PERTEC Application Notes

Soft-Sector Formatting
For
PERTEC Flexible Disk Drives

FOREWORD

This Application Note provides the reader with a method of configuring a soft-sectored format for the PERTEC Flexible Disk Drives. The method is based on the IBM 3740 compatible format. The parameters which contribute to sector format considerations are described and details of the IBM 3740 format are given to serve as a guideline. The recommended PERTEC phase-lock-loop technique is presented for use in data/clock bit separation and address-mark (missing-clock situation) detection.

TABLE OF CONTENTS

NOTES

I. INTRODUCTION

The PERTEC Flexible Disk Drive is a compact random-access memory device suitable for use in data processing applications. The recording medium employed is a circular mylar disk referred to as a diskette.

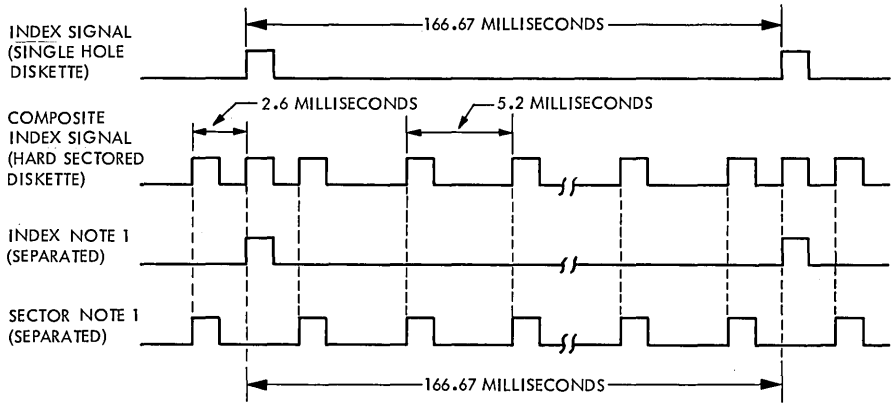
The standard diskette is designed for use with a format in which the sector information is pre-recorded. In this case, a single hole on the diskette serves as a reference point. The detection of this hole is accomplished by a transducer which is made up of a phototransistor/LED combination. The transducer produces an index pulse once per revolution of the diskette.

A diskette that has equally spaced fixed sector holes on the same radius as the index hole is referred to as being *hard sectored*. Sector timing is accomplished by sensing these holes. This configuration is shown in Figure 1 along with a timing diagram which compares the hard-sectored diskette with 32 sector holes and the single-hole diskette.

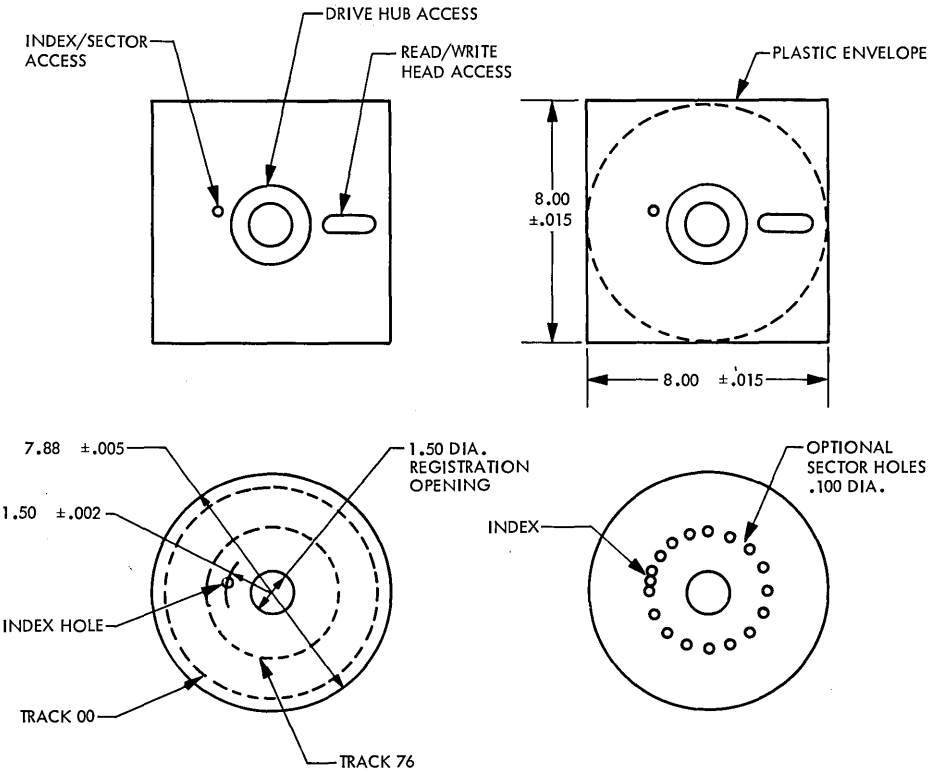
The format in which sector information is pre-recorded on the diskette is referred to as the *soft sectored* format. An address-mark method is normally used to pre-record the sector information. This method is described in detail in Section III of this document. Section II provides some background information on the double-frequency encoding scheme, the phase-lock-loop technique for the address-mark (missing-clock situation) detection, and data/clock separation. Section IV provides guidelines to aid the reader in the design of a typical soft-sectored formatter.

Appendix A provides schematics and a general circuit description of the PERTEC phase-lock-loop data separator which is recommended for use with a soft-sectored formatter. Appendix B provides the content of data fields in each sector on track zero.

All format data contained in this document are based on the IBM 3740 compatible format and are intended to aid the user in configuring a soft-sectored format for his specific applications.



NOTE: 1. USER SYSTEM SEPARATES INDEX AND SECTOR PULSES.
2. ALL TIME VALUES NOMINAL.



NOTE: ALL DIMENSIONS IN INCHES.

Figure 1. Diskette Comparison

II. BACKGROUND

This section provides the user with a basic understanding of the data encoding scheme, and phase-lock-loop method for data/clock bit separation and address-mark detection.

The information stored on the diskette surface is organized in concentric tracks. Each track consists of a continuous string of sectors, each of which contains a group of bytes comprising one record of data. Data are recorded in a sector on a bit-serial basis. Bits of information to be stored are first encoded, then recorded in a specific sector. The user's system will, however, determine the particular data encoding scheme to be used for data storage, depending on the type of recording medium and the bandwidth limitations of the read channel.

The term *information*, as used in this document, refers to any sequence of flux transitions written on the diskette consistent with the data encoding scheme employed by the user. The unit of information is the bit; a group of eight bits comprises one byte.

2.1 DATA ENCODING

For clarification purposes, the terminology associated with data encoding is presented.

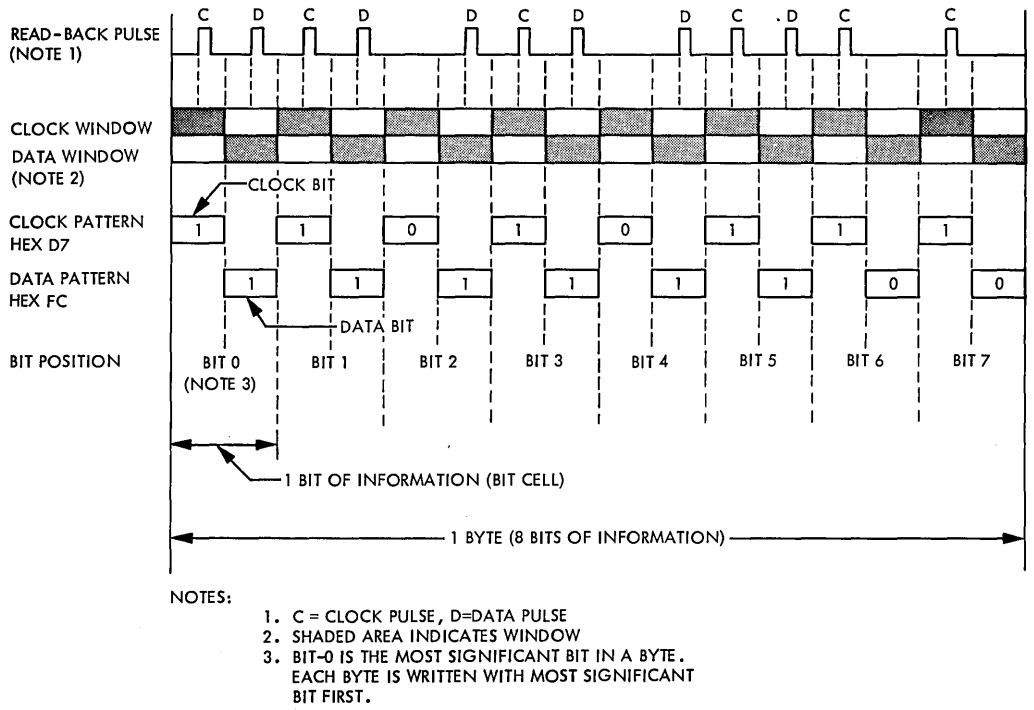
An information bit is considered to occupy a bit-cell which contains a data-bit preceded by a clock-bit. In the general case, each clock-bit and data-bit has a value of either one or zero; the presence of a magnetic flux transition (with its associated read-back pulse) represents a binary one and the absence of a transition (and read-back pulse) represents a binary zero.

Each byte is written starting with the most significant clock-bit first, then the most significant data-bit, and so on until the least significant data-bit is written. For convenience, the byte is defined by two hexadecimal numbers representing the eight clock-bit sequence and eight data-bit sequence, respectively. Figure 2 illustrates the general case of a byte of information consisting of a clock-bit pattern of hex D7 and a data-bit pattern of hex FC.

With the exception of special mark bytes, every byte of information utilizes the double-frequency encoding scheme in which a clock transition is provided for every encoded bit of information. In this case there are eight clock transitions per byte and the clock-bit pattern is hex FF. A mark byte is an encoded byte of information which has missing clock transitions in its clock-transition pattern. Therefore, a mark byte is specified by assigning a fixed data-bit pattern along with its associated clock-bit pattern (other than hex FF). This unique combination of data-bit and clock-bit pattern is not allowed in an address field, data field, or gap. Hence, mark bytes can be used as flags for the beginning of a track, address field, or data field, respectively referred to as an Index Mark, an Address Mark, or a Data Mark. The use of mark bytes in pre-recording sector information on a track is described in detail in Section III.

2.2 DATA DECODING

The disk drive provides a pulse (200 nanoseconds, nominal) for each flux transition recorded on the medium. The leading edge of these pulses represent the flux transitions recorded on the medium.



NOTES:

1. C = CLOCK PULSE, D=DATA PULSE
2. SHADED AREA INDICATES WINDOW
3. BIT-0 IS THE MOST SIGNIFICANT BIT IN A BYTE. EACH BYTE IS WRITTEN WITH MOST SIGNIFICANT BIT FIRST.

Figure 2. Double Frequency Encoding Scheme

Flux transitions recorded on the rotating magnetic medium undergo time distortion during playback; the amount of distortion becomes more pronounced at higher recording densities. This time distortion is commonly referred to as *peak-shift effect* because it has the tendency to shift the analog peaks in the read amplifier output.

Any decoding circuit designed to recover data recorded on the disk surface must take into account the speed variation of the disk drive and the peak-shift in the playback signal. Both of these parameters can adversely affect the decoder margins.

A decoder circuit must be designed to ensure that the contribution of the circuit to decoder tolerances is negligible compared to peak-shift and speed variation effects. The speed variation of the disk drive contributes to deviation of the data rate from its nominal value; however, the use of a phase-lock-loop circuit in decoding data significantly reduces the effect of speed variation on decoder tolerances.

The double-frequency code used in the disk drive possesses a self-clocking property in that it provides at least one flux transition (a pulse of 200 nanoseconds, nominal) for every bit cell. The self-clocking property is fundamental in the use of a phase-lock loop for data decoding.

The phase-lock loop produces a clock signal which is phase locked to the data, and whose frequency is equal to the maximum flux transition rate for the data encoding scheme

employed. The maximum flux transition rate for the double-frequency code is twice the bit data rate. PERTEC's Data Separator circuit utilizes a phase-lock loop to accomplish the data/clock-bit separation, and the address-mark detection. This phase-lock loop design is optimized to provide sufficient bandwidth to lock onto data within the preamble (6 bytes — 192 μseconds); however, it is not so wide as to track phase errors produced by the peak-shift effects.

Data decoding consists of separating the composite read data and clock waveform into data bits and clock bits. This is accomplished through data and clock windows and their associated clock waveforms provided by the phase-lock loop. Once the data and clock bits are separated, the detection of mark bytes, e.g., address mark, is achieved by comparing the data-bit and clock-bit patterns stored in the data and clock shift registers. The following paragraph discusses a possible logic implementation of the data encoding, data and clock bit separation, and address-mark detection scheme.

2.3 LOGIC IMPLEMENTATION

Functional logic block diagrams and their timing diagrams are presented to illustrate the implementation of data encoding and decoding. The 500 kHz oscillator shown in Figure 3 provides the Clock-Bit Write Clock and Data-Bit Write Clock which are used for encoding data and also serve as the system clock during a write operation.

Figure 4 is a simplified block diagram of the phase-lock loop used in data decoding. The Data Decoder Synchronizer circuit provides the Data Window, Clock Window, and their associated clock waveforms used in separating data and clock bits. The Phase Control circuit provides the adjustable time delay to obtain proper phase relationship between the Composite Read Data, and the Data Window and Clock Window waveforms.

The two multiplexers shown in Figure 5 provide the Data-Bit Clock and Clock-Bit Clock by multiplexing the write and read clocks, depending on the operation selected.

Figure 6 is the overall functional block diagram showing the logic implementation of data encoding, data/clock separation, and address-mark detection. A Data Shift Register and a Clock Shift Register are used for parallel-to-serial and serial-to-parallel conversion of data-bit and clock-bit patterns, respectively. During a Write operation (Write Enable true), byte parallel data from the controller are loaded into the Data Shift Register when its Load Control (Data-Bit-Pattern) input goes true. Similarly, clock-bit patterns (always hex FF except when writing four mark bytes) are loaded in parallel into the Clock Shift Register when its Load Control (Clock-Bit-Pattern) input goes true. Data-bit and clock-bit patterns are serialized and combined through a multiplexer to form the Composite Write Data and Clock Waveform.

Figures 7 and 8 are detailed timing diagrams for data encoding and data decoding, respectively. During a Read operation (Read Enable true), decoded data-bits and clock-bits are serially shifted into Data Shift Register and Clock Shift Register, respectively. When a data byte is assembled in the Data Shift Register, it is transferred to the controller in byte parallel form. Similarly, clock-bit patterns are assembled in the Clock Shift Register. Parallel outputs from the Data Shift Register and Clock Shift Register are also applied to the Address-Mark Detection Logic. As each data and clock byte are assembled, a comparison is made to detect data-bit and clock-bit patterns corresponding to different mark bytes. When a particular mark byte is detected, the indication is transmitted on the corresponding output to the control logic.

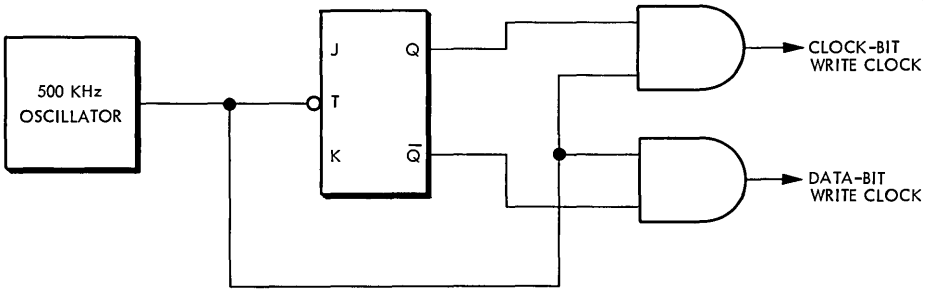


Figure 3. Data-Bit and Clock-Bit Write Clocks

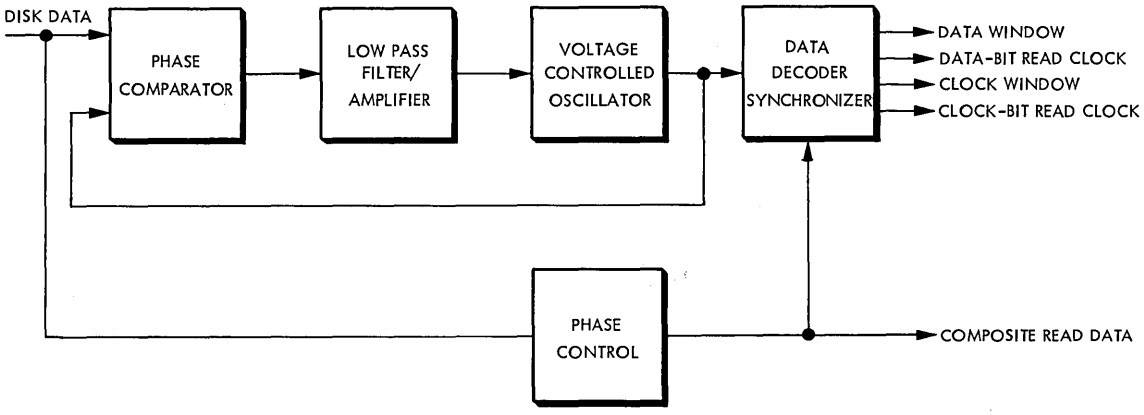


Figure 4. Phase-Lock Loop Block Diagram

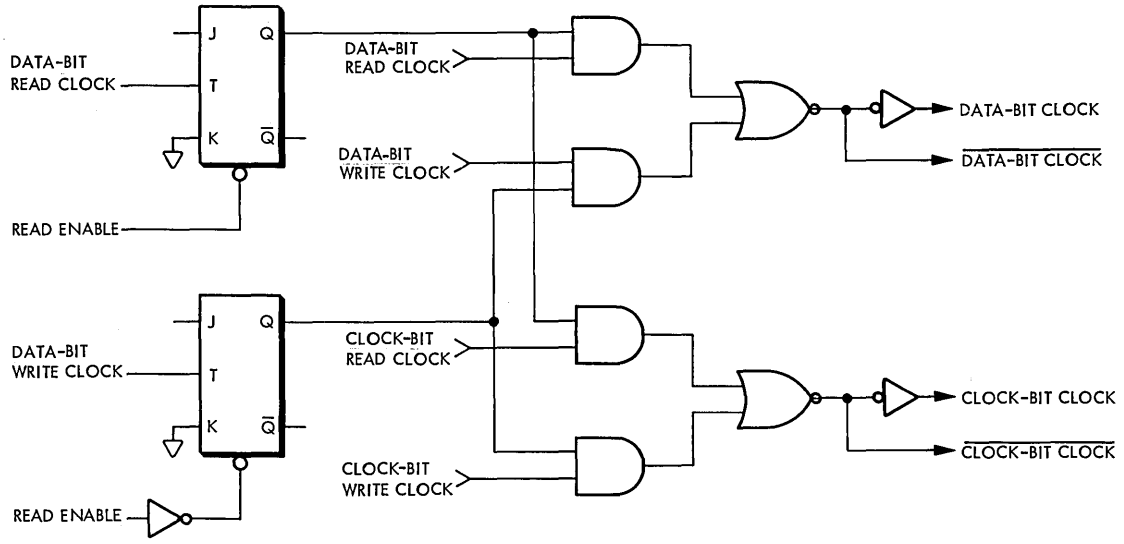


Figure 5. Read/Write Clock Multiplexer

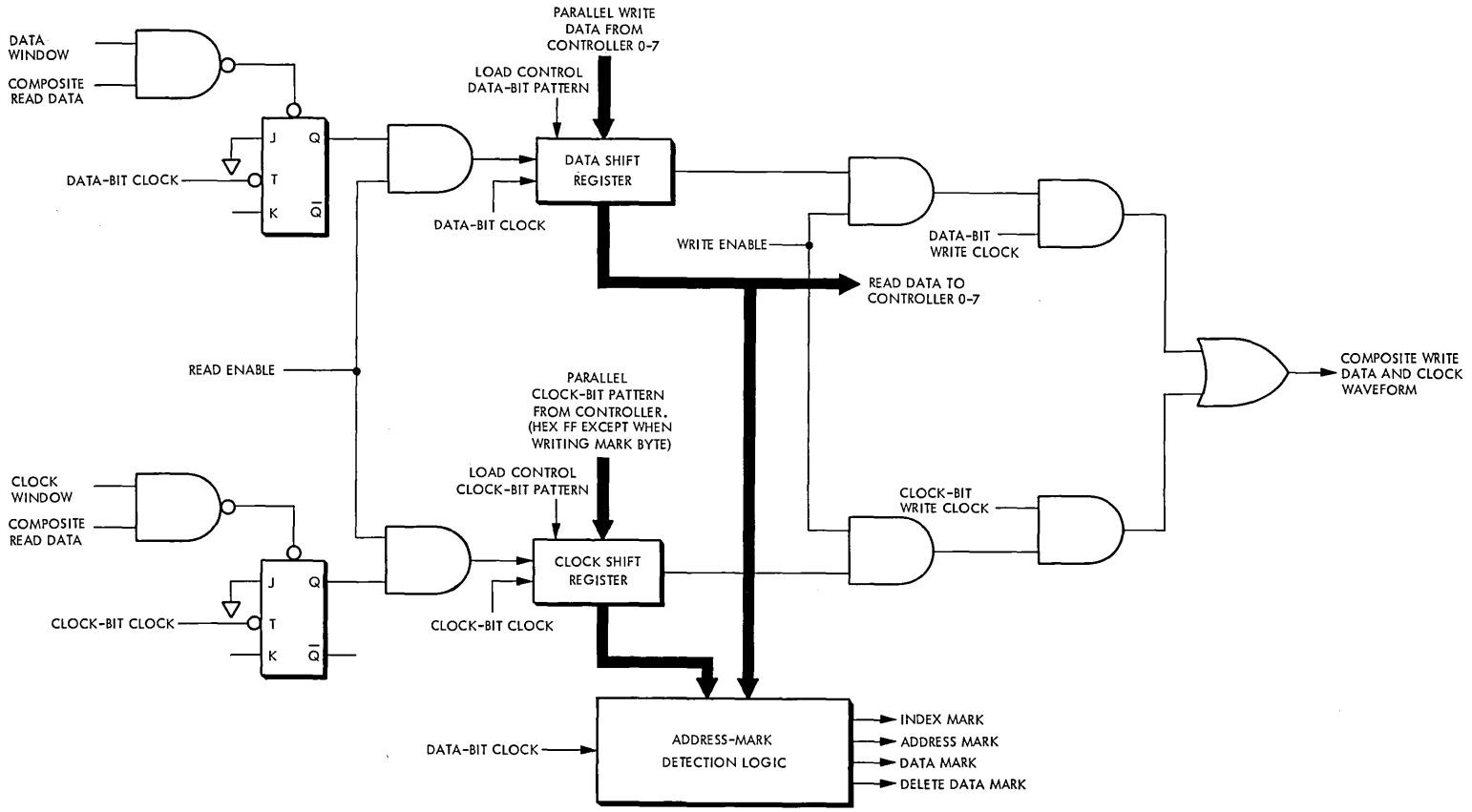


Figure 6. Logic Implementation Data Encoding, Data/Clock Bit Separation, Address-Mark Detection

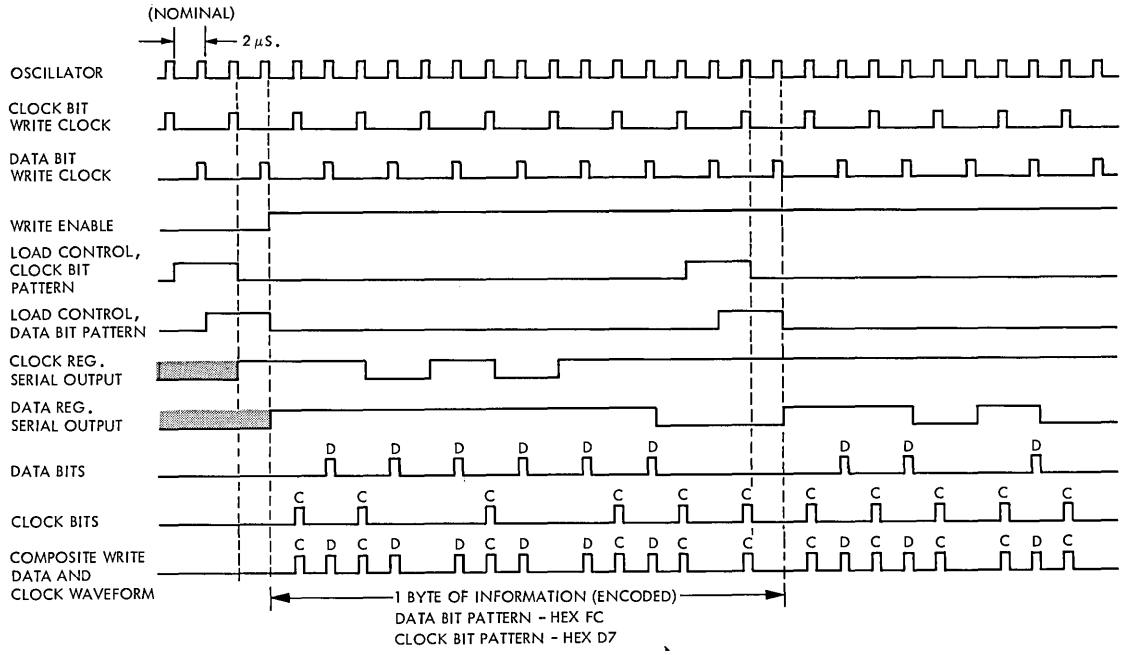


Figure 7. Double Frequency Encoding Scheme, Timing Diagram

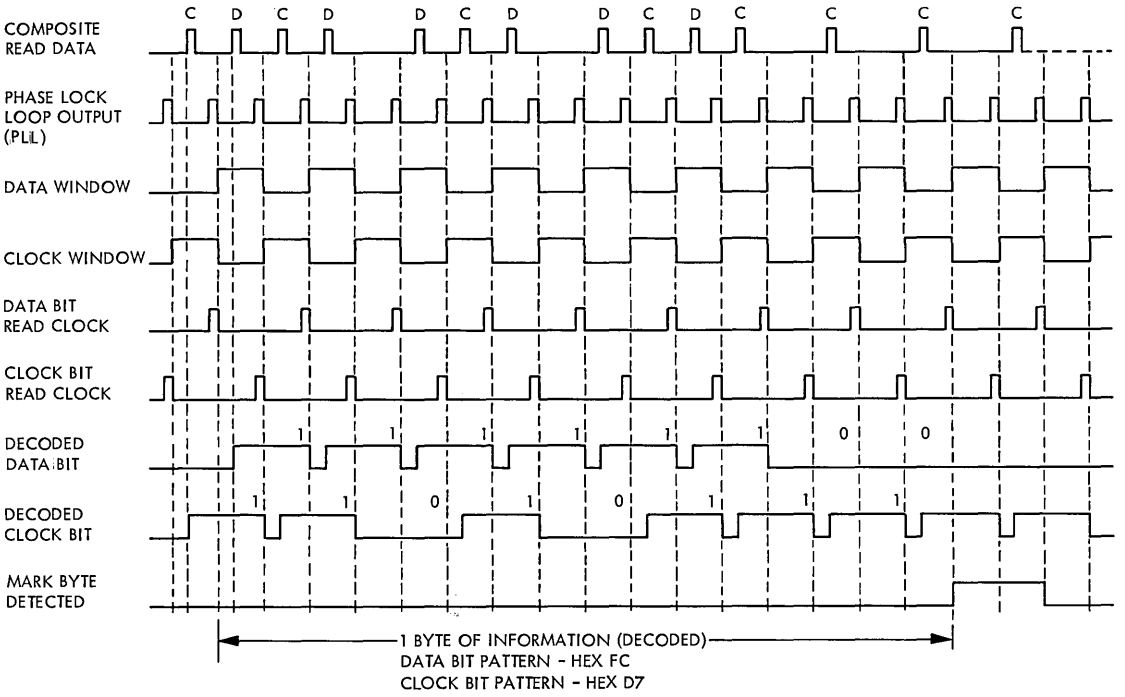


Figure 8. Phase-Lock Loop, Data/Clock Bit Separation Timing Diagram

III. TRACK FORMAT

This section describes a method of organizing diskette data into sectors by pre-recording the sector information on the track, using the IBM System 3740 diskette format as a guide.

The basic parameters are:

- Disk speed — 360 rpm
- Data packing density — 3268 bpi (innermost track)
- Sectors per track — 26
- Data bytes per sector — 128

Each diskette must be initialized before it can be used for data storage. The initialization operation consists of pre-recording sector information on the diskette. During this operation, each track is written on in one continuous operation from index to index. The leading edge of an index pulse is always used as the starting point for the initialize operation.

The following paragraphs describe the software and hardware aspects of the diskette format and their interaction with the user's system.

3.1 LOGICAL DATA FORMAT

The term *logical data format* refers to that aspect of diskette format which determines its software interaction with the user's system.

Starting from the outermost track, each diskette is divided into one index track (number 00), 73 data tracks (numbers 01 through 73), two alternate tracks (numbers 74 and 75), and one spare track (number 76). The spare track is not currently used in the IBM System 3740.

Each track is divided into 26 sectors. The logical record length of each sector can vary from 1 to 128 bytes.

The format used to initialize each track on the diskette is presented in Table 1. Each byte is written starting with the most significant clock bit first, then the most significant data bit, and so on until the least significant data bit is written. Bit position zero in each byte contains the most significant bit. It should be noted, however, that the data field for track 00 (Index Track) differs from the others. This difference is indicated by reference to Appendix B at specific points within Table 1.

3.2 PHYSICAL DATA FORMAT

The term *physical data format* refers to that aspect of the diskette format which determines its hardware interaction with the user's system.

The components of physical data format are shown in Figure 9. These components are: Mark Bytes (i.e., Index Mark, Address Mark, Data Mark), Address Field, Data Field, and Gap Fields. Mark bytes are used to signal the beginning of a track, address field, or data field. As previously described, mark bytes are unique combinations of data-bit and clock-bit patterns which are not allowed in address fields, data fields, or gaps. A gap is that portion of a track, filled with zeros and/or ones during the initialize operation, which

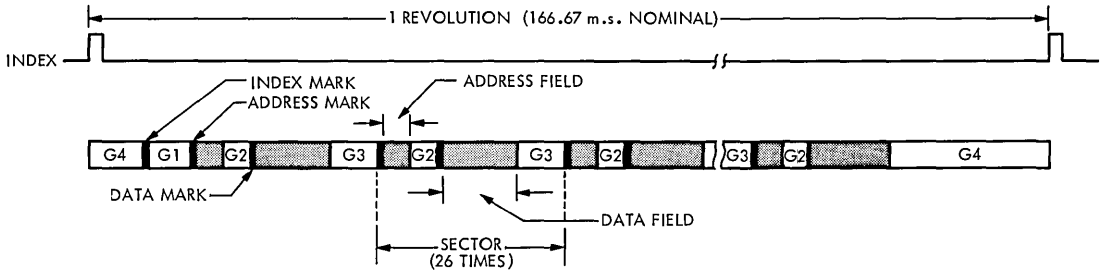
Table 1

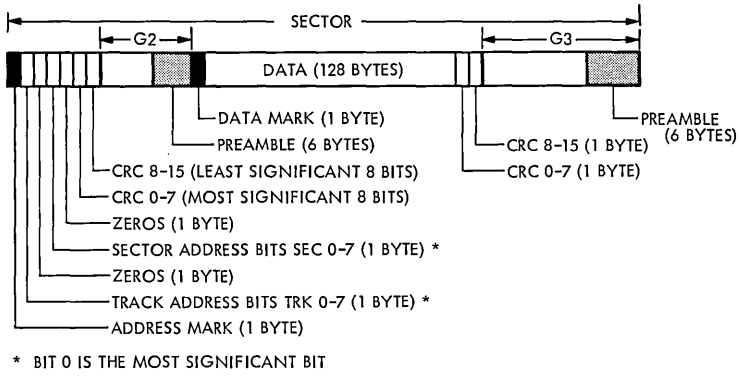
Track Initialization Format

	Number of Bytes	Hex Value of Byte Written	Remarks
	40	00 or FF	Hex FF Recommended
	6	00	Read Preamble
	1	Index Mark	Data-bit pattern of hex FC, clock-bit pattern of hex D7
	26	00 or FF	Hex FF recommended
Write Bracketed Data 26 Times, i.e., 26 Sectors	6	00	Read preamble
	1	Address Mark	Data-bit pattern of hex FE, clock-bit pattern of hex C7
	1	Track Address	Bits 0 — 7 for the track being written
	1	00	
	1	Sector Address	Bits 0 — 7 for the sector being written
	1	00	
	1	CRC	Computed CRC bits 0 — 7 for the address field being written
	1	CRC	Computed CRC bits 8 — 15 for the address field being written
	11	00 or FF	Hex FF recommended
	6	00	Read preamble
	1	Data Mark	Data-bit pattern of hex FB, clock-bit pattern of hex C7
	128	Data Bytes[1]	Initialized as hex E5
	1	CRC[2]	Bits 0 — 7, initialized as hex 5D
	1	CRC[3]	Bits 8 — 15, initialized as hex 30
	27	00 or FF	Hex FF recommended
	247	00 or FF	The length of this field may vary due to tolerances

Notes:

1. Refer to Appendix B for contents of this field for track 00 (Index Track); all other tracks use hex E5 for initialization.
2. Refer to Appendix B for contents of this field for track 00 (Index Track); all other tracks use hex 5D for initialization.
3. Refer to Appendix B for contents of this field for track 00 (Index Track); all other tracks use hex 30 for initialization.





* BIT 0 IS THE MOST SIGNIFICANT BIT

GAP FIELDS

G1 = 26 BYTES HEX FF
6 BYTES HEX 00
G2 = 11 BYTES HEX FF
6 BYTES HEX 00
G3 = 27 BYTES HEX FF
6 BYTES HEX 00
G4 =314 BYTES HEX FF
6 BYTES HEX 00

Figure 9. Sector Format

provides for a preamble and sufficient buffer space. This offsets the effects of deviations from the nominal data density on the track due to tolerances of disk speed and write frequency.

The following paragraphs describe the components of the physical data format.

3.2.1 MARK BYTES

The following mark bytes are defined.

- Index Mark

  Indicates the beginning of a track. It contains a data-bit pattern of hex FC and a clock-bit pattern of hex D7.

- Address Mark

  Indicates the beginning of an address field. It precedes an address field and contains a data-bit pattern of hex FE and a clock-bit pattern of hex C7.

- Data Mark

  Indicates the beginning of a data field. It precedes a data field and contains a data-bit pattern of hex FB and a clock-bit pattern of hex C7.

- Delete Data Mark

  Indicates the beginning of a deleted data field. It precedes a deleted data field and contains a data-bit pattern of hex F8 and a clock-bit pattern of hex C7.

3.2.2 ADDRESS FIELD

An address field contains a group of 6 bytes described as follows.

Byte Position in the Field	Value
1	Track Address Bits 0 — 7
2	Hex 00
3	Sector Address Bits 0 — 7
4	Hex 00
5	CRC Bits 0 — 7
6	CRC Bits 8 — 15

3.2.3 DATA FIELD

A data field contains a group of 128 bytes which comprises one record of data followed by two Cyclic Redundancy Check (CRC) bytes as follows.

Byte Position in the Field	Value
1 — 128	Data Bytes
129	CRC Bits 0 — 7
130	CRC Bits 8 — 15

3.2.4 GAPS

Specific gaps are defined as follows.

- Post-Index Gap (G1)

  The Post-Index Gap (G1) is defined as the 26 bytes of hex FF followed by 6 bytes of hex 00 between the Index Mark and the Address Mark for the first sector.

- Address Field Gap (G2)

  The Address Field Gap (G2) is defined as the 11 bytes of hex FF followed by 6 bytes of hex 00 between an address field and the Data Mark for the following data field.

- Data Field Gap (G3)

  The Data Field Gap (G3) is defined as the 27 bytes of hex FF followed by 6 bytes of hex 00 between a data field and the Address Mark for the following sector with the exception of the gap which follows the data field for the last sector.

- Pre-Index Gap (G4)

  The Pre-Index Gap (G4) is defined as the 314 bytes of hex FF followed by 6 bytes of hex 00 between the data field for the last sector and the Index Mark.

The Post-Index Gap (G1) is not affected by any updating of data fields. However, the length of the Address Field Gap (G2) and the Data Field Gap (G3) may vary after data fields have been updated. The Pre-Index Gap (G4), after initialize, may vary due to tolerances of diskette speed and write frequency. The Pre-Index Gap (G4) is further affected by the updating of data fields.

The last six bytes (hex 00) in the gaps, described in the preceding paragraphs, serve as the read preamble. The read preamble is used by decoder circuits to acquire phase lock and synchronization. The synchronization to a field (address or data) is achieved when the mark byte following the read preamble is detected.

3.3 CRC GENERATION

Each field (address or data) on the diskette is appended with CRC bytes. These two bytes are generated from a cyclic permutation of data bits starting with bit zero of the mark byte and ending with bit seven of the last byte within a field (address or data) excluding the CRC bytes. This cyclic permutation is the remainder from the division of data bits in a field (represented as an algebraic polynomial) by a generator polynomial G(X). For all fields recorded on an IBM System 3740 Diskette, this generator polynomial is:

$$G(X) = X^{16} + X^{12} + X^5 + 1$$

When a field is read back from a diskette, the data bits (from bit zero of the mark byte to bit seven of the second CRC byte) are divided by the same generator polynomial G(X). A non-zero remainder indicates an error within the data read back from the diskette whereas a remainder of zero indicates that the data have been read back correctly from the diskette or that a undetectable error has occurred.

The necessary division can be accomplished by use of the circuit shown in Figure 10. The same circuit is used to generate the CRC bytes during a write operation and to check data for a CRC error (non-zero remainder) during a read operation. In order to avoid CRC bytes of all zeros, all flip-flops shown in Figure 10 are preset to one prior to shifting data through the circuit for division by the generator polynomial G(X).

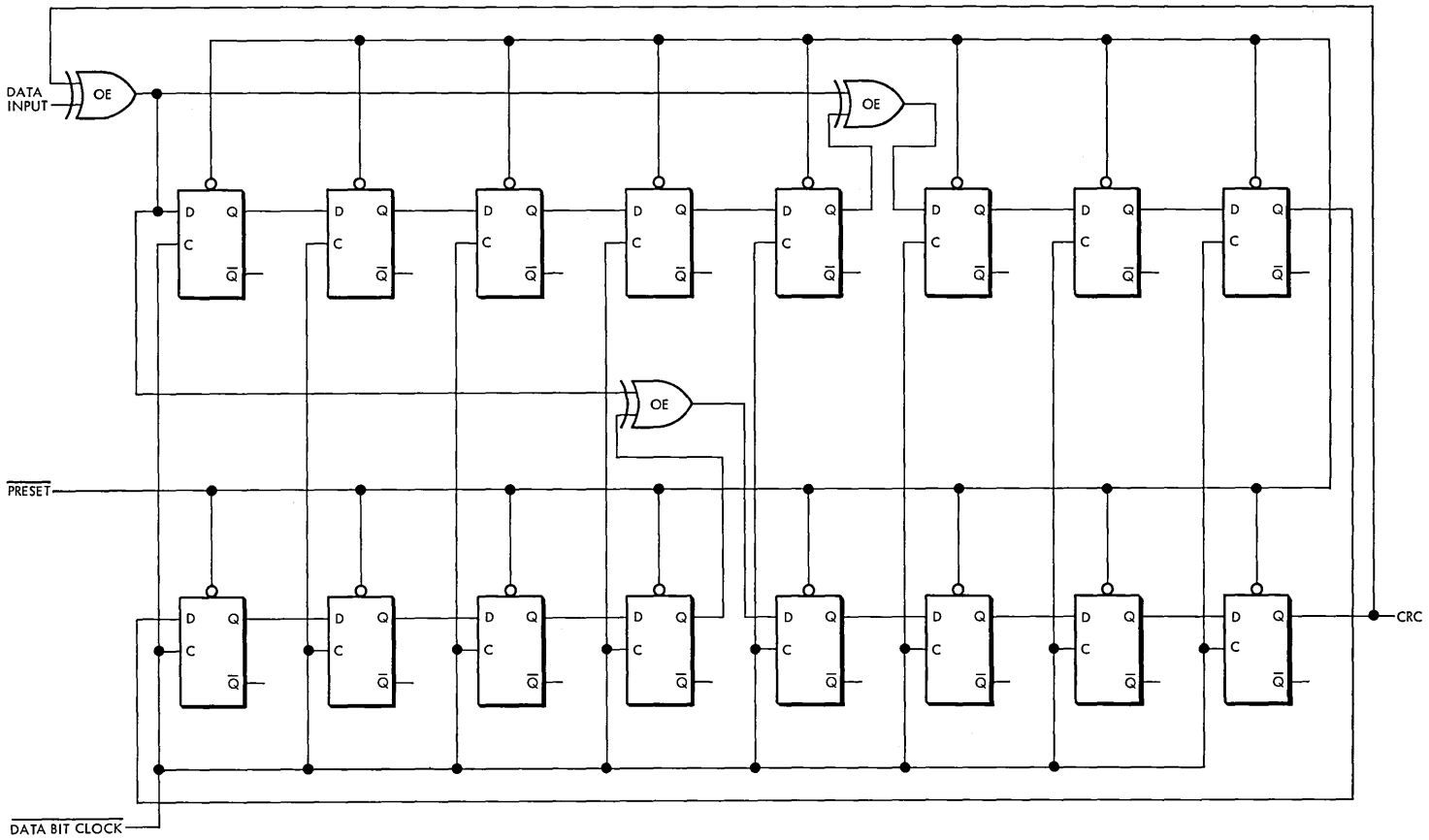


Figure 10. CRC Generation

IV. FORMATTER

This section presents the functional characteristics of a typical soft-sectored formatter. Design guide lines are provided to aid the reader in his effort; however, the specific hardware interface for the formatter will depend upon his application.

4.1 FUNCTIONAL DESCRIPTION

The formatter, when used in conjunction with the disk drive, provides all control and timing necessary to form a data storage/retrieval system suitable for use in data processing applications. In addition to read and write commands, formatter commands are provided for data search, read-after-write data checking, and write initialize operations. The formatter also performs automatic checks on data, sector addressing, and various programming and hardware error conditions. IBM System 3740 compatible format is used for organizing diskette data.

Although the formatter can be designed to function over a wide range of parameters, the following are used as the basis for a standard configuration.

- Disk speed — 360 rpm
- Data packing density — 3268 bpi (innermost track)
- Sectors per track — 26
- Data bytes per sector — 128

The fixed header format which is described in the IBM Diskette Original Equipment Manufacturer's Information document is used to initialize the diskette. The *address-mark* method is used to divide the diskette into sectors as described in Section III of this document. A crystal oscillator, whose frequency is two times the bit-data-transfer rate, controls all timing in the formatter logic and is used for encoding write data, clocking, and in the generation of various delays. Data transfers and timing functions are organized on a byte (8-bit) basis and individual delays are formed by counting the equivalent number of byte times.

A Phase-Lock Loop (PLL) is used in data decoding; it also provides the system clock during a Read operation.

4.2 OPERATION

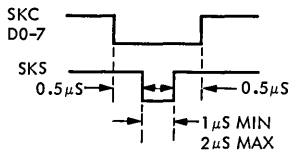
The formatter is capable of executing commands listed in Figure 11; these are grouped into disk commands and formatter commands. Commands should not be issued while the formatter is busy or when the disk drive is not ready.

4.2.1 DISK COMMANDS

The SEEK and RESTORE disk commands cause the positioner in the selected disk drive to move to the required track address. Disk commands do not cause the formatter to go busy.

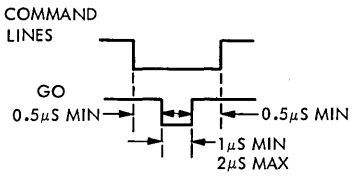
The track address can be specified on the data bus (D0 — 7) when a Seek Strobe (SKS) is issued to the formatter. This address is copied into the track address register in the formatter and the formatter performs a validity check on the address. A valid address causes the positioner to move to the required track. The Seek Status signal (SKG) is set true while the disk drive is seeking, but if an illegal address is detected, the positioner will not move and an Illegal Track Address Status (TIL) will be reported to the controller. A

	Line	
Command	SKC	D0-D7
SEEK	1	Cylinder Address
RESTORE	0	Not Applicable

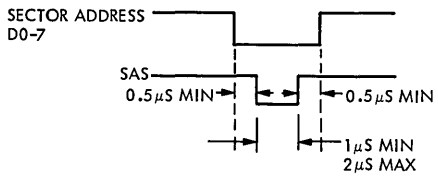


Disk Commands

	Line					
Command	WRT	VFY	SNH	SNL	HCI	DRC
READ					A/R	N/A
WRITE	1				A/R	A/R
VERIFY		1			NA	NA
SCAN HIGH			1		A/R	NA
SCAN LOW				1	A/R	NA
SCAN EQUAL			1	1	A/R	NA



Formatter Commands



Sector Address

Figure 11. Disk and Formatter Command Coding

successful SEEK command must be issued before a data transfer (formatter command) is attempted. However, once the requested track has been reached, no further SEEK commands are necessary until the next track address is called for.

The RESTORE command causes the positioner in the selected disk drive to move to track zero. This command is used when initializing the system; it is also used when the positioner becomes *lost*. The RESTORE command is always issued after a power-up sequence. The SKG signal is set true while the disk drive is executing the RESTORE command.

4.2.2 FORMATTER COMMANDS

These commands involve the transfer of data to or from the diskette. They can be issued while the selected disk is performing a SEEK command.

There are six basic formatter commands, each of which can be conditioned as shown in Figure 11. Figure 12 is a general command timing diagram which should be referred to in conjunction with the following discussion.

The required starting sector address for a data transfer can be specified on the data bus (D0 — 7) and is copied into the formatter by the Sector Address Strobe (SAS). The SAS may occur either before or at the same time that the command is initiated. A formatter command can be initiated by issuing a GO strobe. The SAS and GO strobes can be paralleled if it is desired to transmit the starting sector address at the same time that a command is initiated. When the command is received, the command lines are copied into the formatter, the Formatter Busy (FBY) goes true, and the formatter performs the control and timing functions necessary to execute the command.

The Data Busy (DBY) signal informs the controller as each sector is processed. DBY goes true during the time of data transfer for each sector processed. The controller can set the Last Sector (LST) signal true during the last sector of data transfer as shown in Figure 12. At the end of each sector processed, the formatter samples the state of LST. If LST is true, FBY and DBY are reset and the command is terminated. If LST is false, the next sector is processed. Under certain error conditions, e.g., Programming Error (PER), Address Error (AER), or Format Error (FER), the command is aborted immediately. These conditions are described in Paragraph 4.2.3.

A Data Error (DER) or Scan Found (SNF) condition does not cause the command to be aborted, but both conditions are reported at the end of each sector before DBY goes false. This allows the controller to terminate the command as a result of a DER or SNF detected during the current sector.

The following paragraphs describe the sequence of events which occur during the execution of formatter commands.

4.2.2.1 READ Command

This command consists of the following sequence.

(1) Validate the specified starting sector address by ensuring that it has a value less than or equal to the maximum number of sectors on a track, i.e., 26 sectors for the standard configuration.

(2) If necessary, wait for the selected disk drive to complete its SEEK operation.

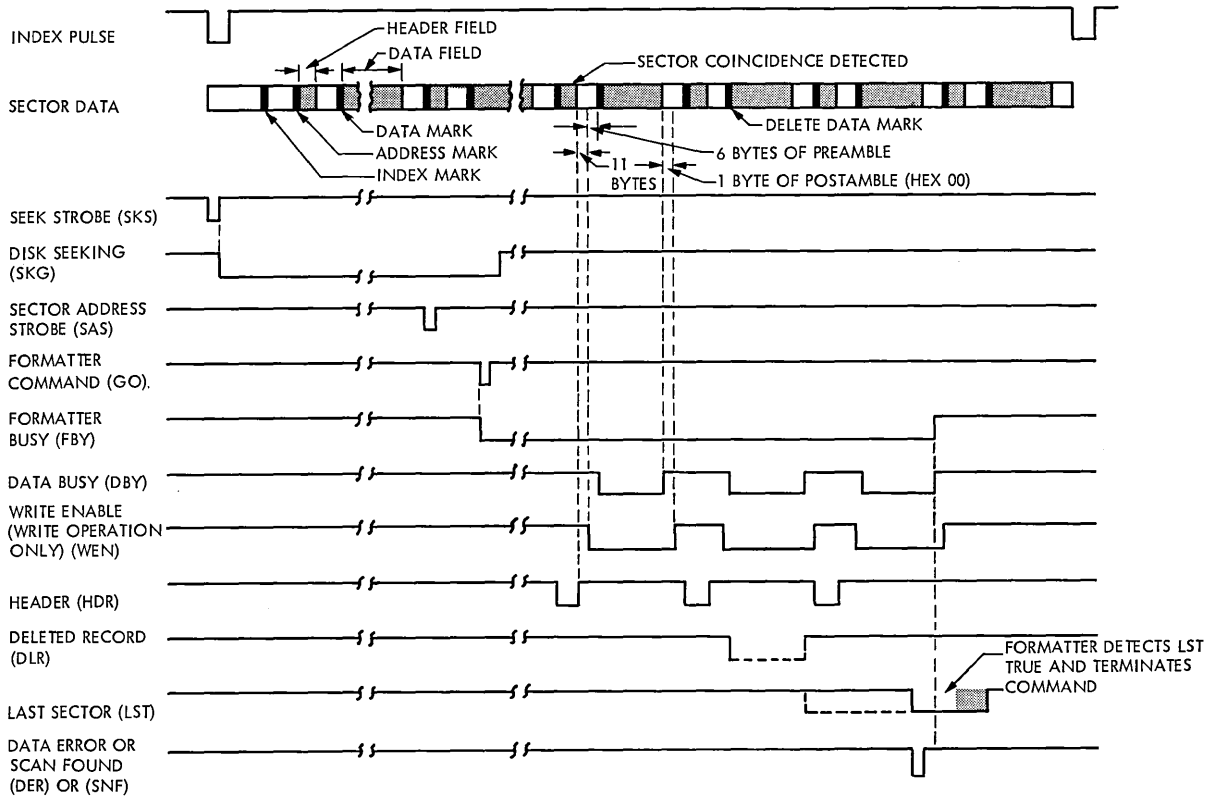


Figure 12. Command Timing

(3) Begin a READ operation, wait for the detection of an address mark, then read the header, check the CRC, and compare this information with the specified track and starting-sector addresses. Wait for coincidence between the specified sector and the sector address contained in the header read from the diskette.

(4) Wait for the detection of a data mark, assemble and transmit data byte-by-byte to to the controller. Compare the CRC read from the diskette with the re-computed CRC.

(5) Test for last sector by sampling LST. If LST is true, terminate the command; if false, wait for the detection of the address mark for the next sector, then read the header, check the CRC, then return to Step (4) of this sequence.

4.2.2.2 WRITE Command

This command consists of the following sequence.

(1) Validate the specified starting sector address by ensuring that it has a value of less than or equal to the maximum number of sectors on a track, i.e., 26 sectors for the standard configuration.

(2) If necessary, wait for the selected disk drive to complete its SEEK operation.

(3) Begin a READ operation, wait for the detection of an address mark, then read the header, check the CRC, and compare this information against the specified track and starting sector addresses. Wait for coincidence between the specified sector and the sector address contained in the header read from the diskette.

(4) Begin a time delay, equivalent to 11 bytes, from the CRC bit last read in Step (3), or Step (5). Write 6 bytes of zeros, a data-mark byte, and 128 bytes of data followed by 2 bytes of CRC. Write 1 byte of zeros for the postamble.

It should be noted that data transfers between the controller and the formatter are byte-serial. Also, the CRC is automatically generated by the formatter from the data field.

(5) Test for last sector by sampling LST. If LST is true, terminate the command; if false, wait for the detection of an address mark for the next sector, then read the header, check the CRC, then return to Step (4) of this sequence.

4.2.2.3. VERIFY Command

This command is intended for read-after-write data checking. It is identical to a READ command except that read data are not transferred to the controller. Normal error checks described in Paragraph 4.2.3 are performed. It is recommended that a WRITE command be followed by a VERIFY command to ensure system data reliability.

4.2.2.4 SCAN Commands (HIGH, LOW, EQUAL)

These commands provide an efficient means of searching the diskette surface for a particular file or class of files. The commands are executed in a manner similar to a READ command except that Step (4) of the READ command (Paragraph 4.2.2.1) is replaced with the following sequence.

The data field read from the diskette is compared by hardware, byte-by-byte, against a data field provided by the controller. A special mask character, hex FF, can be used to inhibit comparison. Whenever this character occurs in the controller field, the comparison for the corresponding byte position is not performed. This comparison is performed for each sector of data transfer. A Scan Found (SNF) condition is reported to the controller shortly before the trailing edge of DBY and does not abort the current command.

If all unmasked bytes in a sector meet the following conditions, a SNF condition is reported at the end of the sector.

- SCAN HIGH: Data Field ≥ Controller Field
- SCAN LOW: Data Field ≤ Controller Field
- SCAN EQUAL: Data Field = Controller Field

In the case where all bytes of a sector are masked, a SNF condition can always be reported.

4.2.2.5 Header Check Inhibit (HCI)

The WRITE Initialize operation is performed when the Header Check Inhibit command line (HCI) is used in conjunction with a WRITE operation. The controller sends the track and sector addresses to be recorded in the header field and the data bytes to be recorded in the data field. The formatter records the header information and the computed CRC into each header field. The tracks are initialized one at a time, each track requiring one Write Initialize operation.

If the Header Check Inhibit line is used during a READ operation, the formatter transmits the header information to the controller in addition to the data for each sector processed.

4.2.2.6 Delete Record (DRC)

The Delete Record operation is performed when the Delete Record command line (DRC) is used in conjunction with a WRITE operation. The formatter performs the Delete Record operation by writing a delete-data mark preceding the data field. When reading the record preceded by a delete-data mark, Deleted Record (DLR) goes true during the time the sector is processed. This is shown in Figure 12. With the foregoing exception, the deleted sector is processed in the same manner as a normal read/write operation.

4.2.3 ERROR CHECKING

This paragraph summarizes the various error and status checks performed by the formatter.

(1) TIL (Illegal Track)
- An illegal track address has been detected by the formatter during a SEEK command.

(2) PER (Program Error)
- Illegal starting sector address.
- Disk overrun (data transfer extends past last sector on diskette).

(3) FER (Format Error)
- Sector coincidence not established.

(4) AER (Address Error)
- Expected track address does not compare with the track address contained in the header, or a CRC error is detected in the header field.

(5) DER (Data Error)
- CRC read from the diskette does not compare with the recomputed CRC for the data field.

(6) SNF (Scan Found)

- A successful comparison has been made between the diskette and controller data during a SCAN command.

The occurrence of a PER, FER, or AER condition aborts the current command. A DER or SNF condition does not cause the current command to be aborted; however, the error is reported at the end of the sector being processed before DBY goes false.

APPENDIX A
PHASE LOCK LOOP

A.1 BASIC PRINCIPLES OF OPERATION

The purpose of the Phase-Lock Loop (PLL) is to provide a clock signal which maintains a fixed phase relationship with the incoming signal. The PLL is a feedback system consisting of a phase comparator, a low-pass filter, an error amplifier in the forward signal path, and a Voltage-Controlled Oscillator (VCO) in the feedback path.

Figure A-1 is a block diagram of the basic PLL system. With no input signal applied the error voltage is equal to zero and the voltage-controlled oscillator operates at a center frequency, f_o, which is called the *free-running* frequency. When an input signal, f_s, is applied to the system the Phase Comparator compares the phase and frequency of the input with the VCO frequency, f_o, then generates an error voltage, $V_e(t)$, that is related to the phase and frequency difference between the two signals. This error voltage is filtered, amplified, and routed to the control input of the VCO. Thus, the control voltage, $V_d(t)$, forces the VCO frequency to vary in a direction that reduces the frequency difference between f_o and the input signal. If the input signal, f_s, is sufficiently close to f_o, the feedback nature of the PLL causes the VCO to synchronize or lock with the incoming signal. Once locked, the VCO frequency is identical to the input signal, except for a finite phase difference. This net phase difference is necessary to generate the corrective error voltage to shift the VCO frequency from its free-running value to the input signal frequency, f_s, thus keeping the PLL locked. This tracking ability allows the PLL to follow the frequency changes of the input signal once it is locked. The range of frequencies over which the PLL can maintain phase-lock with an input signal is defined as the *lock range* of the system. This range is always larger than the band of frequencies over which the PLL can acquire phase-lock with an incoming signal. This latter range of frequencies is called the *capture range* of the system.

The total time taken by the PLL to establish lock-on is called the *pull-in* time. Pull-in time depends on the initial frequency and phase difference between the two signals, as well as the overall loop gain and the bandwidth of the Low Pass Filter.

The Low Pass Filter attenuates the high frequency components at the output of the Phase Comparator, thereby enhancing the noise rejection characteristics of the circuit. Also, it provides a short-term memory for the PLL and ensures a rapid recapture of the signal if the system loses phase-lock due to a noise transient. The Low-Pass Filter, therefore, controls the capture and the transient-response characteristics of the PLL. As the filter bandwidth is decreased the capture process becomes slower, thereby increasing the pull-in time. As the capture range decreases, the interference-rejection properties of the PLL improve and the transient response of the loop becomes under-damped, resulting in a smaller value of the damping factor. The latter effect brings about a practical limitation on the Low-Pass Filter bandwidth.

In a specific PLL application the capture range is minimized. However, it must allow the PLL to acquire phase-lock within the specified lock-up time under worst case fluctuations of the incoming signal frequency. This bandwidth should be sufficiently narrow to ensure that the phase errors in the incoming signal (produced by the peak-shift effect) are not tracked. The value of the damping factor for the PLL is chosen between 0.7 and 1.0 for optimum loop performance. These criteria are applied to the design of the phase-lock loop described in Paragraph A.2.

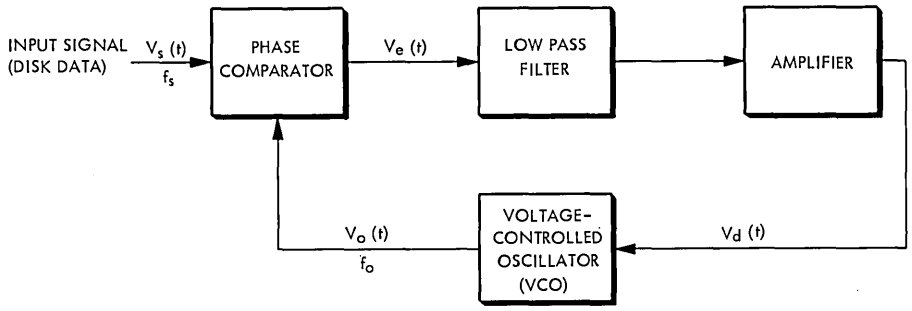


Figure A-1. Phase-Locked Loop Block Diagram

A.2 CIRCUIT DESCRIPTION

Figures A-2 and A-3 are schematic drawings of a PLL which contain the PERTEC data/clock-bit separator circuit. Disk Data are applied to the one-shot circuit at U1 to obtain a pulse of 500 nanoseconds (nonimal) for every flux transition read from the diskette. The output at U1/5 is applied to the Phase Comparator circuit which is comprised of flip-flop U4/15 and NAND gates U3/11 and U3/8. The Phase Comparator compares the phase of the input signal Disk Data with the VCO ÷ 2 signal. The Phase Comparator timing diagram is shown in Figure A-4. Outputs Control A and Control B are applied to the balanced lead-lag-type Low Pass Filter formed by resistors R7, R9, and R10, and capacitors C4 and C5 on one leg, and resistors R12, R14, and R15, and capacitors C6 and C7 on the other leg. The output of the balanced filter is applied to the positive and negative inputs of the operational amplifier U5.

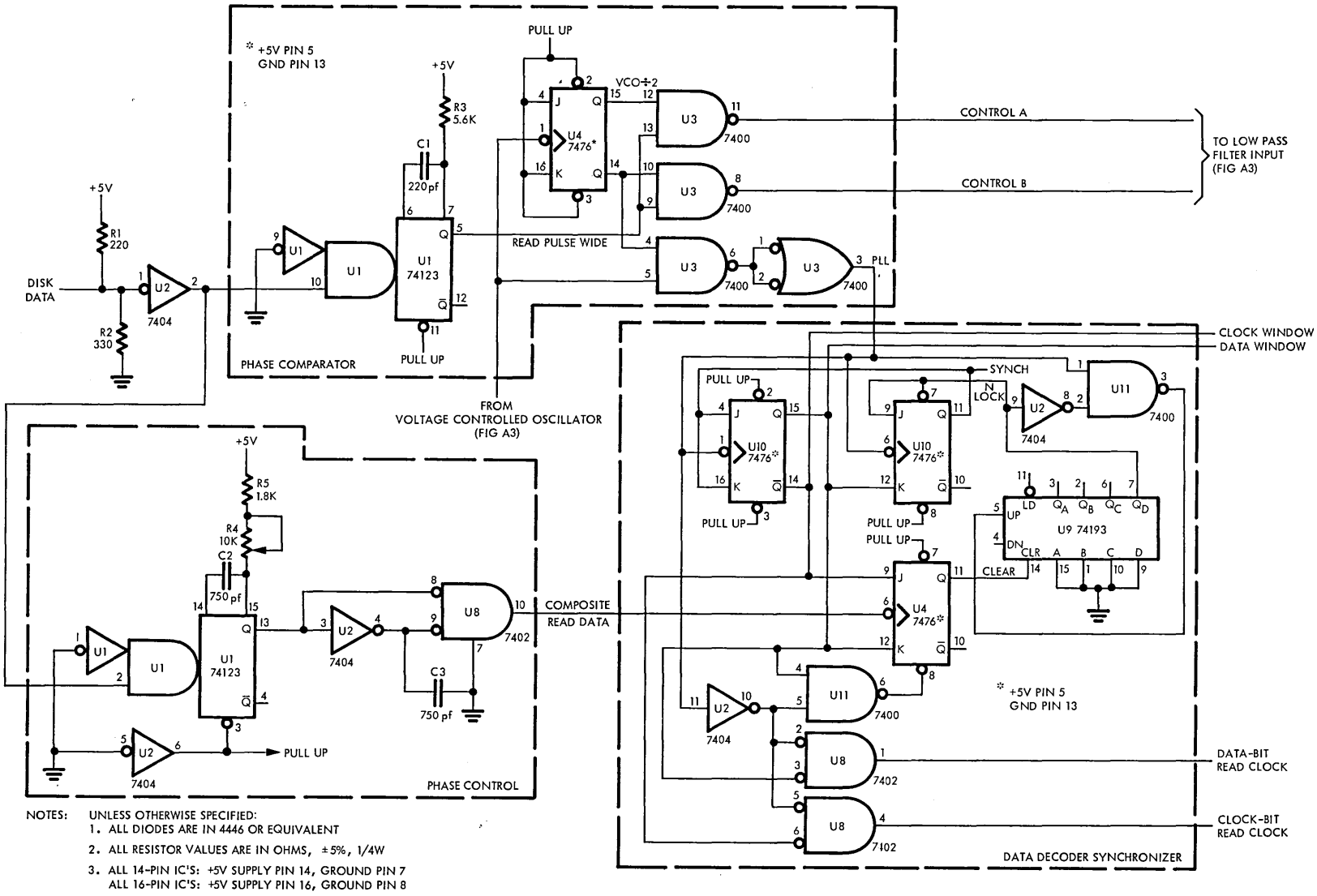
The output of amplifier U5 is applied to the VCO formed by the current source Q1, capacitors C12 and C13, and the voltage comparator U6. Values of capacitors C12, C13, the current source Q1, and the threshold voltage of the comparator U6 determine the free-running frequency of the VCO. This frequency is varied by potentiometer R23. The output of the VCO is applied to the Phase Comparator to generate an error voltage which is proportional to the phase difference between the input signals Disk Data and VCO ÷ 2.

The PLL output is applied to the Data Decoder Synchronizer circuit along with the Composite Read Data signal which is the output of the Phase Control. Potentiometer R4 in the Phase Control circuit provides the means to obtain proper phase relationship between the Composite Read Data signal, the Data Window, and Clock Window waveform. A detailed timing diagram for the Data Decoder Synchronizer circuit is shown in Figure A-5.

The four outputs of the Data Decoder Synchronizer circuit are the Data Window and Clock Window waveforms, and their associated clock waveforms, Data-Bit Read Clock and Clock-Bit Read Clock.

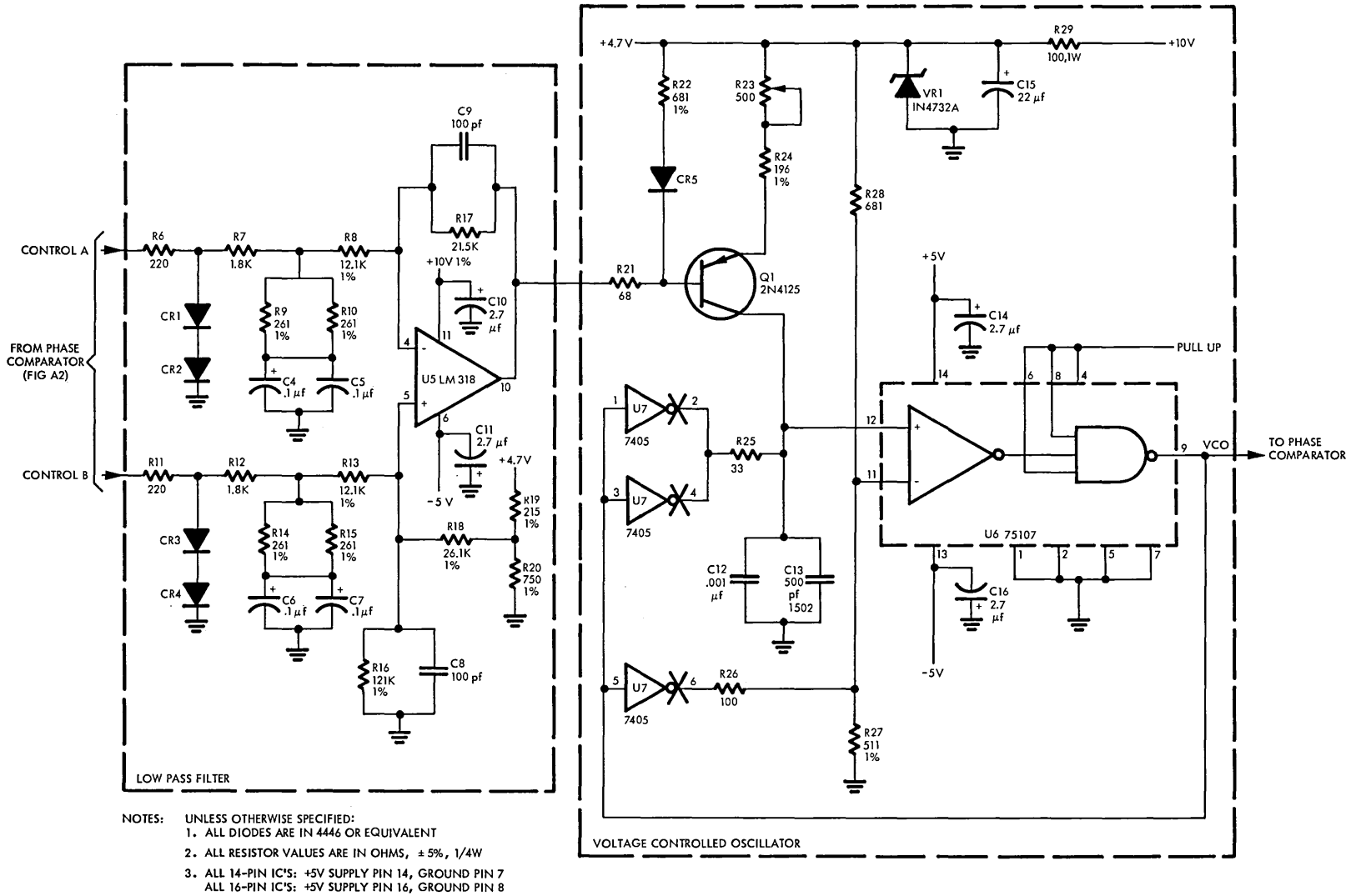
The center frequency of the PLL circuit shown can be adjusted by potentiometer R23. With a frequency counter connected at U4/15 (VCO ÷ 2), R23 should be adjusted until the frequency counter indicates 500 KHz. The Data Disk line at U2/1 must be in the false (high) state during this adjustment.

In the circuit shown, the proper phase relationship between the Composite Read Data signal and the Data Window can be obtained by adjusting potentiometer R4. Using a dual-trace oscilloscope, the Data Window can be observed at U10/15 (Channel 1) and the Composite Read Data signal at U8/10 (Channel 2). With the oscilloscope set to a vertical mode 'add' condition and all-zero data on the Disk Data line, the phase relationship should be as shown in Figure A-5.



NOTES: UNLESS OTHERWISE SPECIFIED:
1. ALL DIODES ARE IN 4446 OR EQUIVALENT
2. ALL RESISTOR VALUES ARE IN OHMS, ±5%, 1/4W
3. ALL 14-PIN IC'S: +5V SUPPLY PIN 14, GROUND PIN 7
   ALL 16-PIN IC'S: +5V SUPPLY PIN 16, GROUND PIN 8

Figure A-2. PLL (Phase Comparator, Phase Control, Data Decoder Synchronizer), Schematic Diagram



NOTES: UNLESS OTHERWISE SPECIFIED:
1. ALL DIODES ARE IN 4446 OR EQUIVALENT
2. ALL RESISTOR VALUES ARE IN OHMS, ±5%, 1/4W
3. ALL 14-PIN IC'S: +5V SUPPLY PIN 14, GROUND PIN 7
   ALL 16-PIN IC'S: +5V SUPPLY PIN 16, GROUND PIN 8

Figure A-3. PLL (Low Pass Filter, Voltage Controlled Oscillator), Schematic Diagram

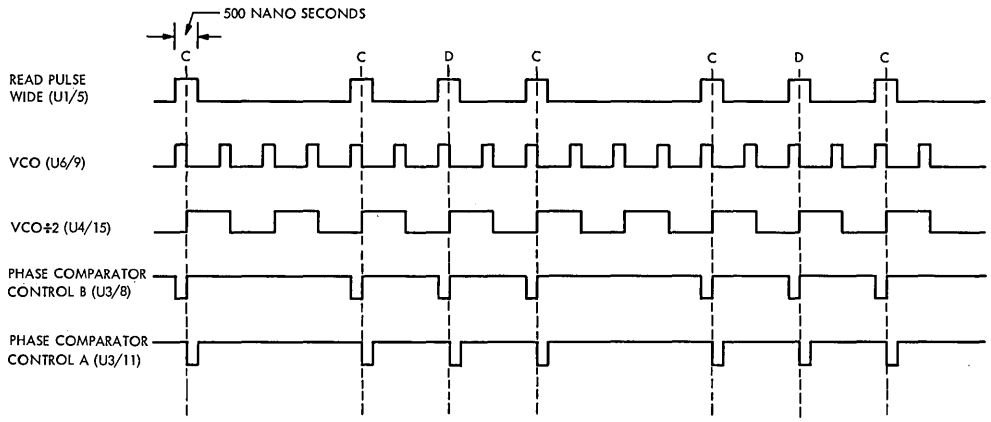


Figure A-4. Phase Comparator Timing (Phase Locked Condition)

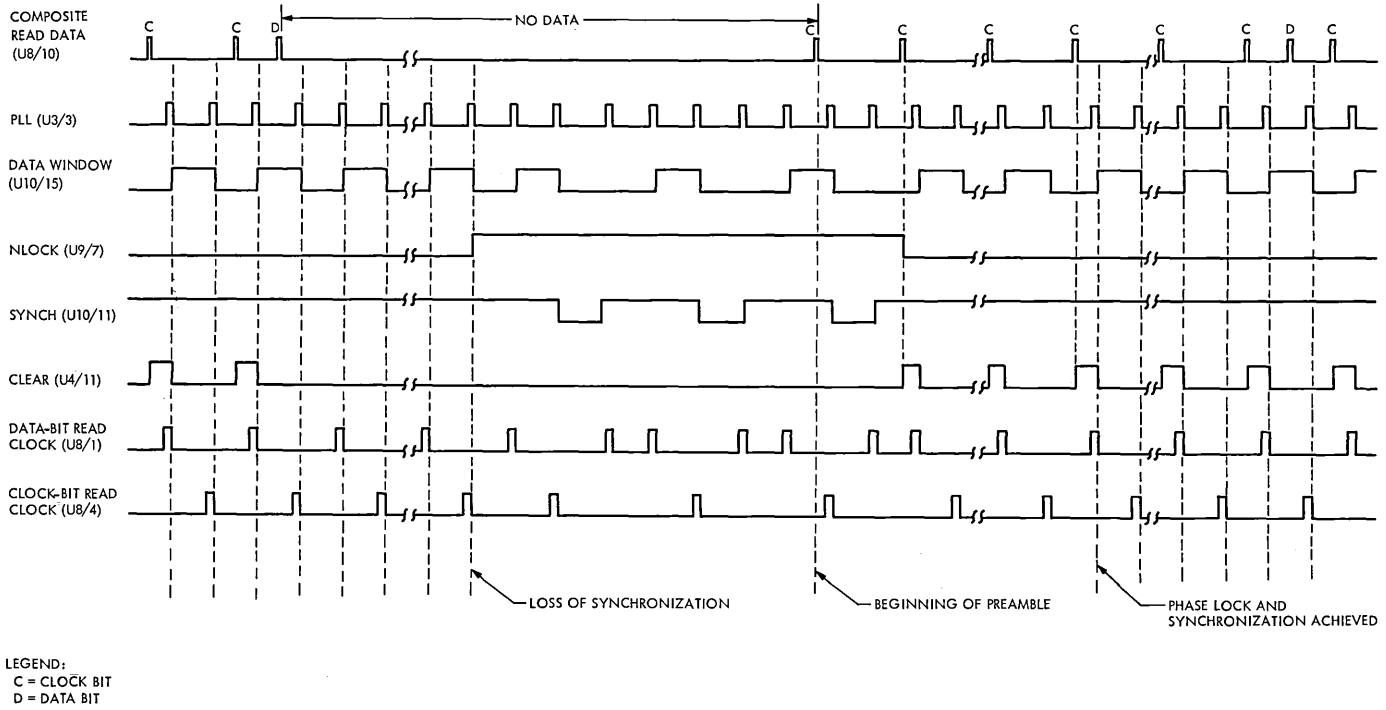


Figure A-5. Decoder Synchronizer Timing Diagram

APPENDIX B
INDEX TRACK INITIALIZATION

This Appendix identifies the data fields for all sectors of the index track (00) on a diskette. It lists the characters and their sequence as they should be written, when initialized. This initialization is accomplished by writing the track from beginning to end (index to index), without interruption using the following characters.

Sector	Position	Remarks
01 — 04	1 — 80	Hex 40
	81 — 128	Hex 00
	129	CRC. Use hex 59 for data.
	130	CRC. Use hex D9 for data.
05	1 — 5	ERMAP position 6 — 80 blanks (hex 40). If a bad track is found on the disk, an alternate track may be substituted.
	7 — 8	Indicates first bad track number. Alternate track 74 will be used in its place.
	9	Hex 00.
	11 — 12	Indicates the second bad track number. Alternate track 75 will be used in its place.
	13	Hex 00.
	14 — 80	Hex 40
	81 — 128	Hex 00
	129	CRC. Use hex E5 for data.
	130	CRC. Use hex 0E for data.
06	1 — 80	Hex 40.
	81 — 128	Hex 00.
	129	CRC. Use hex 59 for data.
	130	CRC. Use hex D9 for data.
07	1 — 4	Vol 1 (Initialized Vol. 1. Required for 3740 system.)
	5 — 10	Vol ID. (initialized on IBM diskette as IBMIRD.) The volume ID may be changed by the system user.
	11	Accessibility — any non-blank character means that the disk is not accessible. (Initialized as a blank character, hex 40. This is rarely used in the 3740 system.)
	12 — 76	Hex 40.
	77, 78	Sector sequence information (initialized as hex 40). This allows for initializing non-sectors on a track, e.g., a 02 in these positions will sequence the sectors: 1, 3, 5, 7, . . . etc. The sector sequence information characters provide for interleaving of sectors.
	79	Hex 40.

Sector	Position	Remarks
	80	Initialized as W. Required by the 3740 system.
	81 — 128	Hex 00.
	129	CRC. Use hex 26 for data.
	130	CRC. Use hex 53 for data.
08 — 26		Data Set Labels defines logical bounderies of data. Up to 19 data sets may be defined on a diskette.
	1 — 4	Header. Sector 8 is initialized as HDR1. initialized Sector 9 through 26 contain deleted records with DDR1 in Positions 1 — 4.
	5	Initialized hex 40.
	6 — 13	Data Set Name. May be user defined. (Initialized with DATA in position 6 — 9.Initialized Sectors 9 through 26 contain records with the sector number recorded in positions 10 — 11. positions 12 and 13 initialized hex 40.)
	14 — 24	Initialized hex 40 in positions 14 — 24.
	25 — 27	Logical record length (maximum 128). Record length must be equal to 080 on the 3742 or greather than 000, less than 128 on the 3741 or on the 3742 with the 128 feature. (Initialized as 080.)
	28	Initialized as hex 40.
	29 — 33	Beginning of Extent (BOE). Identifies address of first sector of a data set. Positions 29 and 30 contain track number, position 31 must be 0, position 32 and 33 contain sector number. (Initialized sector 8 has 01001 in position 29 — 33, sector 9 — 26 has 74001 in position 29 — 33.)
	34	Initialized as hex 40.
	35 — 39	End of Extent. Identifies the address of last sector reserved for data set. (Initialized whith EBCDIC 73026 in position 35 — 39 for sector 8 — 26.)
	40	Initialized as hex 40.
	41	Bypass data set. If this field contains a 'B', then the 3747 data converter will ignore the data set. If the field contains a blank, then the data set will be processed. (Initialized as hex 40.)
	42	Accessibility. Field must contain a blank for processing in data set. (Initialized as hex 40.)
	43	Data Set Write Protect. If the field is blank, then reading and writing in data set is permitted. If the field contains a 'P', the data set is write protected.
	44	Initialized as hex 40.
	45	Multi-volume Indicator. A blank indicates that data set is not continued on or from another diskette volume. A 'C' indicates a data set which is contiuned on another diskette, and an 'L' indicates that this diskette is to be the last on which the data set resides. (Initialized as hex 40.)

Sector	Position	Remarks
	46 — 72	Initialized as hex 40.
	73	Verify mark. A 'V' in this field indicates that the data set has been verified. (Initialized as hex 40.)
	74	End of data. Indicates the address of the next unused sector of the data set. (Initialized in sector 8 as 01001.) (Sectors 9 — 26 initialized with EBCDIC 74001.)
	80	Initialized as hex 40.
	81 — 128	Hex 00.
	129	CRC. Refer to Note 1.
	130	CRC. Refer to Note 2.

NOTES:

1. Use the following for data in position 129 of sectors 8 through 26.

Sector/Hex Value	Sector/Hex Value
8/C6	18/BB
9/A9	19/7B
10/DB	20/BD
11/1B	21/7D
12/4B	22/2D
13/8B	23/ED
14/EB	24/8D
15/28	25/4D
16/7B	26/1D
17/BB	

2. Use the following for data in position 130 of sectors 8 through 26.

Sector/Hex Value	Sector/Hex Value
8/F0	18/4D
9/0A	19/46
10/D3	20/26
11/D8	21/2D
12/E4	22/11
13/EF	23/1A
14/9C	24/69
15/97	25/62
16/AB	26/5E
17/A0	

NOTES

PERTEC

PERTEC reserves the right to change specifications at any time. It is PERTEC policy to improve products as new techniques and components become available.

9600 IRONDALE AVENUE • CHATSWORTH, CALIFORNIA 91311 • PHONE (213) 882-0030 • TWX (910) 494-2093

75607 (Revised 2-77)