GROSVENOR SOFTWARE
2 Beacon Close,
Seaford, E. Sussex, BN25 2JZ.

GROSVENOR SOFTWARE
2 Beacon Close,
Seaford, E. Sussex, BN25 2JZ.

DRAGONDOS
A PROGRAMMER'S GUIDE

DRAGONDOS - A PROGRAMMER'S GUIDE

ABBREVIATIONS:

$	Hexadecimal
#	Number
FSN	File sector # (relative to file)
LSN	Logical sector # (relative to disk)
FIB	File Information Block

Dragon DOS supports 5 1/4 inch drives, single or double sided, with 40 or 80 tracks. 18 sectors per track, 256 bytes per sector. defaults: single sided, 40 tracks.

Directory track: 20
Alternate Directory track: 16
Number of "FIB's": 10 (i.e. max number of open files)

FLOPPY DISK CONTROLLER CHIP: WD 2797

Addresses of 2797 registers:

$FF40	Command reg.
$FF41	Track reg.
$FF42	Sector reg.
$FF43	Data reg.
$FF48	Latch reg.

DIRECTORY TRACK FORMAT:

Sector 1 - Bit Map etc.

bytes 0-179	bit map of free sectors (bit=1 = free)
byte 252	No. of tracks
byte 253	Sectors/~~side~~TRK
byte 254	Complement of no. of tracks
byte 255	Complement of sectors/~~side~~TRK

Sectors 3 onwards - DIRECTORY ENTRIES

Each sector of the directory track contains up to 10 DIR entries of 25 bytes each.

Directory entry layout:

+0	flag
+1-8	file name
+9-11	file name extension
+12-14	#1 extent
+15-17	#2 extent
+18-20	#3 extent
+21-23	#4 extent
+24	bytes in last sector ($00 = 256) or ptr to next directory entry

DIRECTORY TRACK cont.

Format of each extent entry

+0,1 first LSN in this extent
+2 Length (no. of sectors)
unused entries have length=0

Format of extra extent block

+0 flags
+1 extents (7 * 3 bytes)
+22 unused
+24 pointer

Format of flags byte

bit 0 sector unit
0=file name, 1=extension
bit 1 protection
1=protected
bit 3 end of dir
1=end
bit 5 continued?
1=more extents
bit 7 valid?
1=not valid

DOS MEMORY MAP

PAGE ZERO VARIABLES:

ADDRESS hex	CONTENTS
EA	Command
EB	Drive # (1-4)
EC	Track
ED	Sector
EE-EF	Buffer Adress
F0	Status
F1	Current FIB #
F2	No. bytes in buffer
F3	No. bytes to read/write
F4	($00 if length not important)
F5	Read/Write ($00=Read)
F6	IRQ lock out

FORMAT OF DISK COMMAND BLOCK ($0600-$06BD)

600-604	Work
605	Motor Timer
606	Disk option
607	Latch temp.
608	$00=verify on
609	2797 error mask
60a	Default drive #
60B-60C	FWRITE buffer pointer
60D-60E	Buffer start
60F-610	Buffer end
611	Run/Load flag
612	FREAD/FLREAD flag

DOS MEMORY MAP cont.

Variables for AUTO line numbering

613	AUTO flag
60D-60E	Current line
60F-610	Increment

Variables for 'ON ERROR'

614	Trap ON flag
615-616	Line no. to execute on error
617-618	Line in which error occured
619	Error Number
61A-61B	Pointer to bad statement
61C-621	Drive 1 info.
622-627	Drive 2 info.
628-62D	Drive 3 info.
62E-633	Drive 4 info.

Format of drive info:

+0,1	Dir LSN
+2	Frelen
+3,4	Free extent ptr.
+5	number of open files

Index of I/O buffers

634-63A	Buffer 1 info.
63B-641	Buffer 2 info.
642-648	Buffer 3 info.
649-64F	Buffer 4 info.

Format of buffer info.

+0,1	LSN in page buffer
+2	valid flag 0=invalid, not used 1=valid, clean $FF=valid dirty $FE=valid, dirty, grace expired
+3	drive
+4	least recently used counter (1=oldest)
+5,6	buffer address

650-65A	Temp file name
65B	Temp Drive no.
65C-65D	LSN counter
65E-65F	SAVE buffer adr
660	extent found
661-663	work
664-666	FWRITE address
667-668	Page buffer adr
669-66A	current sector
66B-66C	total sectors found
66D-66E	FSN sought
66F-670	current sector
671-696	work

DOS MEMORY MAP cont.

Drive Descriptor Table
each of the following holds 1 byte per drive

697-69A	valid flags
69B-69E	current tracks
69F-6A2	step rates
6A3-6A6	work
6A7-AA	sectors/track

FILE INFO. BLOCKS (FIB's) $6BD-$7F2 10 entries of 31 bytes each

An active FIB exists for each open file.

6BD-6DB	FIB #1
6DC-6FA	FIB #2
6FB-719	FIB #3
71A-738	FIB #4
739-757	FIB #5
758-776	FIB #6
777-795	FIB #7
796-7B4	FIB #8
7B5-7D3	FIB #9
7D4-7F2	FIB #10

Format of each FIB:

(first byte=0 if no entry)

+0-7	file name
+8-10	extension
+11	drive (1-4)
+12-14	next read byte
+15	directory flags
+16-18	length of file
+19-20	FSN extent 1
+21-22	LSN extent 1
+23	sectors in extent 1
+24-25	FSN extent 2
+26-27	LSN extent 2
+28	sectors in extent 2
+29	number of dir entry
+30	dir no. of last entry

DISK I/O BUFFERS $0800-$0BFF
(4 buffers of 256 bytes each)

GRAPHICS PAGES

As DRAGONDOS uses RAM from $0600 to $0BFF, the graphics space is relocated 1.5k higher in RAM. Note that this does not affect the normal text screen space which still occupies $0400 to $05FF.

THE DOS ROM

The DRAGONDOS software occupies a single 8k Eprom (type 2764) in cartridge memory space from $C000 to $DFFF.

ENTRY POINTS TO DOS ROUTINES:

C000	Constant "DK"
C002	Branch to DOS initialize routine
C004	Vector to low level I/O routine
C006	Address of command byte ($00EA)

DOS INDIRECT JUMP TABLE
The following can be used by m/code application programs

C00B	PARSE - scan / verify file name
C00A	SEARCH - Locate or make a FIB
C00C	CREATE - create a new file, backup old
C00E	LENFIL - report length of file
C010	CLOSAL - close all files
C012	CLOSE1 - close 1 file
C014	READ from file
C016	WRITE to file
C018	GETFRE - get free space
C01A	DELETE a file
C01C	PROTECT or unprotect file
C01E	RENAME file
C020	GETDIR get directory entry
C022	READSB read sector to buffer
C024	CLEANUP buffers and copy directory
C026	READ sector to user buffer
C028	WRITE sector from user buffer

ERROR CODES returned in the B register by various routines:
(N.B. all routines that can return an error code end with a TSTB instruction so a BNE can be used immediately on return).

'B'		
$80	NR	Not ready
$82	SK	Seek
$84	WP	Write protect
$86	RT	Record type
$88	RF	Record not found
$8A	CC	CRC check
$8C	LD	Lost data
$8E	BT	Boot
$90	IV	Invalid directory
$92	FD	Directory full
$94	DF	Disk full
$96	FS	File spec.
$98	PT	Protection
$9A	PE	Read past end
$9C	FF	File not found
$9E	FE	File exists
$A0	NE	Non-existant file
$A2	TF	Too many files open
$A4	PR	Parameter
$A6	??	Undefined

DESCRIPTION OF DOS ROUTINES

PARSE file name ($C008) C7B3

This routine validates a name for drive, body and extension. The default extension, and the drive no. from $60A are added if omitted.

On Entry:
- X->name
- B=name length
- Y->default extension

On Exit:
- name stored in $650
- drive in $65B
- B = error code (0=OK)

SEARCH for a FIB ($C00A) C87C

The FIBs are searched for a match with the file name in $650. If found, the FIB # is returned. Else a FIB is created and the disk is searched for the named file.

On entry:
- $0650 file name
- $065B drive (1-4)

On exit:
- A = FIB #
- X-> read pointer
- B = error code
  - $00=ok

CREATE a new file ($C00C) CF14

Any existing file with the same name is renamed as ".BAK". Any existing ".BAK" file is deleted.

On entry:
- A = FIB #

On exit:
- B = error code

LENFIL - report file length ($C00E) CE79

This routine can be used to determine the current length of a file, before using WRITE to extend the file.

On entry:
- A = FIB #

On exit:
- B= error code

A, U - file length.

DOS ROUTINES cont.

CLOSAL close all files ($C010)

This routine closes all files and cleans up the buffers for one drive.

On entry: $EB = Drive# (1-4)

On exit: B = error code

CLOSE1 close one file ($C012)

On entry: A = FIB #

On exit: B = error code

READ from file ($C014)

On entry:
- A = FIB #
- X-> user RAM buffer
- Y = No. of bytes to be read
- U = file sector #
- B = byte # within sector

note: U and B together hold the byte displacement into the file from which to read data.

On exit: X = No. bytes NOT read (if B = EOF error)
- B = error code

WRITE to file ($C016)

On entry:
- A = FIB #
- X-> user RAM buffer
- U = No. of bytes
- Y = File sector #
- B = Byte # within sector

On exit: B = error code

GETFRE report No. of free sectors on drive. ($C01B) P14D

On entry: $EB = Drive #

ON exit: X = No. free sectors
- B = error code

DELETE file ($C01A)

On entry: A = FIB #

On exit: B = error code

$0AB7 7 U, A = 3 Byte No to put

PROTECT on or off ($C01C)

This routine handles a Basic statement PROTECT ON "file" or PROTECT OFF "file". It is not easily used from machine code.

RENAME filename. ($C01E)

This routine handles a Basic statement RENAME "file". Not easily used from machine code.

GETDIR Get directory entry ($C020) $01D4

On entry:
$00EB = Drive #
B = entry #

On exit: X-> entry (in a buffer)
$066F-0670 = LSN
~~$0671~~ = buffer ptr.
$066F-$0680

READSB Read sector to DOS buffer ($C022)

A buffer is allocated from the 4-buffer pool. Preference:

(1) sector already in buffer
(2) an empty buffer
(3) reallocate the least recently used buffer

On entry: Y = LSN
$EB = Drive #

On exit:
X-> buffer info. table entry
U preserved
B = error code

BACKDR Backup directory to the alternate track. ($C024)

No calling arguments.

READ a sector to user's buffer. ($C026)
WRITE a sector from user's buffer ($C028)

On entry:
$00EB = Drive #
X-> user Ram buffer
Y = LSN

On exit: X preserved
B = error code

KNOWN ERRORS IN DRAGON DOS and solutions.

1) The IRQ handling routine which determines when to stop the disk motor spinning, uses Direct addressing mode to check the IRQ lock flag at $00F6 but does not set up the DP reg. Hence programs which alter DP or which disable IRQ can cause the disk to continue spinning. One solution is to use a pre-loader / exec routine to wait for the motor to stop (by monitoring the contents of location $0605).

e.g.:
```
10 LOAD "name.BIN"
20 IF PEEK(&H0605) > 0 THEN 20
30 EXEC
```

2) When re-saving a program to disk, DOS sometimes gives an FE error, instead of taking the normal backup up copy etc. This is due to a coding error in the CREATE routine in the DOS. The bad coding is a TST instruction at $CF3C:

```
    $CF3C   TST   15,X
which should have been
    $CF3C   TST    3,X
```

hence, for those having facilities for programming 2764 Eproms, the solution is to reprogram the DOS ROM, changing the byte at $CF3D from $0F to $03.

PROGRAMMING EXAMPLES

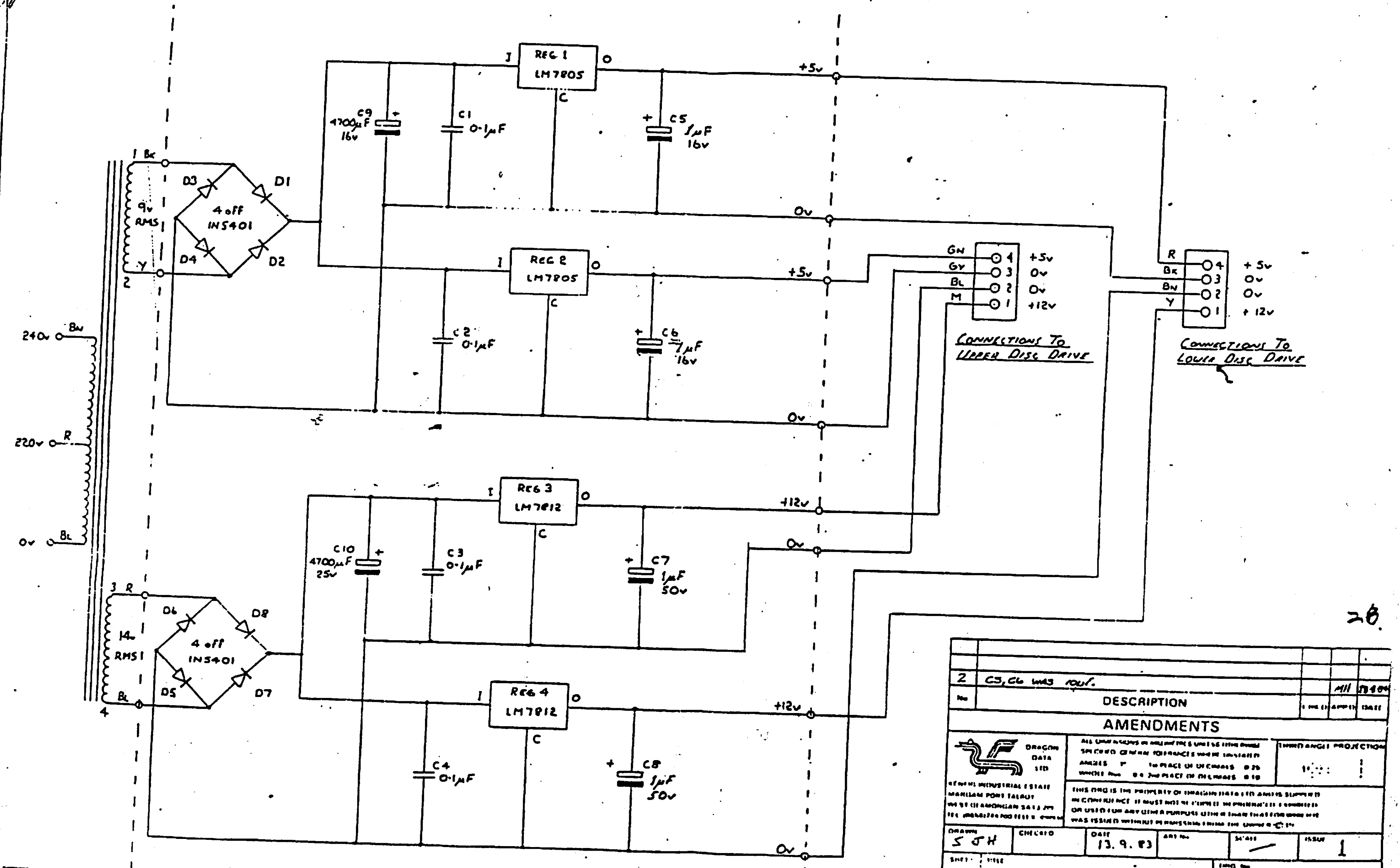
1) To list the disk directory on a printer.

This Basic program redirects the 'character output hook' to refer to the printer output routine, does a DIR command, and restores the character output hook.

```
10 A=PEEK(&H0168):B=PEEK(&H0169)
20 POKE&H0168,&HB0:POKE&H0169,&H0F
30 DIR
40 POKE&H0168,A:POKE&H0169,B
50 END
```

2) This program illustrates the use of the CLOSAL routine from within a Basic program. Without lines 30 and 40, an NE error would occur on line 50.

```
10 SAVE "PROG",&H6000,&H6FFF,0
20 SAVE "PROG",&H6000,&H6FFF,0
30 POKE &HEB,1: REM SET DRIVE #
40 EXEC(256*PEEK(&HC010)+PEEK(&HC011)): REM CLOSAL
50 PROTECT ON "PROG.BAK"
60 PROTECT ON "PROG.BIN"
70 DIR: END
```



No	Description		
2	C5, C6 was 10μF.	MH	28404

AMENDMENTS

DRAGON DATA LTD
KENFIG INDUSTRIAL ESTATE
MARGAM PORT TALBOT
WEST GLAMORGAN SA13 2PE

THIRD ANGLE PROJECTION

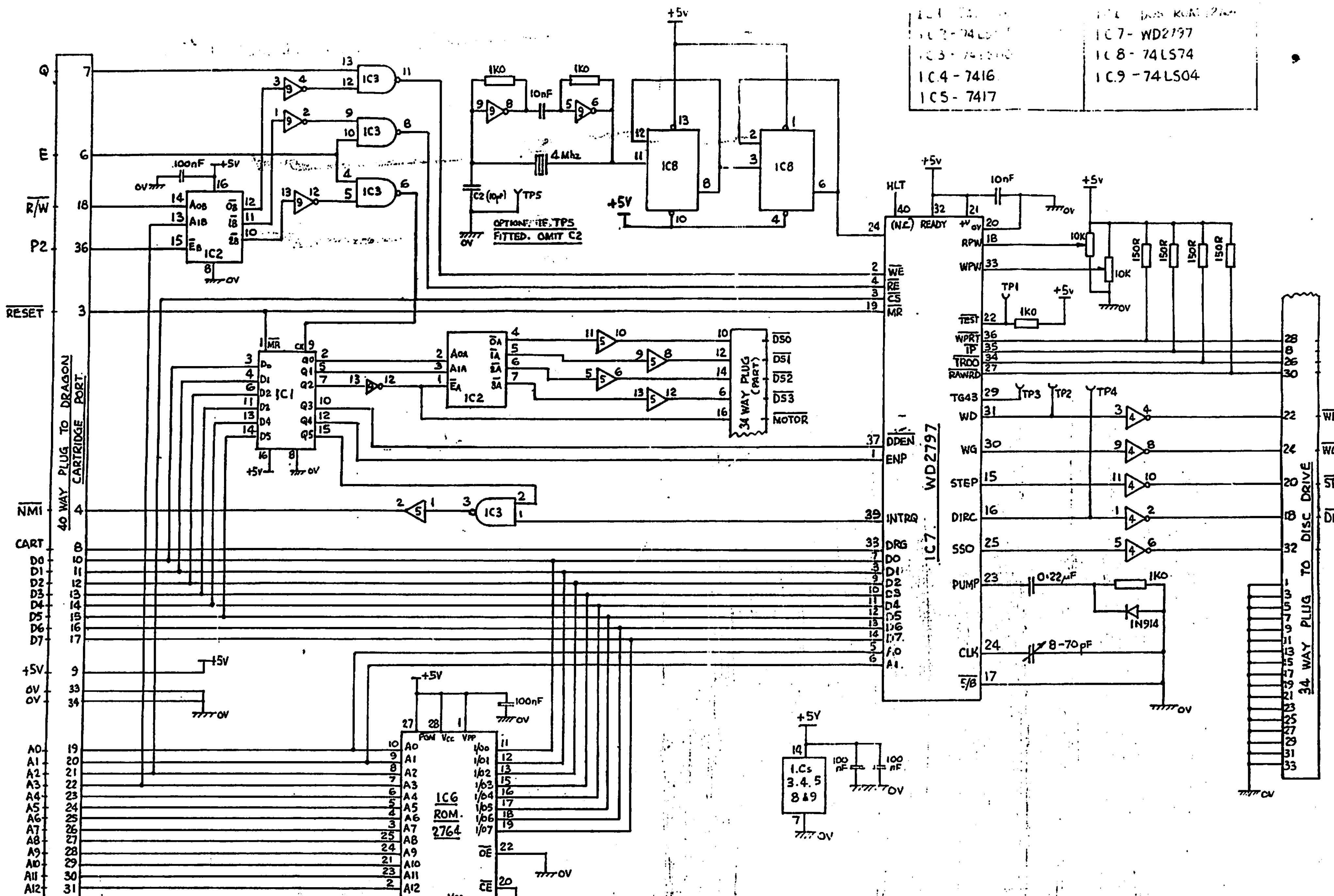
THIS DRG IS THE PROPERTY OF DRAGON DATA LTD AND IS SUPPLIED IN CONFIDENCE. IT MUST NOT BE COPIED, REPRODUCED, EXHIBITED OR USED FOR ANY OTHER PURPOSE OTHER THAN THAT FOR WHICH IT WAS ISSUED WITHOUT PERMISSION FROM THE OWNER ©

Drawn	Checked	Date		Scale	Issue
S J H		13. 9. 83		—	1

Title: SCHEMATIC-DISC CONTROLLER PSU

Drg No: CD 48248

28



CIRCUIT DIAGRAM FOR DRAGON DISC CONTROLLER

REDRAWN FOR N.D.U.G FROM ORIGINAL DATED 20-6-83

BY FRANK PEARSON. 21-12-88