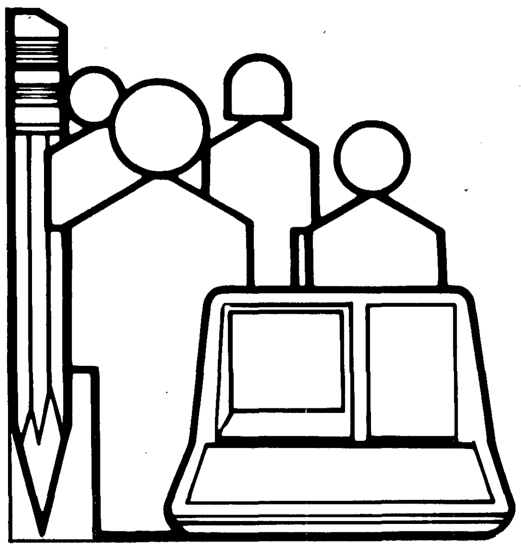


RL11/01 DISK SUB-SYSTEM TRAINING HANDOUT

digital

Santa Clara Educational Center

2525 Augustine Drive
Santa Clara, CA 95051

(408) 748-4048

Los Angeles Educational Center

4311 Wilshire Boulevard
Suite 400
Los Angeles, CA 90010

(213) 937-3870

Dallas Educational Center

12100 Ford Road
Suite 110
Dallas, Texas 75234

(214) 620-2051

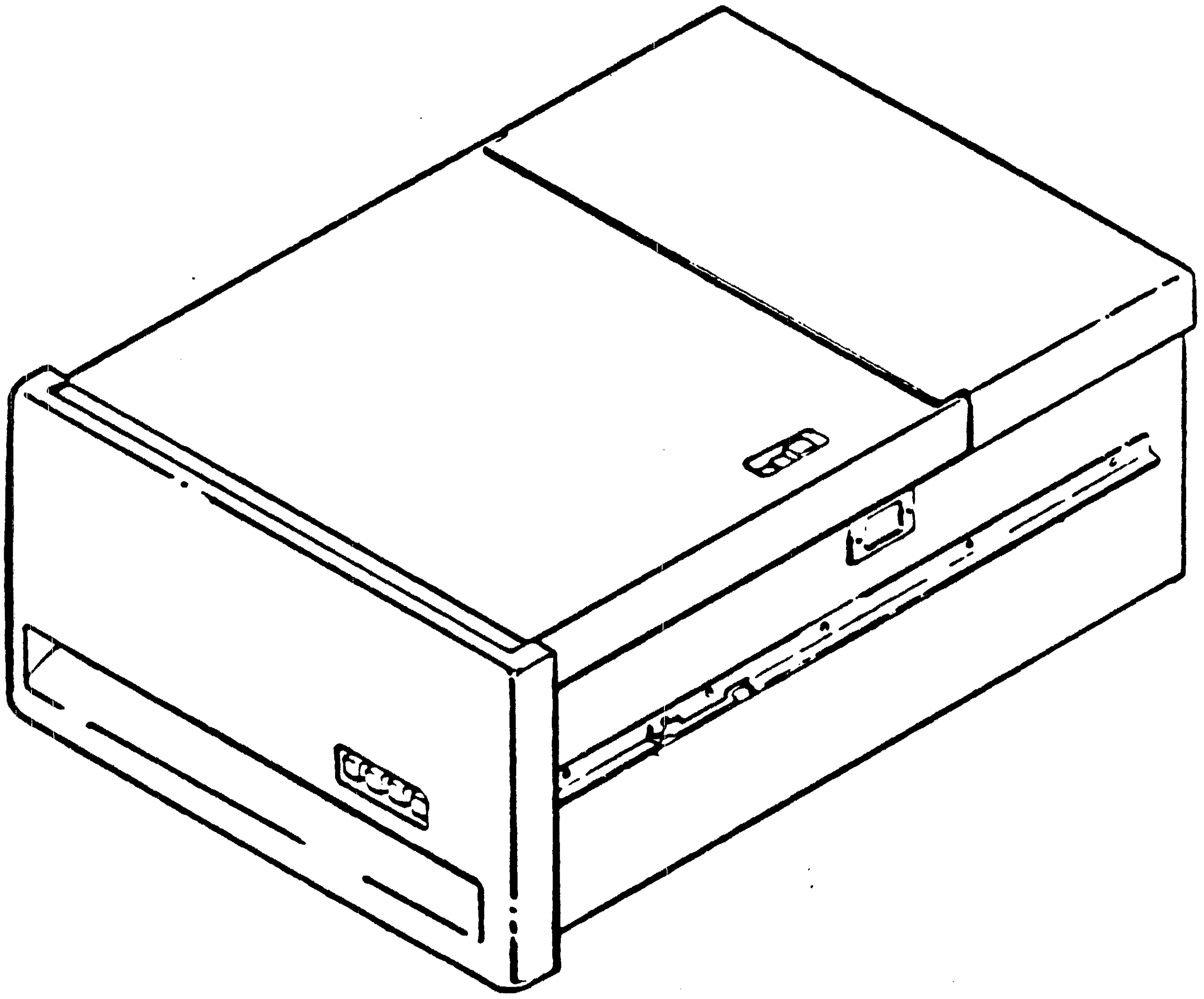


TABLE OF CONTENTS

INTRODUCTION TO THE RL DRIVE

INTRODUCTION TO THE SUBSYSTEM

The RL Disk Drive is a low cost, compact five megabyte (RL01) or ten megabyte (RL02) disk drive. It is utilized on a variety of DEC processors and by a large OEM market.

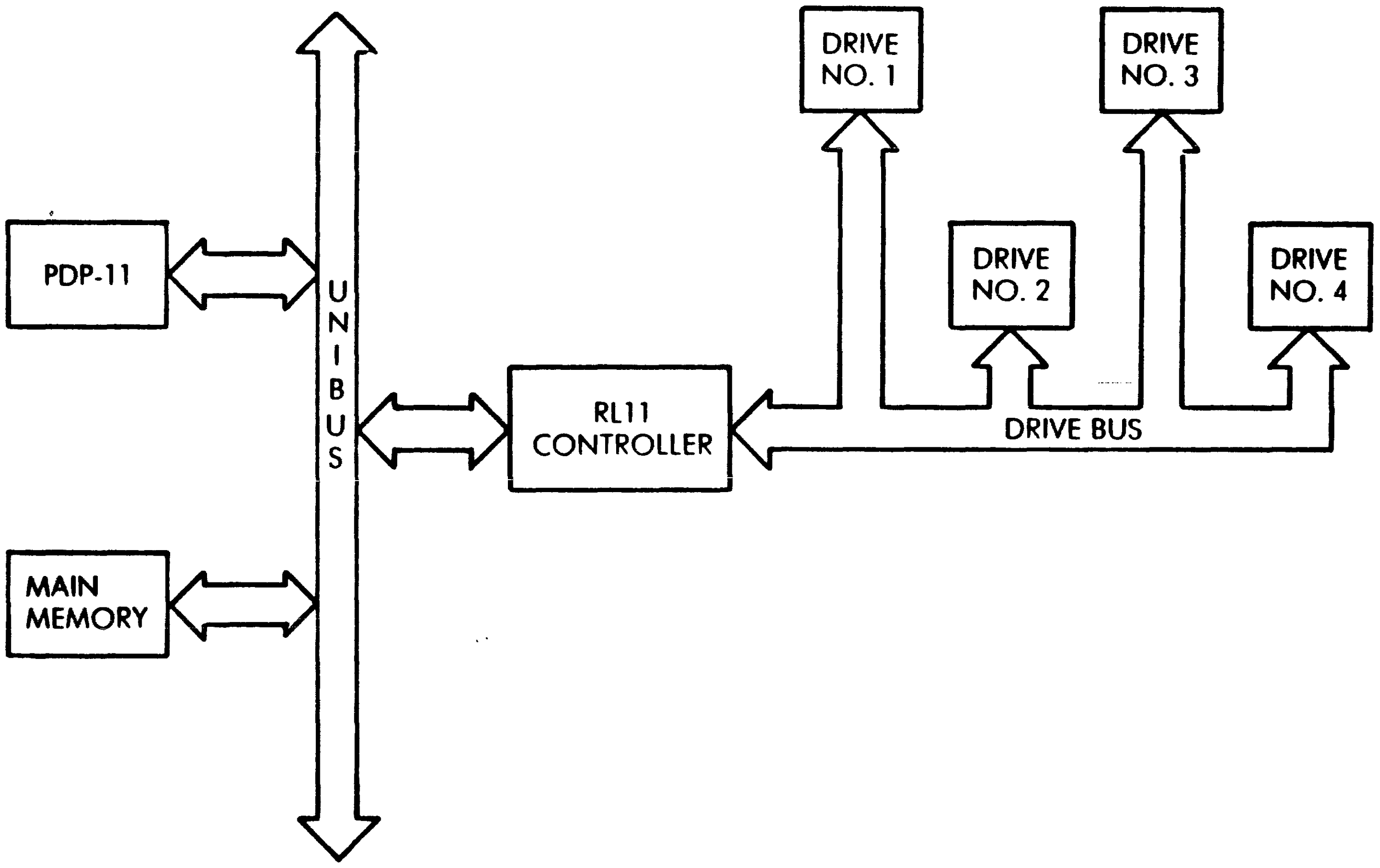
The drive has only two adjustments. All replacement items are relatively simple and can be quickly installed. The RL drive media are modified IBM 5440-type top-loading cartridges. These are single platter devices whose platters originate from the RK06/07 cartridges. One of the changes to the standard 5440 media is the modification to the sector-slot ring on the hub of the disk. This was changed from a 20-sector format to the DEC-required 40-sector format.

Another modification of 5440 media consists of writing disk formatter position servo information on the surface to aid the carriage-moving logic to position the heads accurately.

Figure 1 shows a typical subsystem.

Note that only four drives may be utilized on any of the controllers. If more drives are desired on an 11-CPU, a different set of addresses are selected to allow the presence of another controller with up to four more drives on the system.

The controller for standard PDP-11 processors is the RL11 and consists of one extended hex-height module. The M7762 PC board is a multi-layered, ROM-controlled device. It is installed wherever a hex-height Small Peripheral Controller (SPC) slot is available. Troubleshooting is done only to the board level since a logic analyzer would be required to find faulty components.



CZ-2062

Figure 1 Typical RL11 Mass Storage Subsystem Configuration

2

SUBSYSTEM ERRORS

DESCRIPTION

CE — Clock Error

This error occurs when the 8.2 MHz clock (generated by the controller for Write/Status strobing) is found to be missing. This error is not indicated by a software bit. The error that is flagged is Drive Error. (Illuminating the FAULT lamp on the drive).

CHE — Current in Heads Error

When write current is detected as flowing through the R/W heads without Write Gate being asserted, the result is a Current in Heads Error.

DCRC — Data CRC Error

This is an error that indicates a CRC error was detected while reading data.

DE — Drive Error

This is a reflection of the drive error interface line. If asserted, it indicates that the selected drive has flagged an error. To find out which one will require the execution of a Get Status command.

DLT — Data Late Error

This is an error that indicates a Silo-empty condition when writing; thus no word was available. If it occurs in a Read Data mode, the Silo memory was full, and unable to store another word from the drive.

DSE — Drive Select Error

This error indicates that more than one drive has the same unit number plug inserted.

ERR — Composite Error

This error indicates that one or more of the error bits (10-14) of the CSR is set (RL11/RLV11). In the RL8-A it indicates that one or more of the error bits in the Error Register is set. Errors that occur during command execution cause that function to terminate.

HCRC — Header CRC Error

This is an error that indicates a CRC error occurred when reading the header.

HNF — Header Not Found

This error occurs when the header does not match the contents of the DAR within the time allowed. Successive Header Reads will occur when the controller searches for the correct sector to read or write and will continue until a "match" is found or the error HNF occurs.

NXM — Non-existant Memory (RL11/RLV11 Subsystems Only)

This error indicates that the waiting for Slave Sync on the Unibus during an NPR took longer than the allotted 20 microsecond (SSYNC Time-Out).

OPI — Operation Incomplete Error

This error indicates that the current command was not completed within the allotted time.

SKTO — Seek Time-Out Error

This error occurs when:

- The Seek operation takes too long
- "Ready to Read/Write" was lost while locked on-track. (The servo is unstable.)

SPE — Spin Error

This error could mean either of the following:

- The disk motor failed to get up to speed during a head load sequence
- After the heads had been loaded, the disk motor was spinning faster than the engineering specification allows.

VC — Volume Check

This error is the only one that does not light the FAULT lamp on the drive. It occurs when the drive was cycled up to a heads loaded state, indicating that a possible cartridge ("volume") change has taken place. (Get Status command with Reset bit "ON" clears this for RL11/RLV11 subsystems; Reset command will clear this on an RL8-A subsystem.)

WCHK — Write Check Error

This error indicates that a failure has occured during a Write Check command. The data being read from the disk did not compare with the data from memory.

WDE — Write Data Error

This error indicates that no Write Data Pulses were detected by the drive Write logic after Write Gate was asserted.

WGE — Write Gate Error

This error is a result of Write Gate being asserted during one or more of the following times:

- The drive is not "Ready to Read/Write" (on-track signal)
- The drive is Write Protected
- The drive is in the midst of sector pulse time
- Drive has another error asserted

HEAD RETRACTION ERRORS

In addition to being able to test for the above errors, note that the power supply continuously monitors the status of the DC voltages. If any are found to be out of specification, "low" or failing, then an emergency retract of the heads occurs. The result is that no lamps on the front panel are lit, i.e., there is no power to light the lamps).

The heads will be retracted from the disk cartridge on four other occasions. These occasions involve four of the drive errors previously described:

1. Write Data error
2. Clock error
3. Current in Heads error
4. Result of the disk motor speed control circuits detecting the cartridge spinning too slowly.

IMBEDDED SERVO INTRODUCTION

RL01 DESCRIPTION

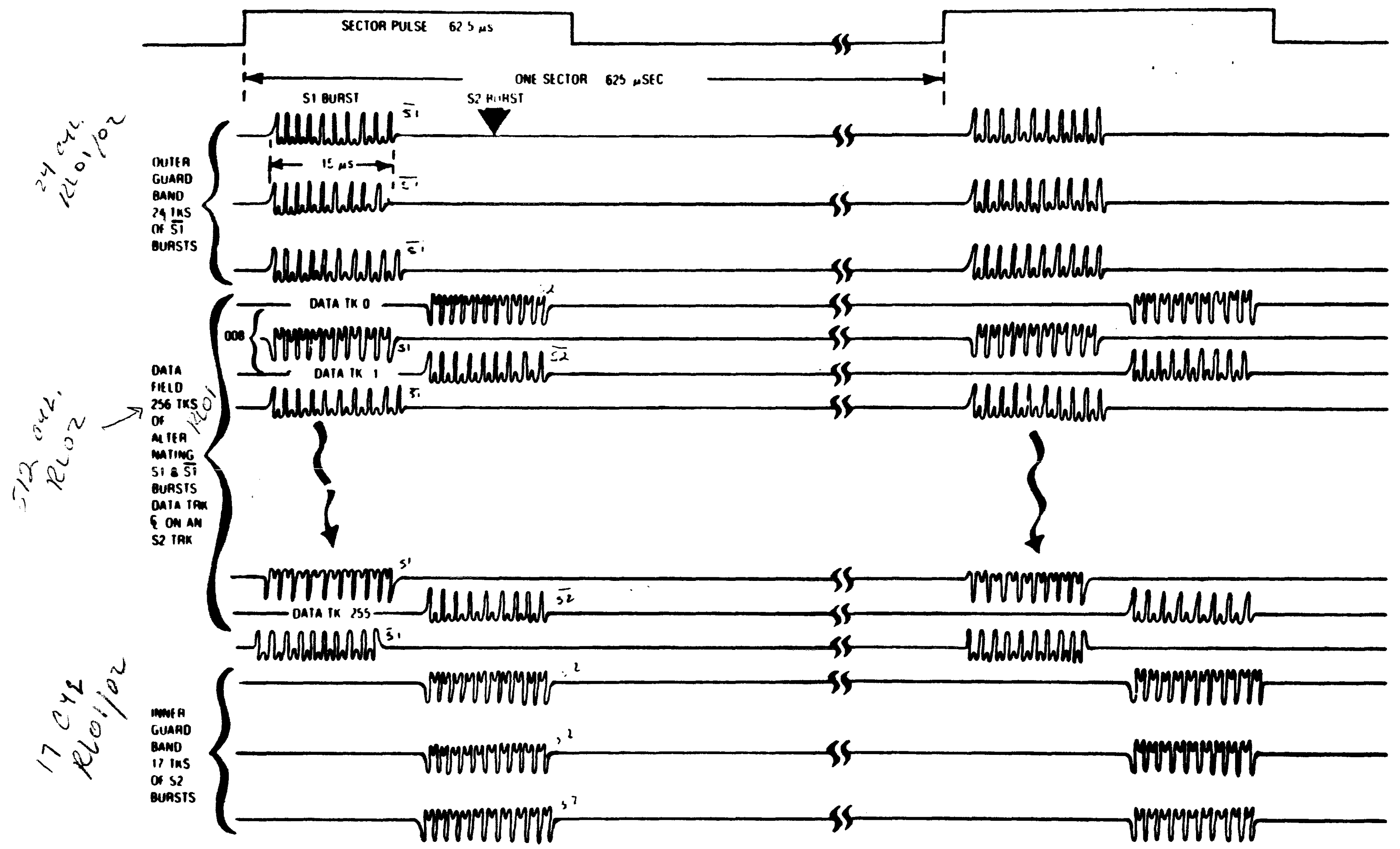
What Is Servo Data?

Figure 1 shows what the disk formatter writes. Using Figure 1, notice that under each sector pulse are two columns of waveforms. In the first "column" are the S1 bursts. They consist of two components which are given the mnemonics S1 and $\overline{S1}$, and are identified by their opposite polarities. The second "column" of waveforms are the S2 bursts, which contain the components S2 and $\overline{S2}$.

In the illustration, note that there are three tracks of information at the top contained within the sector pulses. These three tracks actually represent 24 tracks of consecutive $\overline{S1}$ servo data bursts. ($\overline{S1}$ servo data has more of the signal positive than negative.) These 24 tracks of $\overline{S1}$ servo data are grouped together to form what is known as the Outer Guard Band.

The Outer Guard Band is a 0.2 inch space on the disk, and is called the head-loading zone.

Following the head-loading zone is an area on each surface consisting of alternating S1 and $\overline{S1}$ bursts. The alternating of these bursts comprises the 256 track data area of the disk surface (about two inches). Also found in the data area are S2/$\overline{S2}$ bursts which are displaced in time from, and written between, the S1/$\overline{S1}$ bursts. S2 bursts start in the data area and continue into the Inner Guard Band area (where S1/$\overline{S1}$ bursts are nonexistent).


SECTOR PULSE 62.5 μs
ONE SECTOR 625 μSEC
S1 BURST
S2 BURST
15 μs
OUTER GUARD BAND 24 TKS OF $\overline{S1}$ BURSTS
DATA TK 0
008
DATA TK 1
DATA FIELD 256 TKS OF ALTERNATING S1 & $\overline{S1}$ BURSTS DATA TRK ℄ ON AN S2 TRK
DATA TK 255
INNER GUARD BAND 17 TKS OF S2 BURSTS
24 cyl. RL01/02
512 cyl. RL02
RL01
17 cyl. RL01/02

S1 Burst is Between Even Track going ODD

Figure 1 RL01 Servo Data Format

$\overline{S1}$ " is Between ODD Track Going Even

S2's are found on Track Centers for DATA

S2 is Even DATA TRK

$\overline{S2}$ is ODD DATA TRK

How Are These Servo Bursts Handled?

The following examples illustrate how the R/W heads handle these servo bursts. If a Read head is centered directly over an S1 burst, the head waveform appears as shown in Figure 2.

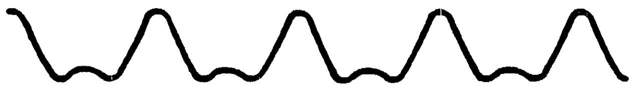


Figure 2 $\overline{S1}$ Burst

If the carriage moves over an S1 burst, the Read signal will look similar to Figure 3.

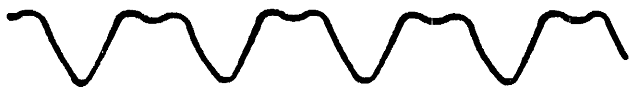


Figure 3 S1 Burst

Figure 4 shows what happens to the read signal when the R/W head is centered directly between the the S1 and $\overline{S1}$ bursts.

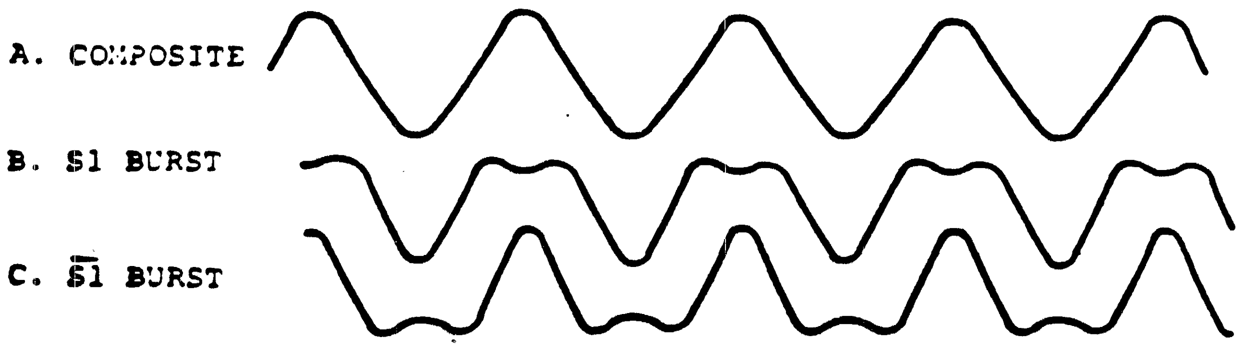


Figure 4 Composite Burst

RL11 BLOCK DIAGRAM

INTRODUCTION

The RL11 is a single hex-height module that can be installed in any Unibus Small Peripheral Controller (SPC) slot. This controller can communicate with up to four drives, although it cannot transfer data to more than one drive at a time. Electrical connections between the controller's J1 and the drives are made in a daisy-chain fashion.
are:

- Issuing head positioning commands
- Controlling the flow of data between the Unibus and the drives
- Controlling the flow of status between the selected drive and controller
- Providing error flags and fault indications related to overall subsystem operation.

Figure 2, the RL11 Basic Block Diagram, is an overview of these functions. This diagram illustrates the various circuit components used in the RL11. This will aid you in subsystem troubleshooting by separating the tasks performed into two categories: controller and subsystem drive tasks.

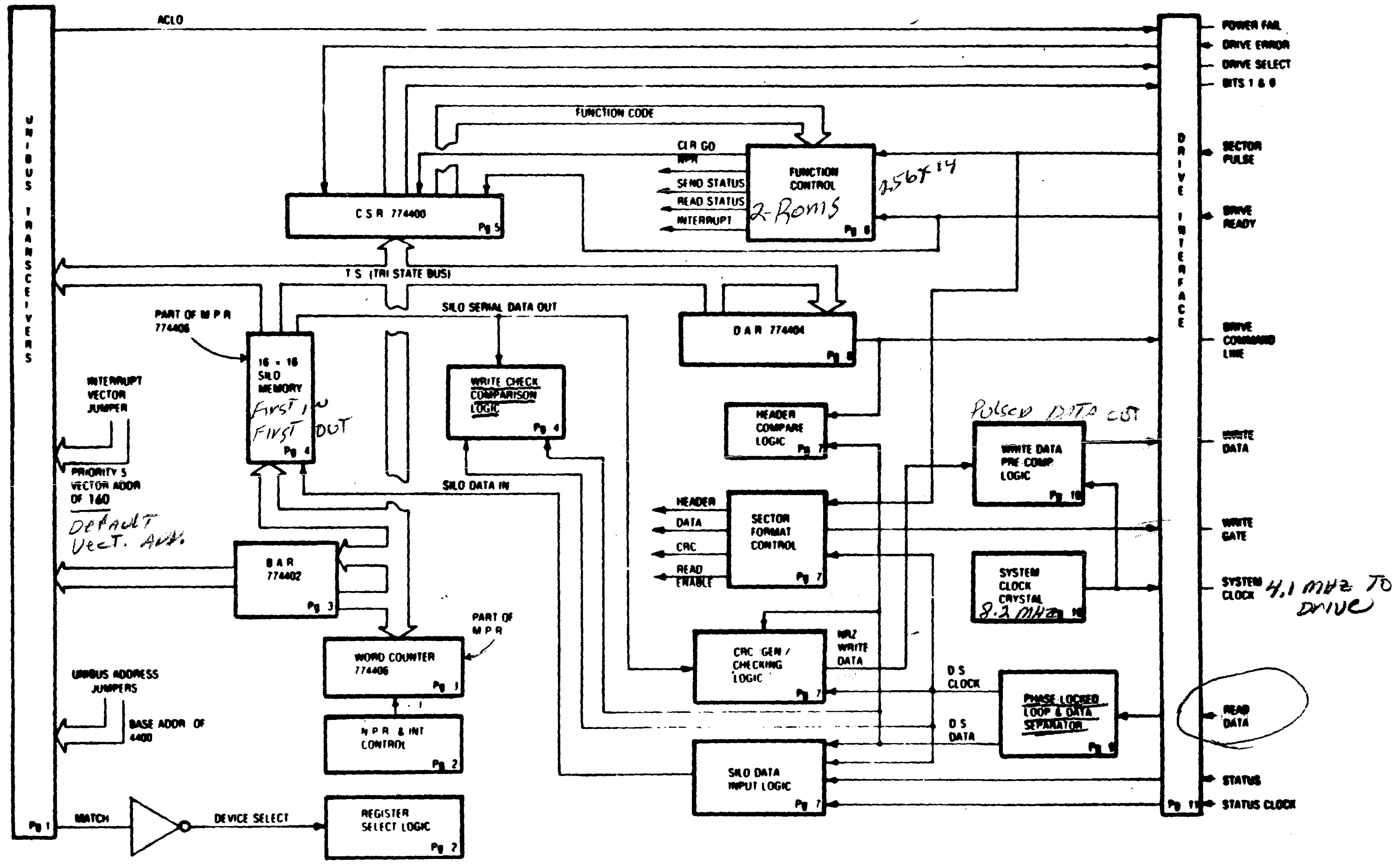


Figure 2 M7762 RL11 Basic Block Diagram

VECTOR ADDRESS - 160

10

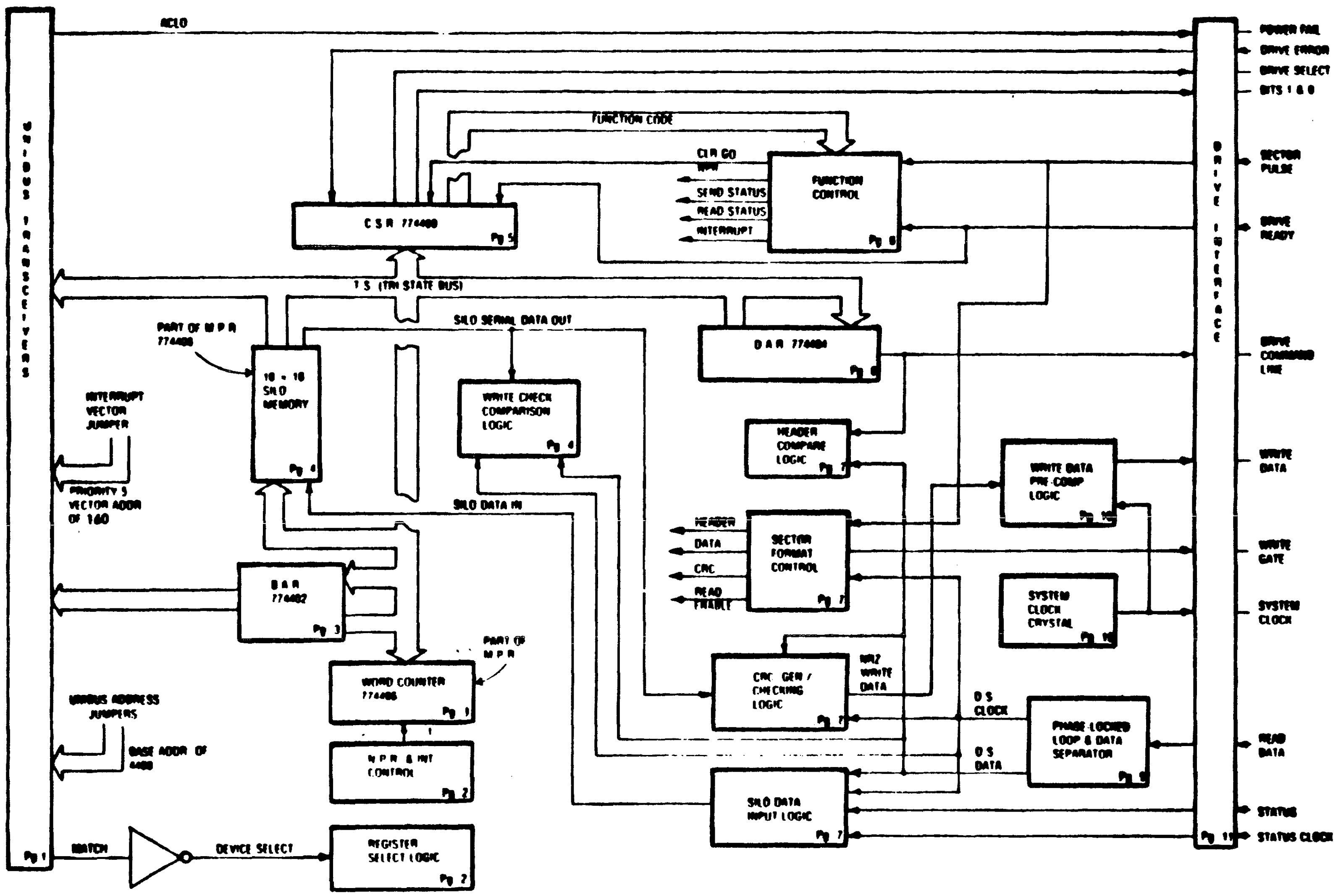


Figure 2 M7762 RL11 Basic Block Diagram

12

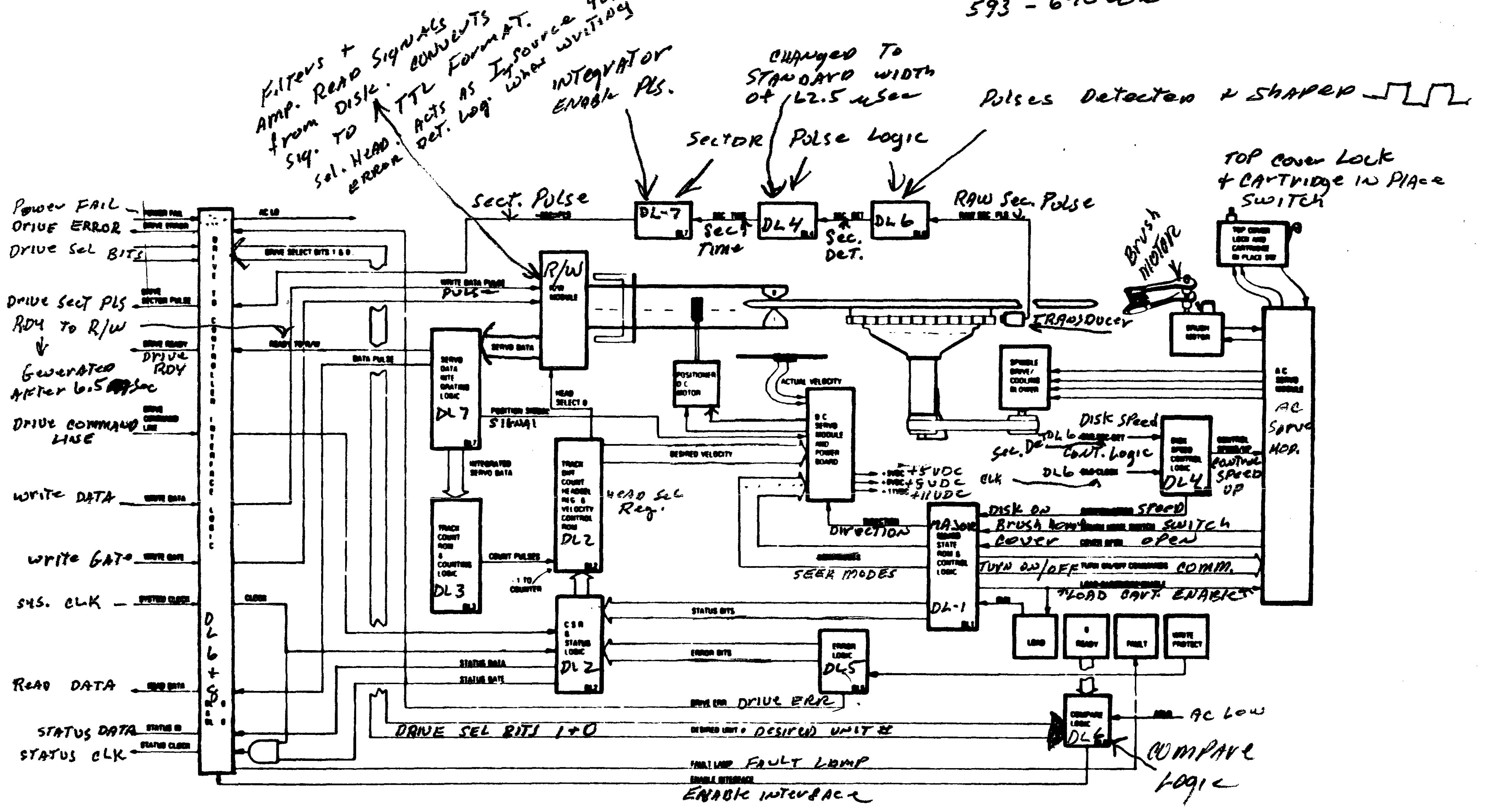
RL DRIVE BLOCK DIAGRAM

INTRODUCTION

The RL Disk Drive consists of Field Replaceable Units ranging from PC boards to the head positioning unit. The block diagram discussed in this module will give you, as a Field Service Engineer, an understanding of the major components that comprise the RL Drive. If you tie in what you have learned in the past modules about the controller, you will know enough about the subsystem to aid you in troubleshooting.

The RL Drive basic block diagram is shown in Figure 1. It contains all of the components needed for an overview of this disk drive. The written material helps you identify the function(s) of these individual components. On the block diagram there are print set page numbers, to help you correlate the block diagram with the logic circuits.

593 - 640 usec for Sect. Speed



+5 VDC is generated from +8VDC

Figure 1 RL Drive Block Diagram

Seek modes - Velocity mode
Position mode

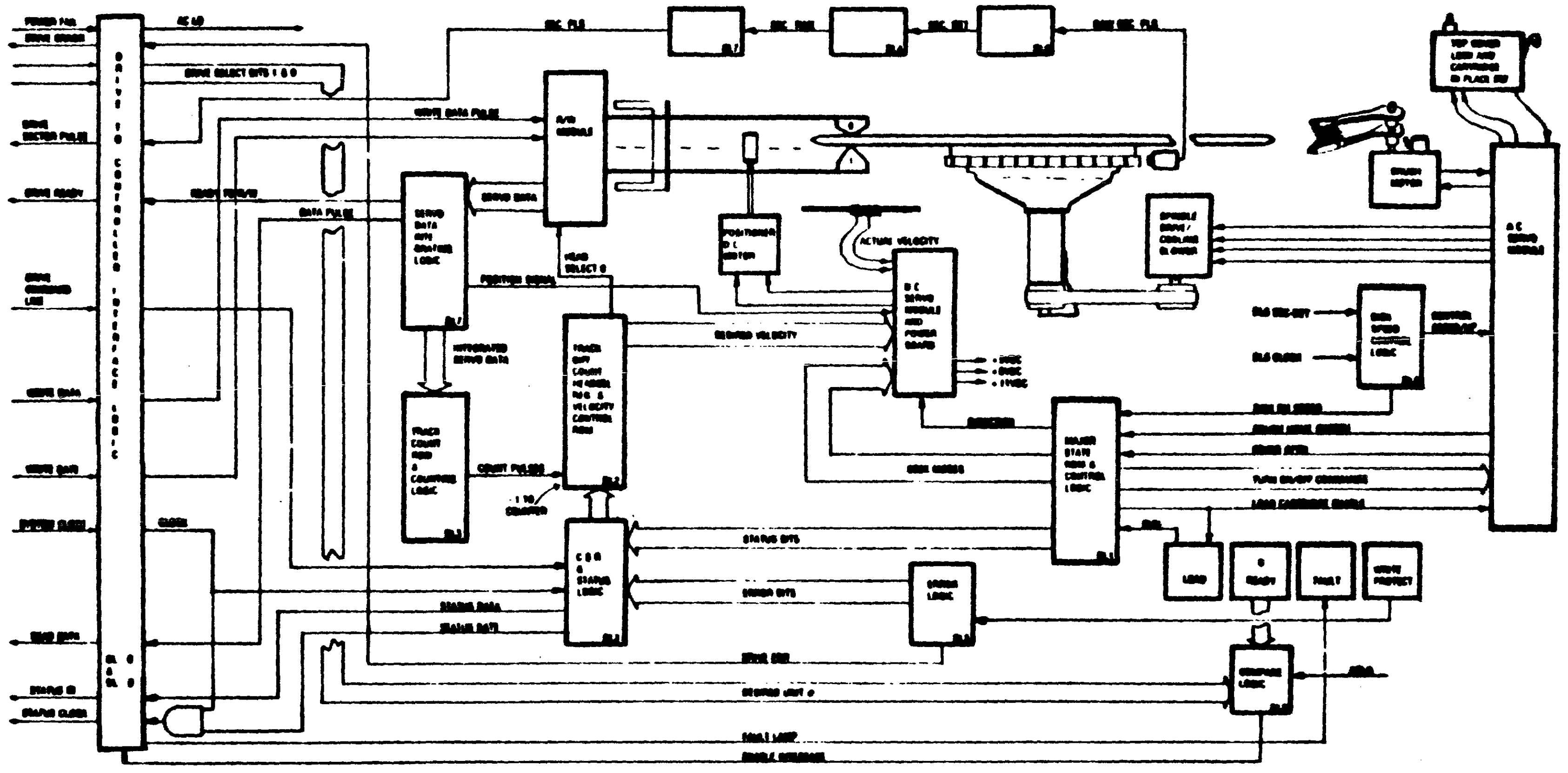


Figure 1 RL Drive Block Diagram

SERVO BLOCK DIAGRAM

POSITION MODE

To understand this mode, first review the servo data waveforms, that were covered in the "Imbedded Servo Introduction" module. Refer to Figure 1.

Under the sector pulse are two vertical columns of waveforms: the S1/$\overline{S1}$ and S2/$\overline{S2}$ servo bursts. The customer data tracks are centered between the S1/$\overline{S1}$ bursts. The drive integrates these bursts to provide the carriage "position" signal.

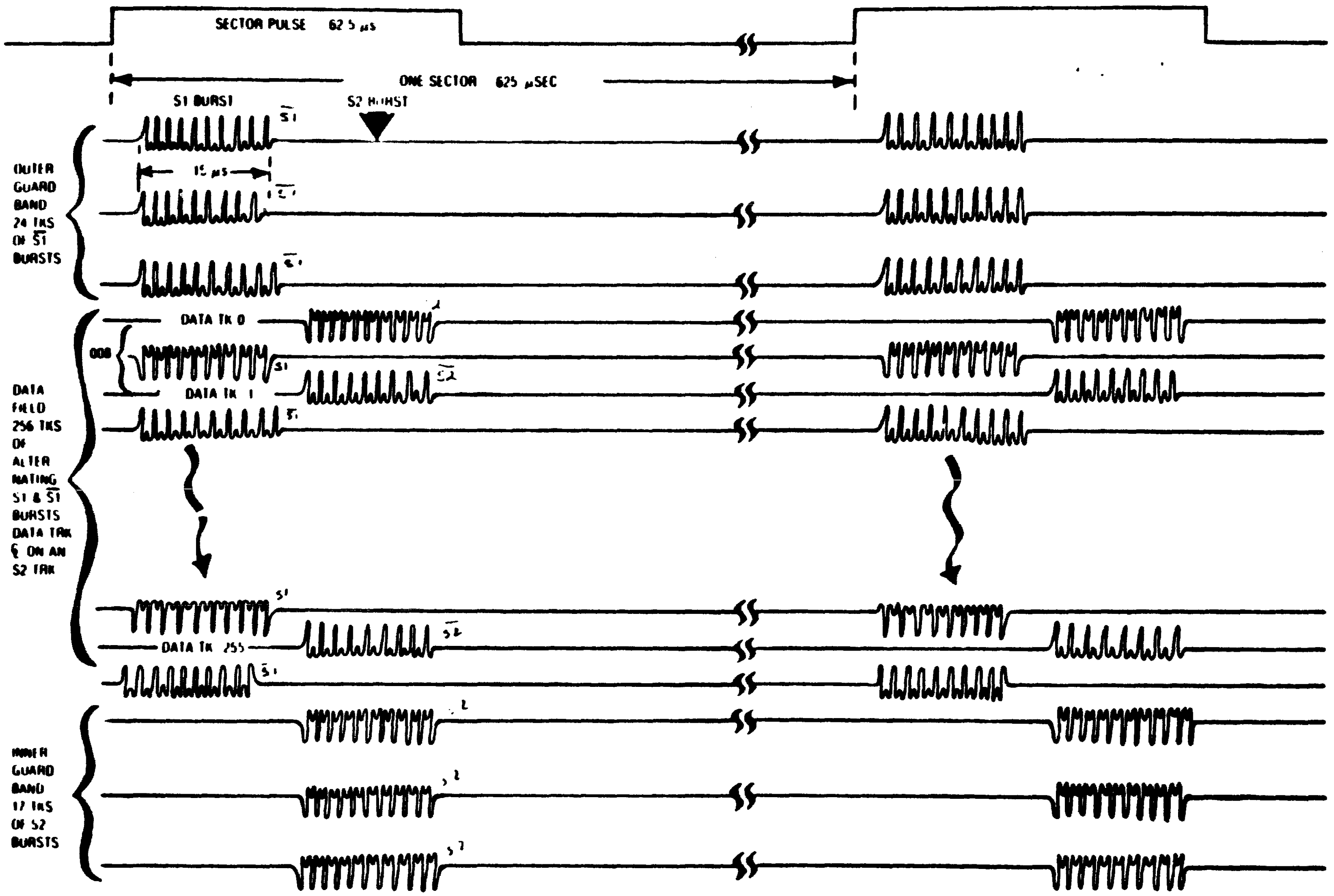
The S2/S2 bursts on the data track centerline provide the proper polarity (carriage direction) to the servo system correction signal.

While reading over this description, refer to Figure 2. The description will encompass two examples of Position Mode operation. In the first, the carriage will be "on track", while the second illustrates what happens when the carriage is "off track".

Example 1:

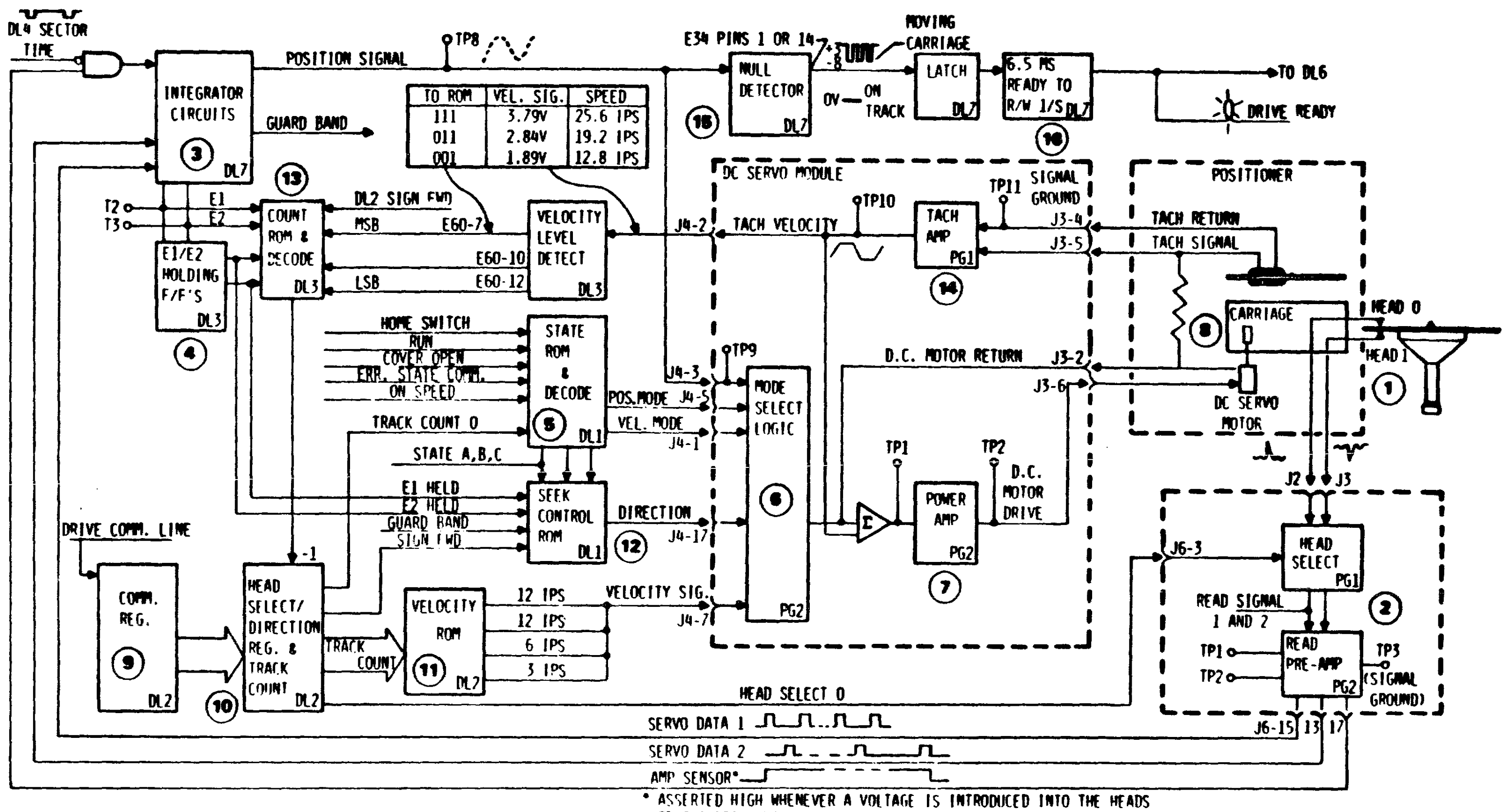
While on track, the servo signals are being sensed by the selected read head (1). The resultant composite waveform is shown in Figure 3.

From the read head, the signals go to the head select/read pre-amp sections of the R/W module (2). The circuit block labeled "Head Select" contains a differential amplifier that outputs a differential signal pair for each analog signal sensed. The outputs are arbitrarily named Read Signal 1 and Read Signal 2. These amplified and filtered signals are then applied to a pair of zero-volt crossover detectors, which convert the analog waveshapes into two digital pulse trains called Servo Data 1 and Servo Data 2 (See Figure 3).



SECTOR PULSE 62.5 µs
ONE SECTOR 625 µSEC
S1 BURST
S2 BURST
15 µs
OUTER GUARD BAND 24 TKS OF S̄1 BURSTS
DATA TK 0
DATA TK 1
ODD
DATA FIELD 256 TKS OF ALTERNATING S1 & S̄1 BURSTS DATA TRK ℄ ON AN S2 TRK
DATA TK 255
INNER GUARD BAND 17 TKS OF S2 BURSTS

Figure 1 RL01 Servo Data Format



TO ROM	VEL. SIG.	SPEED
111	3.79V	25.6 IPS
011	2.84V	19.2 IPS
001	1.89V	12.8 IPS

* ASSERTED HIGH WHENEVER A VOLTAGE IS INTRODUCED INTO THE HEADS OF A SUFFICIENT AMPLITUDE AS TO CONSTITUTE A "REAL" SIGNAL.

Figure 2 Track — Following Servo Block Diagram

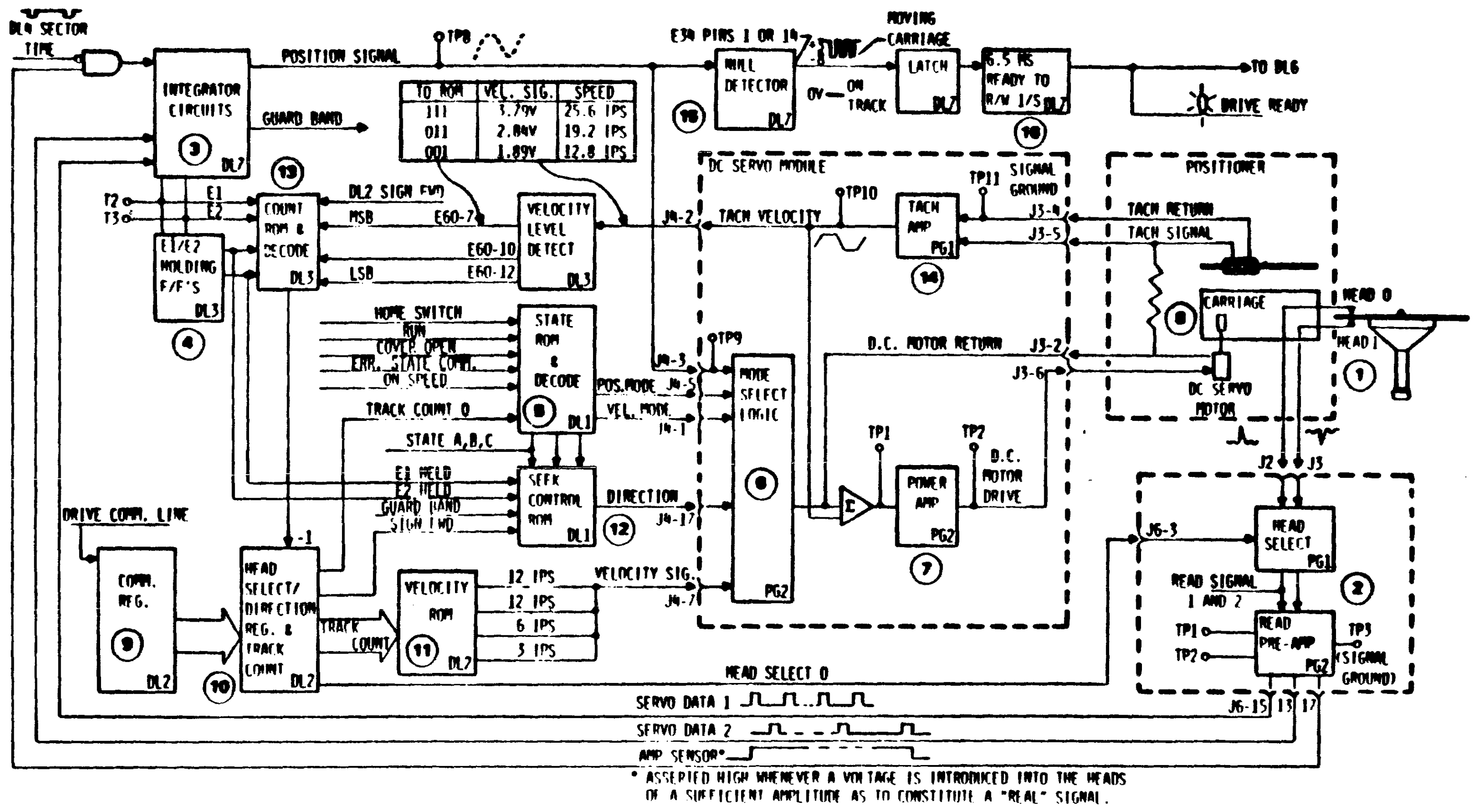


Figure 2 Track — Following Servo Block Diagram

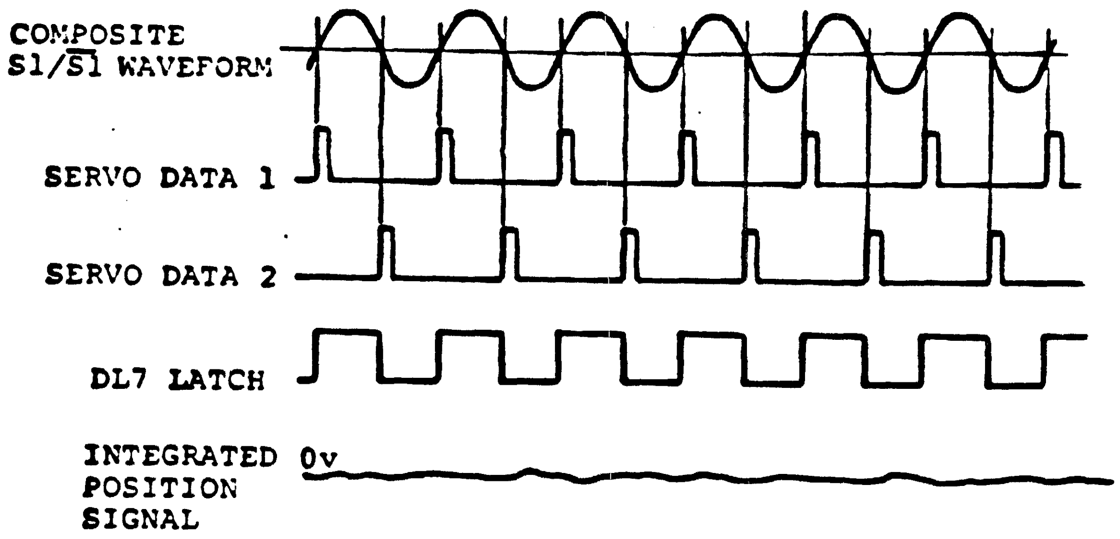

COMPOSITE S1/$\overline{S1}$ WAVEFORM
SERVO DATA 1
SERVO DATA 2
DL7 LATCH
INTEGRATED POSITION SIGNAL
0v

Figure 3 Integration of S1/$\overline{S1}$ Composite

The servo system uses these pulse trains for the positioning of the carriage. The pulse trains are then integrated ((3) in Figure 2). Entering the integrator circuits (3) is another signal from the R/W module. AMP SENSOR is a high-true level whenever a voltage coming from the selected head is of sufficient amplitude to constitute a "real" signal (instead of noise spikes, glitches, etc.). This signal enables the start of the integration process.

The servo data 1 and servo data 2 pulse trains alternately set and clear a latch in the integrator logic ((3) on Figure 2), producing a waveform like the one in Figure 3 (DL7 Latch). It will only be square (as illustrated) if the pulse trains are exactly 180 degrees out of phase with each other. This happens when the selected head is directly between an S1 and an $\overline{S1}$ burst. (The head will be sensing equal portions of both waveforms). The resulting DL7 latch squarewave output is sent to the integrator, whose R-C network alternately charges and discharges, producing a DC voltage level called the position signal. In our example, the heads are centered between the servo data, resulting in a position signal of zero volts. (See Figure 3).

As shown on the block diagram, the position signal leaves the right side of the integrator block and descends the page to the DC Servo Module (6). It is then routed through to the summing amplifier because the major state ROM (5) has asserted position mode. Position mode is active when track count = Ø. The summing amp does not modify this position signal because there is no active carriage velocity. The result is the driving signal to the power amplifier (7). In our example the position signal is zero, which indicates that no change in the carriage position is desired.

Example 2: In this example, the carriage will be off-track centerline by 1/2 track.

NOTE

On a working system the carriage could not get this far off-track. This is a learning example whose waveforms are easier to visualize because they are exaggerated.

Refer to Figure 4.

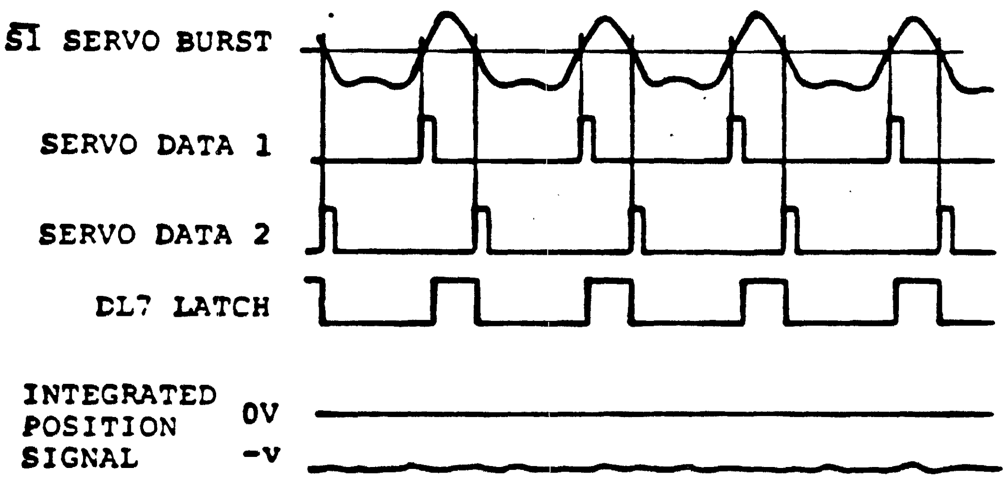


Figure 4 Integration of an $\overline{S1}$ Servo Burst

The two pulse trains generated are not symmetrical. The DL7 latch does not set and clear at even intervals. Thus the integrator's R-C network charges more in one direction than the other. The position signal resulting is a negative DC voltage. This position signal is fed to the DC Servo Module (6) as a servo error correction signal.

In parallel with that, the integrated position signals are converted into TTL logic signals via a pair of voltage comparators and given new names.

Example: $\underline{S1}$ = $\underline{E1}$ in Logic "1"
$\overline{S1}$ = $\overline{E1}$ in Logic "Ø"

The E2 signal is developed the same way using another comparator.

Example: $\underline{S2}$ = E2 in Logic "1"
$\overline{S2}$ = $\overline{E2}$ in Logic "Ø"

Both of these signals (E1/E2) are then routed to a pair of holding flops (4) whose output names are "E1 Held" and "E2 Held". The E2 Held signal is routed down the page to the Seek Control ROM (12). This causes the ROM to assert or negate the "direction" signal. This means that the S2 bursts control the direction of the carriage movement when in position mode. Without them, the mode Select Logic (6) would always apply the error correction in the same direction to the summing amps and power amps.

VELOCITY MODE

This is the mode that moves the carriage from track to track during Seek commands. The controller loads the drive's command register (9) prior to the seek with the following:

- Track difference
- Head address (or number)
- Carriage movement direction
- Seek code

On page DL2, the logic transfers this information to the appropriate holding flops (1Ø). The new track count negates "track count Ø". This causes the state ROM (5) to negate position mode and assert velocity mode.

The track count is sensed by the velocity ROM (11), which has a D/A converter at its output. The resulting velocity signal is applied to the DC Servo Module (6), along with the velocity mode and direction signals. SIGN FWD, from the direction register (1Ø), generates the signal "direction" (Seek Control ROM (12)).

All the signals necessary to initiate head movement are now in place. The summing amplifier turns on the power amplifier (7), causing current to flow into the DC motor (8).

With the heads now moving, two separate operations take place. In one, the tachometer (14) monitors the carriage velocity by inducing current in a coil that moves with the carriage. The faster the carriage moves across the flux lines of the permanent magnet below it, the more current is induced. This current is then amplified in the DC Servo Module and applied back to the summing amplifier to govern the desired velocity. While the velocity ROM only informs the logic of how fast it should go, the tach feedback ensures that the DC motor moves the carriage at that speed.

This tachometer signal is simultaneously applied to an A/D converter in the Drive Logic Module. This decoder outputs a 3-bit code representing the velocity to the track-counting ROM (13). This input, along with the information about the direction of carriage movement and the past and present E1/E2 codes, allows the ROM to decide how many tracks have been crossed between servo burst samples. (Servo bursts only occur once per sector, approximately 560 microseconds apart.)

The second major operation taking place is obtaining the E1/E2 codes. The R/W heads route the servo data to the R/W Module (2). In the R/W Module only one of the head signals is selected and differentially amplified. It is then converted to a TTL digital format and sent to the Drive Logic Module for integration (3). The resulting integrated signal is then converted to a binary code. Recall that E1 is the result of an S1 burst and that when S2 is then integrated, it is converted to E2. The E1 and E2 signals are then stored in holding flops (4).

The next available servo burst that the selected head can read may be on the same track or as far away as three tracks. The E1/E2 codes and the tachometer feedback help to clarify just how many tracks were crossed so that the appropriate number of count pulses are applied to the track difference counter (10).

As the track difference counter value goes down, the velocity signal to the DC Servo Module is modified to reflect the new desired velocity. The new velocity is sensed by the summing amplifier and is compared with the tachometer feedback. The power amplifier then applies the new current value to the positioner motor.

This cycle continues: track crossings decrement the difference counter, a new desired velocity slows the carriage, and the tachometer and servo bursts combine to calculate the number of count pulses to decrement the counter.

When the track difference counter reaches zero, the major state ROM switches the output from velocity mode to position mode. This disables all use of desired velocity for carriage positioning. The servo position signal now has the responsibility for controlling the carriage.

The carriage is stopped when position mode is reached at a track count of zero. The first time the position signal crosses the desired track center line, a null detector (15) fires the Ready to Read/Write one-shot (16). At the completion of this 6.5 millisecond time period the controller is notified that the drive is ready.

RL02 DIFFERENCES

The RL02 has a new design for the Drive Logic, DC Servo and R/W modules. These new modules were designed to be installed in a RL01 or RL02 with on-board jumpers cut or installed at the factory. When replacing these boards from spares, make sure the jumpers are cut property for that drive. Figure 2 is still representive of the RL02 design with the exception that the test points have changed for the new board. Only a few circuits have been added to accommodate the double density disk.

Subsystem Commands

LESSON 1: GET STATUS COMMAND

GENERAL INFORMATION

This command can be used to obtain data about an error that has just occurred. The controller issues the Get Status instruction which is relayed to the drive. The drive responds by loading a register with its error/status information. This information is then serially shifted back to the controller, where the software is notified that the status word is ready. The software can then read this status word to find the source of the drive error. This command can be performed at any time the controller is "Ready". The drive, however, does not have to be "Ready" (on track).

PROGRAMMING

To be an effective troubleshooter, you must know all facets of the subsystem, including the software. Before the controller or drive can respond to a command, the software must assemble the instructions it is to perform.

The Get Status command consists of these three steps:

1. Writing into the DAR to make a status request
2. Writing into the CSR to transmit the request to the drive
3. Reading the MPR to obtain the status word.

Listed below are sample steps that could be used to obtain this status word:

```
      Mov#000003,@#774404    Write in the DA register the
                             code to make a status word
                             request.

      Mov#000004,@#774400    Write in the CS register the
                             command and go bit.

Loop: TSTB@#774400           Test controller and loop
      BPL Loop .             ready.

      Mov@#774406,R0         Get the status word from
                             controller and hold in R0.

      HLT
```

Get Status Command

HARDWARE

Now that you know the steps involved in retrieving a status word from the disk, you will next learn what happens in the hardware. Use the listed program and Figure 2 as references for this lesson.

The code ØØØØØ3 (status word request) arrives at the left side of Figure 2 on its way to the DAR from the Unibus transceivers (1). The status word request is loaded into the DAR (2). The next instruction loads the CS register (3) with the Get Status function code. This code and the Go bit activate the function control (4). Function Control initiates the timing to shift the DAR (2) serially to the drive's CSR (5) via the drive command line.

When the drive's logic recognizes the status word request, it clears and then parallel-loads the CSR (5) with the status and error bits (6). By referring to Figure 3, you can see status and error bits which are loaded into the CSR in the performance of the Get Status command.

When the CSR (5) is loaded, the status clock (7) shifts the status data to the controller. The incoming status data is multiplexed by the silo data input logic (8) and then deposited into the Silo (9).

When the status word reaches the Silo, the Controller Ready bit is set, and the program executes the last instruction. When this last MOV is executed, the status word moves out of the Silo (9) onto the tristate bus and through the Unibus transceiver (1) to the CP.

NOTE

If bit Ø3 of the status word request is set, then the status/error bits in the drive are cleared before shifting to the controller. Using this bit in a Get Status command allows determination of hard errors or clearing of Volume Check status.

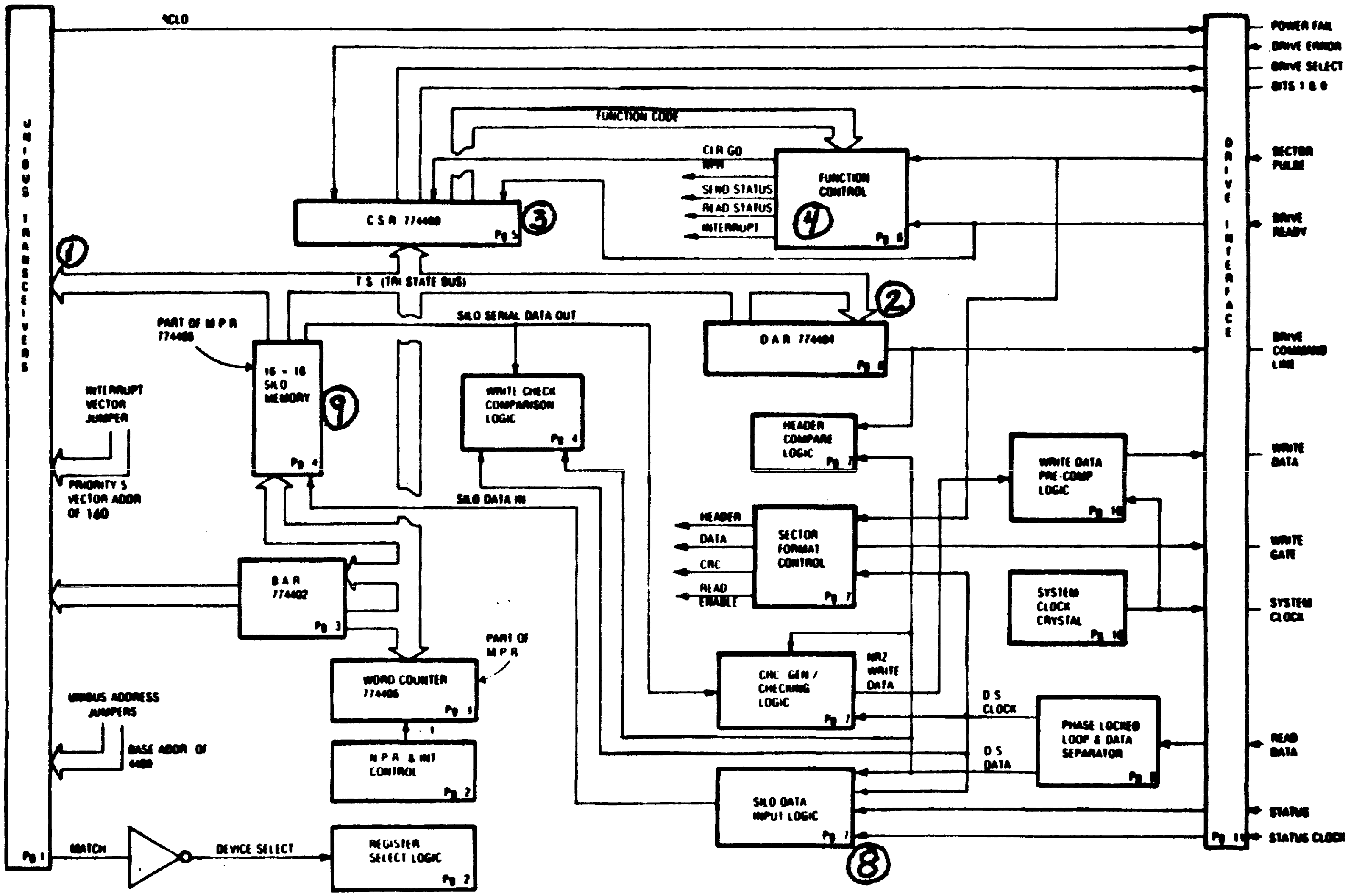


Figure 2 M7762 RL11 Basic Block Diagram

29

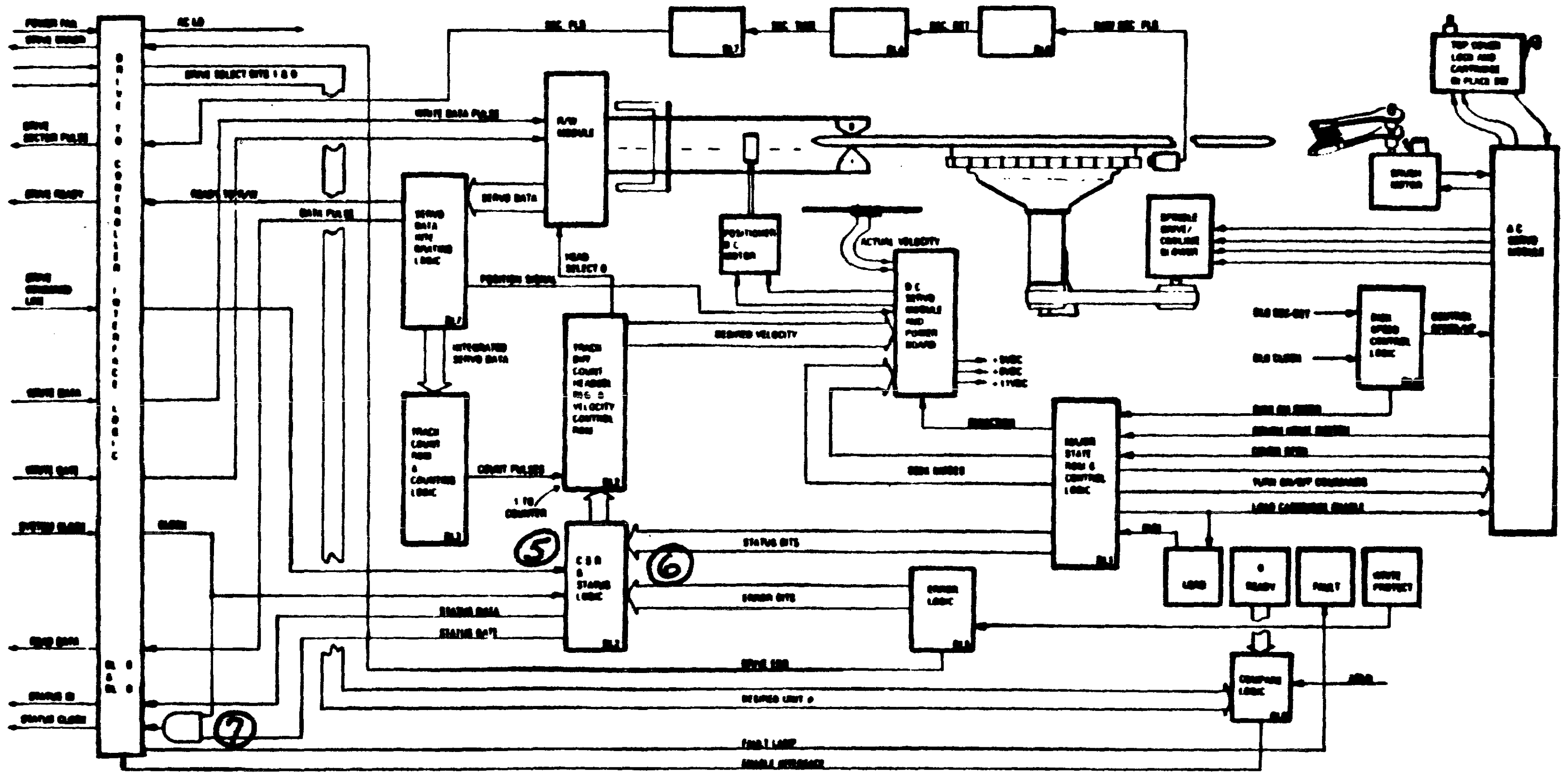


Figure 2 RL Drive Block Diagram

30

LESSON 2: SEEK COMMAND

GENERAL INFORMATION

Because of the simplicity of the controller and drive logic, no implied Seeks are performed by this subsystem. Thus, the Seek command must be issued in order to move the carriage from one track to the next and/or to select another head. Once the drive has the necessary data to accomplish this seek, it may be de-selected. The controller is then free to select another drive. It is entirely possible for all the drives to be seeking at one time.

PROGRAMMING

Before issuing the Seek command, the software must calculate the head selection, the cylinder difference, and the direction of carriage movement. We will not discuss the software that is necessary to perform this calculation here, although you will be made aware of what the software must do to initiate the Seek command itself.

The following program will supply the drive with the cylinder difference, head selection, and carriage direction:

MOV#DIFF,@#774404	Load the DAR with the track difference, direction, head number and Seek code, i.e., 000625 = move forward 3 cylinders using lower head.
MOV#000106,@#774400	Load the CSR with a Seek command, selecting Drive 0 and enabling interrupts. Clear the Controller Ready bit (Go).

These two instructions initiate the Seek. When the drive receives the Seek data, the controller raises the interrupt flag to let the CPU know that the subsystem is free to perform commands on other drives. At the completion of the Seek, the Drive Ready bit in the CSR is asserted, but no interrupt occurs.

Seek Command

HARDWARE

Figure 4 illustrates what the hardware does in the performance of a Seek command. This lesson discusses only the initiation of the Seek. In-depth study of how the drive actually moves the carriage and controls its motion is covered in the "Servo Block Diagram" module.

The first instruction loads the DAR (1) with the track difference, head number, direction of travel bit, and the Seek code. The second instruction loads the CSR (2) with the Seek command.

Clearing the Controller Ready bit causes the function control (3) to start the timing necessary to shift the seek data serially out of the DAR (1) onto the drive command line to the drive.

The path taken for a seek is the same as the Get Status command. The drive command line is clocked to the drive's CSR (4). As soon as the function control shifts all the bits out of the DAR the interrupt flag is raised and Controller Ready is re-asserted.

Because the first bit (0) in the drive command word is a one, the drive's logic recognizes when the CSR is filled. The logic then inspects bit one of the register. If it is a zero, then a Seek code is recognized instead of a Get Status code (in which bit one equals a one).

At this point the track difference, head number, and direction bits are parallel-transferred to their respective holding registers. The drive logic now starts the seek.

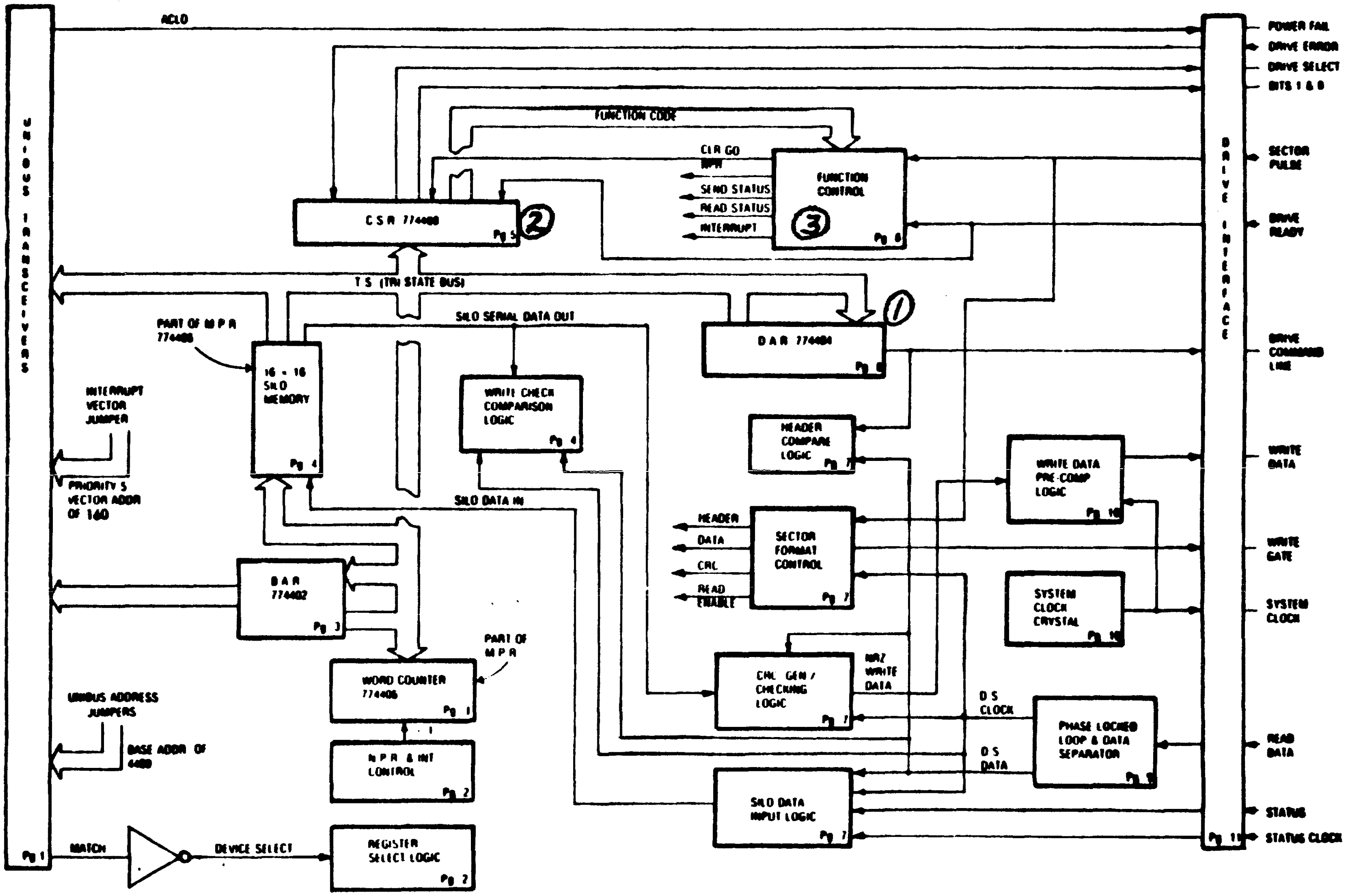


Figure 4 M7762 RL11 Basic Block Diagram

W
W

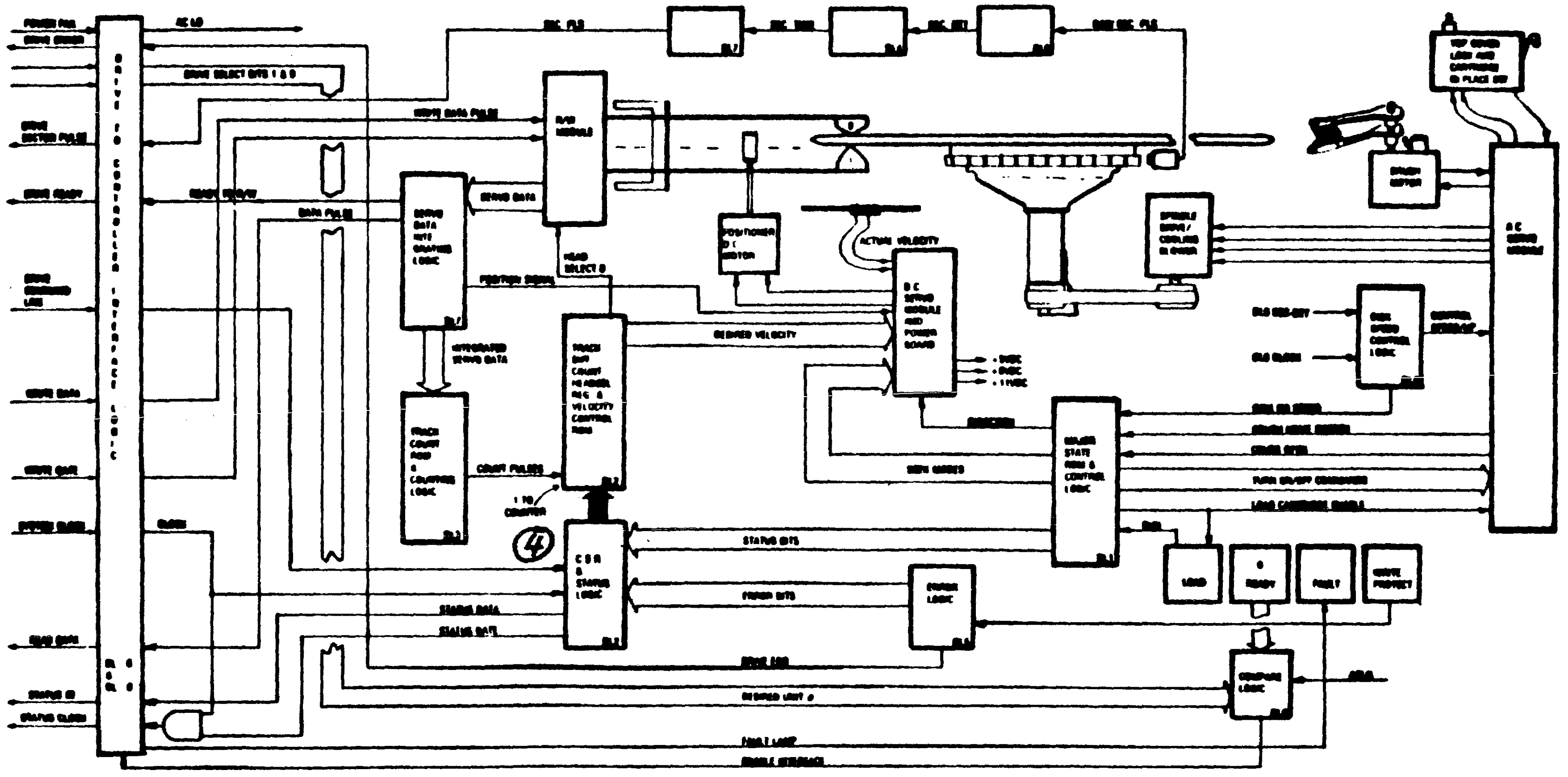


Figure 4 RL Drive Block Diagram

34

LESSON 3: READ DATA COMMAND

GENERAL INFORMATION

The drive is functionally capable of performing only three tasks:

1. Reading
2. Writing
3. Moving the carriage.

If the drive has been selected, the interface logic has also been enabled, allowing the data read to pass to the controller. Since Write Gate is unasserted, reading automatically occurs. The controller then is responsible for computing when the read data is desired. Additional duties of the controller include interpreting what the read data is, i.e., Header, Header CRC, Data, Data CRC, etc.

PROGRAMMING

The previous lesson showed that if the carriage is <u>not</u> over the correct track, or the wrong head is selected, the software must do a Seek command to select the new head and/or move the carriage before the Read Data Command can begin.

Following is a sample routine to be used to read one sector of a customer's data, assuming that the heads are over the desired track.

Mov#2000,@#774402	Load 2000 (first memory location) into BA register.
Mov#000303,@#774404	Load the disk address (cylinder 1, head 1, and sector 3).
Mov#177600,@#774406	Load two's complement of one sector worth of words into a Word Counter.
Mov#000114,@#774400	Load CS register with Read Data command, clear Controller Ready, set Interrupt Enable, and select drive 0.

At this time the software can execute another program, or wait for a Controller Ready condition. The Read Data command is reading more than one sector at a time, although it cannot go beyond the selected track. Data transfers that exceed the last sector of any track must be reprogrammed. The Status register can then be checked at interrupt request time to see if any errors were encountered (if desired).

→

HARDWARE

Using Figure 5, note that circled numbers occur beside some of the unshaded blocks as well as the shaded. The shaded blocks are in the direct path for the execution of the command, while the unshaded blocks denote the logic that must be present in order to execute the command. Unlike the Seek command, the drive has very little to do during Read Data in comparison to the role played by the controller.

This lesson will concentrate on the controller--its initiation of the command as well as its keeping track of the timing to completion of the command.

Use the routine generated at the beginning of this lesson as well as Figure 5.

The instructions load the following registers via the tristate bus:

1. Bus Address register
2. Disk Address register
3. Word Counter
4. Control and Status register

In loading the CSR, the Ready bit in the register is cleared (Go) and the command initiated. The Drive Select bits engage the desired device (one of four), which releases read data and sector pulses to the controller. Sector pulses enter the function control logic (5) and sector format control (13) to begin execution of the command.

In the drive, the data from the disk surface is read by the selected head (selected via the previous Seek command). It is amplified, filtered, and reshaped by the R/W module (6). The module's output signals are named <u>servo data,</u> regardless of what information is being read at the time. Any Servo Bursts that are read are applied to the integrating circuits where they are used to position the carriage. The header and data words are passed through the integrating logic block where it becomes <u>data pulse</u> to the interface logic. Another name change occurs here, and the <u>read data</u> is applied to the controller. The read data could consist of header information, customer's data, or CRC checkwords.

At location (8) on the block, the data read is processed through the PLL/data separator. This PLL, upon detection of the trailing edge of a sector pulse, will lock its oscillator to the rate of the preamble zero areas of each sector. Sector format control, (13) via its counting circuits, establishes the time for enabling the data separator (8). When the marker (Sync bit) is detected, header comparison begins. The data being read is now the header from the disk and is compared (9) to the contents of the DAR (2). Simultaneously, these data bits are utilized in the CRC generator/checker block (10) to check the accuracy of the read.

If the Header Compare logic (9) does not find a match on the first header word read, the controller goes into a wait mode until the next sector pulse, when another attempt will be made.

At some point on the revolving disk, one of the headers being read should match. This indicates that the correct sector address has been found, that the positioner is over the desired track, and that the desired surface has been selected.

When the header word compares, the second and third header words are then read into the CRC block (1Ø). The data separator is turned off (8), and a check made for a header CRC error.

If a Header CRC error is detected, bits 1Ø and 11 of CSR (4) are set, along with bit 15. This will return control of the operation to Function Control (5), terminating the command.

If no CRC error is detected on the header, the PLL (8) is again activated to allow the preamble to synchronize the PLL. The data separator logic looks for the marker (Sync bit) at the end of the preamble zero field. When found, data bits are then shifted via the input logic (11) to the Silo (12). Simultaneously, a serial shifting of the bits to the CRC block occurs (1Ø).

The Silo accepts these serially shifted inputs and, when a full word has arrived, parallel transfers it to its internal stack. The Silo memory transfers the word upward through its locations until it cannot go any further. Then the NPR logic initiates a cycle.

From now on, the data words from the drive are shifted into the Silo and CRC logic continuously and synchronously with the DS clock from the data separator (8).

NPR cycles remove the words from the Silo, while the disk attempts to fill the Silo. If the Silo ever fills, then a Data Late error is generated, (i.e., bit 12 of the CSR will be set).

At the end of the 128 word data field, the CRC word from the disk is shifted in to the CRC block (1Ø). The data separator and PLL (8) are then inhibited. A comparison is made between the CRC word read from the disk and the word already generated in the CRC logic. If a Read Data CRC error is detected, then bits 11 and 15 of the CSR are set to flag the error, thus terminating the command. If no CRC error is detected, the DAR (2) increments the sector address preparing to read the next sector (if it is desired).

Nonprocessor Request Operation

The Marker bit of the data preamble is detected and data words are shifted serially into the Silo (12) and CRC circuits (1Ø). As a data word becomes available at the output of the Silo, the controller issues a NPR for the DATO transfer. The CPU acknowledges receipt of the request and, at the appropriate time, transfers the data word. The controller initiates the parallel transfer of data words from the Silo to the memory location specified by the Bus Address register (1). After each word is transferred, the Bus Address register and the Word Counter (3) are incremented. As long as there are new data words available from the Silo and the Word Count register has not overflowed, new NPR cycles will be executed. Since read data is being shifted into the Silo during the data field of each sector, the Silo is being filled and emptied simultaneously. If data words enter the Silo at a faster rate than they exit, the Silo overflows, a Data Late condition is detected, and the command is terminated.

After all data has been emptied from the Silo onto the Unibus, a command termination decision is made. The Read Data command is terminated as a result of a CRC error or a Word Count register overflow. Otherwise, the command will continue onto the next sector(s). During a multiple-sector Read operation, the process of Header Comparison, Read Data transfers, CRC checks, and Disk Address register incrementation is identical for each sector.

Two other commands use the Read circuitry: Read Header and Read Data without Header Check. They are similar to the Read Data command and are discussed below.

Read Header Command (Code of 4)

The function of the Read Header command is to read the first header encountered on the selected drive. The logic will then store the header information (two header words and a Header CRC word) in the Silo and generate an interrupt. The header information from the data separator is shifted into the Silo and CRC logic. If any errors are encountered, flags are raised. The header words can then be extracted from the Silo (by reading the MPR) to determine the present head position.

The software involved will load the Control and Status register with the command and proper bits to select the drive and enable the controller.

Read Data Without Header Check Command (Code of 7)

This command is the same as a Read Data command, with the exception that no header comparison takes place prior to reading data from the disk drive into the Silo. The first sector encountered is the sector that is read into the Silo. No check is made for a Header CRC error, although a Data CRC check is made.

The software involved is similar to that of the Read Data command, with the exception that the DAR is not needed. Use this command to recover customer data if the header word has been destroyed and is not readable. The Read Header command would be used to find the sector prior to the bad sector. When it is found, the registers are reconfigured for a Read Data Without Header Check command. At sector pulse time, the header is ignored, and the data is transferred to main memory.

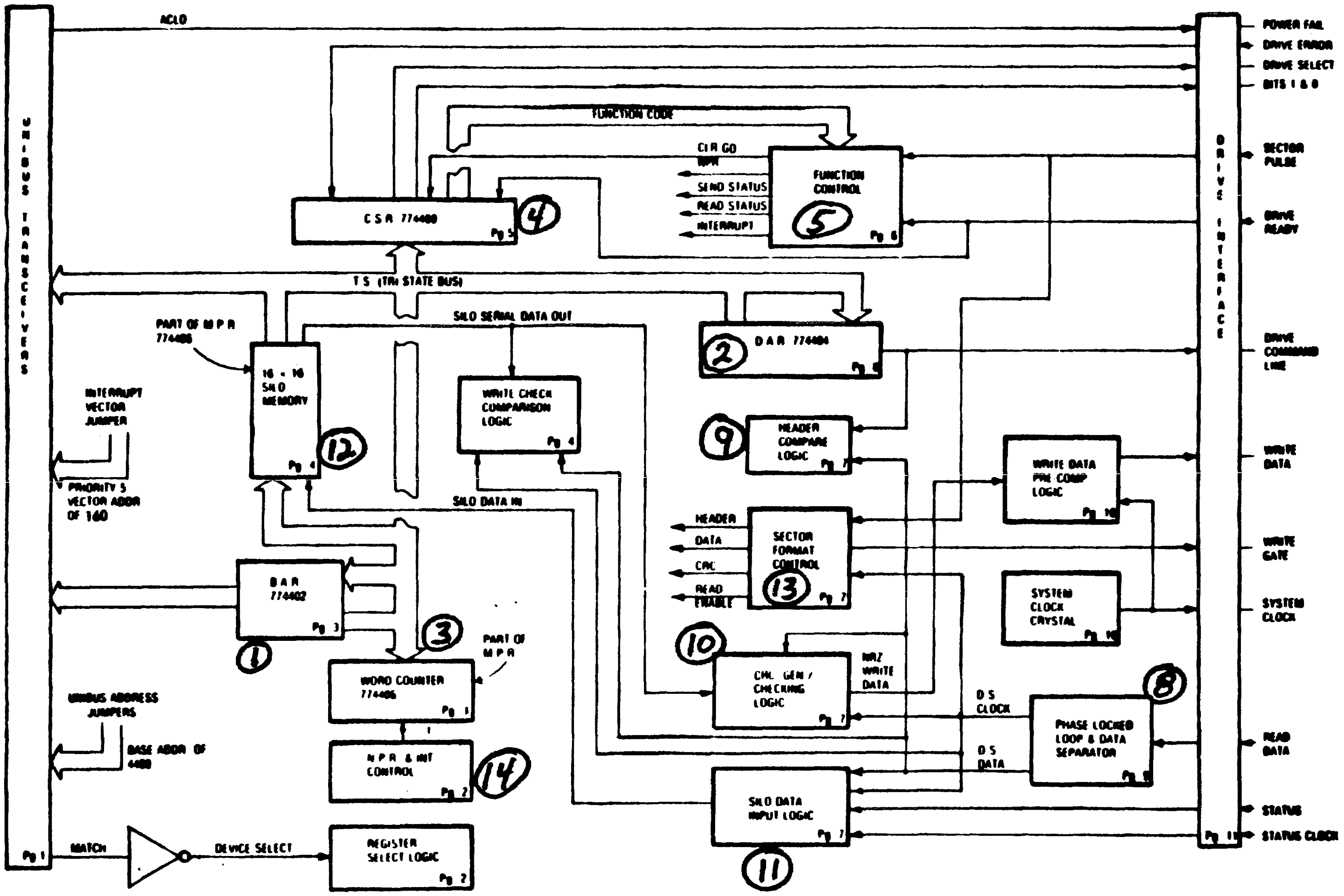


Figure 5 M7762 RL11 Basic Block Diagram

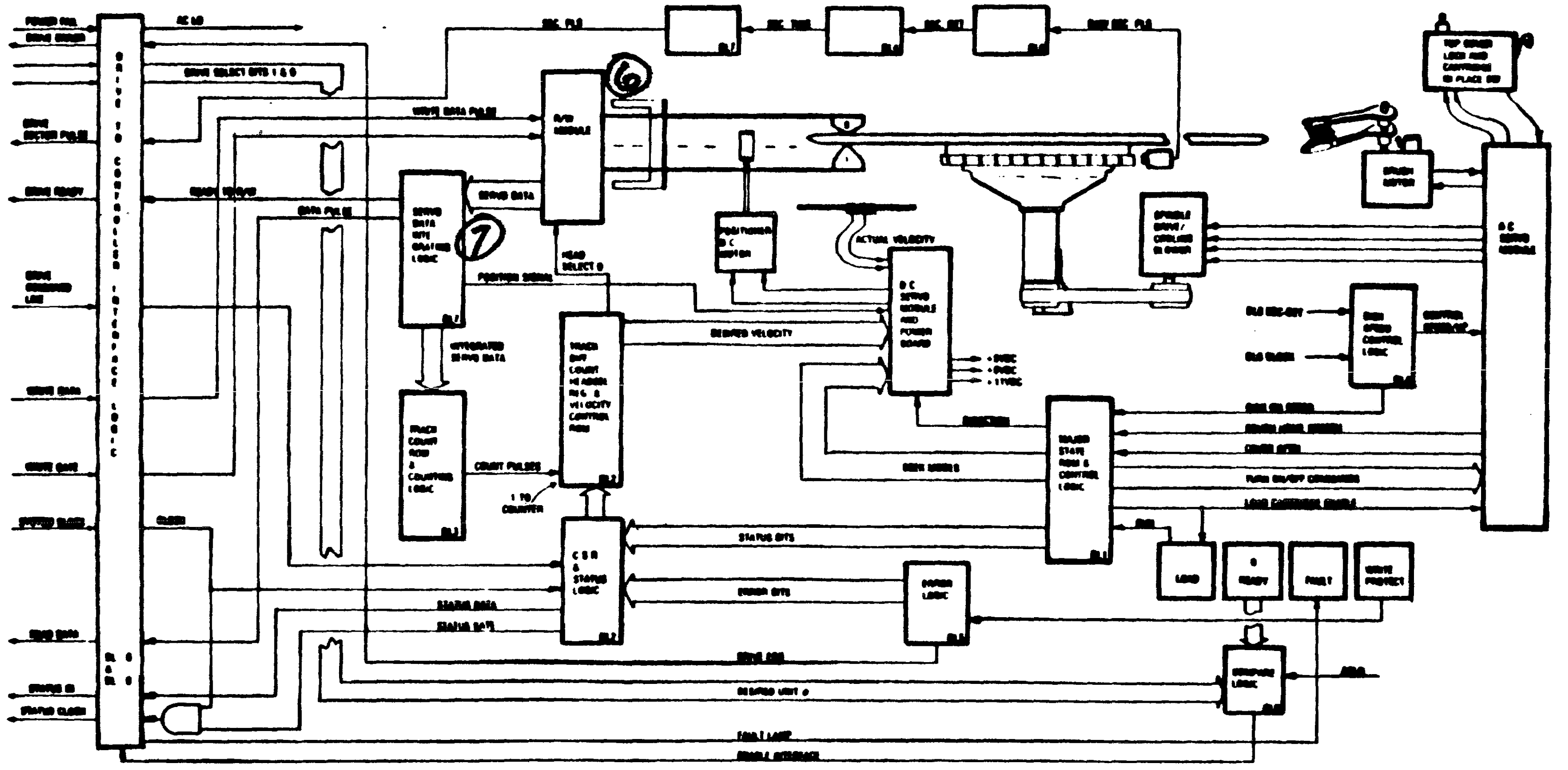


Figure 5 RL Drive Block Diagram

LESSON 4: WRITE DATA COMMAND

GENERAL INFORMATION

This is a command that transfers parallel data from main memory into the controller's Silo memory. From there the Silo shifts the data out in serial form to the disk drive and then onto the disk surface.

Before this transfer can begin, the header must be read and a comparison made of that header and the contents of the DAR. No writing will begin until a match is made between the two. (A header comparison also preceded the Read Data command.)

Assuming that a proper header match is completed before the Header Not Found timer expires, a sector of data will be written. If the word count overflows before the sector is completely written, then the remainder of that sector is filled with zeros.

In addition to writing the 128 data words per sector, the write operation also writes the data preamble and marker bit, and the data CRC and postamble words.

PROGRAMMING

The software associated with this command is similar to that of the Read Data command. The Seek (if desired) must be programmed first to get the R/W heads over the desired track (i.e., cylinder 3, head 1 and sector 7). Following that is the Write Data Command.

`MOV#003000,@#774402`	Load 3000 (first memory location of NPR's) into BAR.
`MOV#000707,@#774404`	Load the disk address (cylinder 3, head 1 and sector 7)
`MOV#177600,@#774406`	Load the two's complement of one sector's worth of data into the Word Count register (MPR).
`MOV#000512,@#774400`	Load the CSR with the Write Data command, clearing Controller Ready (Go bit), setting Interrupt Enable, and selecting drive 1.

Because this controller is an NPR device, the operating system software is free to perform other tasks and/or wait until the sector has been written and the interrupt received. This command is capable of multisector writes, but cannot write beyond the last sector of the selected track without first performing a Seek operation. Re-programming must then occur to continue any of the data transfer commands.

HARDWARE

The controller operation is divided into four phases:

1. Perform the header comparisons until the desired sector is found
2. Become Unibus master, and transfer data words from the Unibus into the Silo
3. Shift the data words from the Silo to the selected drive
4. Generate and append a CRC word at the end of each sector.

As noted earlier, the software must load the following registers with the appropriate information. (Use the block diagram in Figure 6 to aid you in following the written description.)

1. Bus Address
2. Disk Address
3. Word Counter
4. Control and Status

The CSR (4) receives data from the Unibus, selecting a drive. The drive, in turn, enables its interface logic. The sector pulses and Read Data line from the selected drive are then sensed by the controller. Function control (5) senses the Write Data command and initiates NPR cycles to fill the Silo (6). Like the Read Data command, a Header Comparison (12) will take place before the Write begins. The sequence follows.

On the trailing edge of the sector pulse, the PLL (11) is enabled, allowing the header preamble to synchronize the oscillator to the data bit rate. The Read Data separator (11) is also enabled in order to detect the marker bit. When the marker bit is detected, the Header Word is then shifted to the Header Comparison logic.

If no header match is made, the controller enters a "wait" state for the next sector pulse, at which time it will repeat the comparison procedure. If a match is found before the Header Not Found timer times out, the CRC logic attempts to match the generated CRC checkword and the just-read CRC checkword. The PLL/Data Separator is then inhibited to prevent the reading of any more data.

If an error is detected, bits 10 and 11 of the CSR (4) are set, as well as bit 15, to terminate the command. If no error is detected, the sector format control (7) asserts Write Gate on the interface cabling. The drive senses this change in the interface and turns on the write current source drivers on the Read/Write module (10). This causes the 47 data preamble zeros, followed by the marker bit, to be written. To do this, the CRC output logic negates the NRZ Write Data line (8) for the time period established by a bit/word counter in the Sector Format Control block (7). These zeros are routed to the Write Data Precompensation logic (9) to be Miller-encoded and properly clocked out on the Write Data interface line on its way to the R/W Module (10).

As long as Write Gate is asserted, the logic in the drive monitors the Write Data line, delivering data to the disk heads.

All data words take the path from the Silo (6) to the CRC logic (8). The CRC logic deliver the data to the Write Precompensation logic for encoding and clocking to the disk. All clocking originates in the 8.2 MHz crystal. This clock strobes counters in the Sector Format Control (7) and keeps track of the number of words being written. Once 128 words are written, data is shut off from the Silo, and the CRC output is enabled to the NRZ Write Data line. At this time the CRC-generated checkword is written. Following that, a word of zeros is written and the Write Gate is negated. The DAR is then incremented in anticipation of writing another sector.

At this point the Silo (6) is empty because of word count overflow (3). No more data will be written, and the Function Control logic (5) terminates the command. If the Silo contained more data, writing into the next sector(s) would occur until the word count overflows. Note that no writing can continue process past the last sector of any track. This drive/controller combination cannot automatically advance the heads or the carriage. Consequently, when the end of a track is reached, a Seek must be programmed to select the new head or increment the carriage position.

During any multiple-sector Write operations, the process of performing header comparisons followed by writing the preamble and data is identical for each sector.

At this point the discussion of the command function is completed, with the exception of the NPR sequence. At the beginning of the command (when the Silo was empty) the controller issued a request for a DATI transfer. The CPU had acknowledged receipt of the request and given control of the bus to the RL subsystem. When the controller became bus master, the first word coming from memory was then transferred to the controller's Silo (6). After this transferral, the BAR (1) and Word Count registers (3) were incremented. Transfers continued until the Silo became full (in which case no DMA's occurred until an opening became available in the Silo due to the shifting of words to the disk). These transfers began again as space became available in the Silo and continued on a space-available basis until word count overflow occurred. If the Unibus transfer rate into the Silo ever becomes slower then the exit rate from the Silo (to the disk), then a Data Late error will occur, terminating the command.

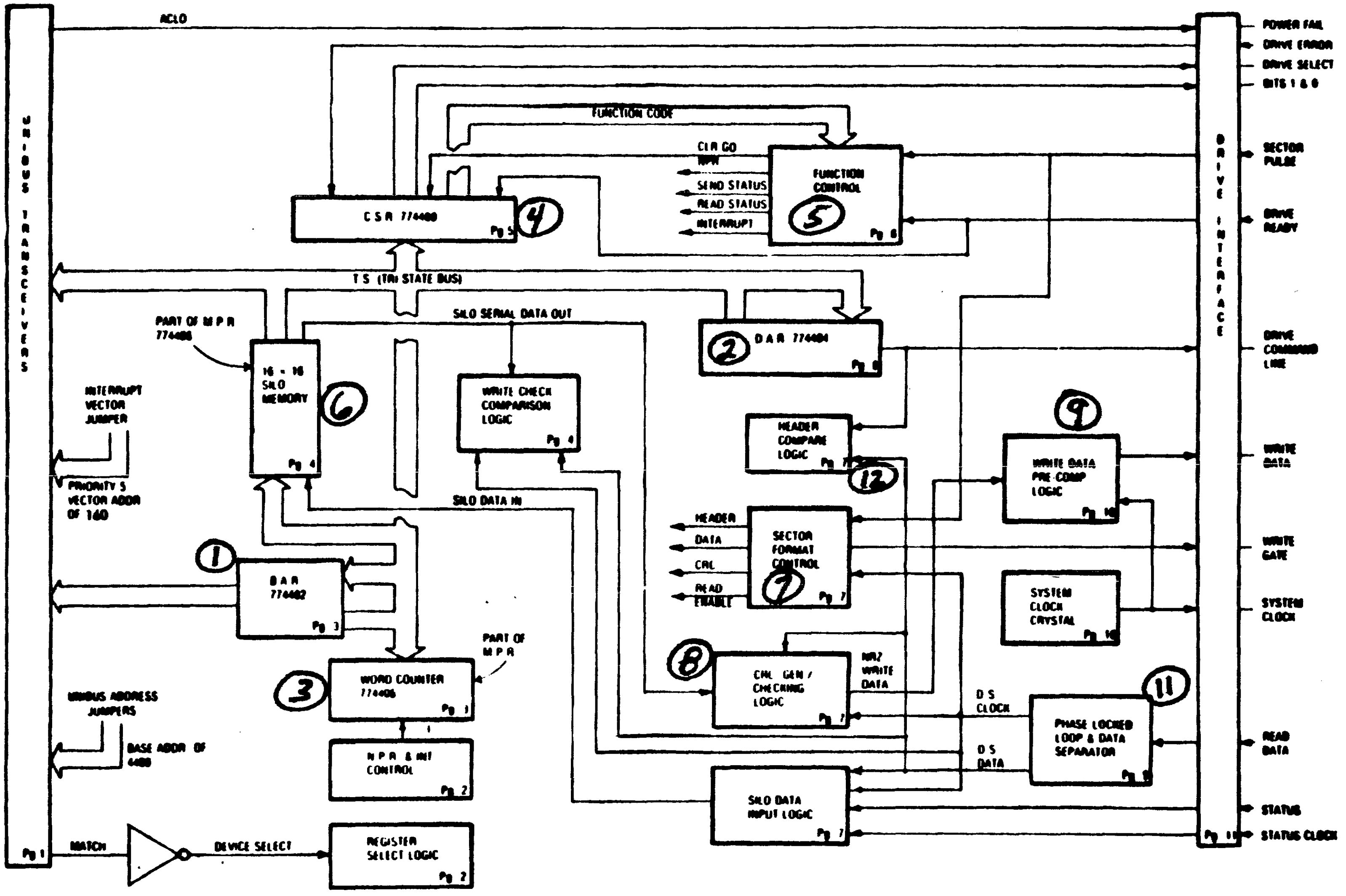


Figure 6 M7762 RL11 Basic Block Diagram

47

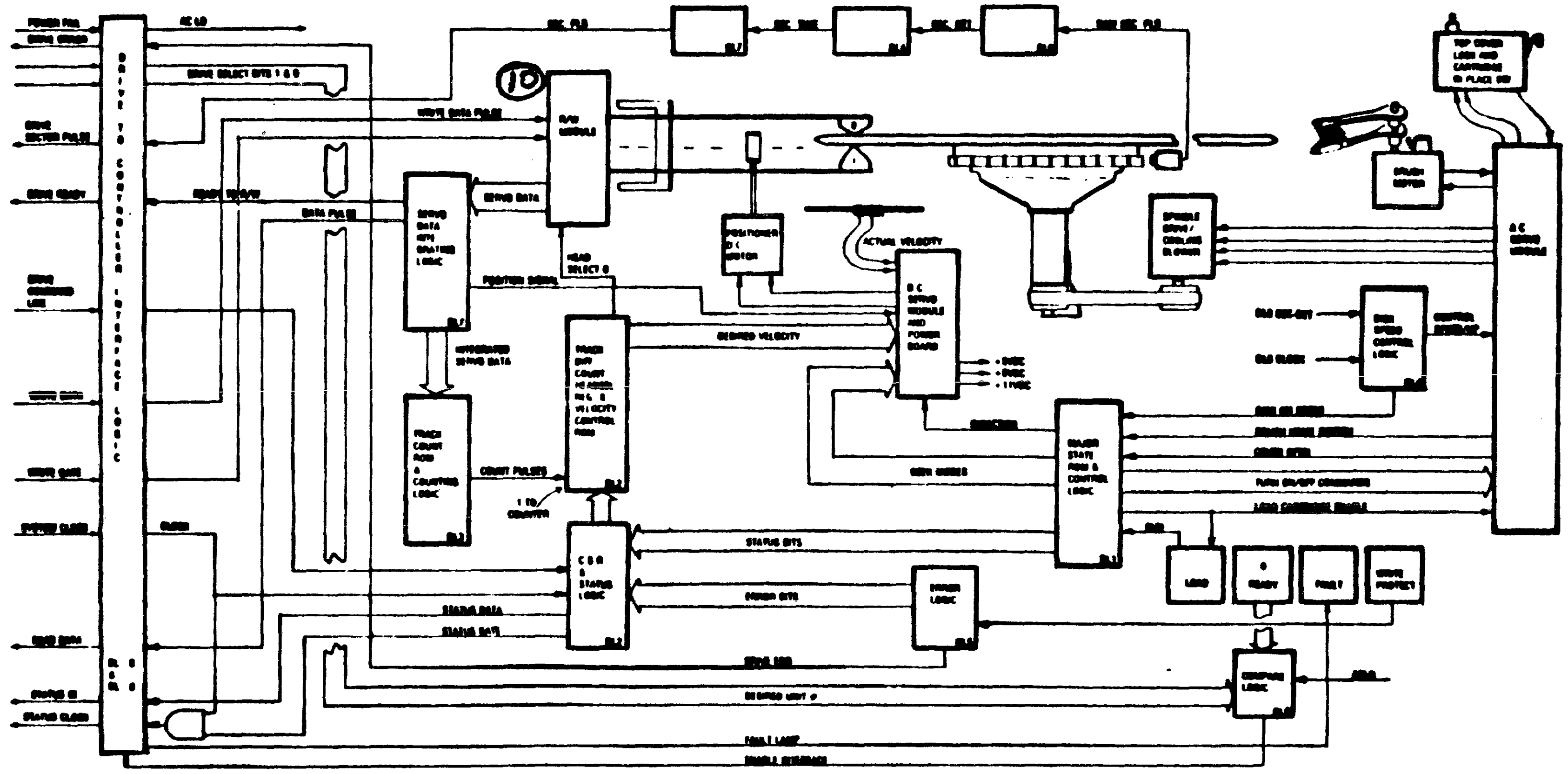


Figure 6 RL Drive Block Diagram

48

LESSON 5: WRITE CHECK COMMAND

GENERAL INFORMATION

This command will verify that the data previously written on the disk is correct. It is performed by commanding the disk drive to read while the controller performs DATI cycles from memory. This results in the words coming in from memory being matched against the words being read from the disk.

PROGRAMMING

The Write Check command is used following the Write command with the original programming parameters for the Write Data. The original word count must be remembered, as well as the starting bus address and disk address. If the transfer crossed track boundaries, then a new Seek command must be issued to re-orient the R/W heads to the original starting track and sector of the Write Data command. Once this is done, the registers are loaded with the appropriate data, and the command is initiated. Following is a sample program illustrating the similarities between Write Check Data and Write Data.

MOV#003000,@#774402	Load 3000 (first memory location) into BAR.
MOV#000707,@#774404	Load the DAR with the desired disk address (cylinder 3, head 1 and sector 7).
MOV#177600,@#774406	Load two's complement of one sector's worth of data into Word Count register (MPR).
MOV#00502,@#774400	Load CSR with Write Check command, clear Controller Ready, se the Interrupt Enable bit, and select drive 1.

Then the drive/controller team takes control, freeing the operating system to perform another task or to wait for the end of operation interrupt.

Write Check Command

HARDWARE

In Figure 7 notice that the shaded areas are a combination of what you have already seen as Read Data (Figure 5) and Write Data (Figure 6). The difference is that a new component is being used--the Write Check Comparison Logic (1Ø). The drive is put into Read Data mode while the Silo (5) is receiving words from core memory (like the Write Data command).

The previous program should have loaded the following registers via the tristate bus:

1. BAR
2. DAR
3. Word Counter
4. CSR.

When the CSR (4) has sensed both the new command and Go, the Drive Select bits enable the desired drive for communication. The sector pulses and Read Data bits from the selected drive are sent back to the controller. The Function Control logic initiates the NPR cycles to fill the Silo (5).

As noted in previous lessons, the proper sector is located by performing reads of the disk headers and comparing these headers with the desired address in the DAR.

At the trailing edge of sector pulse, the PLL (7) is enabled to permit the Read Data preamble zeros to synchronize the oscillator. At the same time, the Data Separator (7) is also enabled to watch for the marker bit. When it is sensed, the header being currently read is compared (6) against the DAR contents (2). When it is found, the CRC is checked. If they match, the PLL is again enabled (7) to allow the Data Preamble to synchronize the oscillator. (Two separate synchronizations are required because the header was written by a different device.) When the data field marker bit is found, data bits are serially shifted to the Data Separator (7) and simultaneously to the CRC logic (9) and Write Check Comparator (1Ø).

Simultaneously, logic has caused the Silo (5) to shift words to this comparator. These data words are being checked bit-by-bit for accuracy. If any do not compare, then bit 11 of the CSR is set to flag the error.

As each word is shifted out of the Silo, the Silo logic moves a new word into the last position, emptying a location within the Silo. This empty location will "bubble" down to the input as words are being moved up to fill the empty position. When the input location becomes empty, another NPR will be generated to fill it.

SUMMARY

This command is used to verify that correct data has been written on the disk. It would normally occur after a Write Data command and before the data written is destroyed in main memory. It reads the data from the disk and compares it to the memory-stored words. The controller's NPR circuits and Silo function like a Write Data command, while the disk performs as if a Read Data command had been issued. Errors set the CRC error flag (bit 11, CSR), so that when interpreting an error print-out, inspect the CSR for the function code to determine the error's identity.

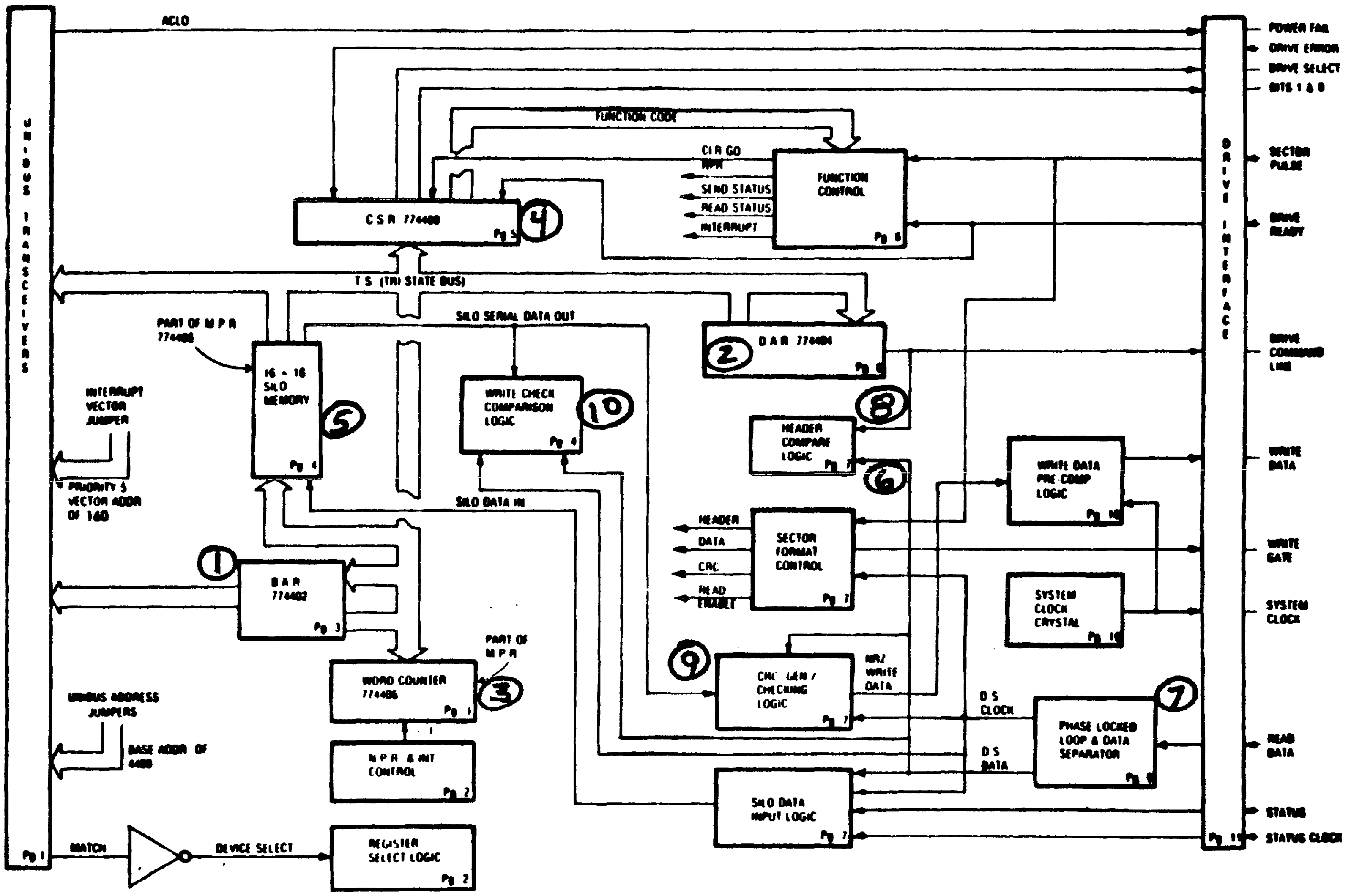


Figure 7 M7762 RL11 Basic Block Diagram

52

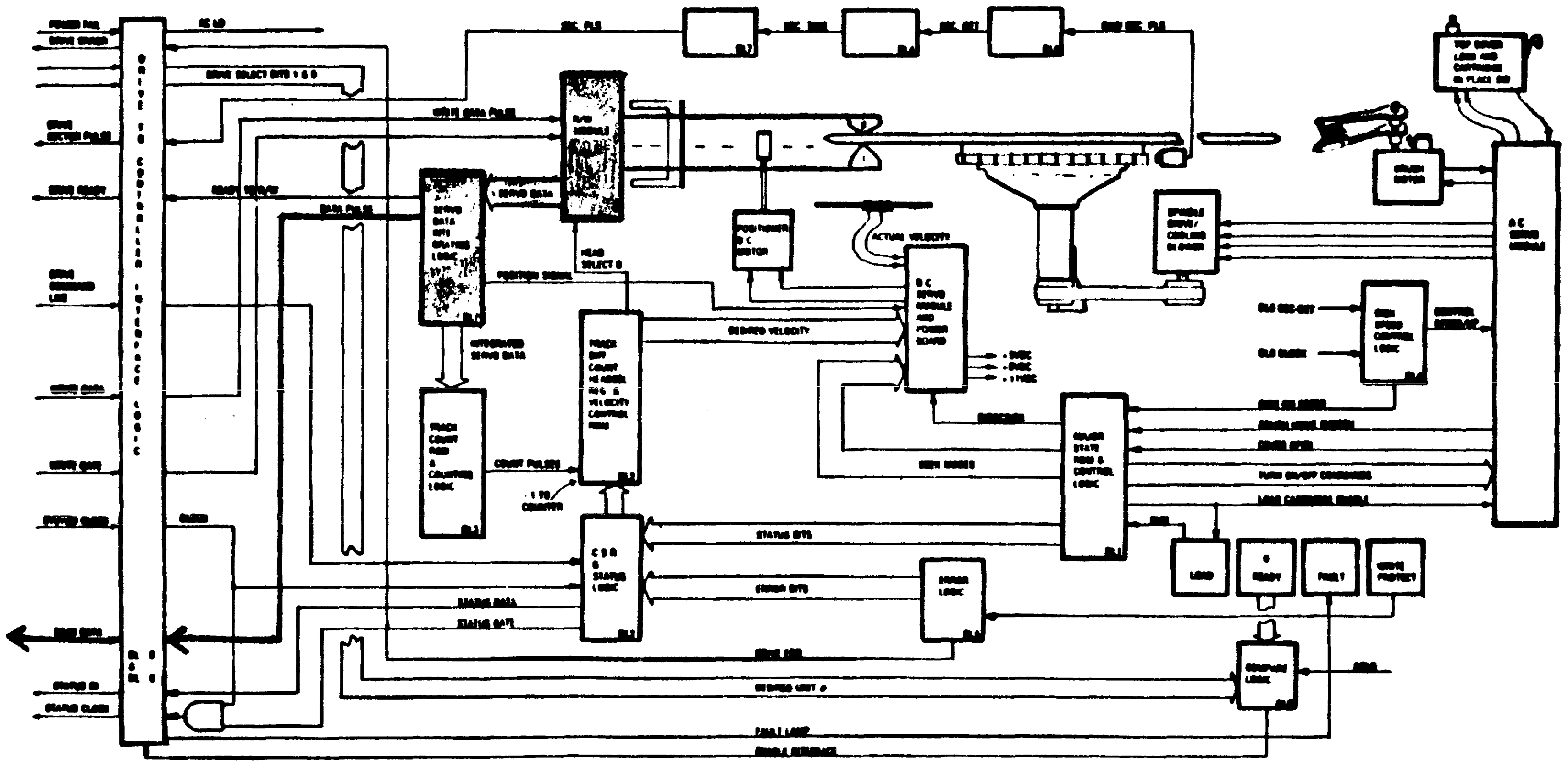


Figure 7 RL Drive Block Diagram

53

RL DIAGNOSTICS

DIAGNOSTIC MODE

These diagnostics build on one another. Thus, they should be run in sequence. Note that they do not find faults for you; they identify failing functions.

CVRLA

The original RL01/RL11/RLV11 diagnostics were labeled DZRLA through F. An additional test of DVRLA was created for the RL01/RLV11 diskless testing. These tests have all been superseded by the designations CZRLG through M. The RLV diskless test description is now CVRLA. These new diagnostics run on both RL01 and RL02 drives.

If an 11/03 CPU is interfacing with an RLV11 controller, then run this diagnostic first. If an RL11 controller is being used, then start with the next diagnostic.

CVRLA uses the RLV11's maintenance mode capabilities to exercise the controller in a diskless environment.

This is an RL11/RLV11 controller test (1 of 2) responsible for testing:

- Interface logic (drive cabled and powered on)
- Register set/clear accuracy
- Commands:

  1. No-Op
  2. Get Status
  3. Read Header
  4. Seek

This test prints out an error or end-of-pass message within 45 seconds. Neither a performance nor a progress report is printed out with it.

Following is a sample of an error report that was received from a unit with a problem in the Drive Logic Module. The interface to the controller had never been enabled due to a bad unit number plug switch assembly.

```
CZRLG DVC FTL ERR 00300 TST 025 SUB 000 PC: 012540

CONTROLLER: 174400 DRIVE: 0
RLCS CONTAINED FOLLOWING ERROR(S):
 COMP OPI
GET STATUS OPERATION-FLAG MODE
BEFORE COMMAND: CS: 000204 BA: 000002 DA: 000013
 MP:  010421
TIME OF ERROR: CS: 102204 BA: 000002 DA: 000013
 MP:  010421 010421 010421
```

The first line of the error report comes from the supervisor. It tells us that the program is DZRLA and that a Device Fatal Error has been encountered. Error number 0300 refers to the software mnemonic for that kind of error. It tells us that the error occurred while performing Test 025 and that it was not in a subroutine. The error (0300) occurred at PC location 012540.

The next lines come from the diagnostic. They tell us that the controller's first address is 174400 and that drive 0 is being tested.

Next we learn that the key error is an OPI that occurred when a Get Status command was being tested. The results of the CSR simply say the same thing. The results of the MPR should reflect the status of the drive, although in this case it is trash. We know this because the OPI occurred indicating that the command was not completed. If the contents of MPR is interpreted as status, we have the drive in spin-up mode with the heads extended and a device-select error asserted. In addition, note that a seek time-out error has occurred, in all, an unlikely combination.

Notice also that the MPR is printed three times because of the error-reporting structure. This type of error calls for the same error-reporting scheme that a Read Header uses. This results in a duplication of the MPR.

Following is another example of an error report from test G.

```
CZRLG DVC FRL ERR 00044 TST 037 SUB 000 PC: 020774
BAD SEEK-TEST OF DIFFENCE WORD
CONTROLLER: 174400 DRIVE: 0
BEFORE COMMAND: CS: 000211 BA: 000002 DA: 000205
 MP:  170005
TIME OF ERROR: CS: 000211 BA: 000002 DA: 000205
 MP:  000042 000000 074012
LAST: 000000 PRES: 000000 EXP'D: 000200
```

Another fatal error is encountered here: error 44 on test 37 (test difference word transmission). Everything needed to interpret the error is printed for you. If you were to look up test 37, error 44, you would find what is stated in the second line of the report.

The tests that Diagnostic G perform are summarized in the Diagnostic Document, Section 6. This section is useful for finding the test number of a routine you may want to loop on in the course of troubleshooting this subsystem.

CZRLH

This program is the second part of the RL11/RLV11 controller diagnostic. It is responsible for the following commands:

- Write Data
- Read Data
- Write Check
- Read Data Without Header Check

Run time is approximately 90 seconds. Error reports are similar to that of the "G" diagnostic. See the following example.

```
CZRLH DVC FTL ERR 00100 TST 001 SUB 000 PC: 021126
CONTROLLER TIMED OUT
CONTROLLER: 174400 DRIVE: 0
GET STATUS OPERATION-FLAG MODE
```

As in the case of the "G" diagnostic, no performance or progress reports are given.

Test summaries are in Section Six of the diagnostic document, if specific tests are desired for scope looping.

CZRLI

This is an RL drive test (1 of 2) responsible for the exercising of the following:

- Basic drive logic
- Get Status Command
- Get Status with Reset
- Seek commands of no cylinder difference
- Read Header command

In addition to the above it includes the Head Alignment Support Routine used at the following times:

- Read signal amplitude checks
- Positioner radial alignment
- Head alignment

This diagnostic can be run two different ways. It can check out all the switches and interlocks on the drive with manual intervention routines. If manual intervention is not desired, then bypass it by not specifying it. See example.

Following is a sample of the questions that this program asks:

```
CZRLI
L-CLK (L) N ?
P-CLK (L) N ?
LSI (L) N ?
LPT (L) N ?
MEM (K) (D) 16 ? 28

TYPE 2 CHAR 4 SEC APART
DS-C>STA
# UNITS (D) ? 1

UNIT 1

RL11 (L) Y ?
BUS ADDRESS (Ø) 174400 ?
VECTOR (Ø) 160 ?
BR LEVEL (Ø) 5 ?
DRIVE (Ø) Ø ? 1

CHANGE SW (L) ? Y

EXECUTE DRIVE SELECT TESTS (L) N ?
EXECUTE HEAD ALIGNMENT SUPPORT (L) N ? Y
EXECUTE MANUAL INTERVENTION TESTS (L) N ?
SPECIFY ERROR LIMIT (D) 2Ø ?
DROP DRIVE IF NO RESPONSE (L) N ?

BUS ADD=1744ØØ DRV=1
HEAD ALIGN. RSET WRT LCK TO SEL HD Ø, SET FOR HD 1
TYPE "CTL C" & "CONT" TO CONTINUE TESTING^C
```

By typing a "Y" for yes to the question "Change SW (L) ?", you get additional questions. Both the head alignment support and the manual intervention tests must be requested. This example asked for the head alignment routine. The printout informs us that if the Write Protect switch were to be depressed to the ON condition, then head number 1 will be selected. If the switch were depressed to the OFF state, then head Ø will be selected. Typing the control/C gets you back out of the routine. The Continue following that executes the remainder of the diagnostic.

If the manual intervention tests are run, the testing will take approximately three minutes. If both the head alignment and manual intervention tests are bypassed, then testing will last only about three seconds.

This program does not issue any performance or progress reports.

Below is a sample error report for CZRLI:

```
CZRLI HRD ERR 01203 TST 012 SUB 001 PC: 030574

HD SWITCH TEST
OPERATION: SEEK
            FROM 000 DIFF 000 SGN 0 HD 1
RESULT: DRV RDY IS 0 SB 1 IN SEEK W/O MOTION
BUS ADD=174400 DRV=2
             RLCS      RLDA      RLBA      RLMP    CYL   HD
OP INIT =  001106    000021    000000    000000
OP DONE =  001304    000003    000000    000135   000    1
```

In this report the contents of the MPR is the status from the drive. It shows that head one was selected, but for some reason, the drive never became ready.

Section Six in the diagnostic document contains the test summaries. In addition to the brief description of each test, there is also a list of suggested failing components in the document which results in a troubleshooting table which you should use.

CZRLJ

This program is part two of the RL drive test. It is responsible for:

- Testing the interface and drive logic
- Seek testing
- Data transfers

Loading procedures are the same for this test as for the three previous ones. Again, no progress or performance reports are available. Run time is approximately eight minutes for one pass and 20 minutes for subsequent passes.

Example error printout:

```
CZRLJ HRD ERROR 10015 TST 002 SUB 001 PC: 017720

ROUTINE TRACE SEQ (IN SEQ CALLED):
    024746
    015700
    020706
DIFF OF 1 SEEK TEST
OPERATION: READ HEADER
RESULT: INTRPT TO LATE
BUS ADD=174400 DRV=0
           RLCS   RLDA   RLBA   RLMP   CYL   HD
OP INIT = 000110 000000 000000 000000
OP DONE = 000311 000000 000000 000003  000    0
```

This report calls out a routine trace sequence because these programs are composed of tests made up of subroutines which may call up other subroutines. If an error occurs during any of this, the diagnostic attempts to trace the routine sequence to find out where the error occurred. Notice also that in the first line of the error report, Sub 001 is called out. In the other programs this referred to a subtest or a subroutine number. Not in this program, however. Instead it refers to the number of times a subtest has been executed within a test. In this error report the subtest was executed once, the error PC indicating that the error occurred during a Read Header operation (written in line 7).

The routine trace sequence ends up as addresses of the JSR's that the test ran through. The first is Go Do a Seek. The second is Go Find the Disk Position, and the third Go Do a Read Header. It is here that the error occurred.

CZRLK

This program is the performance exerciser for the drive. It can test two controllers with up to four drives each. It is purely a system exerciser. Thus, it is difficult to use it to troubleshoot. (Unless diagnostics G, H, I and J all ran successfully.)

In executing this program, the cartridge is written first and then randomly passed through the following series of tests:

- Varied Seek lengths
- Get Status functions
- Read header functions
- Read Data
- Write Data
- Write Checks

```
UNIT 1
RL11 (L) Y ?
BUS ADDRESS (Ø) 174400 ?
VECTOR (Ø) 160 ?
BR LEVEL (Ø) 5 ?
DRIVE (Ø) Ø ?

CHANGE SW (L) ? N

WRITING PACK RLCS: 174400 DRIVE: Ø
TESTING STARTED
^C
DS-C>PRI

              ** RLØ1 PERFORMANCE REPORT **

TIME: ØØ:ØØ:ØØ RLCS: 174400 DRIVE: Ø RUNNING
PACK SERIAL #: ØØØØØØ37Ø3
SEEKS:      2685
BITS READ:         ØØ7287545 (*16)
BITS WRITTEN:      Ø11513676 (*16)

ERRORS
DRIVE:      Ø SEEK:      Ø TRACK:      Ø DATA:      Ø
HARD:       Ø SOFT:      Ø
DCK:        Ø HCRC:      Ø NXM:        Ø HNF:       Ø
DLT:        Ø OPI:       Ø
```

61

Above is a sample printout of a good pass through the diagnostic. Notice on the eighth line it typed out "Writing Pack". After the pack has been written, the diagnostic informs us of when the testing started. Then the drives are randomly exercised. In the example Control/C was typed, bringing the diagnostic back to command mode, when the printing of a performance report was requested (PRI). If the test had run long enough, the report would have been typed automatically.

Within the performance exerciser are 31 discrete and distinct questions regarding Seek lengths, data patterns, error limits, etc. These questions begin in Section 2.3.13 in the diagnostic document under Software Parameters (CHANGE SW(L)?) and are used to vary the length as well as the emphasis on the type of testing.

CZRLL

This diagnostic is the drive compatibility program that will test the interchangeability of cartridges among RL drives. It performs Writes, Reads, Overwrites and adjacent cylinder Writes to prove compatibility. To run it, all previous tests must have been successful and you must have two to four dives on which to test compatibility. It will ask the operator to sequence the same RL cartridge between the drives at various intervals. Section 6 in the document describes in detail more about what occurs in this test.

CZRLM

CZRLM is a utility program that allows the factory and field files to be reported. In addition, the program allows modification of the field Bad Sector File. This program uses the diagnostic supervisor and has seven commands that may be executed:

1. Report the contents of both the factory and field Bad Sector Files
2. Add a sector to the field Bad Sector File
3. Remove a sector from the field Bad Sector File
4. Read the pack (to find bad spots)
5. Write the pack with the worst case pattern, then read back to find the bad spots
6. Generate a new header for a destroyed factory Bad Sector File to allow the pack to be utilized as an F.E. "scratch" pack
7. Print all the available commands.

This is only a utility program. This does not diagnose any hardware problems. Therefore the program assumes a working system with at least 16K of memory.

TOGGLE IN PROGRAMS

This RLØ1/Ø2 Program will write from memory to the disk, the data contained in memory locations 1ØØØ thru 12ØØ, then read the disk placing data into memory locations 2ØØØ thru 22ØØ. The data in locations 1ØØØ-12ØØ can then be compared with the data in locations 2ØØØ-22ØØ. If data compares disk can read and write properly.

ADDRESS	MACHINE CODE	ASSEMBLY CODE	COMMENTS EXTERNAL
17776	ØØØØØ5		Reset I/O Ins.
2ØØØØ	Ø32737	Bit #2Ø1, @#1744ØØ	Test for cntrller rdy.
2ØØØ2	ØØØ2Ø1		
2ØØØ4	1744ØØ		
2ØØØ6	ØØ1774	BEQ -4	Loop until ready
2ØØ1Ø	Ø12737	MOV #1ØØØ,@#174402	Load bar with
2ØØ12	ØØ1ØØØ		memory start address
2ØØ14	1744Ø2		
2ØØ16	Ø12737	MOV#1776ØØ,@#1744Ø6	Load word Cnt
2ØØ2Ø	1776ØØ		2ØØ into MPR
2ØØ22	1744Ø6		(Word counter)
2ØØ24	Ø12737	MOV #Ø,@ #1744Ø4	Set up DAR sector
2ØØ26	ØØØØØØ		Ø, cylinder Ø,
2ØØ3Ø	1744Ø4		Head Ø.
2ØØ32	Ø12737	MOV #12, @#1744ØØ	Write data code
2ØØ34	ØØØØ12		to CSR plus GO
2ØØ36	1744ØØ		CMD.
2ØØ4Ø	Ø32737	Bit #2Ø1,@#1744ØØ	Test for cntrller
2ØØ42	ØØØ2Ø1		Rdy. and drive Rdy.
2ØØ44	1744ØØ		
2ØØ46	ØØ1774	BEQ -4	Loop until ready
2ØØ5Ø	Ø12737	MOV#2ØØØ,@#1744Ø2	Load Bar
2ØØ52	ØØ2ØØØ		
2ØØ54	1744Ø2		
2ØØ56	Ø12737	MOV#1776ØØ,@#174406	Load MPR
2ØØ6Ø	1776ØØ		(word count)
2ØØ62	1744Ø6		
2ØØ64	Ø12737	MOV #Ø, @#1744Ø4	Set up DAR
2ØØ66	ØØØØØØ		
2ØØ7Ø	1744Ø4		

DMA WRITE TO DISK DRIVE

63

ADDRESS	MACHINE CODE	ASSEMBLY CODE	COMMENTS EXTERNAL
2ØØ72	Ø12737	MOV #14, @#1744ØØ	Read CMD code
2ØØ74	ØØØØ14		to CSR plus
2ØØ76	1744ØØ		GO card
2Ø1ØØ	Ø32737	BIT #2Ø1, @#1744ØØ	Test for cntrller
2Ø1Ø2	ØØØ2Ø1		and drive rdy.
2Ø1Ø4	1744ØØ		
2Ø1Ø6	ØØ1774	BEQ -4	Loop until ready
2Ø11Ø	ØØØØØØ	HALT	HALT

Once halted, data should now be recorded on disk (surface Ø, cylinder Ø, sector Ø) and all in memory locations 1ØØØ thru 12ØØ and 2ØØØ thru 22ØØ. If 2ØØØ-22ØØ compares to 1ØØØ thru 12ØØ then read and write must have worked!

NOTE: This program can be modified to read different locations in memory by changing the address loaded into BAR at location 2ØØ12 and 2ØØ52. Similarly the word count can be changed by placing a different count in locations 2ØØ2Ø and 2ØØ6Ø.

11/34 RL BOOTSTRAP PROGRAM

Ensure that the heads are over cylinder Ø and head Ø is selected by releasing the LOAD switch, waiting for the LOAD indicator to light, then depressing the LOAD switch. After the drive is READY, initialize the controller with a system INITIALIZE. Perform a bit status clear. Load the following program into memory.

LOC	CONTENTS	COMMENTS
1ØØØØ	Ø12737	Load CSR
1ØØØ2	ØØØØ14	
1ØØØ4	1744ØØ	
1ØØØ6	ØØØØØ1	Wait

Start the program at 1ØØØØ and allow it to run for a few seconds. Halt the program and restart at ØØØØØ.

digital