A Small Operating System: OS65D The Kernel

Part 2 of 3

Tom R. Berger
School of Math
University of Minnesota
Minneapolis, MN

Subroutine Descriptions

Table 3 is a short memory map of the kernel. In this section we examine some of the subroutines in the map in more detail because they are either useful or interesting. The operating system input/output section will be discussed in some detail in another article, however, three subroutine addresses are vital for understanding the kernel subroutines. These are listed below.

$2339 Input a character without echo to output.
$2340 Input a character with echo to output.
$2343 Output a character.

Input and output for these subroutines is set by the I/O command.

Most programs are greatly enhanced if they can: (1) give instructions or state questions for users; (2) receive replies or input from users; and (3) convert ASCII hex input to binary and vice versa. The kernel contains subroutines to perform these functions. Below are some of the useful routines in the kernel.

Carriage return, line feed ($2D6A)

This routine sends a carriage return followed by a line feed to the output ($2343). It preserves the X-and Y- registers and uses no Z-page locations.

Output a string of embedded text ($2D73)

Assume we have the code listed below.

```
XX00 20732D JSR $2D73
XX03 484921 HI!
XX06 00
XX07 A200 LDX #$00
```

Suppose this segment of code is embedded in our machine language program and the computer is executing instructions just prior to address $XX00. When $XX00 is encountered, the computer jumps to the kernel subroutine at $2D73. This subroutine treats every byte from $XX03 onward as ASCII text to be sent as output ($2343) until the next $00 is encountered. The code above sends the message 'HI!'. Once output is stopped with a $00 (in this case at address $XX06), control is returned to the main program at the next address (in this case $XX07) where execution continues.

Both the Y-register and the Accumulator are destroyed by this routine, but the X-register remains intact. Z-page locations $E3 and $E4 point to the address (low byte-high byte) before the beginning of the embedded text ($XX02 in the example above). Thus, up to 254 characters may be sent out by this routine. More characters may be sent by repeatedly calling the subroutine.

Line buffer input ($2C98)

The buffer is in $2E1E to $2E2F. The subroutine begins with a carriage return ($0D) and line feed ($0A). Further, a carriage return terminates input and is stored in the buffer. Therefore, the user may input up to 17 additional characters in the buffer. Backarrow ($5F) is the standard erase character used by OSI so that from the polled keyboard (shift-locked) Shift-O erases a character. If you disassemble this subroutine you will see a clever use of the routine $2D73. It is used to output backspaces and spaces in order to erase characters on output. Input is obtained via the subroutine $2340 and subroutine $2D6A is called to send out a carriage return followed by a line feed.

This program destroys all registers. It uses only Z-page locations via $2D73. At $2C9B it resets the line buffer output terminator at $2CED.

Line buffer output ($2CE4)

Each time this routine is called, it returns the next character in the line buffer in the Accumulator. The line buffer pointer ($2CE5) is the operand of an LDY #NN instruction at $2CE4. Locations $E1 and $E2 in Z-page point to the beginning of the line buffer and the Y-register is used to index the buffer. After the seventeenth character the buffer will return a carriage return in the Accumulator. The subroutine leaves only the X-register intact.

ASCII hex to binary nibble ($2D3D,$2D40)

If entered at $2D3D, this routine will read the next buffer character ($2CE4), or you may enter the subroutine at $2D40 with an ASCII hex digit in the Accumulator. It will return with a binary number (0-15) in the first four bits of the Accumulator and 0's in the upper four bits. If entered at $2D40, it uses no Z-page locations and leaves the X- and Y-registers intact, provided there is no error. If something other than an ASCII hex digit is read, subroutine $2CA4 is called to output an error Number 7 (Syntax Error). Further, return will occur to the controlling software system via the link set in the jump at $2A4E.

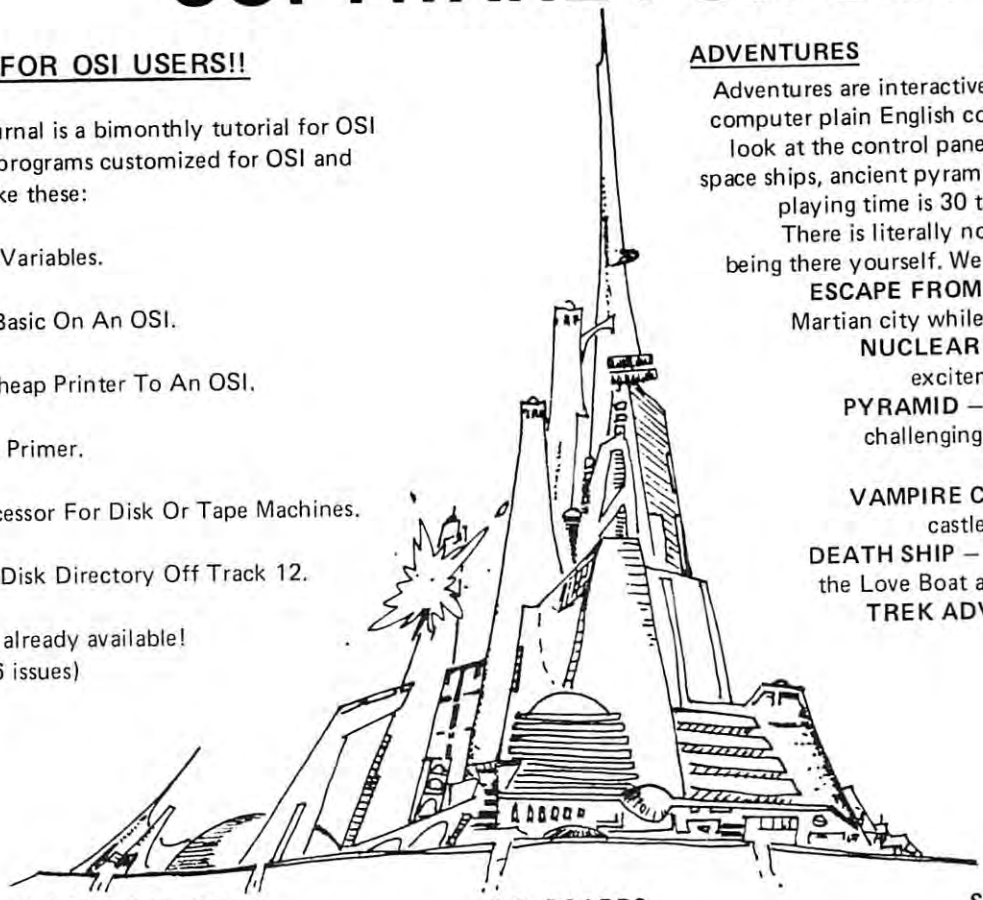
OSI SOFTWARE FOR OSI OSI

A JOURNAL FOR OSI USERS!!

The Aardvark Journal is a bimonthly tutorial for OSI users. It features programs customized for OSI and has run articles like these:

1) Using String Variables.
2) High Speed Basic On An OSI.
3) Hooking a Cheap Printer To An OSI.
4) An OSI Disk Primer.
5) A Word Processor For Disk Or Tape Machines.
6) Moving The Disk Directory Off Track 12.

Four back issues already available!
$9.00 per year (6 issues)



ADVENTURES

Adventures are interactive fantasies where you give the computer plain English commands (i.e. take the sword, look at the control panel.) as you explore alien cities, space ships, ancient pyramids and sunken subs. Average playing time is 30 to 40 hours in several sessions. There is literally nothing else like them – except being there yourself. We have six adventures available.

ESCAPE FROM MARS – Explore an ancient Martian city while you prepare for your escape.

NUCLEAR SUBMARINE – Fast moving excitement at the bottom of the sea.

PYRAMID – Our most advanced and most challenging adventure. Takes place in our own special ancient pyramid.

VAMPIRE CASTLE – A day in old Drac's castle. But it's getting dark outside.

DEATH SHIP – It's a cruise ship – but it ain't the Love Boat and survival is far from certain.

TREK ADVENTURE – Takes place on a familiar starship. Almost as good as being there.

$14.95 each

NEW SUPPORT ROMS FOR BASIC IN ROM MACHINES

C1S – for the C1P only, this ROM adds full screen edit functions (insert, delete, change characters in a basic line.), Software selectable scroll windows, two instant screen clears (scroll window only and full screen.), software choice of OSI or standard keyboard format, Bell support, 600 Baud cassette support, and a few other features. It plugs in in place of the OSI ROM. NOTE: this ROM also supports video conversions for 24, 32, 48, or 64 characters per line. All that and it sells for a measly $39.95.

C1E/C2E for C1/C2/C4/C8 Basic in ROM machines.
This ROM adds full screen editing, software selectable scroll windows, keyboard correction (software selectable), and contains an extended machine code monitor. It has breakpoint utilities, machine code load and save, block memory move and hex dump utilities. A must for the machine code programmer replaces OSI support ROM. Specify system $59.95

DISK UTILITIES

SUPER COPY – Single Disk Copier
This copy program makes multiple copies, copies track zero, and copies all the tracks that your memory can hold at one time – up to 12 tracks at a pass. It's almost as fast as dual disk copying. – $15.95

MAXIPROSS (WORD PROCESSOR) – 65D polled keyboard only - has global and line edit, right and left margin justification, imbedded margin commands, choice of single, double or triple spacing, file access capabilities and all the features of a major word processor – and it's only $39.95.

P.C. BOARDS

MEMORY BOARDS!! – for the C1P. – and they contain parallel ports!
Aardvarks new memory board supports 8K of 2114's and has provision for a PIA to give a parallel ports! It sells as a bare board for $29.95. When assembled, the board plugs into the expansion connector on the 600 board. Available now!

PROM BURNER FOR THE C1P – Burns single supply 2716's. Bare board – $24.95.

MOTHER BOARD – Expand your expansion connector from one to five connectors or use it to adapt our C1P boards to your C4/8P. - $14.95.

ARCADE AND VIDEO GAMES

ALIEN INVADERS with machine code moves – for fast action. This is our best invaders yet. The disk version is so fast that we had to add selectable speeds to make it playable.
Tape - $10.95 – Disk - $12.95

TIME TREK (8K) – real time Startrek action. See your torpedoes move across the screen! Real graphics – no more scrolling displays. $9.95

STARFIGHTER – a real time space war where you face cruisers, battleships and fighters using a variety of weapons. Your screen contains working instrumentation and a real time display of the alien ships. $6.95 in black and white - $7.95 in color and sound.

MINOS – A game with amazing 3D graphics. You see a maze from the top, the screen blanks, and then you are in the maze at ground level, finding your way through on foot. Realistic enough to cause claustrophobia. – $12.95

SCREEN EDITORS

These programs all allow the editing of basic lines. All assume that you are using the standard OSI video display and polled keyboard.

C1P CURSOR CONTROL – A program that uses no RAM normally available to the system. (We hid it in unused space on page 2). It provides real backspace, insert, delete and replace functions and an optional instant screen clear. $11.95

C2/4 CURSOR. This one uses 366 BYTES of RAM to provide a full screen editor. Edit and change lines on any part of the screen. (Basic in ROM systems only.)

FOR DISK SYSTEMS – (65D, polled keyboard and standard video only.)

SUPERDISK. Contains a basic text editor with functions similar to the above programs and also contains a renumberer, variable table maker, search and new BEXEC* programs. The BEXEC* provides a directory, create, delete, and change utilities on one track and is worth having by itself. – $24.95 on 5" disk - $26.95 on 8".

AARDVARK IS NOW AN OSI DEALER!

Now you can buy from people who can support your machine.

–THIS MONTH'S SPECIALS–

Superboard II	$279
C1P Model II	429
C4P	749

. . . and we'll include a free Text Editor Tape with each machine!

Video Modification Plans and P.C. Boards for C1P as low as $4.95

VISA

This is only a partial listing of what we have to offer. We now offer over 100 programs, data sheets, ROMS, and boards for OSI systems. Our $1.00 catalog lists it all and contains free program listings and programming hints to boot.

master charge

OSI

Aardvark Technical Services • 1690 Bolton • Walled Lake, MI 48088
(313) 699-3110 or (313) 624-6316

OSI

A much more useful routine which does the same thing occurs in the ROM machine monitor at $FE93. This latter routine is entered with the hex digit in the Accumulator. It returns with the same data as before except in the case of an error, where $80 is returned in the Accumulator. The ROM subroutine leaves the X- and Y-registers unchanged and uses no Z-page locations.

Full byte binary buffer read ($2D2E)
This routine reads two hex digits from the line buffer and returns with a binary byte in the Accumulator. It calls $2D3D and therefore, has the error procedure of that routine. It uses $E0 as a temporary storage location and affects other registers via subroutine $2CE4.

Full binary address read ($2D23)
By calling $2D2E twice, this subroutine reads four hex digits from the line buffer and stores them as a two byte binary address in Z-page locations $FE and $FF (low byte-high byte).

Nibble to hex digit ($2D9B)
This subroutine converts the first four bits in the Accumulator into an ASCII hex digit and outputs this digit via $2343. It returns with the hex digit in the Accumulator, uses no Z-page addresses, and leaves the X- and Y-registers the same.

One byte binary to two hex digits ($2D92)
By calling $2D9B twice, this routine outputs via $2343 the contents of one full byte binary (in the Accumulator) as an ASCII hex two digit number. It preserves the X- and Y-registers and uses no Z-page locations. The Accumulator is destroyed.

Error output ($2AC4)
If called, this subroutine will reset the 10 flags to the default value, it will disengage the disk head, and it will output "Error # N" where N is a hex digit equal to the first four bits in the Accumulator. Presumably, since an error has occurred, it does not matter which registers have changed.

Stack and Z-page swapper ($2CF7)
This subroutine swaps locations $0000-$01FF (Z-page and the stack) for locations $2F79-$3278 respectively. It returns with the Accumulator and Y-register changed and the X-register equal to 0. When BASIC is resident, OS65D keeps a Z-page and stack separate from BASIC. When the Extended Monitor and Assembler are resident, OS65D and the Extended Monitor keep a Z-page and stack separate from the Assembler.

Shall we swap? ($2D50)
If the contents of $00 are zero the swapper is called, otherwise this subroutine returns with the contents of $00 in the Accumulator and no other changes. BASIC and the Assembler keep nonzero values in $00 while OS65D and the Extended Monitor keep 0 in $00. Thus software can recognize whether or not to swap Z-page and the stack.

Symbol checker ($2D58, $2D5B, $2D5E)
This subroutine reads the buffer to see if the next character is ' = ' ($2D58), ',' ($2D5B), or '/' ($2D5E). If an error occurs the routine behaves as ($2D3D) does, returning to system software control after error Number 7 (Syntax Error). It calls subroutine $2CE4 and uses Z-page location $E0 for temporary storage. This routine uses a standard programming trick of masking 2-byte OPcodes by using a 3-byte BIT instruction.

This concludes a description of the more useful subroutines in the kernel. Most routines are not difficult to decipher. A few have mildly complex flow. The three most involved are: $2A84, The command processor; $2C98, The line buffer input; and $2DA6, The DIRECTORY search. These subroutines are described via flowcharts in Figures 2 to 4. These flowcharts should make it possible to understand disassemblies of the corresponding subroutines.

TABLE 3
MAP - OS65D KERNEL

2A4B
Output an OS65D error # then return to linked software (link is via a jump at 2A4E).

2A51
OS65D Start-up address.

2A7D
Set up the return to software address at 2A4E. Set to 20D7 at 20D1 in BA. Set to 1532 at 152C in ASM. Set to 1756 at 1F31 in in EM. Set to 2A51 at 2A54 in OS65D.

2A84
OS65D Command Processor: called by 2A51. Commands in a table at 2E30 - 2E77.

2AC0
Output ERR# 7: 'SYNTAX ERROR IN COMMAND LINE.'

2AC4
Error message. Enter with error # in accumulator. Resets I/O flags. Disengages disk head.

2ADE
Command AS. Load Tracks 5, 6, and 7, then run the Assembler. Jumps to start at 1300.

2AE6
Command BA. Load Tracks 2, 3, and 4, then run BASIC. Jumps to start at 20E4.

2AEE
Load from the disk the track numbers requested by a command routine starting at 0200 and continuing for 3 tracks.

2B11
Command CA. Call a track and sector from the disk to memory.

2B1A
Engage head, read a sector to memory, then disengage the head.

2B23
Command D9. Disable error #9 in the disk routines. This routine is not called in my version of OS65D. It may be called by changing the address in the COMMAND DIRECTORY.

2B29
Command DI. Give a sector map of a track.

2B2F
Command EM. Load Tracks 5, 6, and 7, then run Extended Monitor. Jumps to start at 1700.

2B37
Command EX. Load an entire disk track to memory for examination.

2B46
Command GO. Start a machine program at specified address.

2B55
Command IN. Initialize a track or the whole disk.

2B68
Text: 'ARE YOU SURE?'

2B83
Command IO. Change the I/O flags.

2BA7
Command LO. Load a named disk file to memory.

2BC6
Command ME. Sets the vectors for memory input and output.

2BDD
Command PU. Puts named file on disk.

2BFD
Command RE. Returns from OS65D to linked software. If software is not in memory, return set to 2AC0 for error #7 out. Settings as follows: ASM to 1303; EM to 1700; BA to 20C4; and M to FEFC (which jumps to FE00).

2C22
Command XQ. Load (starting at 3179) and execute (starting at 317E) a named program

2C28
Command SA. Save memory on a specified sector and track of the disk.

2C43
Command SE. Select a disk drive (A,B,C,D,).

2C60
Get the disk ready for a read or write on a given sector and track.

2C70
Buffer loader. Set the disk start vector to 3179. Engage the disk head.

2C83
Advance head one track. Check for the last track in a file. Report error #D if a read goes beyond the last track of the file.

2C98
Carriage return, line feed, then:

2C9B
Enter and edit a line in the OS65D line buffer at 2E1E - 2E2F.

2CD3
Three empty bytes.

2CD6
Routes input to the Indirect File.

2CE4
Read a line from the OS65D line buffer software, one character at a time.

2CF7
Swapper routine. Switches 0-page and stack for 2F79 - 3178.

2D23
Read 4 ASCII hex digits from the buffer and convert to 2 bytes of binary. Store in FE, FF.

2D2E
Read 2 ASCII hex digits from the buffer and convert to 1 byte binary in accumulator.

2D3D
Read 1 ASCII hex digit from buffer and convert to 1/2 low byte binary in accumulator. Enter at 2D40 with digit in accumulator to skip buffer read.

2D50
Swapper flag check. Initialize for a return to BASIC after an error message. (See BA addresses 20D7 and 20C7).

2D58/2D5B/2D53
Check character to see if it is ' = ', ',', or '/'. Three entry points. Two hidden by BIT instructions.

2D6A
Carriage return and line feed.

2D73
Display embedded text. Display text from the JSR 2D73 instruction until the next null (00).

2D92
1 byte binary in accumulator is converted to 2 ASCII hex digits and displayed in order.

2D9B
Low half byte binary in accumulator is converted to 1 ASCII hex digit and displayed.

2DA6
Directory search. The code from 2DA6-2E1D searches the DISK DIRECTORY to match a file name in the OS65D Buffer with one in the DIRECTORY. When a match is found, the track numbers of the file are saved: last track in 00E5; first track in the accumulator. If a track number (rather than a file name) is given then the track number is read from the line buffer. This routine is used by PU and LO to process the DISK DIRECTORY.

2E1E-2E2F
OS65D Line buffer.

2E30-2E77
OS65D Command directory. 4 bytes per command. First two bytes = First two ASCII letters of Command. Second two bytes = Address of routine - 1.

2E79-2F78
DISK DIRECTORY buffer.

2F79-3078
Buffer for Swapper. Swapped 0-page and stack put here.

3179-317A
Source file start address. (317F if no disk buffers, 3D7F for one buffer, and 497F for two buffers. Address as low byte - high byte.

317B-317C
Source file end address. Address as low byte - high byte.

317B-317C
Source file end address. Address as low byte - high byte.

317D
Number of disk tracks needed to store source file.

317E
Null (00).

OS65D COMMAND DIRECTORY

LOCATION	COMMAND	ROUTINE	ADDRESS
2E30	AS	DD	2A
2E34	BA	E5	2A
2E38	CA	10	2B
2E3C	D9	BF	2A
2E40	DI	28	2B
2E44	EM	2E	2B
2E48	EX	36	2B
2E4C	GO	45	2B
2E50	HO	62	26
2E54	IN	54	2B
2E58	IO	82	2B
2E5C	LO	A6	2B
2E60	ME	C5	2B
2E64	PU	DC	2B
2E68	RE	FC	2B
2E6C	XQ	21	2C
2E70	SA	27	2C
2E74	SE	42	2C

COMMAND PROCESSOR ($2A84)

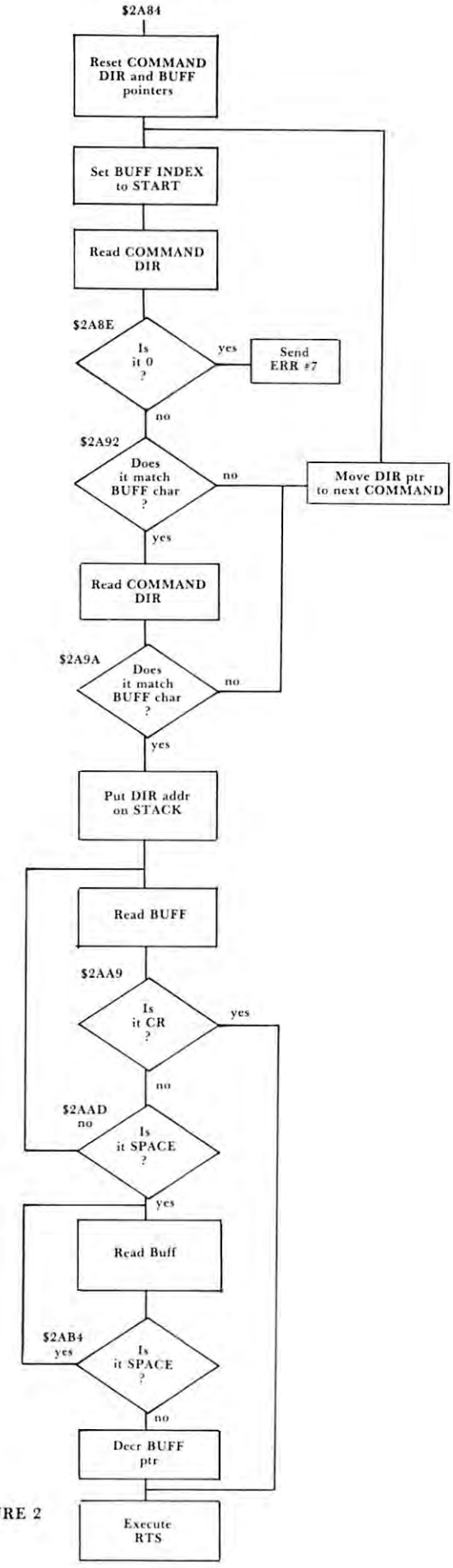


FIGURE 2

OPERATING SYSTEM ORGANIZATION

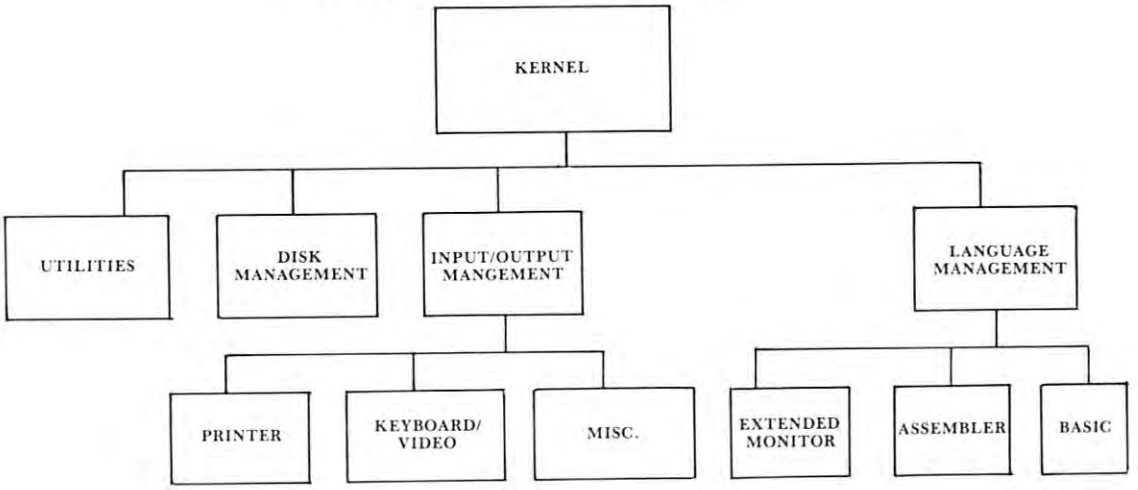


FIGURE 1

LINE BUFFER FILL ($2C98)

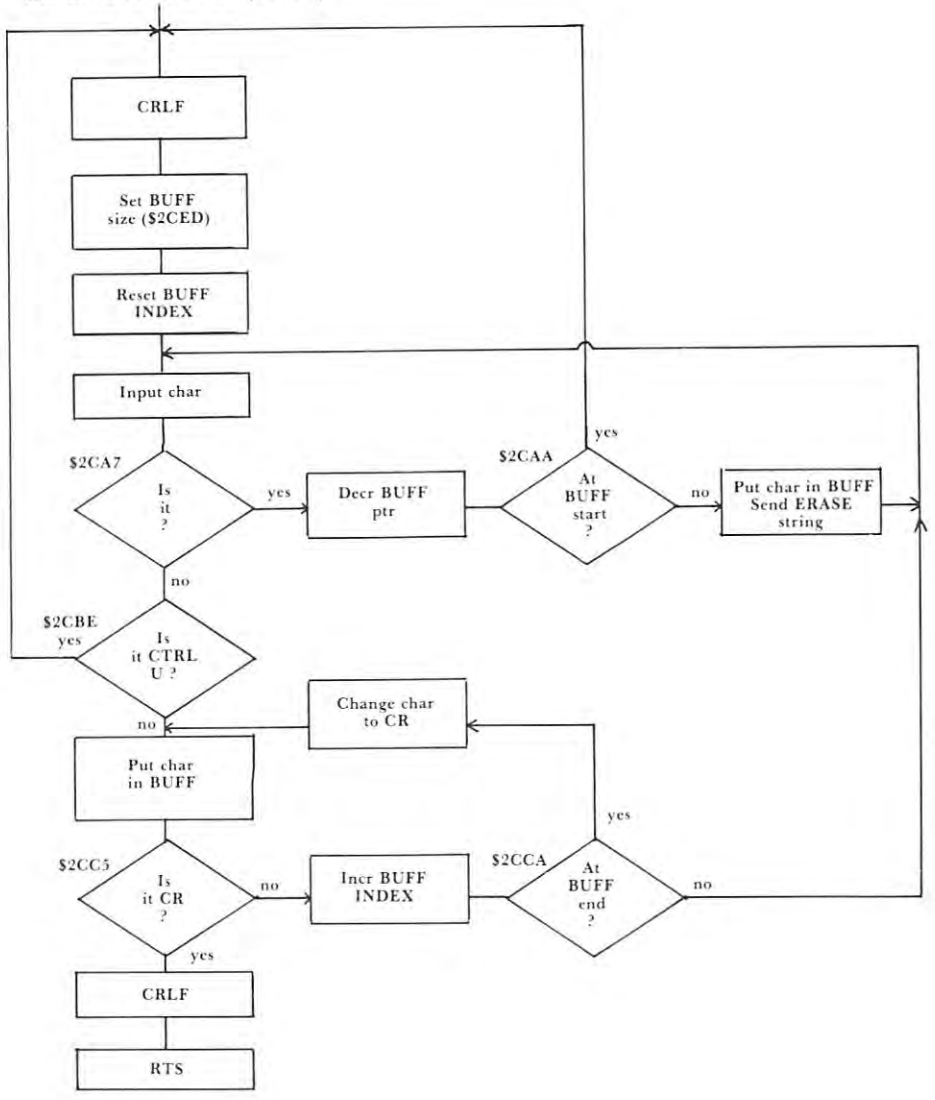


FIGURE 3

DIRECTORY SEARCH ($2DA6, $2DCE)

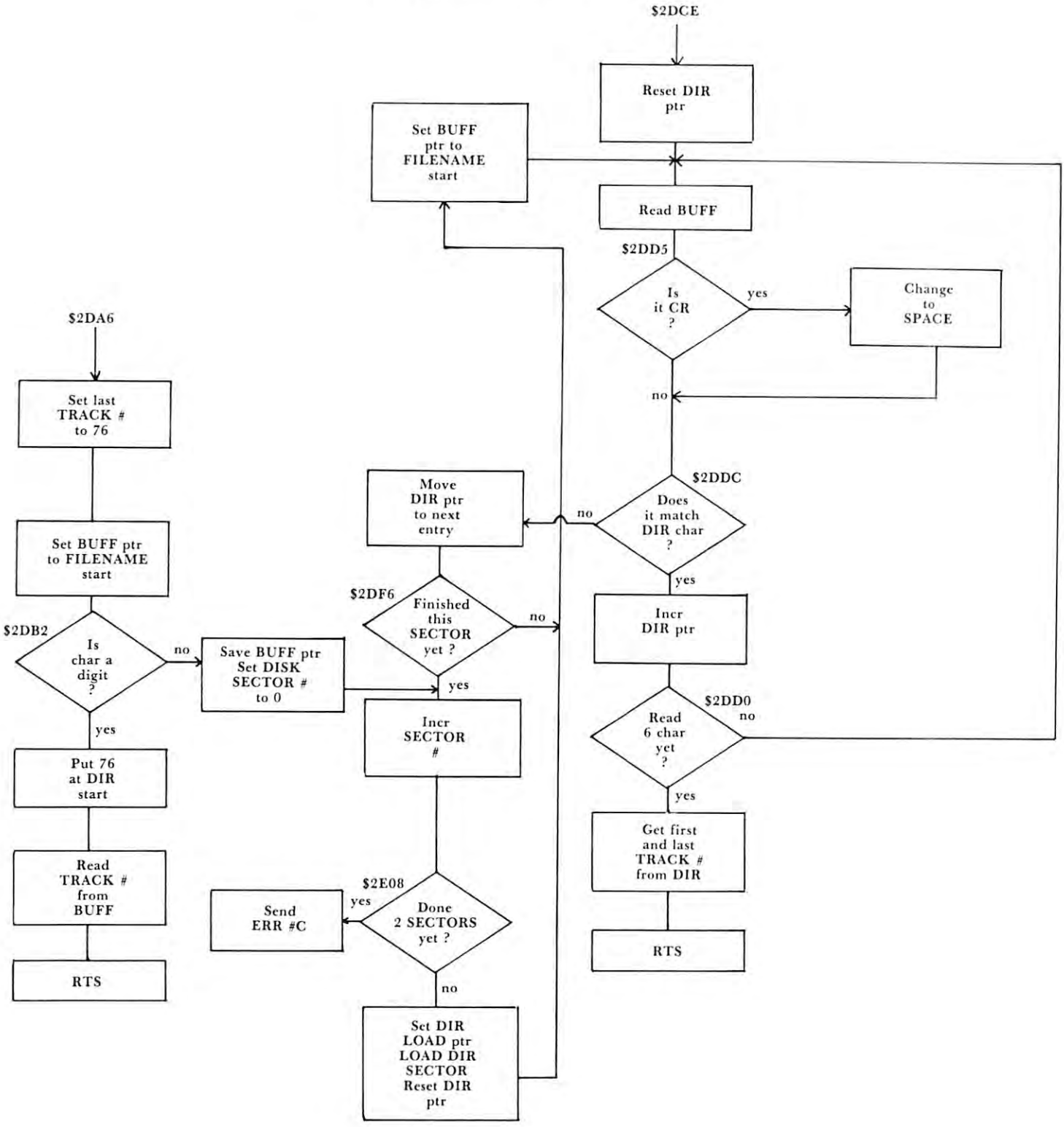


FIGURE 4