Build a Super Simple Floppy-Disk Interface, Part 1

James Nicholson and Roger Camp
1046 Gaskill
Ames IA 50010

For personal-computer users, a floppy-disk system represents the ultimate in mass storage because of its speed and capacity. The floppy-disk controller described in this article provides all the capabilities found in commercial systems, yet it is simple and economical because it requires only ten integrated circuits. Fundamental software will be provided (in the second part of this article) to control and perform data transfers, and discussion of file structuring and alternate hardware will give the experimenter ideas for improvements.

This system uses the FD400, an 8-inch floppy-disk drive manufactured by the Pertec Computer Corporation, and the popular Western Digital 1771 floppy-disk controller integrated circuit (which allows such special features as variable block size, soft sectoring, IBM compatibility, and much more). Although the specifics shown are for microcomputers based on the MOS Technology 6502 microprocessor, the controller could be adapted to other microprocessors with some care at a few crucial points. The 6502 offers some speed advantages and a programming ease not afforded by the others.

Fundamentals

The data recorded on floppy disks is logically arranged in concentric rings called *tracks*, with each track composed of blocks of data called *sectors*. The computer must be able to tell where a sector begins, and there are two ways of doing this. Each sector can be distinguished by its position relative to holes punched in the disk (this is called *hard sectoring*), or it can be distinguished by special sequences of information recorded on the disk (*soft sectoring*). In either case, the disk has one hole that is used as an index to signal the start of the first sector on all tracks.

This controller is simple and economical because it requires only ten integrated circuits.

The most common 8-inch floppy-disk format provides for 77 tracks of 26 sectors each, with 128 bytes recorded in each sector. The address of each sector, in the form of a track number (0 through 76) and a sector number (1 through 26), is recorded on the disk at the start of the sector itself.

The disk drive has two motors: one that spins the disk at 360 rpm (revolutions per minute), and one that moves the head from track to track on command. Each drive also has a printed-circuit board to control both motors. The inputs and outputs of this circuit board (see figure 1) follow a standard set by Shugart Associates, manufacturer of one of the first popular floppy-disk drives.

A single pulse on either the STEP-IN line or the STEP-OUT line moves the head one track toward the center of the disk (track 76) or toward the

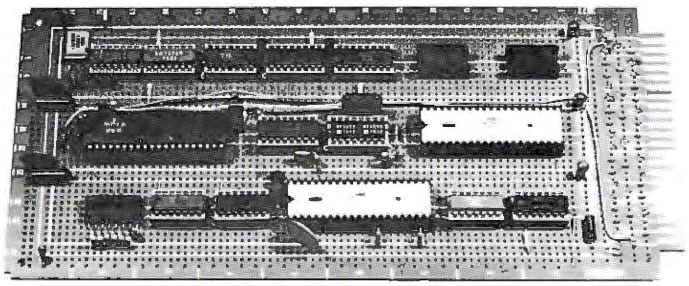


Photo 1: *The authors' wire-wrapped floppy-disk controller board.*

About the Authors

Roger Camp is a Professor of Electrical Engineering at Iowa State University. He is the author of several technical papers and patents, and his most recent book is Micro-Processor Systems Engineering.

James Nicholson, currently Project Manager, Business Recovery Planning, has been involved in large data-center activities for Donnelley Marketing. He has designed and built several microcomputer systems in the last five years.

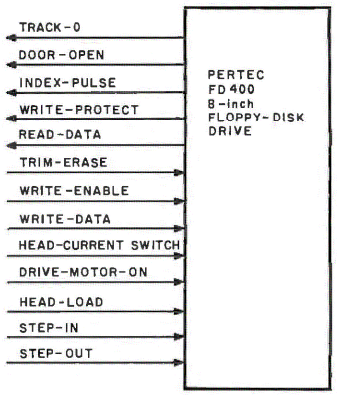


Figure 1: *Input and output lines available for controlling a Pertec FD400 8-inch floppy-disk drive. These signals are the same as those found on any Shugart-compatible drive, so nearly any drive may be substituted for the FD400.*

outside (track 0), respectively. When the head is positioned over track 0, the outermost track, the TRACK-0 output is activated. To turn on the spindle motor, the DRIVE-ON input must be activated, and the disk door in the front of the drive must be closed (this deactivates the DOOR-OPEN output line). As the disk rotates, a photoelectric sensor in the drive detects the index hole in the disk; this generates the INDEX signal that allows the system to begin counting sectors at the first one.

To read data, the HEAD-LOAD line is activated to force the head to contact the rotating disk surface. A mixture of data and clock bits are then detected and amplified by the drive's electronics; these appear as logic levels on the DATA-READ output at the rate of 250 K-bits per second.

To write data on the disk, the head must be loaded, the WRITE-ENABLE line must be activated, and the data must be sent to the drive on the WRITE-DATA line. (This must occur with very specific timing.) If the WRITE-PROTECT output has been activated, the drive has detected the presence of a write-protect notch in the disk's envelope.

Obviously, communication at this level between a disk drive and a microcomputer is possible but not desirable. The microcomputer would spend much of its time catering to the needs of the disk rather than computing. The purpose of the FD1771 (actually a microprocessor in its own right) is to act as a high-level communications interface between the two.

When instructed to *seek* (move the head) to track 30, the 1771 will generate the appropriate number of STEP-IN or STEP-OUT pulses to move the head from its current position, wherever it may be, to track 30. Another example of the 1771's capabilities is the process of reading a specific sector: the 1771 will search a given track for the proper sector address; when located, the data following the address is transferred to the microprocessor. Simultaneously, the 1771 can maintain synchronization with the disk drive and check for errors. Therefore, using the 1771 floppy-disk controller circuit results in a greatly simplified hardware and software design.

Software must be an integral part of the design of any computer subsystem—a subroutine of about 256 bytes is required to communicate the proper commands to the disk controller. Additional software is required to handle complex data-file structures (this software and various structuring techniques will be discussed in part 2 of this article).

Disk Format

Figure 2 schematically describes the format of recorded data on a soft-sectored disk. The pulse generated by the index hole passing the sensor provides a physical reference point to determine the beginning and the end of a track. The diagram represents 16 256-byte sectors (the authors' choice for format) rather than the usual 26

sectors containing 128 bytes.

The disk rotates once every 166.67 ms, which allows the drive to read 41,665 bits of information; that is, a byte every 32 μs. Each track contains 5208 bytes (divided into data and control bytes), as well as gaps between sectors. (The gaps are required to allow sufficient time to turn write-head current on and off without destroying valid data.)

The IAM (index-address mark) that provides a recorded indication of the beginning of the track has 16 sectors recorded after it. The sectors consist of two records: the ID (identification record) and the DATA (data record). The ID contains information on the track number and the sector number of the DATA that follows. Each of the records begins with an AM (address mark). In addition, each record is ended with a 2-byte CRC (cyclic-redundancy-check) code.

Each byte of data recorded on the disk consists of interleaved clock and data bits. The clock bits convey information used for synchronization and for the identification of AMs. AMs always have clock bits corresponding to hexadecimal C7 (D7 in the case of the IAM); all other bytes of information have clock bits corresponding to hexadecimal FF. In other words, some clock bits are omitted in AMs. This scheme allows the data bits of a data-address mark (hexadecimal FB) to be distinguished from a hexadecimal FB recorded as data.

Figure 2 also illustrates that these data and clock bits are recorded as a single stream. When reading from the disk, the 1771 separates the data and clock bits (although our system uses discrete components to achieve greater reliability).

As a general rule, the larger the sector, the greater the total amount of data that can be recorded on one disk. This is due to the reduced amount of area necessary for gaps and indexing information. Using 16 256-byte sectors, 315,392 bytes of data can be recorded. The usual configuration of 256-byte sectors allows tracks with only 15 sectors; however, it has been found that sufficient space is available to reliably record 16 sectors.

Western Digital's 1771 Floppy-Disk Controller

This device is essentially a microprocessor dedicated to the specific task of controlling disk drives (see figure 3). It has five programmable registers and accepts a number of commands through various combinations of them. For economic reasons, there is a desire to connect multiple drives to a single 1771, but, since the device "remembers" the track the head was last positioned to, switching from one drive to another would place an added burden on the driving software. A case can be made for complete duplication of the controller electronics for each disk drive.

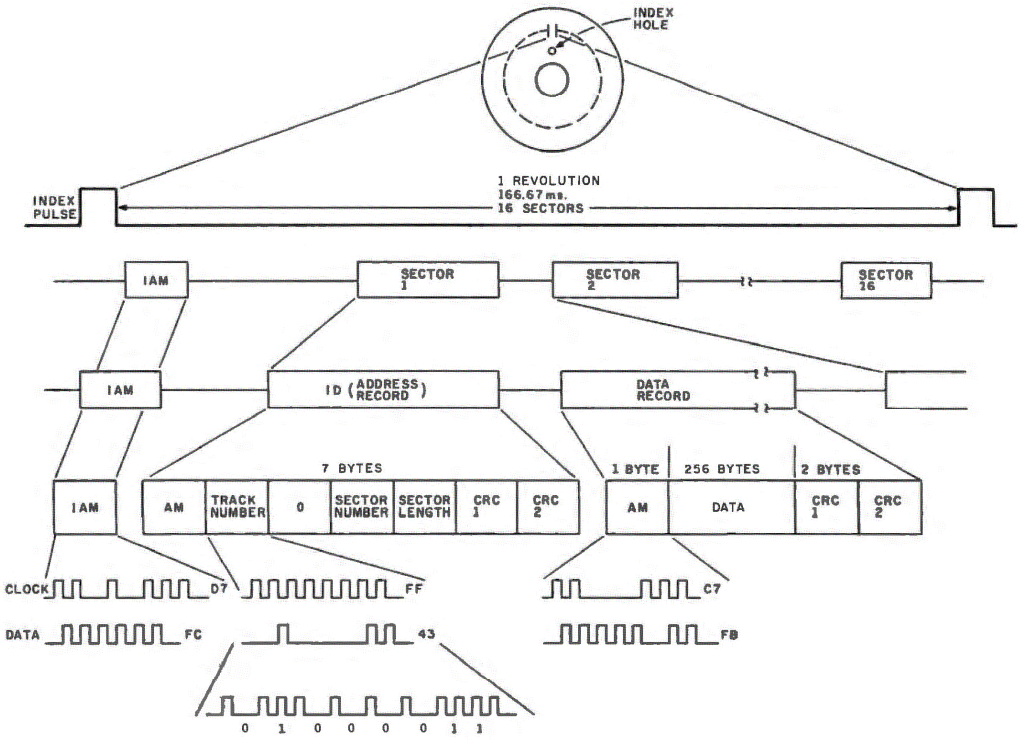


Figure 2: *The format of recorded data on one track of a soft-sectored floppy-disk drive. The IAM (index-address mark) marks the beginning of each track. See the text for details.*

The registers in the 1771 that can be programmed by the user are the data, track, sector, and command registers—there is also a status register that can be read from but not written to. These 8-bit registers form the basis for software control of any disk drive:

- Data register: In disk-reading operations, this register receives 8 bits of data in parallel from the disk via the shift register. The data is held until the computer can accept it, allowing the shift register to be ready for the next byte. During disk-writing operations, 8 bits of data are transferred in parallel from the computer to this register and held until they can be accepted by the shift register for transfer to the disk. When executing the seek command, the data register holds the address of the desired track.
- Track register: This register holds the track number of the current head position. The value is incremented by one for every track the head is stepped in (toward track 76), and decremented by one for every track the head is stepped out (toward track 0). The contents of the register are compared with the track number recorded in the ID field of sectors on the disk.
- Sector register: During read or write operations, the contents of this 8-bit register are compared with the sector number recorded in the ID field of sectors on the disk. The contents should not be changed while the device is busy.
- Command register: This register holds the command currently being executed. The register should not be loaded while the 1771 is busy unless the current command is to be overridden (this action causes an interrupt to be generated). The eleven commands understood by the 1771 are divided into four types, shown in table 1, according to the way their flag bits are defined.
- Status register: Information about the status of the controller can be read from this register. The meaning of the status bits may change depending on the current command.

Registers are accessed by placing the proper logic levels on the A0, A1, $\overline{RE}$, and $\overline{WE}$ lines, as shown in table 2. Other logic levels in the 1771 perform functions to:

- Generate and check the 16-bit CRC code
- Increment, decrement, and compare register values
- Detect ID, data, and index-address marks
- Provide control signals based on an external 2.0 MHz clock

A typical disk operation includes the following steps. First, the soft-

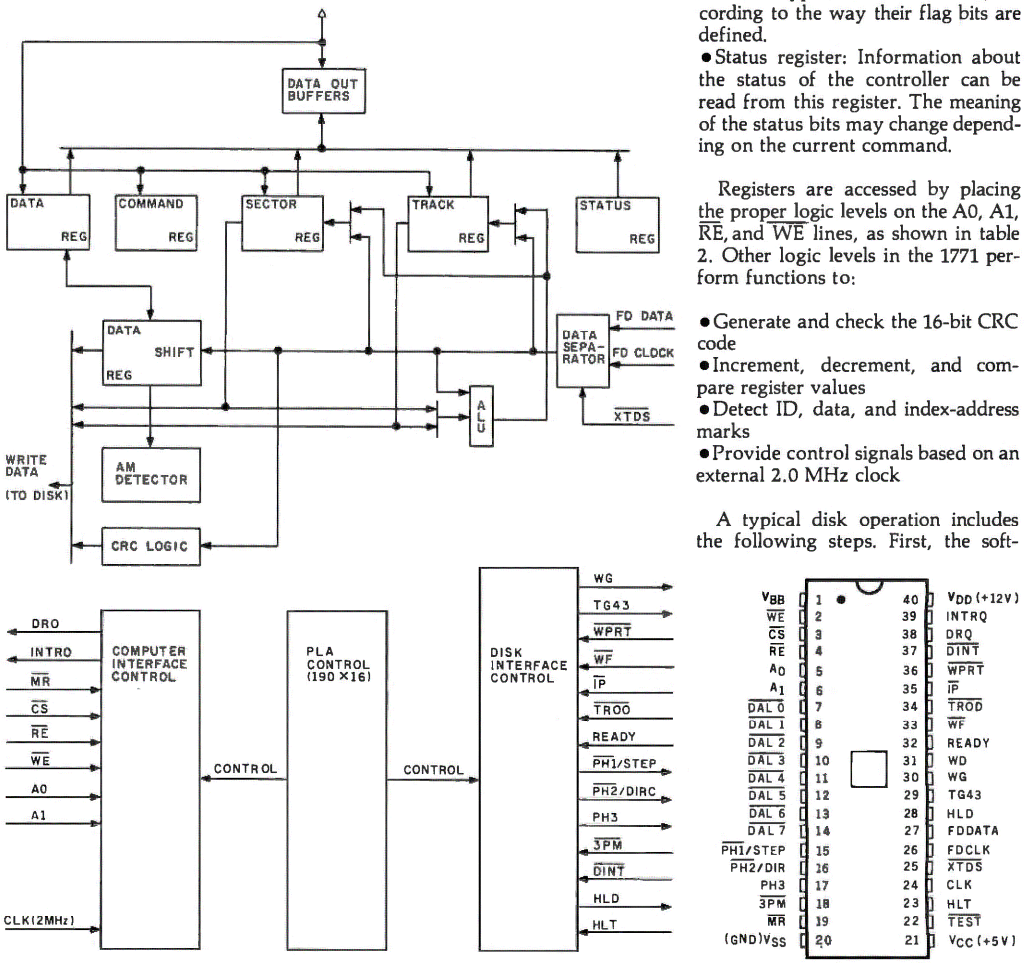


Figure 3: *Internal architecture and pinout diagram of the Western Digital FD1771 floppy-disk controller. The four programmable registers and eleven commands of the 1771 allow any microprocessor to control a disk subsystem using high-level instructions, thus removing a significant burden from the disk-driving software. See table 1 for a summary of the commands.*

TYPE	COMMAND	BITS 7	6	5	4	3	2	1	0
I	Restore	0	0	0	0	h	V	r_1	r_0
I	Seek	0	0	0	1	h	V	r_1	r_0
I	Step	0	0	1	u	h	V	r_1	r_0
I	Step In	0	1	0	u	h	V	r_1	r_0
I	Step Out	0	1	1	u	h	V	r_1	r_0
II	Read Command	1	0	0	m	b	E	0	0
II	Write Command	1	0	1	m	b	E	a_1	a_0
III	Read Address	1	1	0	0	0	1	0	0
III	Read Track	1	1	1	0	0	1	0	$\overline{s}$
III	Write Track	1	1	1	1	0	1	0	0
IV	Force Interrupt	1	1	0	1	I_3	I_2	I_1	I_0

(a)

BIT VALUES FOR TYPE 1
h = Head Load flag (Bit 3)
h = 1, Load head at beginning
h = 0, Do not load head at beginning
V = Verify flag (Bit 2)
V = 1, Verify on last track
V = 0, No verify
r_1r_0 = Stepping motor rate (Bits 1 through 0)
r_1r_0 = 11 gives 40 ms step time
u = Update flag (Bit 4)
u = 1, Update track register
u = 0, No update

(b)

BIT VALUES FOR TYPE II
m = Multiple Record flag (Bit 4)
m = 0, Single record
m = 1, Multiple records
b = Block length flag (Bit 3)
b = 1, IBM format (128 to 1024 bytes)
b = 0, Non-IBM format (16 to 4096 bytes)
a_1a_0 = Data Address Mark (Bits 1 through 0)
a_1a_0 = 00, FB (Data Mark)
a_1a_0 = 01, FA (User defined)
a_1a_0 = 10, F9 (User defined)
a_1a_0 = 11, F8 (Deleted Data Mark)

(c)

BIT VALUES FOR TYPE III
$\overline{s}$ = Synchronize flag (Bit 0)
$\overline{s}$ = 0, Synchronize to Address Mark
$\overline{s}$ = 1, Do not synchronize to Address Mark

(d)

BIT VALUES FOR TYPE IV
I_0 thru I_3 = Interrupt Condition flags (Bits 3 through 0)
I_0 = 1, Not Ready to Ready transition
I_1 = 1, Ready to Not Ready transition
I_2 = 1, Index pulse
I_3 = 1, Immediate interrupt
E = Enable HLD and 10 ms Delay
E = 1, Enable HLD, HLT and 10 ms delay
E = 0, Head is assumed engaged and there is no 10 ms delay

(e)

Table 1: *The high-level instructions of the FD1771 disk formatter/controller device. When one of the instructions defined by table 1a is loaded into the command register of the FD1771, the FD1771 executes one or a series of actions. Bits represented by a letter within a command are defined in the bit-value tables for that type of instruction, tables 1b through 1e.*

ware coordinating the disk operation checks to see if the controller is busy from the last command. If it is not, the software writes the desired command into the command register. If data is to be transferred as each byte is assembled (or disassembled) by the shift register, the controller sends a DRQ (*data request*) signal. When the operation is completed, the controller sends an INTRQ (*interrupt request*) signal. The status register can then be checked by the controlling software for seek, write protect, busy, or CRC errors.

Controller Hardware

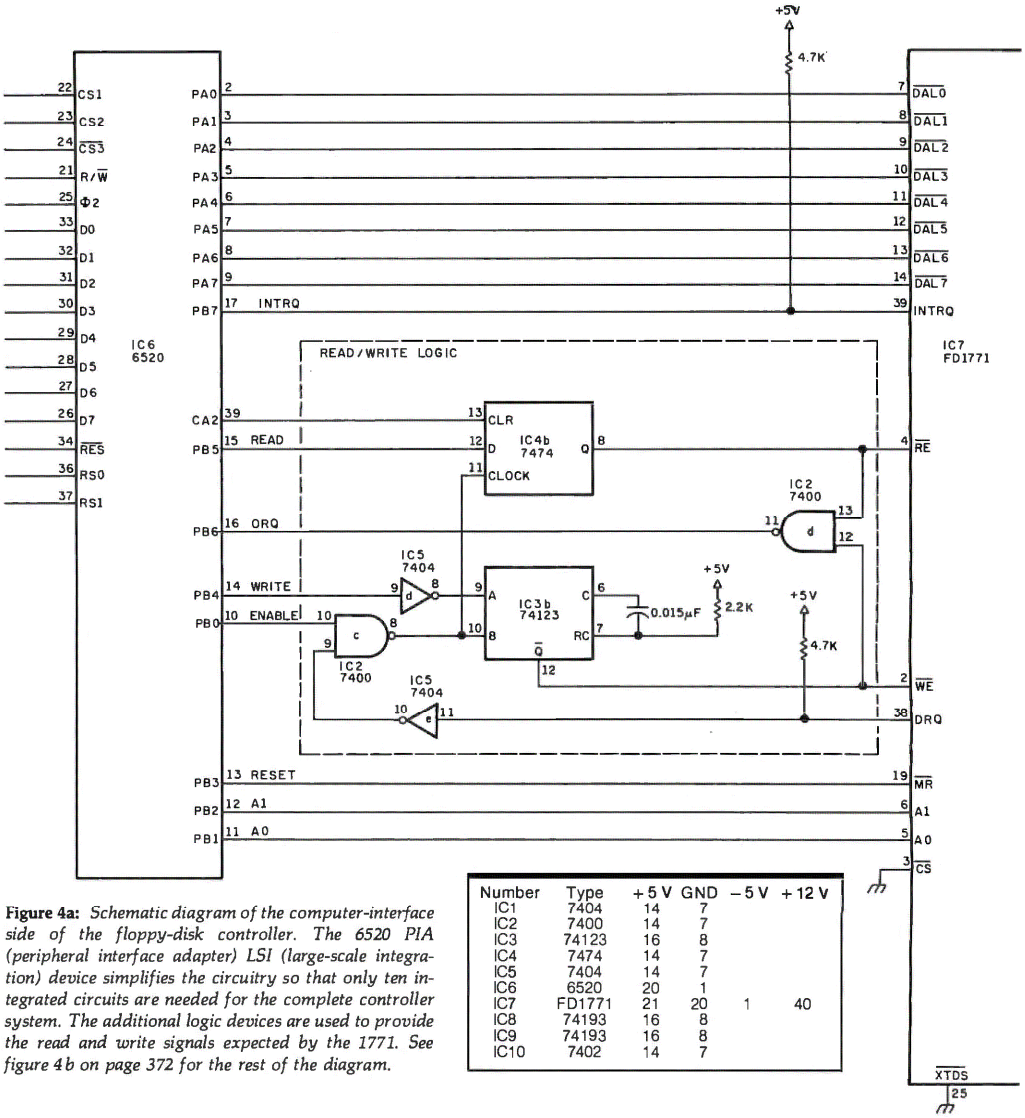
The schematic diagram for the floppy-disk controller is given in figure 4. In addition to the 1771 and the 6520 PIA (peripheral interface adapter), circuitry is included for read/write control, clock and data bit separation, head loading, and inversion of various signals as required by the FD400 disk drive.

Three gates convert the DIR (direction) and STEP signals from the 1771 into the STEP-IN and STEP-OUT signals needed by the FD400 disk drive. The HEAD-LOAD signal is conditioned by a simple one-shot (monostable multivibrator) and an inverter; this guarantees a fixed 40 ms pause allowing the head to load and settle. Once the interval has passed, a signal is sent to the 1771 to acknowledge the fact.

The data-separator and clock circuit was designed by Steve Christiansen of Iowa State University. This circuit contains four of the ten integrated circuits in the system. (If the disk drive you intend to use has sepa-

A1	A0	Register Affected During Read ($\overline{RE}$ = 0, $\overline{WE}$ = 1)	Register Affected During Write ($\overline{RE}$ = 1, $\overline{WE}$ = 0)
0	0	Status Register	Command Register
0	1	Track Register	Track Register
1	0	Sector Register	Sector Register
1	1	Data Register	Data Register

Table 2: *Access to registers within the Western Digital FD1771 disk formatter/controller device. The FD1771 has five internal registers: command, data, sector, status, and track. A given register is read or written by placing the appropriate values on lines A1 and A0 and pulling down either the READ-ENABLE (RE) line for a read operation, or the WRITE-ENABLE (WE) line for a write operation. The sector and track registers specify the sector and track when these parameters are needed by a given command byte. The command register, when filled, causes one of eleven high-level instructions to be executed (see table 1). Data passes between the computer and the disk drive through the data register. After a command has been executed by the FD1771, the status register must be read before another command can be executed.*



Number	Type	+5 V	GND	−5 V	+12 V
IC1	7404	14	7		
IC2	7400	14	7		
IC3	74123	16	8		
IC4	7474	14	7		
IC5	7404	14	7		
IC6	6520	20	1		
IC7	FD1771	21	20	1	40
IC8	74193	16	8		
IC9	74193	16	8		
IC10	7402	14	7		

Figure 4a: *Schematic diagram of the computer-interface side of the floppy-disk controller. The 6520 PIA (peripheral interface adapter) LSI (large-scale integration) device simplifies the circuitry so that only ten integrated circuits are needed for the complete controller system. The additional logic devices are used to provide the read and write signals expected by the 1771. See figure 4b on page 372 for the rest of the diagram.*

rated clock and data signals, you may be able to eliminate some of the circuitry shown. Remember that the 1771 requires a 2.0 MHz clock.)

The clock part of this circuit is a conventional TTL (transistor-transistor logic) crystal oscillator which also drives a divide-by-two stage to produce the 2.0 MHz clock signal. The data-separator part of the circuit inverts the raw signal from the disk drive and gates it out as data or clock information, depending on the state of the QD output of IC9.

There is a certain difficulty in determining, from a serial-bit stream, which bits are clock and which data (the two are interleaved, and some of the clock bits may be missing). The solution relies on the fact that, at most, three clock pulses will be omitted; if four in a row are omitted, the data and clock outputs are switched by the external data-separator circuit.

The read/write circuitry is very compact and plays a major role in the simplicity of the system. It is a subtle solution to a timing problem; the obvious approach of using the outputs of the 6520 to control $\overline{RE}$ and $\overline{WE}$ (the read- and write-enable lines) as input for the DRQ (data-request line) is too slow. The indicated circuitry using the ENABLE line causes each DRQ signal to automatically generate another $\overline{RE}$ or $\overline{WE}$ signal as required.

The 6520 has 20 programmable I/O (input/output) pins (see figure 5),

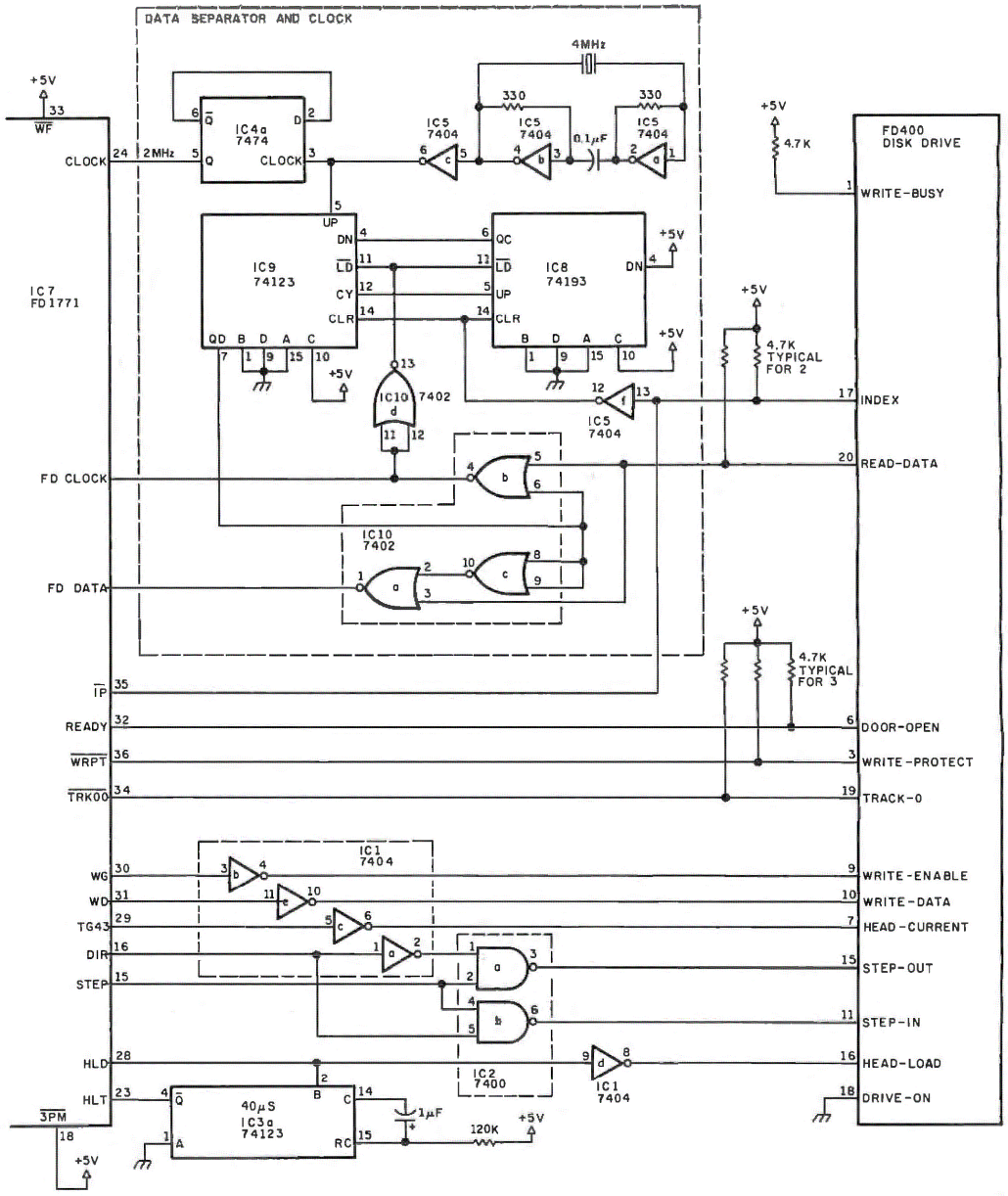


Figure 4b: *Schematic diagram of the drive-interface side of the floppy-disk controller. Clock signals and minor control functions are provided for by the additional circuitry, as well as the separation of recorded data from recorded synchronization pulses.*

of which only 17 are used in this system to interface with the 1771. The A port is programmed as eight bidirectional data lines, and is connected to the 1771's data lines, while the B port pins are programmed as necessary to provide control lines. The data lines of the 6520 can be connected to like lines on the microprocessor, while its three device-select lines can be connected to match whatever address-decoding scheme is appropriate. The

PIA Register Selection and Function

RS1	RS0	R/$\overline{W}$	CRA2	CRB2	Function
0	0	X	0	X	Read or Write. DDRA
0	0	0	1	X	Write into ORA
0	0	1	1	X	Read from A-side input pins
0	1	X	X	X	Read or Write CRA
1	0	X	X	0	Read or Write DDRB
1	0	0	X	1	Write into ORB
1	0	1	X	1	Read from B-side input pins
1	1	X	X	X	Read or Write CRB

X = don't care

Control Register Bit Designations

	7	6	5 4 3	2	1 0
CRA	IRQA1	IRQA2	CA2 Control	DDRA Access	CA1 Control
CRB	IRQB1	IRQB2	CB2 Control	DDRB Access	CB1 Control

Control of CA2 Output Modes

Bit 5	CRA Bit 4	Bit 3	Mode	Description
1	0	0	"Handshake" on Read	CA2 is set high on an active transition of the CA1 interrupt input signal and set low by a microprocessor "read A data" operation. This allows positive control of data transfers from the peripheral device to the microprocessor.
1	0	1	Pulse Output	CA2 goes low for one cycle after a "read A data" operation. This pulse can be used to signal the peripheral device that data was taken.
1	1	0	Manual Output	CA2 set low
1	1	1	Manual Output	CA2 set high

Table 3: *Control codes for the 6520. This device offers 20 pins that may be programmed (either individually or in groups) as input, output, or bidirectional lines.*

6520 controls and modes are listed in table 3.

Construction Notes

The prototype floppy-disk controller was built on a Vector 3677 wire-wrap board (see photo 1). There are no special layout considerations, but adequate power supply bypassing must be observed (i.e., 0.1 μf capacitors across the supply and ground pins of each integrated circuit). A 16-pin DIP (dual in-line package) socket is used to connect the controller to a ribbon cable from the disk drive (use proper terminations).

Debugging

The read/write circuit can be debugged by using a microcomputer. Move the DRQ input (IC5, pin 11 in figure 4) from the 1771 to a convenient 6520 output. With the microcomputer running a diagnostic program, check to see that the $\overline{WE}$ pulse (IC3, pin 12 in figure 4) is about 14 μs.

The data separator can be checked by using a single-pulse input signal in

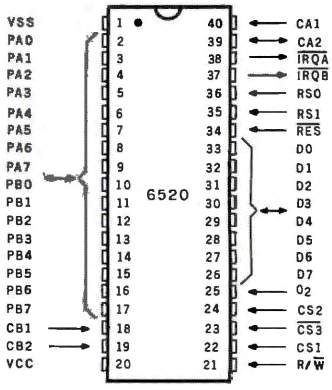


Figure 5: *Pin description of the MOS Technology 6520 PIA. Use of this particular device allows easy interfacing of a disk controller to a 6502-based computer. One I/O port handles control signals; the other is used to transfer parallel bytes of data.*

lieu of the 4.0 MHz crystal oscillator signal. The output of IC9 should count through the full range of 0 through 15, starting at 4, while IC8 should count from 4 through 8.

The INTRQ and DRQ signals were connected to PB6 and PB7 of the 6520 because powerful testing instructions are available for these pins. If problems occur in this area, these instructions will come in handy.

Testimonials

This system has been built by several people and has been proven to work with minimal debugging, using wire-wrap, Slit-N-Wrap, and Super Strip techniques. The circuits are not the simplest possible; we have interfaced a 5-inch disk drive to the KIM and AIM systems using only three integrated circuits. The newer versions of the 1771, which allow the controller to be connected directly to data and address buses, do not need a 6520; but there is a case for isolating the microcomputer from the disk controller through a 6520. Whatever route you choose, this basic design will provide reliable, trouble-free operation.

In Part 2, next month, we will look at the software needed to use this controller. ■

References

1. Allen, David M. "A Floppy Disk Interface." BYTE, January 1978, page 58.
2. Bal, Subhash, and Mike Wright. "LSI Controller." *Electronic Design*, September 1978, page 90.
3. Camp, Roger, Terry Smay and James Triska. *Micro Computer Systems Principles, 2nd Edition*. Matrix Publishers Inc, 1979.
4. Elinoff, Gary. "Floppy Disk Memory Systems." *Electronics Products*, May 1978, page 39.
5. *FD1771 A/B-01 Data Sheet*. Western Digital.
6. Hoeppner, John. "Interface a Floppy-Disk Drive to an 8080A-Based Computer." BYTE, May 1980, page 72.
7. *IBM Diskette General Information Manual*, No. GA 21-9182. International Business Machines Corporation.
8. Nicholson, James H. "A Super Simple Floppy Disk Controller." *Proceedings of the National Electronics Conference*, Volume 33.
9. *Operating and Service Manual No. 600203*. Pertec.
10. Welles, Kenneth B. "Build This Economy Floppy Disk Interface." BYTE, February 1977, page 34.

Build a Super Simple Floppy-Disk Interface

Part 2: Software

James Nicholson and Roger Camp
1046 Gaskill
Ames IA 50010

The first part of this article presented basic floppy-disk technology and a description of a simple controller design with its circuit details. This controller provides a great deal of function and flexibility when combined with some simple software.

Software

The software shown in listing 1 provides disk-formatting, reading, writing, and error-recovery functions. The software can be reassembled to allow relocation of program or page zero variables. Various entry points are shown in table 4.

Before using the FD1771 to read and write data within the sectors on the floppy disk, the disk must be formatted to conform to a certain structure. A program (entry point FORMAT) is supplied that formats all 77 tracks of a standard 8-inch disk in a standard IBM-compatible 128-bytes-per-sector arrangement (each track contains 26 sectors).

The program, when called, initializes all 6520 and 1771 electronic interfaces before writing the standard track. The initialization process guarantees that the head is positioned over the outermost track. Each track is written from a standard pattern contained in programmable memory. A 40 ms delay is generated following a step-in function to move the head to the next track. This guarantees the proper head-settling time required by the floppy-disk drive. This process continues until all tracks have been formatted.

Sector sizes other than 128 bytes can be selected by initializing the 1771 differently. (A sector size other than 128 can lead to incompatibilities with *other* floppy-disk systems.) For sector lengths greater than 128, the FORMAT program must be rewritten to use an entire track image in memory. This is required because of an indexing limit of 256 using the 6502 microprocessor. Our system, using sixteen 256-byte sectors per track, has proven to be a convenient alternative.

When a disk is properly formatted, the basic I/O (input/output) program (entry point FDENT) can be used. If the system has just been turned on, entry point FDENT should be called first to initialize all interface and drive electronics. To perform disk operations, certain variables must be set up before calling FDENT. They include the desired *command*, *track number*, and *sector number*, as well as the address in memory used for data transfer (see table 5).

The program begins by analyzing the command to determine which segment of the program must be used in response. There are three basic command types:

- head movement
- read/write sectors
- read/write raw tracks

In the case of read/write commands, the program ascertains if the head is positioned properly and, if necessary, provides the seek command to move it.

Following execution of the command by the 1771, completion

Name	Purpose
FORMAT	Write proper track format on all 77 tracks
FDINT	Initialize 6520 and 1771 interface
FDENT	Perform basic floppy-disk operations using established variables
FDIO	Uses FDENT, followed by error checking and retry

Table 4: *Entry points for various floppy-disk controller operations.*

Name	Length in Bytes	Purpose
DVCODE	1	Device-selection byte 00 = DVC 0, 80 = DVC 1
ERRCDE	1	FF = Error, 00 = Normal Set by FDIO
COMMAND	1	1771 Command byte
STATUS	1	1771 Completion status
TRACK	1	Desired track value
SECTOR	1	Desired sector value
FDBUF	2	Address of data buffer

Table 5: *Variables used to perform floppy-disk operations. All values are listed in hexadecimal.*

The numbering of all nontext material is continued from part 1 of this article.

analysis is performed to read back and store the status, track number, and sector number from the 1771. The status can then be examined by the user program to determine if the operation was successful. No registers are saved by any of the routines previously discussed.

Although the hardware design has proven to be very reliable, an error occasionally occurs. Since it would be a great burden for each application to concern itself with error recovery, another program has been provided. Using entry point FDIO, a user program can add the error-recovery function to that provided by FDENT.

After storing all the registers, FDIO calls FDENT to perform the requested operation. Following completion, FDIO examines the status to determine if an error occurred, and, if so, the operation may be retried. Generally, read/write operations will be retried up to five times before assuming a "hard" (ie: nontransient) error.

A nonrecoverable error is indicated with hexadecimal FF in the ERRCDE variable (see listing 2). This condition generally causes the application program to terminate so the error can be researched. The STATUS variable provides details about the specific problem.

Certain nonrecoverable conditions will not be retried. For example, a *busy* or *device not ready* condition causes an error condition without retry. The program can be altered to increase the sophistication to any level desired. Errors can be cataloged and recorded on another floppy disk to provide a history of all abnormal conditions.

	Variables (all values in hexadecimal)				
COMMAND	TRACK	SECTOR	BUFADR	BUFADR+1	TEST
02	—	—	—	—	RESTORE
1A	20	—	—	—	SEEK
16	10	—	—	—	VERIFY
8C	10	01	00	10	READ

Table 6: *Values to be set in variables for testing the controller (with the routine in listing 3). All values are listed in hexadecimal.*

Testing

After completing construction of the controller circuit and verifying the proper timing of the 74123 components, some simple tests can be performed to verify proper operation. These tests can be conducted with the aid of a simple program (listing 3) and table (table 6). Set your monitor to begin execution at INIT. When the break occurs, set the variables as shown for each specific test and allow program execution to continue. This procedure requires you to load the software previously discussed. Initial testing requires a *preformatted* IBM-compatible disk. Examination of the status byte following each test helps diagnose any existing problems.

The restore-drive procedure should generate stepping pulses that move the head to the track 0 position. The head-drive lead screw can be moved manually off the track 0 position to verify proper operation.

Directing the head to seek to a specific track requires the desired track value to be set in the data register of the 1771. This test also loads the head but does not attempt to perform a track verification. This test can be repeated several times with different track values to determine if the 1771 properly seeks in both directions.

If the controller moves the head correctly, the third test performs a track verification. Following the seek movement, the head is loaded, and the 1771 reads the address information recorded on the track to verify that it has located the proper track.

The fourth test attempts to read a specific sector. The data is stored beginning at location hexadecimal

Text continued on page 340

Listing 1: *Software to provide fundamental high-level operations for the disk controller (written for the 6502 microprocessor).*

```
   FD400/FD1771B FLOPPY DISK CONTROL                                    PAGE    1

CARD #  LOC     CODE       CARD  10        20        30        40        50        60        70
     2  0000                   .OPT  CNT,XREF,MEM,LIST,ERR,GEN
     3  0000            ;
     4  0000            ;  J.H.NICHOLSON 1-22-79
     5  0000            ;
     6  0000            ;
     7  0000            ;   THIS SEGMEMT PROVIDES BASIC CONTROL FUNCTIONS FOR EXECUTING
     8  0000            ;  COMMANDS TO A PERTEC FD400 FLOPPY DISK DRIVE CONTROLED WITH
     9  0000            ;  A WESTERN DIGITAL FD1771B-01 FLOPPY DISK CONTROLLER. THE BASIC
    10  0000            ;  ROUTINES PROVIDE AN INTERFACE VIA A 6520 PIA TO THE FD400/FD1771B
    11  0000            ;  SYSTEM.
    12  0000            ;   THE FD1771B COMMAND, TRACK, AND SECTOR VALUES ARE PROVIDED IN
    13  0000            ;  PAGE ZERO VARIABLES. THE COMMAND AND OTHER NECESSARY DATA IS
    14  0000            ;  WRITTEN TO THE FD1771B AND THE COMMAND IS EXECUTED. FOLLOWING
    15  0000            ;  THE COMPLETION OF THE COMMAND(INTRQ FROM FD1771B) THE STATUS,
    16  0000            ;  TRACK, AND SECTOR VALUES ARE READ FROM THE FD1771B. NO ERROR
    17  0000            ;  RECOVERY IS PERFORMED BY THIS SEGMEMT. IF THE DEVICE IS BUSY
    18  0000            ;  WHEN THIS SEGMENT IS GIVEN CONTROL, NORMAL COMPLETION ANALYSIS
    19  0000            ;  WILL BE DONE.
    20  0000            ;   TWO ENTRY POINTS TO THIS SEGMENT PROVIDE COMMAND EXECUTION AND
    21  0000            ;  PIA INITIALIZATION.
    22  0000            ;     FDENT.....ENTRY FOR FD1771B COMMAND EXECUTION.
    23  0000            ;     FDINT.....ENTRY FOR INITIALIZATION OF FD400/FD1771B.
    24  0000            ;
    25  0000            ;
    26  0000            ;   ALL FD1771B COMMANDS ARE VALID AND ARE LISTED BELOW BY FUNCTIONAL
    27  0000            ;  CATAGORY AS WELL AS TYPE GROUPS.
    28  0000            ;
    29  0000            ;   BASIC :  RESTORE              TYPE 1.  RESTORE
    30  0000            ;             STEP                          SEEK
    31  0000            ;             STEP IN                       STEP
    32  0000            ;             STEP OUT                      STEP IN
    33  0000            ;             FORCE INTRQ                   STEP OUT
    34  0000            ;
    35  0000            ;   READ   :  READ SECTOR          TYPE 2.  READ SECTOR
    36  0000            ;             READ TRACK                    WRITE SECTOR
    37  0000            ;             READ ADDR
    38  0000            ;                                 TYPE 3.  READ ADDR
    39  0000            ;   WRITE  :  WRITE SECTOR                  READ TRACK
    40  0000            ;             WRITE TRACK                   WRITE TRACK
    41  0000            ;
    42  0000            ;   SEEK   :  SEEK                 TYPE 4.  FORCE INTRQ



   FD400/FD1771B FLOPPY DISK CONTROL                                    PAGE    2

CARD #  LOC     CODE       CARD  10        20        30        40        50        60        70
    44  0000            ;
    45  0000            ;
    46  0000            ;   WHEN GIVEN CONTROL, THIS SEGMENT ANALYZES THE COMMAND TYPE
    47  0000            ;  TO DETERMINE THE FUNCTIONS WHICH MUST BE PERFORMED. THE COMMANDS
    48  0000            ;  CAN BE SEGMENTED INTO FOUR LOGICAL FUNCTION GROUPS WHICH ARE
    49  0000            ;  SIMILAR TO THE FD1771B COMMAND TYPES.
    50  0000            ;
    51  0000            ;
    52  0000            ;    CMD TYPE                  FUNCTION
    53  0000            ;    --------                  --------
    54  0000            ;    TYPE 1(EX.SEEK)           BASIC FUNCTION
    55  0000            ;
    56  0000            ;    TYPE 1(SEEK)              WRITE NEW TRACK, THEN BASIC
    57  0000            ;
    58  0000            ;    TYPE 2                    WRITE SECTOR REGISTER
    59  0000            ;                              SPLIT TO READ OR WRITE
    60  0000            ;
    61  0000            ;    TYPE 3                    SPLIT TO READ OR WRITE
    62  0000            ;
    63  0000            ;    TYPE 4                    BASIC FUNCTION
    64  0000            ;
```

Listing 1 continued on page 308

Listing 1 continued:

```
   65  0000              ;
   66  0000              ;   BASIC FUNCTION :
   67  0000              ;     1. WRITE COMMAND TO THE FD1771B.
   68  0000              ;     2. WAIT FOR COMPLETION(INTRQ).
   69  0000              ;     3. COMPLETION ANALYSIS(READ STATUS, TRACK, AND SECTOR)
   70  0000              ;     4. EXIT
   71  0000              ;
   72  0000              ;   SEEK FUNCTION :
   73  0000              ;     1. WRITE NEW TRACK TO DATA REGISTER.
   74  0000              ;     2. WRITE SECTOR TO SECTOR REGISTER.
   75  0000              ;     3. GO TO BASIC FUNCTION.
   76  0000              ;
   77  0000              ;   READ FUNCTION :
   78  0000              ;     1. SEEK TO PROPER TRACK IF NECESSARY
   79  0000              ;     2. WRITE SECTOR TO SECTOR REGISTER.
   80  0000              ;     3. WRITE COMMAND TO FD1771B.
   81  0000              ;     4. WAIT & LOOP FOR DRQ/INTRQ READING DATA ON DRQ.
   82  0000              ;     5. ON INTRQ DO COMPLETION ANALYSIS(BASIC FCTN, STEP 3)
   83  0000              ;
   84  0000              ;   WRITE FUNCTION :
   85  0000              ;     1. SEEK TO PROPER TRACK IF NECESSARY
   86  0000              ;     2. WRITE SECTOR TO SECTOR REGISTER.
   87  0000              ;     3. WRITE COMMAND TO FD1771B.
   88  0000              ;     4. WAIT & LOOP FOR DRQ/INTRQ WRITING DATA ON DRQ.
   89  0000              ;     5. ON INTRQ DO COMPLETION ANALYSIS(BASIC FCTN, STEP 3)
```

```
   FD400/FD1771B FLOPPY DISK CONTROL                                  PAGE    3

CARD #  LOC   CODE      CARD  10        20        30        40        50        60        70
   91  0000             ;
   92  0000             ;  ******* 6520 PIA
   93  0000             ;
   94  0000             SADD   =$CC0C           6520 PIA A DATA DIRECTION
   95  0000             SAD    =$CC0C           6520 PIA A DATA REGISTER
   96  0000             CRA    =$CC0D           6520 PIA A CONTROL REGISTER
   97  0000             SBDD   =$CC0E           6520 PIA B DATA DIRECTION
   98  0000             SBD    =$CC0E           6520 PIA B DATA REGISTER
   99  0000             CRB    =$CC0F           6520 PIA B CONTROL REGISTER
  100  0000             ;
  101  0000             ;  ******* PIA CONNECTIONS
  102  0000             ;
  103  0000             ;   CA1 <-- UNUSED
  104  0000             ;   CA2 --> PULSE(¬RE CLR)
  105  0000             ;     PA7 <-> DAL7
  106  0000             ;     PA6 <-> DAL6
  107  0000             ;     PA5 <-> DAL5
  108  0000             ;     PA4 <-> DAL4
  109  0000             ;     PA3 <-> DAL3
  110  0000             ;     PA2 <-> DAL2
  111  0000             ;     PA1 <-> DAL1
  112  0000             ;     PA0 <-> DAL0
  113  0000             ;
  114  0000             ;     PB7 <-- INTRQ
  115  0000             ;     PB6 <-- DRQ
  116  0000             ;     PB5 --> READ
  117  0000             ;     PB4 --> WRITE
  118  0000             ;     PB3 --> ¬MR
  119  0000             ;     PB2 --> A1
  120  0000             ;     PB1 --> A0
  121  0000             ;     PB0 --> ¬ENABLE R/W
  122  0000             ;   CB1 <-- UNUSED
  123  0000             ;   CB2 --> DEVICE SELECT
  124  0000             ;
  125  0000             ;  ******* FD1771B COMMANDS
  126  0000             ;
  127  0000             FDRST  =$02             RESTORE
  128  0000             FDSK   =$12             SEEK
  129  0000             FDST   =$22             STEP
  130  0000             FDSTI  =$42             STEP IN
  131  0000             FDSTO  =$62             STEP OUT
  132  0000             FDRD   =$80             READ SECTOR
  133  0000             FDWT   =$A0             WRITE SECTOR
```

Listing 1 continued on page 310

Listing 1 continued:

```
  134  0000            FDRDA   =$C4            READ ADDRESS
  135  0000            FDRDT   =$E4            READ TRACK
  136  0000            FDWTT   =$F4            WRITE TRACK
  137  0000            FDFI    =$D0            FORCE INTERUPT
  138  0000            ;
  139  0000            ;  ******* COMMAND QUALIFIERS
  140  0000            ;
  141  0000            QV      =$04            VERIFY
  142  0000            QH      =$08            LOAD HEAD
  143  0000            QU      =$10            UPDATE TRK REG
  144  0000            QM      =$10            MULTIPLE RECORDS
  145  0000            QB      =$08            IBM FORMAT


   FD400/FD1771B FLOPPY DISK CONTROL                                  PAGE    4

CARD # LOC      CODE      CARD  10        20        30        40        50        60        70
  146  0000            QS      =$01            NOT SYNC TO AM
  147  0000            QI0     =$01            NR TO R TRANS.
  148  0000            QI1     =$02            R TO NR TRANS.
  149  0000            QI2     =$04            INDEX PULSE
  150  0000            QI3     =$08            EACH 10 MS.
  151  0000            QE      =$04            ENABLE HLD + HLT DELAY
  152  0000            QFB     =$00            FB DATA MARK
  153  0000            QFA     =$01            FA DATA MARK
  154  0000            QF9     =$10            F9 DATA MARK
  155  0000            QF8     =$11            F8 DATA MARK
  156  0000            ;
  157  0000            ;  ******* INITIALIZATION CTL BYTES
  158  0000            ;
  159  0000            QCRC    =$F7            WRITE CRC
  160  0000            QIAM    =$FC            INDEX ADDR MARK
  161  0000            QIDM    =$FE            ID ADDR MARK
  162  0000            QAFB    =$FB            FB DATA MARK
  163  0000            QAFA    =$FA            FA DATA MARK
```

Listing 1 continued on page 312

Listing 1 continued:

```
  164  0000              QAF9   =$F9              F9 DATA MARK
  165  0000              QAF8   =$F8              F8 DATA MARK
  166  0000              ;
  167  0000              ;  ******* PIA CONTROL COMMANDS(-MR ON)
  168  0000              ;
  169  0000              READ   =$29              READ FD1771B
  170  0000              WRITE  =$19              WRITE FD1771B
  171  0000              STAT   =$00              A1=0,A0=0   STATUS REGISTER
  172  0000              TRK    =$02              A1=0,A0=1   TRACK REGISTER
  173  0000              SECT   =$04              A1=1,A0=0   SECTOR REGISTER
  174  0000              DATA   =$06              A1=1,A0=1   DATA REGISTER
  175  0000              CMD    =$00              A1=0,A0=0   COMMAND REGISTER
  176  0000              ;
  177  0000              ;  ******* PAGE ZERO VARIABLES/EQUATES
  178  0000              ;
  179  0000              TIME1  *=*+1
  180  0001              TIME2  *=*+1
  181  0002                     *=$E0
  182  00E0              DVCODE *=*+1             DVC/FILE CODE
  183  00E1              ERRCDE *=*+1             ERROR CODE
  184  00E2              COMAND *=*+1 FD1771B COMMAND
  185  00E3              STATUS *=*+1          STATUS
  186  00E4              TRACK  *=*+1          TRACK
  187  00E5              SECTOR *=*+1          SECTOR
  188  00E6              FDBUF  *=*+2          BUFFER PTR
  189  00E8              ;
  190  00E8                     *=$200
```

FD400/FD1771B FLOPPY DISK CONTROL PAGE 5

```
CARD # LOC   CODE       CARD  10        20        30        40        50        60        70
  192  0200              ;
  193  0200              ;  ******* TYPE 1 COMMANDS
  194  0200              ;
  195  0200  A5 E2       TYPE1  LDA   COMAND     IF NOT SEEK
  196  0202  C9 20              CMP   #$20        ASSUME
  197  0204  B0 3A              BCS   BASIC         BASIC
  198  0206  C9 10              CMP   #$10        IF RESTORE
  199  0208  90 36              BCC   BASIC       ASSUME BASIC
  200  020A  A9 1F              LDA   #WRITE+DATA PIA CTL CMD
  201  020C  20 DE 02           JSR   SETUP       SET-UP PIA
  202  020F  A5 E4              LDA   TRACK       TRACK ADDR
  203  0211  C9 4D              CMP   #$4D        IF PAST END
  204  0213  B0 33              BCS   CMPANL      RETURN
  205  0215  20 CD 02           JSR   PULSE       WRITE TRACK
  206  0218  A9 1D              LDA   #WRITE+SECT PIA CTL CMD
  207  021A  20 DE 02           JSR   SETUP       SET-UP PIA
  208  021D  A5 E5              LDA   SECTOR      SECTOR ADDR
  209  021F  20 CD 02           JSR   PULSE       WRITE SECTOR
  210  0222  4C 40 02           JMP   BASIC       CONTINUE
  211  0225              ;
  212  0225              ;  ******* COMMAND ENTRY ANALYSIS
  213  0225              ;
  214  0225  A9 29       FDENT  LDA   #READ+STAT PIA CTL CMD        ** ENTRY **
  215  0227  20 DE 02           JSR   SETUP       SET-UP PIA
  216  022A  20 CD 02           JSR   PULSE       READ STATUS
  217  022D  6A                 ROR   A           IF DEVICE BUSY
  218  022E  B0 18              BCS   CMPANL        DO COMPLETION
  219  0230              ;
  220  0230              ;  ******* DETERMINE COMMAND TYPE
  221  0230              ;
  222  0230  A9 10              LDA   #$10        CMD MASK
  223  0232  24 E2              BIT   COMAND      CHECK FOR
  224  0234  10 CA              BPL   TYPE1         TYPE 1
  225  0236  50 23              BVC   TYPE2         TYPE 2
  226  0238  F0 4A              BEQ   RDATA         TYPE 3 READ
  227  023A  A9 20              LDA   #$20        SEPERATE
  228  023C  24 E2              BIT   COMAND        FORCE INTRQ FROM
  229  023E  D0 63              BNE   WDATA         TYPE 3 WRITE
  230  0240              ;
  231  0240              ;  ******* BASIC COMMAND PROCESS
  232  0240              ;
  233  0240  20 C2 02    BASIC  JSR   WRTCMD      WRITE CMD TO FD1771B
  234  0243  2C 0E CC           BIT   SBD         WAIT FOR
```

Listing 1 continued on page 314

Listing 1 continued:

```
  235  0246  10 FB            BPL   *-3          INTRQ
  236  0248                  ;
  237  0248                  ;  ******* COMPLETION ANALYSIS
  238  0248                  ;
  239  0248  A0 02     CMPANL LDY   #2           LOOP CNT + INDEX
  240  024A  98        CPLP   TYA                USE INDEX TO
  241  024B  0A               ASL   A             SET A1,A0
  242  024C  09 29            ORA   #READ        SET READ
  243  024E  20 DE 02         JSR   SETUP        SET-UP PIA
  244  0251  20 CD 02         JSR   PULSE        READ REGISTER
  245  0254  99 E3 00         STA   STATUS,Y     STORE DATA
  246  0257  88               DEY                DECR INDEX
```



```
CARD #  LOC   CODE       CARD  10        20        30        40        50        60        70
  247  0258  10 F0            BPL   CPLP        CONTINUE
  248  025A  60               RTS               RETURN
  249  025B                  ;
  250  025B                  ;  ******* TYPE 2 VERIFY TRACK
  251  025B                  ;
  252  025B  A9 2B     TYPE2  LDA   #READ+TRK PIA CTL CMD
  253  025D  20 DE 02         JSR   SETUP        SET-UP PIA
  254  0260  20 CD 02         JSR   PULSE        READ TRACK
  255  0263  C5 E4            CMP   TRACK        IF NOT EQUAL
  256  0265  F0 0D            BEQ   TYPE2A        SEEK TO TRACK
  257  0267  A5 E2            LDA   COMAND       SAVE COMMAND
  258  0269  48               PHA                 FOR LATER
  259  026A  A9 12            LDA   #FDSK        SEEK COMMAND
  260  026C  85 E2            STA   COMAND       SET IT
  261  026E  20 25 02         JSR   FDENT        DO SEEK
  262  0271  68               PLA                RESTORE
  263  0272  85 E2            STA   COMAND        COMMAND
  264  0274                  ;
  265  0274                  ;  ******* TYPE 2 COMMANDS
  266  0274                  ;
  267  0274  A9 1D     TYPE2A LDA   #WRITE+SECT PIA CTL CMD
  268  0276  20 DE 02         JSR   SETUP        SET-UP PIA
  269  0279  A5 E5            LDA   SECTOR       SECTOR ADDR
  270  027B  20 CD 02         JSR   PULSE        WRITE SECTOR
  271  027E  A9 20            LDA   #$20         SEPERATE
  272  0280  24 E2            BIT   COMAND        READ
  273  0282  D0 1F            BNE   WDATA          FROM WRITE
  274  0284                  ;
  275  0284                  ;  ******* READ DATA
  276  0284                  ;
  277  0284  20 C2 02  RDATA  JSR   WRTCMD       WRITE COMMAND
  278  0287  A0 00            LDY   #0           BUFFER INDEX
  279  0289  A9 2F            LDA   #READ+DATA PIA CTL CMD
  280  028B  20 DE 02         JSR   SETUP        SET-UP PIA
  281  028E  2C 0E CC  RDL    BIT   SBD          WAIT FOR            4
  282  0291  30 B5            BMI   CMPANL        INTRQ OR           2
  283  0293  50 F9            BVC   RDL            DRQ              2
  284  0295  AD 0C CC         LDA   SAD          GET DATA BYTE       4      25 CYCLES
  285  0298  49 FF            EOR   #$FF         INVERT DATA         2
  286  029A  91 E6            STA   (FDBUF),Y    SAVE BYTE           6
  287  029C  C8               INY                INCR BUFFER PTR     2
  288  029D  D0 EF            BNE   RDL          IF ZERO             3 2
  289  029F  E6 E7            INC   FDBUF+1       INCR BASE AND        5   + 9 CYCLES
  290  02A1  D0 EB            BNE   RDL            CONTINUE           3
  291  02A3                  ;
  292  02A3                  ;  ******* WRITE DATA
  293  02A3                  ;
  294  02A3  20 C2 02  WDATA  JSR   WRTCMD       WRITE COMMAND
  295  02A6  A0 00            LDY   #0           BUFFER INDEX
  296  02A8  A9 1F            LDA   #WRITE+DATA PIA CTL CMD
  297  02AA  20 DE 02         JSR   SETUP        SET-UP PIA
  298  02AD  B1 E6     WTL    LDA   (FDBUF),Y    GET DATA BYTE       6
  299  02AF  49 FF            EOR   #$FF         INVERT DATA         2
  300  02B1  8D 0C CC         STA   SAD          WRITE IT            4
  301  02B4  2C 0E CC  WTL1   BIT   SBD          WAIT FOR            4      25 CYCLES
```

Listing 1 continued on page 317

Listing 1 continued:

```
   FD400/FD1771B FLOPPY DISK CONTROL                                    PAGE     7

CARD #  LOC     CODE        CARD  10        20        30        40        50        60        70
   302  02B7  30 SF             BMI   CMPANL      INTRQ OR           2
   303  02B9  50 F9             BVC   WTL1          DRQ              2
   304  02BB  C8                INY                 INCR BUFFER PTR  2
   305  02BC  D0 EF             BNE   WTL           IF ZERO          3 2
   306  02BE  E6 E7             INC   FDBUF+1       INCR BASE AND      5   + 9 CYCLES
   307  02C0  D0 EB             BNE   WTL           CONTINUE           3
   308  02C2             ;
   309  02C2             ;
   310  02C2             ;  ******* WRITE COMMAND TO FD1771B
   311  02C2             ;
   312  02C2  A9 19      WRTCMD LDA   #WRITE+CMD PIA CTL CMD
   313  02C4  20 DE 02          JSR   SETUP       SET-UP PIA
   314  02C7  A5 E2             LDA   COMAND      GET COMMAND
   315  02C9  20 CD 02          JSR   PULSE         AND WRITE IT
   316  02CC  60                RTS               RETURN
   317  02CD             ;
   318  02CD             ;  ******* ENABLE FD1771B READ/WRITE
   319  02CD             ;             (TRANSFER DATA)
   320  02CD             ;
   321  02CD  49 FF      PULSE  EOR   #$FF        INVERT DATA
   322  02CF  8D 0C CC          STA   SAD         DATA OUT
   323  02D2  CE 0E CC          DEC   SBD         ENABLE
   324  02D5  EE 0E CC          INC   SBD         READ/WRITE
   325  02D8  AD 0C CC          LDA   SAD         DATA IN
   326  02DB  49 FF             EOR   #$FF        INVERT DATA
   327  02DD  60                RTS               CONTINUE
   328  02DE             ;
   329  02DE             ;  ******* SET UP PIA FOR READ/WRITE
   330  02DE             ;
   331  02DE  A2 00      SETUP  LDX   #$00        ASSUME READ
   332  02E0  8D 0E CC          STA   SBD         SET DVC CTL REG
   333  02E3  0A                ASL   A           CHECK
   334  02E4  0A                ASL   A             IF READ AND
   335  02E5  30 01             BMI   SET1          SET FOR INPUT
   336  02E7  CA                DEX               ADJUST DIR TO OUTPUT
   337  02E8  A9 00      SET1   LDA   #0          SET CTL FOR
   338  02EA  8D 0D CC          STA   CRA           DIR REGISTER
   339  02ED  8E 0C CC          STX   SADD        SET DATA DIRECTION
   340  02F0  A9 2C             LDA   #$2C        RESET PIA CTL
   341  02F2  8D 0D CC          STA   CRA           TO DATA REG
   342  02F5  60                RTS               RETURN
   343  02F6             ;
   344  02F6             ;  ******* DEVICE INITIALIZATION
```

Listing 1 continued on page 318

Listing 1 continued:

```
 345  02F6                ;
 346  02F6  A2 00     FDINT  LDX   #$00       A DIR AS INPUT      ** ENTRY **
 347  02F8  20 E8 02         JSR   SET1       SET-UP A SIDE
 348  02FB  AD 0C CC         LDA   SAD        CLEAR -RE
 349  02FE  A0 04            LDY   #$04       CTL FOR B SIDE
 350  0300  8C 0F CC         STY   CRB         DATA REGISTER
 351  0303  86 E0            STX   DVCODE     CLEAR DEVICE CODE
 352  0305  E8               INX              SET B SIDE
 353  0306  8E 0E CC         STX   SBD         DATA REGISTER
 354  0309  CA               DEX              CTL FOR B SIDE
 355  030A  8E 0F CC         STX   CRB         DIR REGISTER
 356  030D  A2 3F            LDX   #$3F       SET B SIDE
```

```
   FD400/FD1771B FLOPPY DISK CONTROL                                    PAGE     8

CARD # LOC      CODE        CARD  10        20        30        40        50        60        70
 357  030F  8E 0E CC          STX   SBDD       DIR REGISTER
 358  0312  A2 3C            LDX   #$3C       SELECT
 359  0314  8E 0F CC          STX   CRB         DEVICE 1
 360  0317  A9 02            LDA   #FDRST     RESTORE CMD
 361  0319  85 E2            STA   COMAND     SAVE IT
 362  031B  20 40 02         JSR   BASIC      RESTORE DEVICE 1
 363  031E  A2 34            LDX   #$34       SELECT
 364  0320  8E 0F CC          STX   CRB         DEVICE 0
 365  0323  4C 40 02         JMP   BASIC      RESTORE DEVICE 0
 366  0326              ;
 367  0326              ;
```

```
   FLOPPY DISK I/O & ERROR RECOVERY                                     PAGE     9

CARD # LOC      CODE        CARD  10        20        30        40        50        60        70
 369  0326              ;
 370  0326              ;
 371  0326              ;
 372  0326              ;    THIS SEGMENT PROVIDES FLOPPY DISK I/O OPERATIONS, USING
 373  0326              ;   THE FDENT ROUTINE, AND ERROR RECOVERY IS PERFORMED PRIOR TO
 374  0326              ;   RETURNING. COMMAND, TRACK, SECTOR, AND BUFFER ADDRESS ARE
 375  0326              ;   SET AS IF FDENT WERE TO BE USED. FDIO WILL SAVE ALL REGISTERS
 376  0326              ;   AS WELL AS PERFORMING ERROR RECOVERY. IF A COMMAND IS
 377  0326              ;   CONSIDERED RECOVERABLE, IT WILL BE RETRIED 5 TIMES PRIOR TO
 378  0326              ;   RETURNING A PERMANENT ERROR INDICATION IN ERRCODE($FF).
 379  0326              ;   A ZERO INDICATES A NORMAL COMPLETION OF THE I/O ACTIVITY.
 380  0326              ;   ADDITIONALLY THE PROPER FLOPPY DRIVE WILL BE SELECTED USING
 381  0326              ;   THE HIGH BIT OF DVCODE.
 382  0326              ;
 383  0326              ;    TYPE      MASK       ERROR
 384  0326              ;
 385  0326              ;    TYPE1     00011000   SEEK ERROR
 386  0326              ;    WRITE     01111000   WRITE PROTECT/FAULT
 387  0326              ;    READ      00011100   ID/REC NOT FOUND
 388  0326              ;                         CRC ERROR
 389  0326              ;                         LOST DATA
 390  0326              ;
 391  0326              ;   ******* INITIALIZE AND EXEC CMD
 392  0326              ;
 393  0326  48        FDIO   PHA                SAVE ACC            ** ENTRY **
 394  0327  98               TYA                SAVE Y
 395  0328  48               PHA                 REGISTER
 396  0329  8A               TXA                SAVE X
 397  032A  48               PHA                 REGISTER
 398  032B  A9 05            LDA   #5           SET ERROR
 399  032D  85 E1            STA   ERRCDE        COUNT
 400  032F  A9 34            LDA   #$34         START W/DVC 0
 401  0331  24 E0            BIT   DVCODE       IF NOT 0
 402  0333  50 02            BVC   SETDVC        SET TO ONE
 403  0335  09 08            ORA   #$08         SET DVC 1
 404  0337  8D 0F CC  SETDVC STA   CRB          SET PIA
 405  033A  A5 E7     RETRY  LDA   FDBUF+1      SAVE ADDR HIGH
 406  033C  48               PHA                 FOR RECOVERY
 407  033D  A5 E5            LDA   SECTOR       SAVE SECTOR
 408  033F  48               PHA                 FOR RECOVERY
 409  0340  20 25 02         JSR   FDENT        EXEC CMD
```

Listing 1 continued on page 320

Listing 1 continued:

```
 410  0343  38              SEC             ASSUME ERROR
 411  0344                  ;
 412  0344                  ;  ******* CHECK FOR BUSY/NOT READY
 413  0344                  ;
 414  0344  A9 01           LDA   #$01      CHECK
 415  0346  24 E3           BIT   STATUS     FOR
 416  0348  D0 3F           BNE   ER1         BUSY OR
 417  034A  30 3D           BMI   ER1          NOT READY
 418  034C                  ;
 419  034C                  ;  ******* DETERMINE CMD TYPE
 420  034C                  ;
 421  034C  A9 10           LDA   #$10      CMD MASK
 422  034E  24 E2           BIT   COMAND    SPLIT INTO
 423  0350  10 19           BPL   TYP1       TYPE 1


   FLOPPY DISK I/O & ERROR RECOVERY                                  PAGE   10

CARD # LOC   CODE       CARD  10        20        30        40        50        60        70
 424  0352  50 29           BVC   TYP2        TYPE 2
 425  0354  F0 37           BEQ   RDT         TYPE 3 READ
 426  0356  A9 20           LDA   #$20       SEPERATE
 427  0358  24 E2           BIT   COMAND     FORCE INTRQ FROM
 428  035A  D0 27           BNE   WRT         TYPE 3 WRITE
 429  035C                  ;
 430  035C                  ;  ******* RETURN
 431  035C                  ;
 432  035C  18         RTN1  CLC             NO ERROR
 433  035D  A9 00           LDA   #0        CLEAR
 434  035F  85 E1      RTN2  STA   ERRCDE     ERROR CODE
 435  0361  68               PLA            CLEAR STACK
 436  0362  85 E5           STA   SECTOR     OF SECTOR
 437  0364  68               PLA              AND ADDR HIGH
 438  0365  68         RTN3  PLA             RESTORE X
 439  0366  AA               TAX             REGISTER
 440  0367  68               PLA            RESTORE Y
 441  0368  A8               TAY             REGISTER
 442  0369  68               PLA            RESTORE ACC
 443  036A  60               RTS            RETURN
 444  036B                  ;
 445  036B                  ;  ******* TYPE 1 RECOVERY
 446  036B                  ;
 447  036B  A9 18      TYP1  LDA   #$18      CHECK FOR
 448  036D  25 E3           AND   STATUS     BOTH CRC AND
 449  036F  F0 EB           BEQ   RTN1
 450  0371  C9 18           CMP   #$18        NOT FOUND
 451  0373  F0 14           BEQ   ER1          ERRORS
 452  0375  A9 30           LDA   #$30      STOP IF
 453  0377  24 E2           BIT   COMAND     STEP IN
 454  0379  D0 0E           BNE   ER1         OR STEP OUT
 455  037B  F0 26           BEQ   RDT1      RETRY SEEK AND RESTORE
 456  037D                  ;
 457  037D                  ;  ******* TYPE 2 SEPERATION
 458  037D                  ;
```

Listing 1 continued on page 322

Listing 1 continued:

```
 459  037D  A9 20     TYP2   LDA   #$20        SEPERATE
 460  037F  24 E2            BIT   COMAND       READ
 461  0381  F0 0A            BEQ   RDT          FROM WRITE
 462  0383            ;
 463  0383            ;  ******* WRITE RECOVERY
 464  0383            ;
 465  0383  A9 60     WRT    LDA   #$60        ERROR MASK
 466  0385  24 E3            BIT   STATUS      STOP IF WRITE
 467  0387  F0 04            BEQ   RDT          PROTECT/FAULT
 468  0389  A9 FF     ER1    LDA   #$FF        SET ERROR CODE
 469  038B  D0 D2            BNE   RTN2        RETURN
 470  038D            ;
 471  038D            ;  ******* COMMON RECOVERY
 472  038D            ;
 473  038D  A9 0C     RDT    LDA   #$0C        ERROR MASK
 474  038F  24 E3            BIT   STATUS      IF ERROR
 475  0391  D0 10            BNE   RDT1         RETRY
 476  0393  A9 10            LDA   #$10        CHECK FOR
 477  0395  24 E3            BIT   STATUS       NOT FND
 478  0397  F0 C3            BEQ   RTN1          NONE RETURN
```



```
CARD # LOC   CODE      CARD  10        20        30        40        50        60        70
 479  0399            ;                       IF MULTIPLE
 480  0399  24 E2            BIT   COMAND       SECTOR OPERATION
 481  039B  F0 06            BEQ   RDT1          CHECK
 482  039D  A9 1B            LDA   #$1B           FOR END OF
 483  039F  C5 E5            CMP   SECTOR          TRACK
 484  03A1  F0 B9            BEQ   RTN1        CALL IT NORMAL
 485  03A3            ;
 486  03A3            ;  ******* CHECK ERROR COUNT
 487  03A3            ;
 488  03A3  C6 E1     RDT1   DEC   ERRCDE      DECR ERROR CNT
 489  03A5  10 05            BPL   RDT2        RETURN
 490  03A7  68               PLA                WITH
 491  03A8  68               PLA                 ERROR
 492  03A9  4C 65 03         JMP   RTN3           CONDITION
 493  03AC            ;
 494  03AC            ;  ******* RETRY OPERATION
 495  03AC            ;
 496  03AC  68       RDT2    PLA               RESTORE
 497  03AD  85 E5            STA   SECTOR       SECTOR
 498  03AF  68               PLA               RESTORE
 499  03B0  85 E7            STA   FDBUF+1      ADDR HIGH
 500  03B2  4C 3A 03         JMP   RETRY       RETRY
```

Listing 1 continued on page 324

Listing 1 continued:

```
   FD400/FD1771B FLOPPY DISK FORMAT                                    PAGE   12

CARD # LOC      CODE        CARD  10        20        30        40        50        60        70
   502  03B5                ;
   503  03B5                ;
   504  03B5                ;   THIS SEGMENT FORMATS AN ENTIRE FLOPPY DISKETTE IN IBM COMPATIBLE
   505  03B5                ;  SOFT SECTORING WITH 128 BYTE SECTORS. EACH SECTOR CONTAINS 80
   506  03B5                ;  BYTES OF ASCII BLANK(X'20') FOLLOWED BY HEX ZEROS FOR THE
   507  03B5                ;  REMAINDER OF THE SECTOR.
   508  03B5                ;
   509  03B5                ;
   510  03B5                ;
   511  03B5                ;  ******* INITIALIZE
   512  03B5                ;
   513  03B5  20 F6 02      FORMAT JSR   FDINT       INIT SYSTEM          ** ENTRY **
   514  03B8  A2 00                LDX   #0          A SIDE
   515  03BA  8E 0D CC             STX   CRA          DIRECTION
   516  03BD  CA                   DEX               SET TO
   517  03BE  8E 0C CC             STX   SADD         OUTPUT
   518  03C1  A2 2C                LDX   #$2C        A SIDE
   519  03C3  8E 0D CC             STX   CRA          DATA
   520  03C6                ;
   521  03C6                ;  ******* SET UP RECORD
   522  03C6                ;
   523  03C6  A9 4C                LDA   #$4C        SET
   524  03C8  8D 00 05             STA   REND         TRACK COUNT
   525  03CB  A9 FF                LDA   #$FF        SET TRK
   526  03CD  8D B3 05             STA   RTN          TO ZERO
   527  03D0  A9 FE           GO   LDA   #$FE        SET SECTOR
   528  03D2  8D B1 05             STA   RSN          TO ONE
   529  03D5  A2 1A                LDX   #$1A        SECTOR CNT
   530  03D7  A0 FD                LDY   #RSTRT-REND WRITE LENGTH
   531  03D9                ;
   532  03D9                ;  ******* ISSUE WRITE TRACK
   533  03D9                ;
   534  03D9  A9 0B                LDA   #255-FDWTT STOR FD1771B
   535  03DB  8D 0C CC             STA   SAD          COMMAND
   536  03DE  A9 19                LDA   #WRITE+CMD STORE PIA
   537  03E0  8D 0E CC             STA   SBD          COMMAND
   538  03E3  CE 0E CC             DEC   SBD         ENABLE
   539  03E6  EE 0E CC             INC   SBD          READ/WRITE
   540  03E9  A9 1F                LDA   #WRITE+DATA STORE PIA
   541  03EB  8D 0E CC             STA   SBD          COMMAND
   542  03EE                ;
   543  03EE                ;  ******* RECORD TRANSFER
   544  03EE                ;
   545  03EE  EE FE 05      WDT    INC   RSTRT+1     DELAY 6 CYCLES
   546  03F1  B9 00 05             LDA   REND,Y      STORE A
   547  03F4  8D 0C CC             STA   SAD          DATA BYTE
   548  03F7  2C 0E CC      WLP    BIT   SBD         WAIT FOR
   549  03FA  30 12                BMI   NEXT         INTRQ
   550  03FC  50 F9                BVC   WLP           OR DRQ
   551  03FE  88                   DEY               DECR INDEX
   552  03FF  D0 ED                BNE   WDT         CONTINUE
   553  0401  CE B1 05             DEC   RSN         INC SECTOR
   554  0404  A0 BA                LDY   #RNORM-REND INDEX VALUE
   555  0406  CA                   DEX               DECR SECTOR CNT
   556  0407  D0 E8                BNE   WDT+3       CONTINUE


   FD400/FD1771B FLOPPY DISK FORMAT                                    PAGE   13

CARD # LOC      CODE        CARD  10        20        30        40        50        60        70
   557  0409                ;
   558  0409                ;  ******* WAIT FOR COMPLETION
   559  0409                ;
   560  0409  2C 0E CC      TRKEND BIT   SBD         WAIT FOR
   561  040C  10 FB                BPL   TRKEND       INTRQ
   562  040E                ;
   563  040E                ;  ******* MOVE TO NEXT TRACK
   564  040E                ;
   565  040E  20 35 04      NEXT   JSR   DELAY       DELAY 40 MS.
   566  0411  A9 B5                LDA   #255-FDSTI-QH STORE FD1771B
   567  0413  8D 0C CC             STA   SAD          COMMAND
   568  0416  A9 19                LDA   #WRITE+CMD STORE PIA
```

Listing 1 continued on page 326

Listing 1 continued:

```
  569   0418  8D 0E CC          STA   SBD          COMMAND
  570   041B  CE 0E CC          DEC   SBD         ENABLE
  571   041E  EE 0E CC          INC   SBD          READ/WRITE
  572   0421  2C 0E CC  SLP     BIT   SBD         WAIT FOR
  573   0424  10 FB             BPL   SLP          INTRQ
  574   0426  20 35 04          JSR   DELAY       DELAY 40 MS.
  575   0429  CE B3 05          DEC   RTN         INCR TRACK
  576   042C  CE 00 05          DEC   REND        DEC TRK CNT
  577   042F  10 9F             BPL   GO          CONTINUE
  578   0431  20 F6 02          JSR   FDINT       RESTORE DRIVE
  579   0434  60                RTS               STOP
  580   0435                ;
  581   0435                ;  ******* DELAY 40 MS.
  582   0435                ;
  583   0435  A9 40       DELAY  LDA   #$40        MAJOR LOOP VALUE
  584   0437  85 00              STA   TIME1       MAJOR LOOP CNT
  585   0439  A9 4A       DL2    LDA   #$4A        MINOR LOOP VALUE
  586   043B  85 01              STA   TIME2       MINOR LOOP CNT
  587   043D  C6 01       DL1    DEC   TIME2       DECR MINOR CNT
  588   043F  D0 FC              BNE   DL1         CONTINUE
  589   0441  C6 00              DEC   TIME1       DECR MAJOR CNT
  590   0443  D0 F4              BNE   DL2         CONTINUE
  591   0445  60                RTS               RETURN
```

```
   FD400/FD1771B FLOPPY DISK FORMAT                                    PAGE   14

CARD #  LOC     CODE        CARD  10        20        30        40        50        60        70
  593   0446                    *=*+255/256*256
  594   0500                ;
  595   0500                ;  ******* RECORD FORMAT
  596   0500                ;           (REVERSED AND INVERTED)
  597   0500  00          REND    .BYTE $00
  598   0501  00                  .BYTE $00,$00,$00,$00,$00,$00,$00,$00
  598   0502  00
  598   0503  00
  598   0504  00
  598   0505  00
  598   0506  00
  598   0507  00
  598   0508  00
  599   0509  00                  .BYTE $00,$00,$00,$00,$00,$00,$00,$00
  599   050A  00
  599   050B  00
  599   050C  00
  599   050D  00
  599   050E  00
```

Listing 1 continued on page 328

Listing 1 continued:

```
599  050F  00
599  0510  00
600  0511  00              .BYTE $00,$00,$00,$00,$00,$00,$00,$00
600  0512. 00
600  0513  00
600  0514  00
600  0515  00
600  0516  00
600  0517  00
600  0518  00
601  0519  00              .BYTE $00,$00,$00
601  051A  00
601  051B  00
602  051C  08              .BYTE $08                                    DATA CRC
603  051D  FF              .BYTE $FF,$FF,$FF,$FF,$FF,$FF,$FF,$FF
603  051E  FF
603  051F  FF
603  0520  FF
603  0521  FF
603  0522  FF
603  0523  FF
603  0524  FF
604  0525  FF              .BYTE $FF,$FF,$FF,$FF,$FF,$FF,$FF,$FF
604  0526  FF
604  0527  FF
604  0528  FF
604  0529  FF
604  052A  FF
604  052B  FF
604  052C  FF
605  052D  FF              .BYTE $FF,$FF,$FF,$FF,$FF,$FF,$FF,$FF
605  052E  FF
605  052F  FF
605  0530  FF
605  0531  FF
605  0532  FF
```

Listing 1 continued on page 330

Listing 1 continued:

```
   FD400/FD1771B FLOPPY DISK FORMAT                                    PAGE   15

CARD # LOC     CODE       CARD  10        20        30        40        50        60        70
  605  0533  FF
  605  0534  FF
  606  0535  FF               .BYTE $FF,$FF,$FF,$FF,$FF,$FF,$FF,$FF
  606  0536  FF
  606  0537  FF
  606  0538  FF
  606  0539  FF
  606  053A  FF
  606  053B  FF
  606  053C  FF
  607  053D  FF               .BYTE $FF,$FF,$FF,$FF,$FF,$FF,$FF,$FF
  607  053E  FF
  607  053F  FF
  607  0540  FF
  607  0541  FF
  607  0542  FF
  607  0543  FF
  607  0544  FF
  608  0545  FF               .BYTE $FF,$FF,$FF,$FF,$FF,$FF,$FF,$FF
  608  0546  FF
  608  0547  FF
  608  0548  FF
  608  0549  FF
  608  054A  FF
  608  054B  FF
  608  054C  FF
  609  054D  DF               .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
  609  054E  DF
  609  054F  DF
  609  0550  DF
  609  0551  DF
  609  0552  DF
  609  0553  DF
  609  0554  DF
  610  0555  DF               .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
  610  0556  DF
  610  0557  DF
  610  0558  DF
  610  0559  DF
  610  055A  DF
  610  055B  DF
  610  055C  DF
  611  055D  DF               .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
  611  055E  DF
  611  055F  DF
  611  0560  DF
  611  0561  DF
  611  0562  DF
  611  0563  DF
  611  0564  DF
  612  0565  DF               .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
  612  0566  DF
  612  0567  DF
  612  0568  DF
  612  0569  DF

   FD400/FD1771B FLOPPY DISK FORMAT                                    PAGE   16

CARD # LOC     CODE       CARD  10        20        30        40        50        60        70
  612  056A  DF
  612  056B  DF
  612  056C  DF
  613  056D  DF               .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
  613  056E  DF
  613  056F  DF
  613  0570  DF
  613  0571  DF
  613  0572  DF
  613  0573  DF
  613  0574  DF
```

Listing 1 continued on page 332

Listing 1 continued:

```
614  0575  DF              .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
614  0576  DF
614  0577  DF
614  0578  DF
614  0579  DF
614  057A  DF
614  057B  DF
614  057C  DF
615  057D  DF              .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
615  057E  DF
615  057F  DF
615  0580  DF
615  0581  DF
615  0582  DF
615  0583  DF
615  0584  DF
616  0585  DF              .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
616  0586  DF
616  0587  DF
616  0588  DF
616  0589  DF
616  058A  DF
616  058B  DF
616  058C  DF
617  058D  DF              .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
617  058E  DF
617  058F  DF
617  0590  DF
617  0591  DF
617  0592  DF
617  0593  DF
617  0594  DF
618  0595  DF              .BYTE $DF,$DF,$DF,$DF,$DF,$DF,$DF,$DF
618  0596  DF
618  0597  DF
618  0598  DF
618  0599  DF
618  059A  DF
618  059B  DF
618  059C  DF
619  059D  04              .BYTE $04                                 DATA AM
620  059E  FF              .BYTE $FF,$FF,$FF,$FF,$FF                 DATA FLD SYNC
620  059F  FF
620  05A0  FF
```



```
CARD #  LOC   CODE      CARD  10        20        30        40        50        60        70
620  05A1  FF
620  05A2  FF
620  05A3  FF
621  05A4  00              .BYTE $00,$00,$00,$00,$00                 DATA
621  05A5  00
621  05A6  00
621  05A7  00
621  05A8  00
621  05A9  00
622  05AA  00              .BYTE $00,$00,$00,$00,$00                  GAP
622  05AB  00
622  05AC  00
622  05AD  00
622  05AE  00
623  05AF  08              .BYTE $08                                 ADDR CRC
624  05B0  FF        RSL   .BYTE $FF                                 SECTOR LENGTH
625  05B1  FE        RSN   .BYTE $FE                                 SECTOR ADDR
626  05B2  FF              .BYTE $FF                                 ZERO
627  05B3  FF        RTN   .BYTE $FF                                 TRACK ADDR
628  05B4  01              .BYTE $01                                 ID AM
629  05B5  FF              .BYTE $FF,$FF,$FF,$FF,$FF                 ADDR SYNC
629  05B6  FF
629  05B7  FF
629  05B8  FF
629  05B9  FF
```

Listing 1 continued on page 334

Listing 1 continued:

```
  629  05BA  FF
  630  05BB            RNORM  =*-1
  631  05BB  00               .BYTE $00,$00,$00,$00,$00,$00,$00,$00
  631  05BC  00
  631  05BD  00
  631  05BE  00
  631  05BF  00
  631  05C0  00
  631  05C1  00
  631  05C2  00
  632  05C3  00               .BYTE $00,$00,$00,$00,$00,$00,$00,$00
  632  05C4  00
  632  05C5  00
  632  05C6  00
  632  05C7  00
  632  05C8  00
  632  05C9  00
  632  05CA  00
  633  05CB  00               .BYTE $00,$00,$00,$00,$00,$00,$00,$00
  633  05CC  00
  633  05CD  00
  633  05CE  00
  633  05CF  00
  633  05D0  00
  633  05D1  00
  633  05D2  00
  634  05D3  00               .BYTE $00,$00
  634  05D4  00
  635  05D5  03               .BYTE $03                                    INDEX MARK
  636  05D6  00               .BYTE $00,$00,$00,$00,$00,$00,$00,$00
```



```
CARD #  LOC   CODE      CARD  10        20        30        40        50        60        70
  636  05D7  00
  636  05D8  00
  636  05D9  00
  636  05DA  00
  636  05DB  00
  636  05DC  00
  636  05DD  00
  637  05DE  00               .BYTE $00,$00,$00,$00,$00,$00,$00,$00
  637  05DF  00
  637  05E0  00
  637  05E1  00
  637  05E2  00
  637  05E3  00
  637  05E4  00
  637  05E5  00
  638  05E6  00               .BYTE $00,$00,$00,$00,$00,$00,$00,$00
  638  05E7  00
  638  05E8  00
  638  05E9  00
  638  05EA  00
  638  05EB  00
  638  05EC  00
  638  05ED  00
  639  05EE  00               .BYTE $00,$00,$00,$00,$00,$00,$00,$00
  639  05EF  00
  639  05F0  00
  639  05F1  00
  639  05F2  00
  639  05F3  00
  639  05F4  00
  639  05F5  00
  640  05F6  00               .BYTE $00,$00,$00,$00,$00,$00,$00,$00
  640  05F7  00
  640  05F8  00
  640  05F9  00
  640  05FA  00
  640  05FB  00
  640  05FC  00
  640  05FD  00
  641  05FE            RSTRT  =*-1
  642  05FE                   .END
```

Listing 1 continued on page 336

Listing 1 continued:

```
END OF MOS/TECHNOLOGY 650X ASSEMBLY VERSION 5
NUMBER OF ERRORS =   0,  NUMBER OF WARNINGS =   0


        SYMBOL TABLE

SYMBOL    VALUE  LINE DEFINED        CROSS-REFERENCES

BASIC     0240         233   197  199  210  362  365
CMD       0000         175   312  536  568
CMPANL    0248         239   204  218  282  302
COMAND    00E2         184   195  223  228  257  260  263  272  314  361  422
                         427  453  460  480
CPLP      024A         240   247
CRA       CC0D          96   338  341  515  519
CRB       CC0F          99   350  355  359  364  404
DATA      0006         174   200  279  296  540
DELAY     0435         583   565  574
DL1       043D         587   588
DL2       0439         585   590
DVCODE    00E0         182   351  401
ERRCDE    00E1         183   399  434  488
ER1       0389         468   416  417  451  454
FDBUF     00E6         188   286  289  298  306  405  499
FDENT     0225         214 . 261  409
FDFI      00D0         137  ****
FDINT     02F6         346   513  578
FDIO      0326         393  ****
FDRD      0080         132  ****
FDRDA     00C4         134  ****
FDRDT     00E4         135  ****
FDRST     0002         127   360
FDSK      0012         128   259
FDST      0022         129  ****
FDSTI     0042         130   566
FDSTO     0062         131  ****
FDWT      00A0         133  ****
FDWTT     00F4         136   534
FORMAT    03B5         513  ****
GO        03D0         527   577
NEXT      040E         565   549
PULSE     02CD         321   205  209  216  244  254  270  315
QAFA      00FA         163  ****
QAFB      00FB         162  ****
QAF8      00F8         165  ****
QAF9      00F9         164  ****
QB        0008         145  ****
QCRC      00F7         159  ****
QE        0004         151  ****
QFA       0001         153  ****
QFB       0000         152  ****
QF8       0011         155  ****
QF9       0010         154  ****
QH        0008         142   566
QIAM      00FC         160  ****
QIDM      00FE         161  ****
QI0       0001         147  ****
QI1       0002         148  ****
QI2       0004         149  ****
QI3       0008         150  ****
QM        0010         144  ****
QS        0001         146  ****


SYMBOL   VALUE  LINE DEFINED        CROSS-REFERENCES

QU        0010         143  ****
QV        0004         141  ****
RDATA     0284         277   226
RDL       028E         281   283  288  290
RDT       038D         473   425  461  467
RDT1      03A3         488   455  475  481
RDT2      03AC         496   489
READ      0029         169   214  242  252  279
```

Listing 1 continued on page 338

Listing 1 continued:

```
REND      0500        597   524   546   576   530   554
RETRY     033A        405   500
RNORM     05BA        630   554
RSL       05B0        624   ****
RSN       05B1        625   528   553
RSTRT     05FD        641   530   545
RTN       05B3        627   526   575
RTN1      035C        432   449   478   484
RTN2      035F        434   469
RTN3      0365        438   492
SAD       CC0C         95   284   300   322   325   348   535   547   567
SADD      CC0C         94   339   517
SBD       CC0E         98   234   281   301   323   324   332   353   537   538   539
                            541   548   560   569   570   571   572
SBDD      CC0E         97   357
SECT      0004        173   206   267
SECTOR    00E5        187   208   269   407   436   483   497
SETDVC    0337        404   402
SETUP     02DE        331   201   207   215   243   253   268   280   297   313
SET1      02E8        337   335   347
SLP       0421        572   573
STAT      0000        171   214
STATUS    00E3        185   245   415   448   466   474   477
TIME1     0000        179   584   589
TIME2     0001        180   586   587
TRACK     00E4        186   202   255
TRK       0002        172   252
TRKEND    0409        560   561
TYPE1     0200        195   224
TYPE2     025B        252   225
TYPE2A    0274        267   256
TYP1      036B        447   423
TYP2      037D        459   424
WDATA     02A3        294   229   273
WDT       03EE        545   552   556
WLP       03F7        548   550
WRITE     0019        170   200   206   267   296   312   536   540   568
WRT       0383        465   428
WRTCMD    02C2        312   233   277   294
WTL       02AD        298   305   307
WTL1      02B4        301   303
```

```
INSTRUCTION  COUNT

    ADC          0
    AND          1
    ASL          3
    BCC          1
    BCS          3
    BEQ         11
    BIT         19
    BMI          5
    BNE         15
    BPL          8
    BRK          0
    BVC          6
    BVS          0
    CLC          1
    CLD          0
    CLI          0
    CLV          0
    CMP          6
    CPX          0
    CPY          0
    DEC          9
    DEX          4
    DEY          2
    EOR          4
    INC          6
    INX          1
    INY          2
    JMP          4
    JSR         27
    LDA         52
    LDX          8
    LDY          6
```

Listing 1 continued on page 340

Listing 1 continued:

```
        LSR          0
        NOP          0
        ORA          2
        PHA          6
        PHP          0
        PLA         10
        PLP          0
        ROL          0
        ROR          1
        RTI          0
        RTS          7
        SBC          0
        SEC          1
        SED          0
        SEI          0
        STA         27
        STX         10
        STY          1
        TAX          1
        TAY          1
        TSX          0
        TXA          1
        TXS          0
        TYA          2

 # SYMBOLS =  101 (LIMIT =  800)     # BYTES =  837 (LIMIT = 8192)


# LINES =  853 (LIMIT = 3000)     # XREFS =  257 (LIMIT = 1600)
```

Listing 2: *Example of a routine that reads disk track 3 into memory, starting at location hexadecimal 1000. This routine also illustrates the use of the ERRCDE variable.*

```
JSR FDINT          Initialize
LDA #$9C           Read multiple
STA COMMAND            sector command
LDA #$03           Request track
STA TRACK              number 3
LDA #0             Set buffer
STA FDBUF              address
LDA #$10                   at
STA FDBUF + 1                  hexadecimal 1000
JSR FDIO           Do I/O
LDA STATUS             check for
BNE ERROR                  error
    .
    .
    .
```

Listing 3: *Simple testing program for a disk controller/6502 microprocessor combination. When the BRK (break) occurs, the variables listed in table 6 can be set to test the various controller functions.*

```
INIT    JSR FDINT
        BRK
        BRK
GO      JSR FDENT
        BRK
        BRK
        JMP GO
```

Text continued from page 304:

1000. The status byte indicates if the read operation was successful. If the read test appears good, various other commands should be attempted to increase your familiarity with the 1771 and drive operation.

Extensions

With the addition of an external multiplexing circuit to switch the floppy-disk control lines, multiple drives can be controlled. Multiple drives, however, add a new software-control problem. Since the 1771 retains the current head location, it is necessary to update the track register when switching between drives. A memory variable to contain the head location of each drive can be used to adjust the 1771's register.

A simplified version of the floppy-disk controller can be used to operate 5-inch disk drives in either single- or dual-density. In addition, this disk design is extensible to a more elaborate controller that uses a dedicated 6502 to communicate over a parallel or serial interface to a host computer.

Conclusion

Floppy-disk drives provide sufficient capacity and performance to meet the needs of most microcomputer users. By combining hardware and software, a floppy-disk system can be constructed economically without sacrificing any function or performance. The 6502 microprocessor, with a few hundred bytes of program, can control head movement and data transfer by utilizing the 1771 controller. The software provides a flexible, yet economic, solution to mass-storage problems. ■