programmed I/O parallel data transfer between controller and host computer allows all types of data access and transmission formats.
- Provides ASCII Text, ASCII HEX object code & executable binary object code storage formats.
- Provides four data access storage transmission formats (direct access, stream access, relative access & punctuated access).
- Controller mechanized with two host computer memory ports.
- Two memory port controller can be interfaced to any host computer via the high speed & bit parallel memory port or via the optional RS232 serial asynchronous memory port.
- Controls up to four PerSci Model 70 Single Diskette Drives or up to two PerSci Model 277 Dual Diskette Drives.
- Can be utilized to control other manufacturer's equivalent floppy disk drives without hardware or PDOS modifications.
- On-board 1K byte static RAM communication data buffer storage memory provides data transfers between host computer and diskette without passing the critical *read/write* timing parameters associated with the disk drive to the host computer.
- On-board microcomputer with ROM Resident PDOS FMF resolves problem of requiring a different DOS for every type of computer CPU interfaced to.
- Host computer requires only a minimal I/O driver routine unique to its CPU to complete the total DOS software requirements (168 bytes in a typical 8080 based microcomputer system).
- Resident PDOS FMF can be customized to special applications without redesigning hardware (although not a standard option, this capability does provides means of customizing the controller to meet special situations for quantity orders).

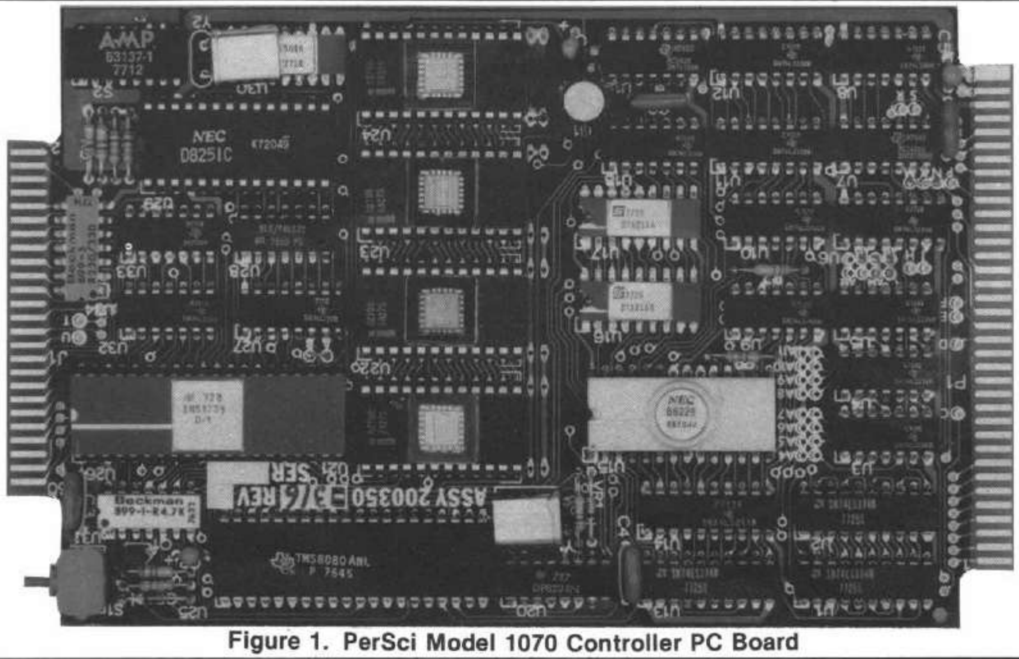


Figure 1. PerSci Model 1070 Controller PC Board

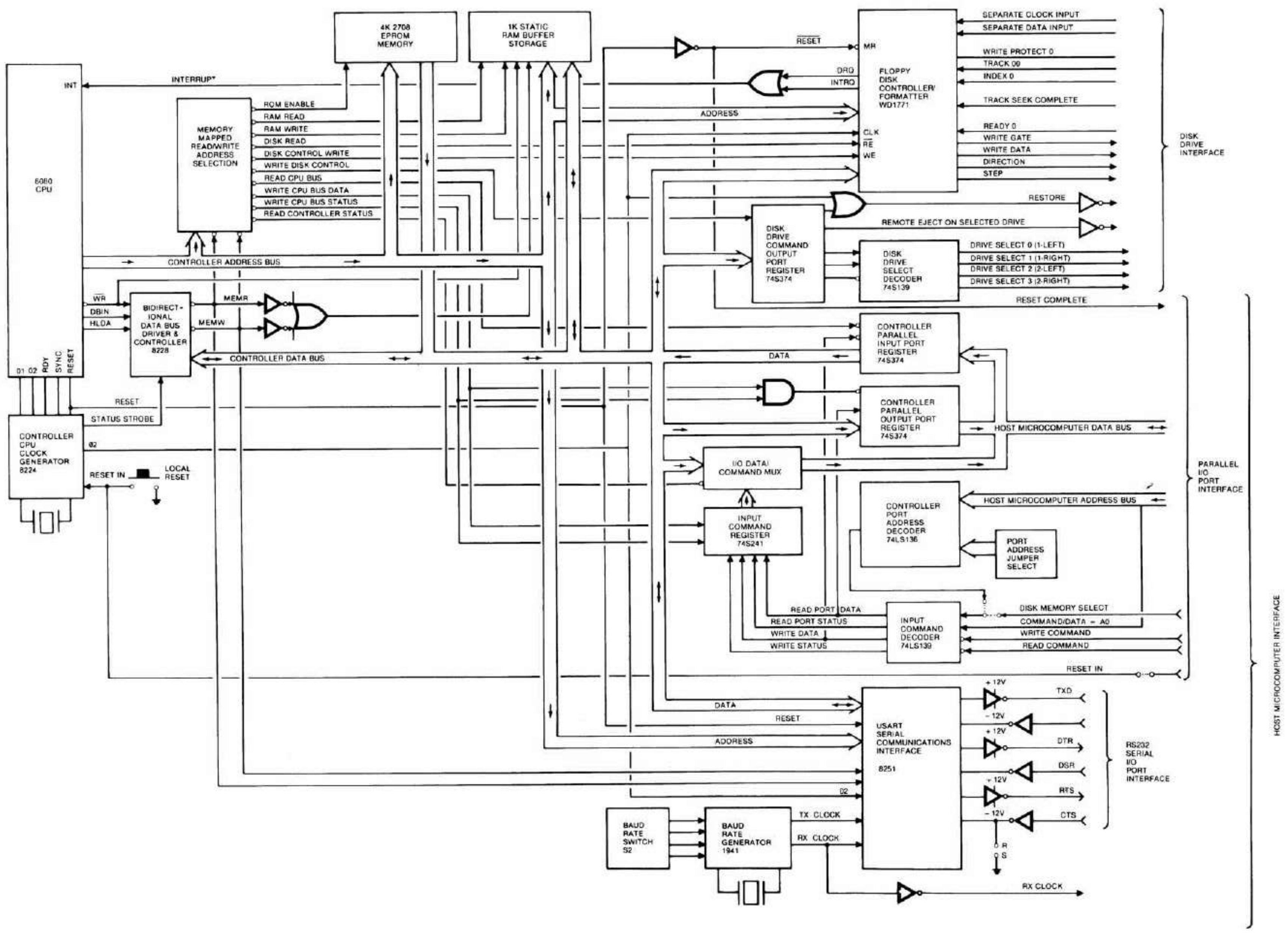


Figure 2. Model 1070 Intelligent Floppy Disc Controller Functional Block Diagram

Reprinted from ***INTERFACE AGE MAGAZINE*** *September 1977*

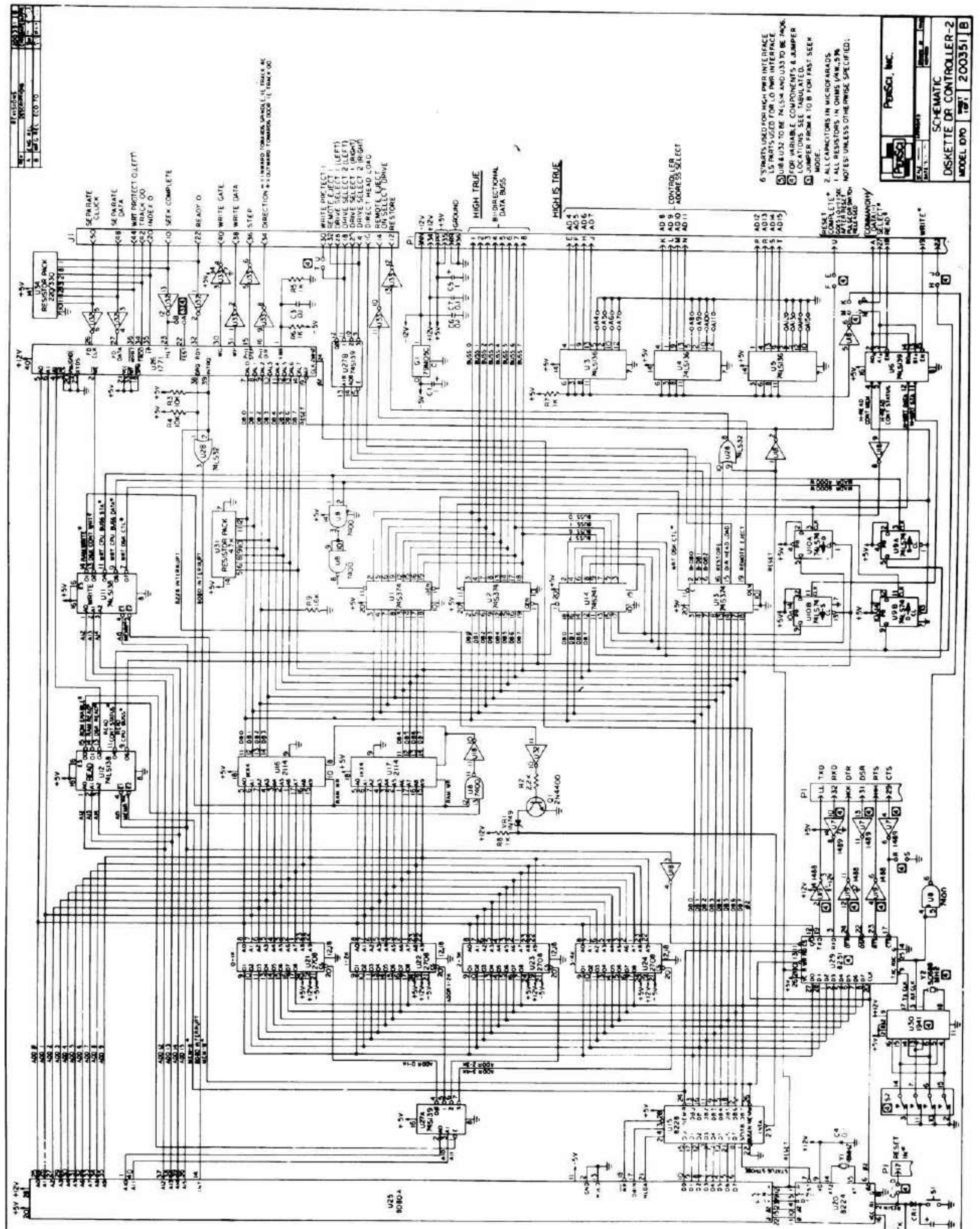


Figure 3. Model 1070 Intelligent Floppy Disk Controller Logical Program

- PerSci's PDOS FMF Modified IBM soft sectored disk storage format allows 76 tracks available for file data storage and permits up to a 100-file index table on a single index track.
- PDOS FMF allows up to 252,928 bytes of data storage/single sided diskette volume.
- IBM 3740 Soft Sectored Disk Storage Format available as an option.
- PDOS FMF Diskette Initialization function provides choice between non-interleaved and twelve optional interleaved sector sequences.
- PDOS FMF allows up to five open files to exist simultaneously.
- PDOS FMF manipulates diskette files by name yet requires no more software support in your host microcomputer than does a paper tape reader or cassette tape recorder storage system.
- PDOS FMF provides Write Error Detection and Auto Retry on soft errors.
- PDOS FMF provides operator error and disk drive diagnostic messages.
- Operates on standard microcomptuer voltage levels (+5V, +12V & −12V).

CONTROLLER FUNCTIONAL DESCRIPTION

The 1070 Controller is in actuality an 8080 general purpose microcomputer configurated with a Western Digital single chip floppy disk controller port, a controller to host computer parallel memory port, an optional controller to host computer RS232 USART serial memory port, 1K byte static RAM data buffer storage memory, and a 4K byte ROM resident PDOS File Management Firmware memory with all addressing via the memory mapped technique. The host computer port interface includes both a memory data port and command-status port. The command-status port provides the means by which the controller CPU can talk to the host computer and vice versa without conflict because of the coordinated handshake technique utilized. For reference see Figure 2, Model 1070 block diagram and Figure 3, Model 1070 logical diagram.

Host Computer Interface

The 1070 Intelligent Disk Controller provides two methods of interfacing a floppy disk memory system to the microcomputer of your choice. An 8-bit parallel data port interface is standard while an RS232 serial interface is available as an added option. The RS232 interface is used to interface the host computer in those cases where the computer MIB bus structure does not provide a natural logical interface.

One of the major features of the 1070 Controller with its own resident DOS firmware is that it allows the Controller to be interfaced to any mini- or microcomputer such as the 8080, Z-80, 6800, 6502, and others with a minimal effort requiring in some cases only a simple host computer I/O driver software routine to be up and running.

Parallel Interface — The parallel interface is a two port (data and command-status ports) programmed I/O or memory mapped addressed full handshaking co-ordinated bidirectional data transfer computer interface. The command-status port with ASCII communication protocol control characters (**EOT, ACK, NAK, SOH & ENO**) co-ordinates data transfers in both directions and provides means of uniquely distinguishing communications control characters from data characters or bytes. The selection between the data port and command-status port is accomplished by the LSB (A_0) address line of the host computer. (A_0 selects the command-status port while A_0 selects the data port). Handshake coordination between the 1070 controller and host computer is accomplished through the command-status port by the four controller status bits shown in the following Figure 4,

STATUS BIT#	STATUS BIT INFORMATION	STATUS BIT CONTROL
0	Computer output control character transferred to controller input register U2	Controller sets up status bits
1	Computer output data character (or byte) transferred to controller input register U2	Computer resets status bits on reading contents of controller output register U1
6	Controller data character in controller output register U1 and ready for transfer to computer	Computer sets up status bits
7	Controller control character in controller ouput register U1 and ready for transfer to computer	Controller resets status bits on reading contents of controller input register U2

Figure 4. Controller Handshaking Status Bits

while Figure 5 defines the handshake coordination between controller and computer for those individuals unfamiliar with handshaking techniques.

The 4-bit status co-ordinated handshake technique provides variable or fixed length, blocked or unblocked random or sequential ASCII text, ASCII HEX object coded or executable binary object coded data to be transferred to and from the floppy diskette storage media.

The parallel port interface includes the following interface signals:

- **8 BIT DATA BUS** (Bus 0-Bus 7)
- **12 BIT ADDRESS BUS** (AD4-AD15)
- **COMMAND/DATA** (Controlled by Address Line A_0)
- **EXTERNAL SELECT**
- **READ**
- **WRITE**
- **RESET COMPLETE**
- **RESET CONTROLLER** (RESET IN)
- **GROUND,** +5 Volts, = 12 Volts & −12 Volts

Typical Parallel Interface — In order to integrate the PerSci controller into your system, using the parallel memory port interface, normally will require three logical IC's and a computer MIB bus to control bus connector adapter. An example of the simplicity of this logical-physical adapter interface is shown in Figure 6.

RS232 Serial Interface Option — An RS232 USART serial interface is provided as an option for interfacing directly to host computers that have a built-in RS232 serial interface. The latest version of PDOS FMF allows both the parallel and serial interface ports to be physically connected but only allows one of the two ports to communicate with the controller at any given time on a first come first served basis. The RS232 serial interface option also includes an on-board USART baud rate selection via a twelve-position DIP rotary switch. The twelve baud rates are as follows:

BAUD RATE	SWITCH SETTING
50	0
75	1
110	2

134.5	3
150	4
300	5
600	6
1200	7
1800	8
2006	9
2400	A
3600	B
4800	C
7200	D
9600	E
19,200	F

The optional RS232 serial interface provides no means either for co-ordinating data transfers through handshaking or for distinguishing between communications control and data characters. Thus, the user must take care not to transmit data to the controller faster than the controller can write it on diskette which depends on the type of operation, type of drive, sector interleave, etc. Furthermore, the user must ensure that the significant communication control character (**SOH, ACK, NAK, ENQ & EOT**) are not embedded in data sent to or from the controller. If binary data file information is to be read or written, the user must provide a suitable escape convention. The RS232 serial interface includes the following interface signals:

- **TxD TRANSMIT DATA**
- **RxD RECEIVE DATA**
- **DTR DATA TERMINAL READY**
- **DSR DATA SET READY**
- **RTS REQUEST TO SEND**

CONTROLLER — COMPUTER COMMUNICATION EXCHANGE	**HANDSHAKING SEQUENCE**
COMPUTER OUTPUTS COMMAND TO CONTROLLER	1. Computer writes output control character status into the controller's control character input status register U10B (status bit #0) true. 2. Computer writes output control character into the controller's input data, register U2 and sets the controller's data input status register U10A (status bit #1) true. 3. Controller reads input control character and resets the controller's status registers U10A (status bit #1) & U11B (status bit #0) to zero.
COMPUTER OUTPUTS DATA TO CONTROLLER	1. Computer writes output data into the controller's data input register U2 and sets the controller's data input status register U10A (bit #1) true. 2. Controller reads input data from register U2 and resets its status registers U10A (status bit #1) and U110B (status bit #0) to zero.
CONTROLLER OUTPUTS COMMAND TO COMPUTER	1. Controller writes output control character into its output register U1 and sets its control character output status register U9A (status bit #7) true. 2. Computer reads the controller's status bits 6 or 7 as data ready status bits and if either one is true, reads in data from the controller's output register U1, resets controller's status bits #6 & 7 to zero, examines controller's control status bit #7 and sets CPU carry flag to 1 if the control bit 7 is true, otherwise input word is a data word.
CONTROLLER OUTPUTS DATA TO COMPUTER	1. Controller writes output data into the controller's output data register U1 and sets the controller's data output status register U9B (status bit #6) true. 2. Computer reads the controller's status bits 6 or 7 as data ready status bits and if either one is true, reads in data from the controller's output register U1, resets controller's status bits #6 & 7 to zero, examines controller's control status bit #7 and sets CPU carry flag to 1 if the control bit #7 is true, otherwise input word is a data word.
COMPUTER READS CONTROLLER OUTPUT STATUS	1. Computer reads the controller's output status register U9A (status bit #6) and U9B (status bit #7).
COMPUTER READS CONTROLLER OUTPUT COMMAND OR OUTPUT DATA	1. Computer reads the controller's status bits 6 or 7 as data ready status bits and if either one is true, reads in data from the controller's output register U1, resets controller's status bits #6 & 7 to zero, examines controller's control status bit #7 and sets CPU carry flag to 1 if the control bit #7 is true, otherwise input word is a data word.

Figure 5. Handshaking Coordination Exchange

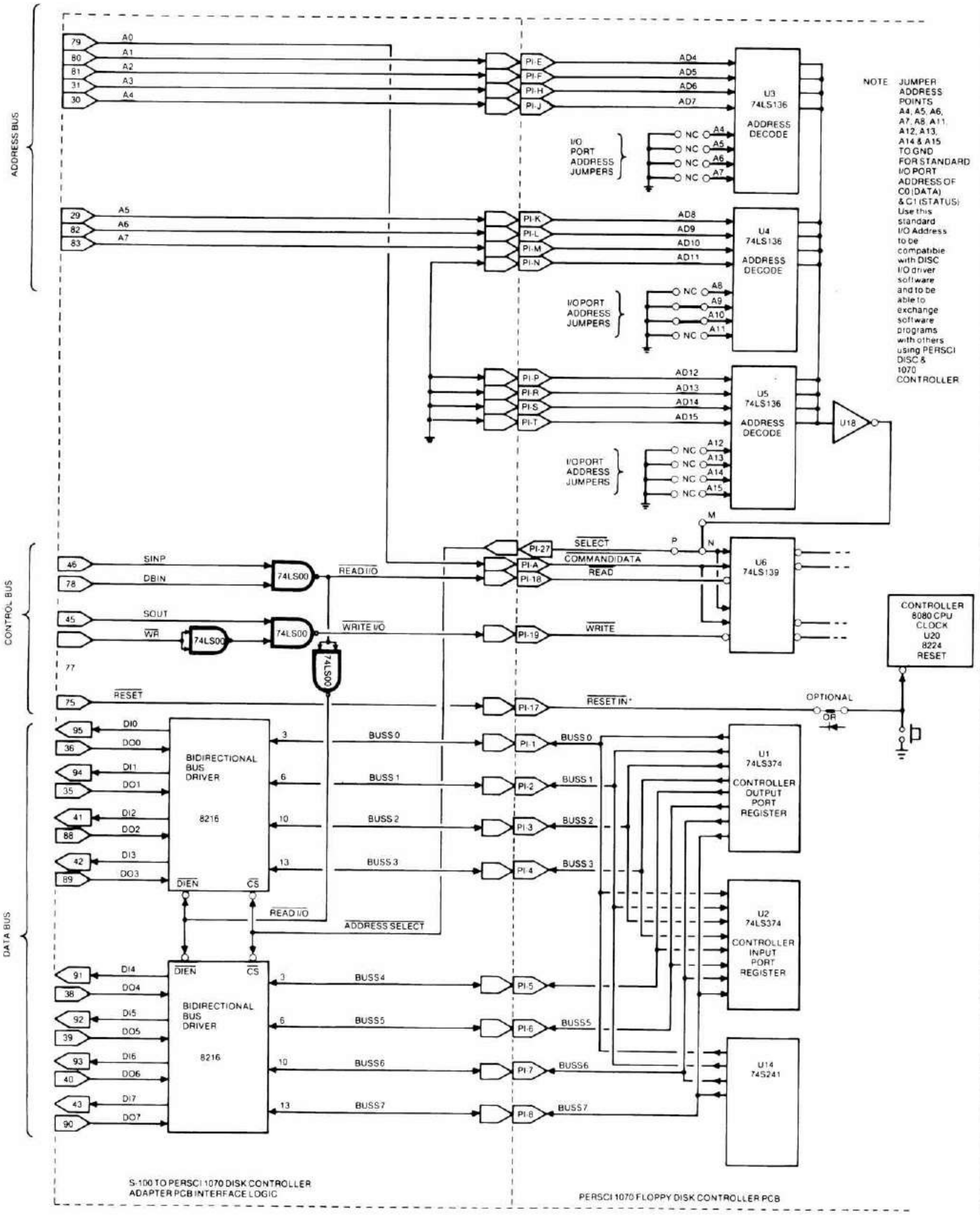

ADDRESS BUS
CONTROL BUS
DATA BUS
A0
A1
A2
A3
A4
A5
A6
A7
AD4
AD5
AD6
AD7
AD8
AD9
AD10
AD11
AD12
AD13
AD14
AD15
U3 74LS136 ADDRESS DECODE
U4 74LS136 ADDRESS DECODE
U5 74LS136 ADDRESS DECODE
I/O PORT ADDRESS JUMPERS
U18
NOTE JUMPER ADDRESS POINTS A4, A5, A6, A7, A8, A11, A12, A13, A14 & A15 TO GND FOR STANDARD I/O PORT ADDRESS OF C0 (DATA) & C1 (STATUS) Use this standard I/O Address to be compatible with DISC I/O driver software and to be able to exchange software programs with others using PERSCI DISC & 1070 CONTROLLER
SINP
DBIN
SOUT
WR
74LS00
READ I/O
WRITE I/O
SELECT
COMMAND/DATA
READ
WRITE
U6 74LS139
CONTROLLER 8080 CPU CLOCK U20 8224 RESET
RESET
RESET IN*
OPTIONAL
OR
BIDIRECTIONAL BUS DRIVER 8216
DIEN
CS
BUSS 0
BUSS 1
BUSS 2
BUSS 3
BUSS 4
BUSS 5
BUSS 6
BUSS 7
ADDRESS SELECT
U1 74LS374 CONTROLLER OUTPUT PORT REGISTER
U2 74LS374 CONTROLLER INPUT PORT REGISTER
U14 74S241
S-100 TO PERSCI 1070 DISK CONTROLLER ADAPTER PCB INTERFACE LOGIC
PERSCI 1070 FLOPPY DISK CONTROLLER PCB

Figure 6. 8080S-100 Bus Controller Interface

- **CTS CLEAR TO SEND**
- **GND GND**

All RS232 interface signals are RS232 voltage levels.

Disk Drive Interface

The floppy diskette drive interface provides the capabilities of controlling four Model 70 single drives or two Model 277 dual drives. The disk drive interface includes the following interface signals:

- **DRIVE SELECT 2-RIGHT** (Drive #3)
- **SEEK COMPLETE**
- **RESTORE**
- **REMOTE EJECT**
- **DRIVE SELECT 2-LEFT** (Drive #2)
- **INDEX**
- **READY**
- **DRIVE SELECT 1-LEFT** (Drive #0)
- **DRIVE SELECT 1-RIGHT** (Drive #1)
- **DIRECTION**
- **STEP**
- **WRITE DATA**
- **WRITE GATE**
- **TRACK 00**
- **WRITE PROTECT**
- **SEPARATE DATA**
- **SEPARATE CLOCK**

CONTROLLER PHYSICAL CONFIGURATION

The Intelligent Controller is mechanized on a single 4.5" x 7" x 0.62" printed circuit board with edge connectors along the 4.5" dimension. Host computer connector interface is via a 72-pin PCB edge connector with .100" conductor center to center spacing while the disk drive connecter interface is via a 50-pin PCB edge connector with 0.100" conductor center to center spacing.

HOST COMPUTER I/O DRIVER PROGRAM

A companion article in this issue provides an 8080/Z-80 PerSci controller I/O driver program while next month's issue of INTERFACE AGE will include a 6800 I/O driver program. Both of these programs provide complete source code, flow diagrams and descriptive documentation to get your diskette system up and running almost without any programming on your part.

PDOS FILE MANAGEMENT FIRMWARE

The 1070 Intelligent Controller software consists of a 4K ROM resident PerSci Disk Operating System (PDOS) File Management Firmware (FMF). The PDOS FMF program resides in ROM starting at memory address 0000H through and including 2FFFH. PDOS includes all functions normally associated with a DOS except for the host computer I/O driver linkage routine which must reside in the host computer's main memory.

Diskette Data Storage Format

The diskette initialization function of the 1070 controller creates a diskette volume that contains 77 tracks with 26 sectors/track and 128 data bytes per sector. The first track is reserved by the controller for use as an index (i.e. a table of contents) for the diskette volume, while the remaining 76 tracks are available for data storage. Tracks are numbered from 00 to 76 (outer to inner) and sectors are numbered from 01 to 26 on each track. Each sector has a header which defines the track and sector number for the soft sectored diskette. Both the sector header and the data itself is written with a 16-bit cyclic redundancy check (CRC) word. Formatted data capacity of each track is 3328 bytes while the capacity of each diskette is 252,928 data bytes, excluding index track. The index track accommodates up to 100 file index references, each consisting of the following index information.

- **NAME** (up to eight alphanumeric characters)
- **VERSION** (up to three alphanumeric characters)
- **TYPE** (single alphanumeric character)
- **START OF FILE ALLOCATION** (4 digit decimal number)
- **LENGTH OF FILE ALLOCATION** (4 digit decimal number)
- **POSITION OF THE FILE EOF MARK** (4 digit decimal number)
- **DATE OF FILE CREATION** (Six digit decimal number)
- **DATA OF LAST UPDATE** (Six digit decimal number)

ACCESS METHOD	DELIMITER	SMALLEST ELEMENT OF DATA TRANSFERRED TO OR FROM MEMORY SYSTEM	PDOS STORAGE & RETRIEVAL COMMANDS
DIRECT	SECTOR #	SECTOR	INPUT OUTPUT
RELATIVE	# OF BYTES	BYTE	READ WRITE
PUNCTUATED	ASCII RS CONTROL CHARACTER	RECORD	READ WRITE
STREAM	FILE NAME	FILE	LOAD SAVE

Note:
Fixed length is defined as the specified length or number and is part of the PDOS command, whereas the length of a variable-length record or file is determined by delimiters, such as RS control characters or file name and EOF control characters.

File Allocation

When a new file is created on a diskette volume, it receives an allocation of contiguous sectors. The minimum file allocation is one sector, and the maximum is 1976 sectors (i.e. 76 tracks of 26 sectors, or 252,928 bytes). The first file created on a newly initialized diskette receives an allocation starting immediately above the index track. Subsequently created files receive an allocation starting immediately above the allocation of the previously created file. The allocation of each file is recorded in the file header information recorded on the index track.

Whenever a file is deleted, its block of contiguous sectors is de-allocated. This leaves a gap in the sequence of allocated sectors on the diskette volume. PDOS provides a GAP command to compress the allocations on a diskette volume, eliminating the gaps caused by previous file deletions.

File Access Methods

The controller PDOS ROM resident File Management Firmware provides the following four methods for accessing and updating files stored on a diskette:

- **STREAM ACCESS**
- **PUNCTUATED ACCESS**
- **RELATIVE ACCESS**
- **DIRECT ACCESS**

Stream Access — The stream access method permits an entire ASCII text, ASCII HEX object program or binary program file to be stored or retrieved as a continuous stream of data bytes (as if the diskette file were a very high speed paper tape). Stream access is ideally suited for the storage and retrieval of executable binary programs, ready to run BASIC application programs and completed ASCII text word processing files. Stream access is performed on opened or closed files using the **LOAD** and **SAVE** controller commands.

Punctuated Access — The punctuated access method treats a file as a sequence of variable-length records separated by the non-printing ASCII control character **RS**. Prior to file access a file reference pointer is positioned via the **POSITION** command to the desired record position.

Variable-length punctuated file records may then be read or written in continuous sequence, or one at a time by using the **READ** or **WRITE** commands if the file is open. Reocrds may span sector boundaries of a diskette (sector boundaries are made transparent by the controller). Punctuated access is appropriate for the storage and retrieval of ASCII text word processing files, BASIC application programs, assembler source programs and HEX object programs which are to be accessed sequentially, one record or line at a time and are in ASCII text or ASCII HEX format. Because of its dependency upon the unique ASCII RS punctuation control character to separate records, punctuated access can not be used to store and retrieve binary data.

Relative Access — The relative access method treats a file as a byte-addressable random access memory. Prior to file access, a file reference pointer is positioned to any desired byte position. Any fixed length number of bytes may then be stored or retrieved by using the **READ** and **WRITE** commands if the file is open. Relative **READ** and **WRITE** operations may span sector boundaries but this is made transparent by the controller. Relative access is ideal for data base oriented applications in which random byte access is required.

Direct Access — The direct access method permits any specified single sector of any specified track of a diskette volume to be read or written directly by using the **INPUT** and **OUTPUT** commands and thereby bypassing the file management functions of the controller altogether. Files need not be open prior to an **INPUT** or **OUTPUT** operation.

The following Table 1 summarizes the characteristics of the four access methods:

File Reference

A file reference identifies a particular file or group of files. File references may be either unique or ambiguous. A unique file reference identifies one file uniquely, while an ambiguous file reference may be satisfied by several different files. File references consist of four identifier fields:

REQUIRED FILE STATUS TO STORE OR RETRIEVE FILE DATA	FILE LENGTH CLASSIFICATION	FILE ACCESS		CLASS OF FILE HANDLED VERSUS COMPUTER I/O PORT INTERFACE			
		RANDOM	SEQUENTIAL	PARALLEL		SERIAL	
				BINARY	ASCII	BINARY	ASCII
OPEN OR CLOSED	FIXED	X	X	X	X		X
OPEN	FIXED	X	X	X	X		X
OPEN	VARIABLE		X		X		X
OPEN OR CLOSED	VARIABLE		X	X	X		X

Table 1. Access Methods Characteristics Summary

- **N** Name of up to eight alphanumeric characters
- **V** Version of up to three alphanumeric characters
- **T** Type specified by a single alphanumeric character
- **D** Drive which is a numeric digit between 0 and 3.

The version, type and drive identifier fields are optional and are set off from the name by means of the unique leading punctuation characters (period, colon, and slash) as shown in the following

NNNNNNNN.VVV:T/D

The period punctuation character denotes start of the version identifier field, colon denotes start of type identifier field, while slash denotes start of the drive identifier field.

The following are examples of the valid file references:

MONITOR	FILE NAMED MONITOR
MONITOR.SRC	FILE NAMED MONITOR, VERSION SRC
MONITOR.OBJ:A	FILE NAMED MONITOR VERSION OBJ, TYPE A
MASTER/2	FILE NAMED MASTER, DRIVE 2
MASTER:$	FILE NAMED MASTER, TYPE $
MASTER.ONE	FILE NAMES MASTER VERSION ONE
STARTREK.BAS/1	FILE NAMED STARTREK VERSION BAS DRIVE 1
STARTREK.X0T	FILE NAMED STARTREK, VERSION X0T
STARTREK:0/3	FILE NAMED STARTREK, TYPE 0, DRIVE 3

The special characters "**?**" and "*****" may be used a file reference ambiguous so that it may match a number of different file. The "?" is used as a "wild-card" character which matches any character in the corresponding position in a file reference. Thus the ambiguous file reference:

PER ????.BA?

matches all of the following unambiguous file references:

PERFECT.BAL
PERSCI.BAS
PERQ.BAX

The character "*****" is used to denote that all characters positions to the right are wild-cards. The following examples illustrate the flexibility which this facility provides:

MONITOR.*	=	**MONITOR.???:?**	Matches all files with name "MONITOR"
***.BAS**	=	**????????.BAS:?**	Matches all files with version "BAS"
Z*	=	**Z???????.???:?**	Matches all files starting with "Z"
*****	=	**????????.???:?**	Matches all files on the diskette

Controller Commands

PDOS FMF commands consist of a single alpha character command identifier followed by one or more command parameters. Parameters must not contain embedded spaces, must be set off from one another by spaces and may optionally be set off from the common letter by spaces. All PDOS commands are initiated by a **CR LF EOT** following the command. PDOS mode of operation is directed by one of 17 commands as summarized in the following:

COMMAND	NAME	COMMAND FUNCTION
• A	ALLOCATE	Allocates an empty file & assigns name
• C	COPY	Copy file or diskette
• D	DELETE	Deletes file or diskette
• E	EJECT	Ejects diskette
• F	FILE	Opens & closes file
• G	GAP	Eliminate deleted files & compress diskette storage
• I	INPUT	Read single sector
• K	KILL	Deletes all diskette files
• L	LOAD	Read entire file
• M	MODE	Set default diskette and date
• N	NAME	Rename file
• O	OUTPUT	Writes single sector
• P	POSITION	Sets open file reference pointer position
• Q	QUERY	List diskette index track
• R	READ	Reads specified number of bytes from open file
• S	SAVE	Creates new file and saves input stream data in it
• T	TEST	Executes one of three resident disk drive diagnostic tests

PDOS FMF Command Description

Each of the 17 PDOS FMF commands are described in detail in the following:

COMMAND	COMMAND NAME	FILE ACCESS METHOD	
A	ALLOCATE	—	Allocates an empty file of N sectors long and assigns a file name. File names may consist of up to eight alphanumeric characters. File references may be either unique or ambiguous.
C	COPY	—	Copies file, collection of files, or complete diskette volume to same or different diskette volume. Requires that all files be closed.
D	DELETE	—	Deletes specified file, collection of files, or complete diskette volume.
E	EJECT	—	Ejects diskette in specified drive (effective only if the diskette drive is equipped with remote eject option).
F	FILE	—	Opens and closes diskette files in drive unit 0-3. A file must be open before punctuated or relative access is permitted by the controller. All open files are equated to logical unit numbers between 1 and 5 to reduce number of alphanumeric characters required to define a file name. A maximum of five files may be open simultaneously.
G	GAP	—	Reallocates and compresses diskette storage to eliminate prior file deletions. Requires that all files be closed.
I	INPUT	DIRECT	Reads single sector of specified tract, sector and drive.
K	KILL	—	Deletes all files on specified diskette without initializing diskette or deletes all files and initializes diskette to any one of 13 sector interleave sequences.
L	LOAD	STREAM	Read entire diskette file.
M	MODE	—	Loads current date and sets defule diskette driver number (0-3) for all subsequent file references and commands which do not include an explicitly specified drive.
N	NAME	—	Change specified file name to new name.

O	OUTPUT	DIRECT	Writes a single sector into an open file of specified drive.
Q	QUERY	—	Reads index track header information from one, group, or all files on specified diskette volume. This header information includes the following file record data: •File name version & type •Start of allocation (decimal track and sector starting address). •Length of allocation (decimal number of sectors). •Position of the end-of-data mark (decimal sector and byte offset). •Date of creation. •Date of last update.
R	READ	RELATIVE OR PUNCTUATED	Reads specified number of bytes from open file of specified drive (fixed-length or variable-length records). Variable-length records delimited by a record separator character **RS**.
S	SAVE	STREAM	Creates a new variable-length file record in specified diskette. File sector allocation size is determined by length of the data stream record.
T	TEST	—	Executes one of three resident diagnostic tests on specified drive. The three diagnostic tests are random seek-verify test, random seek-read test, and incremental seek-read test.
W	WRITE	RELATIVE OR PUNCTUATED	Writes a fixed-length or variable-length record into an open file on specified or default selected drive.

Protocol Software Handshaking Interface

The PDOS Command protocol interface between the host computer and the controller is accomplished by utilizing the standard ASCII communications control characters **SOH, ACK, NAK, ENQ** and **EOT** as controller-host computer communication commands. These protocol communication commands are used under a co-ordinated handshaking technique to control all data communication between the controller and host computer via the controller command-status port. Normally the computer issues a PDOS comand followed by a communication command sequence (**CR LF EDT**), whereupon the controller transmits data or answers back by issuing a single or sequence of communication commands, data or error message if appropriate followed by the terminating communication command sequences **ACK EOT.** This co-ordinated handshake protocol interface command sequence is shown in the following Figure 7, for each PDOS command.

Error Diagnostic Messages

The controller issues fatal and non-fatal error diagnostic messages. Fatal error diagnostic messages transmitted by the controller are always proceeded by a NAK control character and followed by an EOT control character. These error diagnostic messages indicate the premature and unsuccessful termination of a controller command.

Fatal Errors — Fatal error diagnostic messages are as follows:

• **COMMAND ERROR**	Indicates an invalid command or command parameter.
• **DUPLICATE FILE ERROR**	Indicates an attempt to create a new file with the same name as an existing file on the same diskette.
• **NOT FOUND ERROR**	Indicates that a file with the specified name could not be found in the index of the specified diskette.
• **OUT OF SPACE ERROR**	Indicates an attempt was made to exceed the capacity of a diskette or exceeded the index track capacity.
• **READY ERROR**	Indicates an attempt was made to access a diskette drive which is not ready.
• **UNIT ERROR**	Indicates an attempt was made to read, write, or position a logical unit number that is not equated with an open file.
• **HARD DISK ERROR**	Indicates a seek, read, or write error which could not be successfully resolved in five retries.

Note that each fatal error message begins with a unique alpha character, so that an interfacing program need only to analyze the first character following a NAK control character to determine the type of fatal error.

Non-Fatal Errors — Non-Fatal error diagnostic messages are issued for soft disc errors. They are not preceded by a NAK control character, and they contain the following information:

- **TYPE OF DISK OPERATION** (seek, read or write)
- **ERROR RETRY NUMBER** (1 to 5)
- **DISKETTE LOCATION AT WHICH ERROR OCCURRED** (decimal track and sector)
- **TYPE OF ERROR** (protect, write fault, verify, CRC, or lost data)

During the transmission of diskette data (**LOAD, SAVE, READ, WRITE, INPUT** and **OUTPUT** commands), non-fatal error messages are suppressed.

PDOS COMMAND	COORDINATED HANDSHAKE PROTOCOL SEQUENCE
ALLOCATE	COMPUTER TRANSMITS: **PDOS COMMAND CR LF EOT**
EJECT	
FILE	CONTROLLER TRANSMITS: **ACK EOT**
KILL	
MODE NAME	
TEST	
COPY	
DELETE	COMPUTER TRANSMITS: **PDOS COMMAND CR LF EOT**
GAP	
POSITION	CONTROLLER TRANSMITS: **INFORMATION-DATA CR LF ACK EOT**
QUERY	
INPUT	
LOAD	COMPUTER TRANSMITS: **PDOS COMMAND CR LF EOT**
READ	CONTROLLER TRANSMITS: **SOH DISK-DATA ACK EOT**
OUTPUT	COMPUTER TRANSMITS: **PDOS COMMAND CR LF EOT**
SAVE	
WRITE	CONTROLLER TRANSMITS: **ENQ EOT**
	COMPUTER TRANSMITS: **DISK-DATA EOT**
	CONTROLLER TRANSMITS: **ACK EOT**
Note: Controller may terminate command at any time with fatal error diagnostic message by using the following protocol	CONTROLLER TRANSMITS: **NAK** FATAL-ERROR MESSAGE **CR LF EOT** Note: No **ACK** will be transmitted by the controller in this case.

The new generation of Diskette Drives is here and under control.

PerSci has it–a family of diskette drives "design-years" ahead of competitive drives–now available in complete low cost subsystems for interface to 8080, 6800 and other major microprocessors.

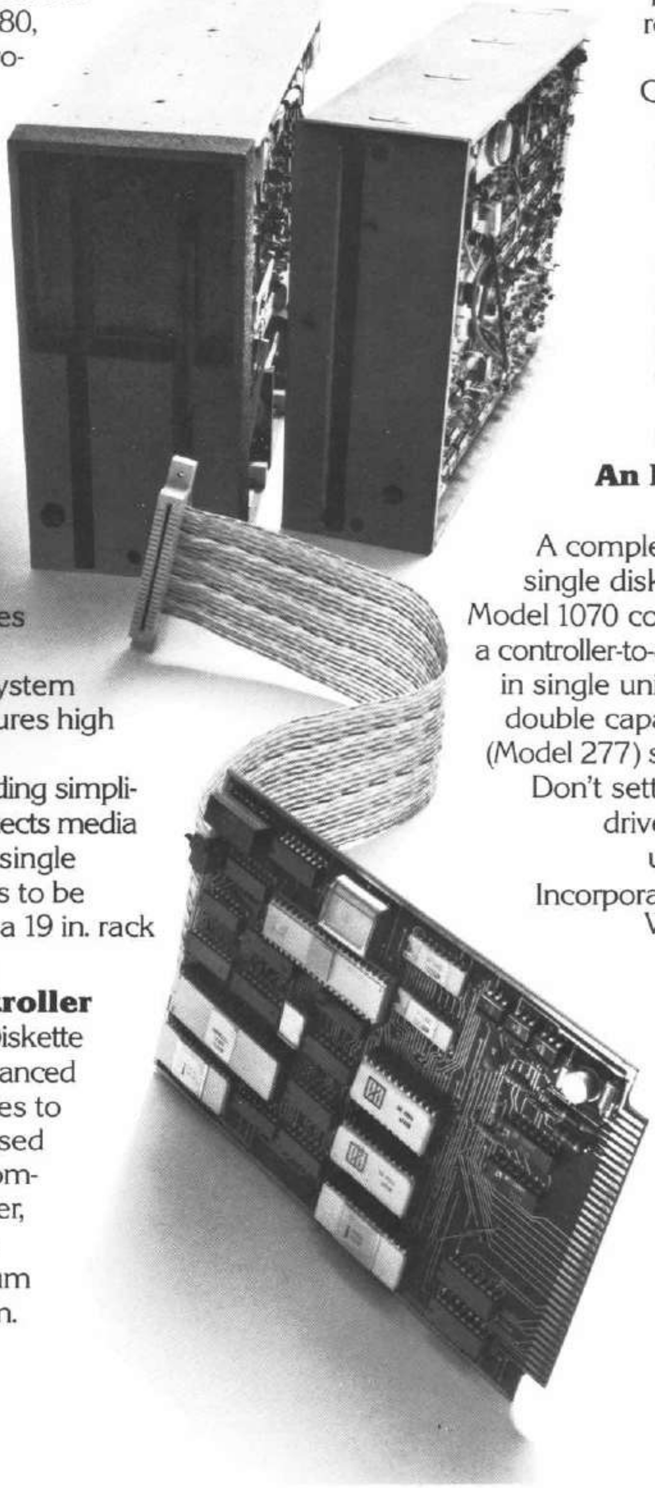
The Highest Performance Diskette Drives:

PerSci diskette drives, both single and dual head units, offer a combination of performance features unique in the marketplace while still maintaining compatibility in existing systems:

- ☐ Voice coil positioning for access speeds seven times faster than competitive drives (76 tracks in 100 ms)
- ☐ A low power all DC system reduces cost and assures high reliability
- ☐ Automatic electric loading simplifies operation and protects media
- ☐ Small size permits 5 single drives or 4 dual drives to be mounted vertically in a 19 in. rack

The Most Powerful Diskette Drive Controller

The PerSci Model 1070 Diskette Drive Controller puts the advanced performance of PerSci drives to work in microprocessor based systems. An IBM format compatible, "intelligent" controller, the Model 1070 will handle from 1-4 drives with minimum demand on the host system.



In fact, with addition of a power supply and keyboard to the PerSci subsystem, the user can perform many floppy disk routines without additional hardware or software.

Controller features include:

- ☐ 8080 microprocessor
- ☐ Formatter/control IC
- ☐ 4K disk operating system
- ☐ 1K data buffer
- ☐ 8080 or 6800 microprocessor interfaces
- ☐ RS-232 serial interface optional
- ☐ IBM data format

An Economical Diskette Drive Subsystem

A complete subsystem including a single diskette drive (Model 70), the Model 1070 controller with interface and a controller-to-disk-drive cable is available in single units or OEM quantities. For double capacity, a dual diskette drive (Model 277) subsystem is also available.

Don't settle for yesterday's diskette drive. Get the new generation under control from PerSci, Incorporated, 12210 Nebraska Ave. W. Los Angeles, CA 90025 (213) 820-3764

PERSCI

Peripherals a Generation Ahead.