บรรณานุกรม

กฤษดา วิศวธีรานนท์/ ยืน ภู่วรวรรณ, ไมโครโปรเซสเซอร์, สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น), 2526.

พิษณุ สถิตศาสตร์, การออกแบบและสร้างโปรเซสเซอร์สำหรับระบบเครื่องเตรียมข้อมูลลงจานแม่เหล็กแบบฟลอปปี, วิทยานิพนธ์ ภาควิชาวิศวกรรมไฟฟ้า บัณฑิตวิทยาลัย จุฬาลงกรณ์มหาวิทยาลัย, 2528.

Rodnay Zaks, Programming The Z80, Third edition, Sybex, 1984.

Madnick/ Donovan, Operating Systems, McGraw-Hill, 1978.

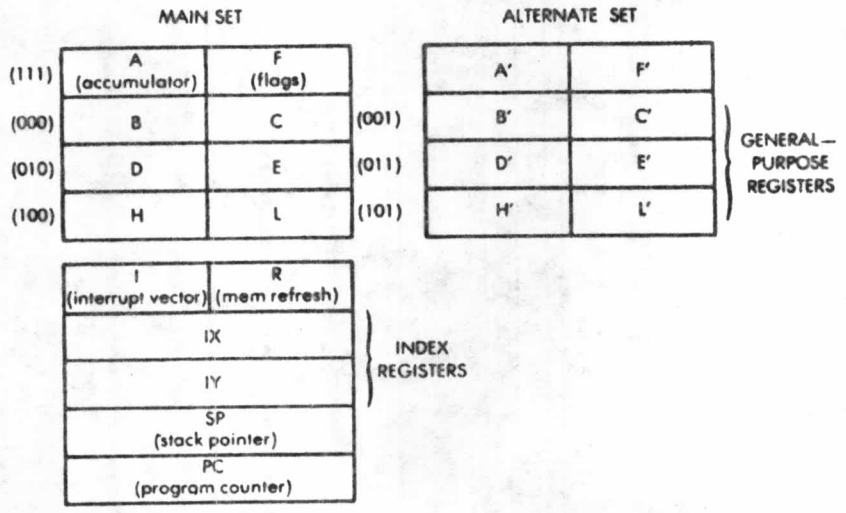
Stephen H Kaisler, The Design of Operating System For Small Computer System, A Wiley Interscience Publication, 1982.

Mark Dahmke, Microcomputer Operating Systems, McGraw-Hill, 1982.

Andy Johnson-Laird, The Programmer's CP/M Handbook, Osborn/ McGraw-Hill, 1983.

ภาคผนวก

ภาคผนวก ก: รีจิสเตอร์และชุดคำสั่งของ Z80



The Z80 instruction set is listed numerically with the corresponding hexadecimal values. The following representations apply:

nn	8-bit parameter
nnnn	16-bit parameter
dd	8-bit signed displacement
*	Instructions common to the 8080

Hex		Mnemonic		Hex		Mnemonic	
00		* NOP		13		* INC	DE
01	nnnn	* LD	BC,nnnn	14		* INC	D
02		* LD	(BC),A	15		* DEC	D
03		* INC	BC	16	nn	* LD	D,nn
04		* INC	B	17		* RLA	
05		* DEC	B	18	dd	JR	dd
06	nn	* LD	B,nn	19		* ADD	HL,DE
07		* RLCA		1A		* LD	A,(DE)
08		EX	AF,AF'	1B		* DEC	DE
09		* ADD	HL,BC	1C		* INC	E
0A		* LD	A,(BC)	1D		* DEC	E
0B		* DEC	BC	1E	nn	* LD	E,nn
0C		* INC	C	1F		* RRA	
0D		* DEC	C	20	dd	JR	NZ,dd
0E	nn	* LD	C,nn	21	nnnn	* LD	HL,nnnn
0F		* RRCA		22	nnnn	* LD	(nnnn),HL
10	dd	DJNZ	dd	23		* INC	HL
11	nnnn	* LD	DE,nnnn	24		* INC	H
12		* LD	(DE),A	25		* DEC	H

Hex		Mnemonic		Hex	Mnemonic	
26	nn	* LD	H,nn	50	* LD	D,B
27		* DAA		51	* LD	D,C
28	dd	JR	Z,dd	52	* LD	D,D
29		* ADD	HL,HL	53	* LD	D,E
2A	nnnn	* LD	HL,(nnnn)	54	* LD	D,H
2B		* DEC	HL	55	* LD	D,L
2C		* INC	L	56	* LD	D,(HL)
2D		* DEC	L	57	* LD	D,A
2E	nn	* LD	L,nn	58	* LD	E,B
2F		* CPL		59	* LD	E,C
30	dd	JR	NC,dd	5A	* LD	E,D
31	nnnn	* LD	SP,nnnn	5B	* LD	E,E
32	nnnn	* LD	(nnnn),A	5C	* LD	E,H
33		* INC	SP	5D	* LD	E,L
34		* INC	(HL)	5E	* LD	E,(HL)
35		* DEC	(HL)	5F	* LD	E,A
36	nn	* LD	(HL),nn	60	* LD	H,B
37		* SCF		61	* LD	H,C
38	dd	JR	C,dd	62	* LD	H,D
39		* ADD	HL,SP	63	* LD	H,E
3A	nnnn	* LD	A,(nnnn)	64	* LD	H,H
3B		* DEC	SP	65	* LD	H,L
3C		* INC	A	66	* LD	H,(HL)
3D		* DEC	A	67	* LD	H,A
3E	nn	* LD	A,nn	68	* LD	L,B
3F		* CCF		69	* LD	L,C
40		* LD	B,B	6A	* LD	L,D
41		* LD	B,C	6B	* LD	L,E
42		* LD	B,D	6C	* LD	L,H
43		* LD	B,E	6D	* LD	L,L
44		* LD	B,H	6E	* LD	L,(HL)
45		* LD	B,L	6F	* LD	L,A
46		* LD	B,(HL)	70	* LD	(HL),B
47		* LD	B,A	71	* LD	(HL),C
48		* LD	C,B	72	* LD	(HL),D
49		* LD	C,C	73	* LD	(HL),E
4A		* LD	C,D	74	* LD	(HL),H
4B		* LD	C,E	75	* LD	(HL),L
4C		* LD	C,H	76	* HALT	
4D		* LD	C,L	77	* LD	(HL),A
4E		* LD	C,(HL)	78	* LD	A,B
4F		* LD	C,A	79	* LD	A,C

Hex	Mnemonic		Hex	Mnemonic	
7A	* LD	A,D	A4	* AND	H
7B	* LD	A,E	A5	* AND	L
7C	* LD	A,H	A6	* AND	(HL)
7D	* LD	A,L	A7	* AND	A
7E	* LD	A,(HL)	A8	* XOR	B
7F	* LD	A,A	A9	* XOR	C
80	* ADD	A,B	AA	* XOR	D
81	* ADD	A,C	AB	* XOR	E
82	* ADD	A,D	AC	* XOR	H
83	* ADD	A,E	AD	* XOR	L
84	* ADD	A,H	AE	* XOR	(HL)
85	* ADD	A,L	AF	* XOR	A
86	* ADD	A,(HL)	B0	* OR	B
87	* ADD	A,A	B1	* OR	C
88	* ADC	A,B	B2	* OR	D
89	* ADC	A,C	B3	* OR	E
8A	* ADC	A,D	B4	* OR	H
8B	* ADC	A,E	B5	* OR	L
8C	* ADC	A,H	B6	* OR	(HL)
8D	* ADC	A,L	B7	* OR	A
8E	* ADC	A,(HL)	B8	* CP	B
8F	* ADC	A,A	B9	* CP	C
90	* SUB	B	BA	* CP	D
91	* SUB	C	BB	* CP	E
92	* SUB	D	BC	* CP	H
93	* SUB	E	BD	* CP	L
94	* SUB	H	BE	* CP	(HL)
95	* SUB	L	BF	* CP	A
96	* SUB	(HL)	C0	* RET	NZ
97	* SUB	A	C1	* POP	BC
98	* SBC	A,B	C2 nnnn	* JP	NZ,nnnn
99	* SBC	A,C	C3 nnnn	* JP	nnnn
9A	* SBC	A,D	C4 nnnn	* CALL	NZ,nnnn
9B	* SBC	A,E	C5	* PUSH	BC
9C	* SBC	A,H	C6 nn	* ADD	A,nn
9D	* SBC	A,L	C7	* RST	0
9E	* SBC	A,(HL)	C8	* RET	Z
9F	* SBC	A,A	C9	* RET	
A0	* AND	B	CA nnnn	* JP	Z,nnnn
A1	* AND	C	CB 00	RLC	B
A2	* AND	D	CB 01	RLC	C
A3	* AND	E	CB 02	RLC	D

Hex	Mnemonic	Hex	Mnemonic
CB 03	RLC E	CB 2D	SRA L
CB 04	RLC H	CB 2E	SRA (HL)
CB 05	RLC L	CB 2F	SRA A
CB 06	RLC (HL)	CB 38	SRL B
CB 07	RLC A	CB 39	SRL C
CB 08	RRC B	CB 3A	SRL D
CB 09	RRC C	CB 3B	SRL E
CB 0A	RRC D	CB 3C	SRL H
CB 0B	RRC E	CB 3D	SRL L
CB 0C	RRC H	CB 3E	SRL (HL)
CB 0D	RRC L	CB 3F	SRL A
CB 0E	RRC (HL)	CB 40	BIT 0,B
CB 0F	RRC A	CB 41	BIT 0,C
CB 10	RL B	CB 42	BIT 0,D
CB 11	RL C	CB 43	BIT 0,E
CB 12	RL D	CB 44	BIT 0,H
CB 13	RL E	CB 45	BIT 0,L
CB 14	RL H	CB 46	BIT 0,(HL)
CB 15	RL L	CB 47	BIT 0,A
CB 16	RL (HL)	CB 48	BIT 1,B
CB 17	RL A	CB 49	BIT 1,C
CB 18	RR B	CB 4A	BIT 1,D
CB 19	RR C	CB 4B	BIT 1,E
CB 1A	RR D	CB 4C	BIT 1,H
CB 1B	RR E	CB 4D	BIT 1,L
CB 1C	RR H	CB 4E	BIT 1,(HL)
CB 1D	RR L	CB 4F	BIT 1,A
CB 1E	RR (HL)	CB 50	BIT 2,B
CB 1F	RR A	CB 51	BIT 2,C
CB 20	SLA B	CB 52	BIT 2,D
CB 21	SLA C	CB 53	BIT 2,E
CB 22	SLA D	CB 54	BIT 2,H
CB 23	SLA E	CB 55	BIT 2,L
CB 24	SLA H	CB 56	BIT 2,(HL)
CB 25	SLA L	CB 57	BIT 2,A
CB 26	SLA (HL)	CB 58	BIT 3,B
CB 27	SLA A	CB 59	BIT 3,C
CB 28	SRA B	CB 5A	BIT 3,D
CB 29	SRA C	CB 5B	BIT 3,E
CB 2A	SRA D	CB 5C	BIT 3,H
CB 2B	SRA E	CB 5D	BIT 3,L
CB 2C	SRA H	CB 5E	BIT 3,(HL)

Hex	Mnemonic	Hex	Mnemonic
CB 5F	BIT 3,A	CB 89	RES 1,C
CB 60	BIT 4,B	CB 8A	RES 1,D
CB 61	BIT 4,C	CB 8B	RES 1,E
CB 62	BIT 4,D	CB 8C	RES 1,H
CB 63	BIT 4,E	CB 8D	RES 1,L
CB 64	BIT 4,H	CB 8E	RES 1,(HL)
CB 65	BIT 4,L	CB 8F	RES 1,A
CB 66	BIT 4,(HL)	CB 90	RES 2,B
CB 67	BIT 4,A	CB 91	RES 2,C
CB 68	BIT 5,B	CB 92	RES 2,D
CB 69	BIT 5,C	CB 93	RES 2,E
CB 6A	BIT 5,D	CB 94	RES 2,H
CB 6B	BIT 5,E	CB 95	RES 2,L
CB 6C	BIT 5,H	CB 96	RES 2,(HL)
CB 6D	BIT 5,L	CB 97	RES 2,A
CB 6E	BIT 5,(HL)	CB 98	RES 3,B
CB 6F	BIT 5,A	CB 99	RES 3,C
CB 70	BIT 6,B	CB 9A	RES 3,D
CB 71	BIT 6,C	CB 9B	RES 3,E
CB 72	BIT 6,D	CB 9C	RES 3,H
CB 73	BIT 6,E	CB 9D	RES 3,L
CB 74	BIT 6,H	CB 9E	RES 3,(HL)
CB 75	BIT 6,L	CB 9F	RES 3,A
CB 76	BIT 6,(HL)	CB A0	RES 4,B
CB 77	BIT 6,A	CB A1	RES 4,C
CB 78	BIT 7,B	CB A2	RES 4,D
CB 79	BIT 7,C	CB A3	RES 4,E
CB 7A	BIT 7,D	CB A4	RES 4,H
CB 7B	BIT 7,E	CB A5	RES 4,L
CB 7C	BIT 7,H	CB A6	RES 4,(HL)
CB 7D	BIT 7,L	CB A7	RES 4,A
CB 7E	BIT 7,(HL)	CB A8	RES 5,B
CB 7F	BIT 7,A	CB A9	RES 5,C
CB 80	RES 0,B	CB AA	RES 5,D
CB 81	RES 0,C	CB AB	RES 5,E
CB 82	RES 0,D	CB AC	RES 5,H
CB 83	RES 0,E	CB AD	RES 5,L
CB 84	RES 0,H	CB AE	RES 5,(HL)
CB 85	RES 0,L	CB AF	RES 5,A
CB 86	RES 0,(HL)	CB B0	RES 6,B
CB 87	RES 0,A	CB B1	RES 6,C
CB 88	RES 1,B	CB B2	RES 6,D

Hex	Mnemonic		Hex	Mnemonic	
CB B3	RES	6,E	CB DD	SET	3,L
CB B4	RES	6,H	CB DE	SET	3,(HL)
CB B5	RES	6,L	CB DF	SET	3,A
CB B6	RES	6,(HL)	CB E0	SET	4,B
CB B7	RES	6,A	CB E1	SET	4,C
CB B8	RES	7,B	CB E2	SET	4,D
CB B9	RES	7,C	CB E3	SET	4,E
CB BA	RES	7,D	CB E4	SET	4,H
CB BB	RES	7,E	CB E5	SET	4,L
CB BC	RES	7,H	CB E6	SET	4,(HL)
CB BD	RES	7,L	CB E7	SET	4,A
CB BE	RES	7,(HL)	CB E8	SET	5,B
CB BF	RES	7,A	CB E9	SET	5,C
CB C0	SET	0,B	CB EA	SET	5,D
CB C1	SET	0,C	CB EB	SET	5,E
CB C2	SET	0,D	CB EC	SET	5,H
CB C3	SET	0,E	CB ED	SET	5,L
CB C4	SET	0,H	CB EE	SET	5,(HL)
CB C5	SET	0,L	CB EF	SET	5,A
CB C6	SET	0,(HL)	CB F0	SET	6,B
CB C7	SET	0,A	CB F1	SET	6,C
CB C8	SET	1,B	CB F2	SET	6,D
CB C9	SET	1,C	CB F3	SET	6,E
CB CA	SET	1,D	CB F4	SET	6,H
CB CB	SET	1,E	CB F5	SET	6,L
CB CC	SET	1,H	CB F6	SET	6,(HL)
CB CD	SET	1,L	CB F7	SET	6,A
CB CE	SET	1,(HL)	CB F8	SET	7,B
CB CF	SET	1,A	CB F9	SET	7,C
CB D0	SET	2,B	CB FA	SET	7,D
CB D1	SET	2,C	CB FB	SET	7,E
CB D2	SET	2,D	CB FC	SET	7,H
CB D3	SET	2,E	CB FD	SET	7,L
CB D4	SET	2,H	CB FE	SET	7,(HL)
CB D5	SET	2,L	CB FF	SET	7,A
CB D6	SET	2,(HL)	CC nnnn	* CALL	Z,nnnn
CB D7	SET	2,A	CD nnnn	* CALL	nnnn
CB D8	SET	3,B	CE nn	* ADC	A,nn
CB D9	SET	3,C	CF	* RST	8
CB DA	SET	3,D	D0	* RET	NC
CB DB	SET	3,E	D1	* POP	DE
CB DC	SET	3,H	D2 nnnn	* JP	NC,nnnn

Hex	Mnemonic		
D3	nn	* OUT	(nn),A
D4	nnnn	* CALL	NC,nnnn
D5		* PUSH	DE
D6	nn	* SUB	nn
D7		* RST	10H
D8		* RET	C
D9		EXX	
DA	nnnn	* JP	C,nnnn
DB	nn	* IN	A,(nn)
DC	nnnn	* CALL	C,nnnn
DD	09	ADD	IX,BC
DD	19	ADD	IX,DE
DD	21nnnn	LD	IX,nnnn
DD	22nnnn	LD	(nnnn),IX
DD	23	INC	IX
DD	29	ADD	IX,IX
DD	2Annnn	LD	IX,(nnnn)
DD	2B	DEC	IX
DD	34dd	INC	(IX+dd)
DD	35dd	DEC	(IX+dd)
DD	36ddnn	LD	(IX+dd),nn
DD	39	ADD	IX,SP
DD	46dd	LD	B,(IX+dd)
DD	4Edd	LD	C,(IX+dd)
DD	56dd	LD	D,(IX+dd)
DD	5Edd	LD	E,(IX+dd)
DD	66dd	LD	H,(IX+dd)
DD	6Edd	LD	L,(IX+dd)
DD	70dd	LD	(IX+dd),B
DD	71dd	LD	(IX+dd),C
DD	72dd	LD	(IX+dd),D
DD	73dd	LD	(IX+dd),E
DD	74dd	LD	(IX+dd),H
DD	75dd	LD	(IX+dd),L
DD	77dd	LD	(IX+dd),A
DD	7Edd	LD	A,(IX+dd)
DD	86dd	ADD	A,(IX+dd)
DD	8Edd	ADC	A,(IX+dd)
DD	96dd	SUB	(IX+dd)
DD	9Edd	SBC	A,(IX+dd)
DD	A6dd	AND	(IX+dd)
DD	AEdd	XOR	(IX+dd)

Hex	Mnemonic		
DD	B6dd	OR	(IX+dd)
DD	BEdd	CP	(IX+dd)
DD	CBdd06	RLC	(IX+dd)
DD	CBdd0E	RRC	(IX+dd)
DD	CBdd16	RL	(IX+dd)
DD	CBdd1E	RR	(IX+dd)
DD	CBdd26	SLA	(IX+dd)
DD	CBdd2E	SRA	(IX+dd)
DD	CBdd3E	SRL	(IX+dd)
DD	CBdd46	BIT	0,(IX+dd)
DD	CBdd4E	BIT	1,(IX+dd)
DD	CBdd56	BIT	2,(IX+dd)
DD	CBdd5E	BIT	3,(IX+dd)
DD	CBdd66	BIT	4,(IX+dd)
DD	CBdd6E	BIT	5,(IX+dd)
DD	CBdd76	BIT	6,(IX+dd)
DD	CBdd7E	BIT	7,(IX+dd)
DD	CBdd86	RES	0,(IX+dd)
DD	CBdd8E	RES	1,(IX+dd)
DD	CBdd96	RES	2,(IX+dd)
DD	CBdd9E	RES	3,(IX+dd)
DD	CBddA6	RES	4,(IX+dd)
DD	CBddAE	RES	5,(IX+dd)
DD	CBddB6	RES	6,(IX+dd)
DD	CBddBE	RES	,(IX+dd)
DD	CBddC6	SET	0,(IX+dd)
DD	CBddCE	SET	1,(IX+dd)
DD	CBddD6	SET	2,(IX+dd)
DD	CBddDE	SET	3,(IX+dd)
DD	CBddE6	SET	4,(IX+dd)
DD	CBddEE	SET	5,(IX+dd)
DD	CBddF6	SET	6,(IX+dd)
DD	CBddFE	SET	7,(IX+dd)
DD	E1	POP	IX
DD	E3	EX	(SP),IX
DD	E5	PUSH	IX
DD	E9	JP	(IX)
DD	F9	LD	SP,IX
DE	nn	* SBC	A,nn
DF		* RST	18H
E0		* RET	PO
E1		* POP	HL

Hex		Mnemonic		Hex		Mnemonic	
E2	nnnn	* JP	PO,nnnn	ED	69	OUT	(C),L
E3		* EX	(SP),HL	ED	6A	ADC	HL,HL
E4	nnnn	* CALL	PO,nnnn	ED	6F	RLD	
E5		* PUSH	HL	ED	72	SBC	HL,SP
E6	nn	* AND	nn	ED	73nnnn	LD	(nnnn),SP
E7		* RST	20H	ED	78	IN	A,(C)
E8		* RET	PE	ED	79	OUT	(C),A
E9		* JP	(HL)	ED	7A	ADC	HL,SP
EA	nnnn	* JP	PE,nnnn	ED	7Bnnnn	LD	SP,(nnnn)
EB		* EX	DE,HL	ED	A0	LDI	
EC	nnnn	* CALL	PE,nnnn	ED	A1	CPI	
ED	40	IN	B,(C)	ED	A2	INI	
ED	41	OUT	(C),B	ED	A3	OUTI	
ED	42	SBC	HL,BC	ED	A8	LDD	
ED	43nnnn	LD	(nnnn),BC	ED	A9	CPD	
ED	44	NEG		ED	AA	IND	
ED	45	RETN		ED	AB	OUTD	
ED	46	IM	0	ED	B0	LDIR	
ED	47	LD	I,A	ED	B1	CPIR	
ED	48	IN	C,(C)	ED	B2	INIR	
ED	49	OUT	(C),C	ED	B3	OTIR	
ED	4A	ADC	HL,BC	ED	B8	LDDR	
ED	4Bnnnn	LD	BC,(nnnn)	ED	B9	CPDR	
ED	4D	RETI		ED	BA	INDR	
ED	4F	LD	R,A	ED	BB	OTDR	
ED	50	IN	D,(C)	EE	nn	* XOR	N
ED	51	OUT	(C),D	EF		* RST	28H
ED	52	SBC	HL,DE	F0		* RET	P
ED	53nnnn	LD	(nnnn),DE	F1		* POP	AF
ED	56	IM	1	F2	nnnn	* JP	P,nnnn
ED	57	LD	A,I	F3		* DI	
ED	58	IN	E,(C)	F4	nnnn	* CALL	P,nnnn
ED	59	OUT	(C),E	F5		* PUSH	AF
ED	5A	ADC	HL,DE	F6	nn	* OR	nn
ED	5Bnnnn	LD	DE,(nnnn)	F7		* RST	30H
ED	5E	IM	2	F8		* RET	M
ED	5F	LD	A,R	F9		* LD	SP,HL
ED	60	IN	H,(C)	FA	nnnn	* JP	M,nnnn
ED	61	OUT	(C),H	FB		* EI	
ED	62	SBC	HL,HL	FC	nnnn	* CALL	M,nnnn
ED	67	RRD		FD	09	ADD	IY,BC
ED	68	IN	L,(C)	FD	19	ADD	IY,DE

Hex		Mnemonic		Hex		Mnemonic	
FD	21nnnn	LD	IY,nnnn	FD	CBdd1E	RR	(IY+dd)
FD	22nnnn	LD	(nnnn),IY	FD	CBdd26	SLA	(IY+dd)
FD	23	INC	IY	FD	CBdd2E	SRA	(IY+dd)
FD	29	ADD	IY,IY	FD	CBdd3E	SRL	(IY+dd)
FD	2Annnn	LD	IY,(nnnn)	FD	CBdd46	BIT	0,(IY+dd)
FD	2B	DEC	IY	FD	CBdd4E	BIT	1,(IY+dd)
FD	34dd	INC	(IY+dd)	FD	CBdd56	BIT	2,(IY+dd)
FD	35dd	DEC	(IY+dd)	FD	CBdd5E	BIT	3,(IY+dd)
FD	36ddnn	LD	(IY+dd),nn	FD	CBdd66	BIT	4,(IY+dd)
FD	39	ADD	IY,SP	FD	CBdd6E	BIT	5,(IY+dd)
FD	46dd	LD	B,(IY+dd)	FD	CBdd76	BIT	6,(IY+dd)
FD	4Edd	LD	C,(IY+dd)	FD	CBdd7E	BIT	7,(IY+dd)
FD	56dd	LD	D,(IY+dd)	FD	CBdd86	RES	0,(IY+dd)
FD	5Edd	LD	E,(IY+dd)	FD	CBdd8E	RES	1,(IY+dd)
FD	66dd	LD	H,(IY+dd)	FD	CBdd96	RES	2,(IY+dd)
FD	6Edd	LD	L,(IY+dd)	FD	CBdd9E	RES	3,(IY+dd)
FD	70dd	LD	(IY+dd),B	FD	CBddA6	RES	4,(IY+dd)
FD	71dd	LD	(IY+dd),C	FD	CBddAE	RES	5,(IY+dd)
FD	72dd	LD	(IY+dd),D	FD	CBddB6	RES	6,(IY+dd)
FD	73dd	LD	(IY+dd),E	FD	CBddBE	RES	7,(IY+dd)
FD	74dd	LD	(IY+dd),H	FD	CBddC6	SET	0,(IY+dd)
FD	75dd	LD	(IY+dd),L	FD	CBddCE	SET	1,(IY+dd)
FD	77dd	LD	(IY+dd),A	FD	CBddD6	SET	2,(IY+dd)
FD	7Edd	LD	A,(IY+dd)	FD	CBddDE	SET	3,(IY+dd)
FD	86dd	ADD	A,(IY+dd)	FD	CBddE6	SET	4,(IY+dd)
FD	8Edd	ADC	A,(IY+dd)	FD	CBddEE	SET	5,(IY+dd)
FD	96dd	SUB	(IY+dd)	FD	CBddF6	SET	6,(IY+dd)
FD	9Edd	SBC	A,(IY+dd)	FD	CBddFE	SET	7,(IY+dd)
FD	A6dd	AND	(IY+dd)	FD	E1	POP	IY
FD	AEdd	XOR	(IY+dd)	FD	E3	EX	(SP),IY
FD	B6dd	OR	(IY+dd)	FD	E5	PUSH	IY
FD	BEdd	CP	(IY+dd)	FD	E9	JP	(IY)
FD	CBdd06	RLC	(IY+dd)	FD	F9	LD	SP,IY
FD	CBdd0E	RRC	(IY+dd)	FE	nn	* CP	nn
FD	CBdd16	RL	(IY+dd)	FF		* RST	38H

ภาคผนวก ข: ตัวอย่างการกำหนดสภาพการทำงานของระบบ

ในการกำหนดสภาพการทำงานของระบบ ทำโดยอาศัยโปรแกรมชื่อ "CONFIG" ซึ่งเป็นโปรแกรมที่เขียนสำหรับใช้กับระบบนี้โดยเฉพาะ ในตัวอย่างนี้ ตัวอักษรตัวหนา หมายถึงข้อความที่ผู้ใช้ป้อนเข้าไป และเครื่องหมาย <RET> หมายถึงกดแป้น RETURN

ขั้นที่ 1 เริ่มต้นทำงาน

ใส่แผ่นจานแม่เหล็กที่มีโปรแกรม "CONFIG" แล้วเรียก "CONFIG" มาใช้งานโดยป้อน

A> CONFIG<RET>

โปรแกรมควบคุมระบบจะอ่านโปรแกรม "CONFIG" เข้ามาในหน่วยความจำ และส่งการควบคุมให้ "CONFIG" เริ่มทำงาน

ขั้นที่ 2 ทำการเลือกดูหรือกำหนดสภาพของระบบ

เมื่อ "CONFIG" เริ่มทำงาน จะแสดงข้อความ

```
  CCCCCCC
 CC     CC
CC
 CC     CC
  CCCCCCC onfig will display all Resident Status and
let you assign the new ones. For being assign, Config will
prompt and wait for your selection one by one.  To assign,
enter your assignment and press <RET>  or only press <RET>
to skip.
     Now press <A> to Assign, else any key to display only <A
```

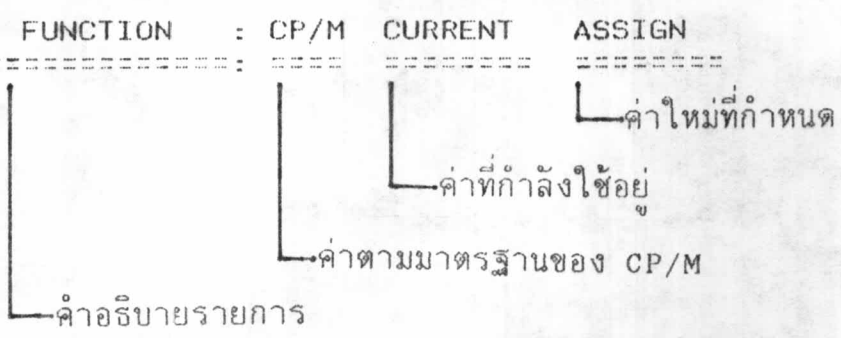
ในขั้นนี้ผู้ใช้มีทางเลือก 2 ทาง คือ ทำการกำหนดสภาพของระบบ โดยกดตัวอักษร "A" หรือทำการดูสภาพของระบบในขณะนั้น โดยกดแป้นใดๆ นอกจาก "A"

ถ้าผู้ใช้เลือกทำการกำหนดสภาพ "CONFIG" จะแสดงค่าที่ใช้อยู่ในขณะนั้น พร้อมกับรอรับค่าใหม่ทีละรายการ แต่ถ้าเลือกดูสภาพ "CONFIG" จะแสดงค่าของทุกรายการที่ใช้อยู่ แล้วกลับสู่ระบบเพื่อรอรับคำสั่งต่อไป

ในตัวอย่างนี้จะแสดงการเลือกกำหนดสภาพ โดยกดตัวอักษร "A"

ขั้นที่ 3 ทำการกำหนดค่าใหม่

จากขั้นนี้ไปให้ดูรูปที่ ข-1 ประกอบ โดยเมื่อเลือกทำการกำหนดสภาพ "CONFIG" จะแสดงข้อความตามบันทัด A ซึ่งมีความหมายดังนี้



ในการกำหนดค่า ให้ป้อนค่าใหม่ที่ต้องการแล้วกด <RET> หรือกด <RET>เลยถ้าต้องการใช้ค่าที่กำลังใช้อยู่เดิม

ขั้นที่ 4 กำหนดพร็อม (PROMPT)

พร็อมที่กำหนดประกอบด้วยชื่อของเครื่องขับจานแม่เหล็ก (DRIVE NAME) และดิสค์ไอดี (DISK ID) ความยาวรวมกันไม่เกิน 8 ตัวอักษร โดยตัวอักษรขวาสุดเป็นดิสค์ไอดีเสมอ กรณีที่กำหนดพร็อมยาวเพียง 1 ตัวอักษร หมายความว่าเลือกกำหนดเฉพาะดิสค์ไอดี ชื่อของเครื่องขับจานแม่เหล็กไม่กำหนด

ตัวอย่าง (ดูบันทัด B ประกอบ)

```
DRIVE NAME : A      DRIVE A    DASD 1<RET>
```

ค่าตามมาตรฐานของ CP/M ใช้ "A"
ค่าที่ใช้อยู่มีชื่อของเครื่องขับจานแม่เหล็กเป็น "DRIVE "
และมีดิสค์ไอดีเป็น "A"
ค่าใหม่ที่กำหนดมีชื่อของเครื่องขับจานแม่เหล็กเป็น "DASD "
และค่าดิสค์ไอดีเป็น "1"

ขั้นที่ 5 กำหนดเครื่องหมายพร็อม (PROMPT SIGN)

การกำหนดเครื่องหมายพร็อมใช้เครื่องหมายอะไรก็ได้ ความยาว 1 ตัวอักษร

ตัวอย่าง (ดูบันทัด C ประกอบ)

```
PROMPT SIGN : >      ]          :<RET>
```

เครื่องหมายพร็อมตามมาตรฐานของ CP/M ใช้ ">"

เครื่องหมายพร็อมที่ใช้อยู่เป็น "]"

เครื่องหมายพร็อมที่กำหนดใหม่เป็น ":"

ขั้นที่ 6 กำหนดชื่อคำสั่งประจำระบบ

ชื่อของคำสั่งประจำกำหนดได้ยาวไม่เกิน 8 ตัวอักษร คำสั่งประจำที่ยอมให้ผู้ใช้กำหนดเองได้ มีทั้งหมด 8 คำสั่ง (คำสั่ง "SHOW" ไม่สามารถเปลี่ยนชื่อได้)

ตัวอย่าง (ดูบันทัด D ประกอบ)

```
LIST FILENAME : DIR   CATALOG   FILES<RET>
```

คำสั่งแสดงชื่อแฟ้มข้อมูล

ชื่อตามมาตรฐานของ CP/M ใช้ "DIR"

ชื่อที่ใช้อยู่เป็น "CATALOG"

ชื่อที่กำหนดใหม่เป็น "FILES"

```
DELETE FILE : ERA   KILL      <RET>
```

คำสั่งลบแฟ้มข้อมูล

ชื่อตามมาตรฐานของ CP/M ใช้ "ERA"

ชื่อที่ใช้อยู่เป็น "KILL"

ชื่อที่กำหนดใหม่ใช้ตามชื่อเดิมคือ "KILL"

ขั้นที่ 7 กำหนดการแสดงภาพโลโก

"CONFIG" จะแสดงว่าขณะนี้ มีการกำหนดโลโกหรือไม่ (Y หรือ N) ซึ่งผู้ใช้สามารถกำหนดใหม่ ให้แสดงภาพโลโกโดยกด "Y" หรือไม่แสดงโดยกด "N"

ตัวอย่าง (ดูบันทัด E ประกอบ)

```
DISPLAY LOGO : Y  <ENTER Y/N> : <N<RET>
```

ค่าเดิมกำหนดให้แสดงภาพโลโก (Y)

ค่าใหม่กำหนดไม่แสดงภาพโลโก (N)

ขั้นที่ 8 กำหนดออโตรัน

ผู้ใช้สามารถเลือกทำออโตรันในโหมดต่างๆ ได้ โดยกด 0, 1, 2 หรือ 3 และสามารถเลือกคำสั่งที่จะใช้ทำออโตรันได้

ตัวอย่าง (ดูบันทัด F ประกอบ)

```
MODE <0=NOT ASSIGN/1=IN CBOOT/2=IN WBOOT/3=BOTH C/WBOOT>
   ASSIGN AUTO MODE : IN CBOOT  <ENTER 0-3> : <3<RET>
CURRENT AUTO COMMAND: CATALOG *.COM
 ASSIGN AUTO COMMAND: FILES *.COM<RET>
```

ค่าเดิมกำหนดทำออโตรันตอนโคล์ดบูท (เลือกหมายเลข 1)

ค่าใหม่กำหนดทำออโตรันทั้งตอนโคล์ดบูทและวอร์มบูท (เลือกหมายเลข 3)

คำสั่งเดิมที่ทำตอนออโตรันเป็น "CATALOG *.COM"

คำสั่งใหม่ที่ทำตอนออโตรันเป็น "FILES *.COM"

หมายเหตุ คำสั่งออโตรันเดิม จะแสดงชื่อแฟ้มทั้งหมด ที่มีชนิดเป็น "COM" ตอนระบบทำโคล์ดบูท ส่วนคำสั่งออโตรันใหม่จะแสดงชื่อแฟ้มทั้งหมดที่มีชนิดเป็น "COM" ตอนระบบทำโคล์ดบูทหรือวอร์มบูท และเหตุที่ต้องเปลี่ยนคำสั่งออโตรันใหม่ เพราะเปลี่ยนชื่อคำสั่งแสดงชื่อแฟ้มข้อมูลจากเดิม "CATALOG" เป็น "FILES"

ขั้นที่ 9 ชนิดของการกำหนด

จากค่าที่กำหนดมาทั้งหมด ผู้ใช้สามารถเลือกได้ว่าจะกำหนดแบบชั่วคราวหรือแบบถาวร การกำหนดแบบถาวรทำโดยกดตัว "P" ค่าที่กำหนดจะเป็นค่าที่ใช้ตลอดไป จนกว่าจะทำการกำหนดใหม่ ส่วนการกำหนดแบบชั่วคราวให้กดแป้นใดๆ นอกจากตัว "P" ค่าที่กำหนดจะมีผลเฉพาะในการทำงานขณะนั้น ถ้าปิดและเปิดเครื่องใหม่ ระบบจะกลับไปใช้ค่าเดิม

ในการกำหนดแต่ละแบบ "CONFIG"จะแสดงข้อความ

++ PERMANENT ASSIGN ++

หรือ ++ TEMPORARY ASSIGN ++

เพื่อยืนยันสิ่งที่ผู้ใช้เลือกกำหนด (ดูบันทัด G ประกอบ)

```
          ++ RESIDENT STATUS ++

   FUNCTION    : CP/M  CURRENT    ASSIGN ---------- (A)
===============: ====  ========   ========
    DRIVE NAME : A     DRIVE A    DASD 1<RET> ------ (B)
  PROMPT SIGN  : >     ]          :<RET> ---------- (C)
LIST FILENAME  : DIR   CATALOG    FILES<RET>
  DELETE FILE  : ERA   KILL       <RET>
  RENAME FILE  : REN   CHGNAM     RENAME<RET>
    TYPE FILE  : TYPE  XEDIT      LIST<RET>
  SAVE MEMORY  : SAVE  XPPUN      STORE<RET>      (D)
SET PARTITION  : USER  PART       <RET>
TRANSFER FILE  : N/A   TRANS      MOVE<RET>

      DISPLAY LOGO : Y  <ENTER Y/N> : <N<RET> ------ (E)

MODE <0=NOT ASSIGN/1=IN CBOOT/2=IN WBOOT/3=BOTH C/WBOOT>
   ASSIGN AUTO MODE : IN CBOOT  <ENTER 0-3> : <3<RET>
CURRENT AUTO COMMAND: CATALOG *.COM
 ASSIGN AUTO COMMAND: FILES *.COM<RET> ------------ (F)

PRESS <P> FOR Permanent ASSIGN ----------
  ELSE ANY KEY FOR Temporary ASSIGN : <T<RET>
                                                   (G)
        ++ TEMPORARY ASSIGN ++ ----------
```

รูปที่ ข-1 ตัวอย่างการใช้โปรแกรม "CONFIG"

ภาคผนวก ค: การสั่งงานโมดุลดอส

To use function of DOS, register C keeps function number and sent parameters are set in register E or DE (except function 38). Then call address 5H.

NO	FUNCTION OPERATION	SENT PARAMETER(S)	RETURNED PARAMETER(S)
00	SYSTEM RESET	NONE	NONE
01	CONSOLE INPUT	NONE	A=ASCII CHARACTER
02	CONSOLE OUTPUT	E=ASCII CHARACTER	NONE
03	READER INPUT	NONE	A=ASCII CHARACTER
04	PUNCH OUTPUT	E=ASCII CHARACTER	NONE
05	LIST OUTPUT	E=ASCII CHARACTER	NONE
06	DIRECT CONSOLE IN	E=FF	A=ASCII CHARACTER
	DIRECT CONSOLE OUT	E=ASCII CHARACTER	NONE
07	GET IOBYTE	NONE	A=IOBYTE
08	SET IOBYTE	E=IOBYTE	NONE
09	PRINT STRING	DE=STRING ADDRESS	NONE
0A	READ STRING	DE=BUFFER ADDRESS	DATA IN BUFFER
0B	GET CONSOLE STATUS	NONE	A=STATUS
0C	GET VERSION NUMBER	NONE	HL=VERSION
0D	RESET SYSTEM DISK	NONE	NONE
0E	SELECT DISK	E=DISK NUMBER	NONE
0F	OPEN FILE	DE=FCB ADDRESS	A=ERROR CODE
10	CLOSE FILE	DE=FCB ADDRESS	A=ERROR CODE
11	SEARCH FIRST	DE=FCB ADDRESS	A=ERROR CODE
12	SEARCH NEXT	DE=FCB ADDRESS	A=ERROR CODE
13	DELETE FILE	DE=FCB ADDRESS	A=ERROR CODE
14	READ SEQUENTIAL	DE=FCB ADDRESS	A=ERROR CODE
15	WRITE SEQUENTIAL	DE=FCB ADDRESS	A=ERROR CODE
16	MAKE FILE	DE=FCB ADDRESS	A=DIR CODE
17	RENAME FILE	DE=FCB ADDRESS	A=DIR CODE
18	GET LOGIN VECTOR	NONE	HL=LOGIN VECTOR
19	GET CURRENT DISK	NONE	A=CURRENT DISK

FUNCTION NO	OPERATION	SENT PARAMETER(S)	RETURNED PARAMETER(S)
1A	SET DMA ADDRESS	DE=DMA ADDRESS	NONE
1B	GET ALLOC ADDRESS	NONE	HL=ALLOC ADDRESS
1C	WRITE PROTECT DISK	NONE	NONE
1D	GET R/O VECTOR	NONE	HL=R/O VECTOR
1E	SET FILE ATTRIBUTE	DE=FCB ADDRESS	A=DIR CODE
1F	GET DISK PARAMS	NONE	HL=DPB ADDRESS
20	GET PARTITION	E=FF	A= PARTITION
	SET PARTITION	E=PARTITION	NONE
21	READ RANDOM	DE=FCB ADDRESS	A=ERROR CODE
22	WRITE RANDOM	DE=FCB ADDRESS	A=ERROR CODE
23	COMPUTE FILE SIZE	DE=FCB ADDRESS	RANDOM FIELD SET
24	SET RANDOM RECORD	DE=FCB ADDRESS	RANDOM FIELD SET
25	RESET DRIVE	DE=RESET DRIVE BIT	A=ERROR CODE
26	TRANSFER FILE	B=PARTITION DE=FCB ADDRESS	A=ERROR CODE
28	WRITE RANDOM (ZERO)	DE=FCB ADDRESS	A=ERROR CODE

ภาคผนวก ง: การสั่งงานโมดุลไอโอ

ADDRESS	ROUTINE	OPERATION
EB00	CBOOT	COLD BOOT
EB03	WBOOT	WARM BOOT
EB06	CONST	CONSOLE INPUT STATUS
EB09	CONIN	CONSOLE INPUT
EB0C	CONOUT	CONSOLE OUTPUT
EB0F	LIST	LIST DEVICE OUTPUT
EB12	PUNCH	PUNCH DEVICE OUTPUT
EB15	READER	READER DEVICE INPUT
EB18	HOME	HOME DRIVE
EB1B	SETDRV	SELECT DRIVE
EB1E	SETTRK	SET TRACK
EB21	SETSEC	SET SECTOR
EB24	SETDMA	SET DMA ADDRESS
EB27	READ	READ SECTOR
EB2A	WRITE	WRITE SECTOR
EB2D	LISTST	LIST DEVICE OUTPUT STATUS
EB30	SECTRN	SECTOR TRANSLATION

```
CROMEMCO CDOS Z80 ASSEMBLER version 02.15                PAGE 0001

                        0001 ;;;  EXTERNAL REFERENCE FROM EXTRA.  ::::::::::::::::
(FD00)                  0002 AGUERR  EQU     0FD00H  ;PRINT 'ARGUMENT ERROR'
(FD08)                  0003 AGUEXC  EQU     0FD08H  ;PRINT 'ARGUMENT EXCEED'
(FCF4)                  0004 CNOFND  EQU     0FCF4H  ;PRINT 'COMMAND NOT FOUND'
(FCF0)                  0005 COMERR  EQU     0FCF0H  ;PRINT 'COMMAND ERROR'
(FC26)                  0006 COMTBL  EQU     0FC26H  ;RESIDENT COMMAND TABLE
(FC00)                  0007 DRVNAM  EQU     0FC00H  ;ADDRESS OF ASSIGN DRIVE NAME
(FC08)                  0008 DSKID   EQU     0FC08H  ;ADDRESS OF ASSIGN DISK ID.
(FCFC)                  0009 FEXIST  EQU     0FCFCH  ;PRINT 'FILE EXISTS'
(FCF8)                  0010 FNOFND  EQU     0FCF8H  ;PRINT 'FILE NOT FOUND'
(FD04)                  0011 NINRNG  EQU     0FD04H  ;PRINT 'NOT IN RANGE'
(FD83)                  0012 PERAUS  EQU     0FD83H  ;PRINT ERA USAGE
(FD86)                  0013 PMOVUS  EQU     0FD86H  ;PRINT MOVE USAGE
(FDC9)                  0014 PRENUS  EQU     0FDC9H  ;PRINT REN USAGE
(FC09)                  0015 PROMPT  EQU     0FC09H  ;ADDRESS OF PROMPT SIGN
(FD70)                  0016 PSAVUS  EQU     0FD70H  ;PRINT SAVE USAGE
(FD8E)                  0017 PTYPUS  EQU     0FD8EH  ;PRINT TYPE USAGE
(FD9C)                  0018 PUSRUS  EQU     0FD9CH  ;PRINT USER USAGE
(FC66)                  0019 SHOW    EQU     0FC66H  ;RESIDENT COMMAND SHOW
                        0020 ;;;  EQUATES WHICH MUST BE CHANGED FROM ONE TO ONE VERS
                             ION :::::
(003E)                  0021 MSIZE   EQU     62      ;MEMORY SIZE OF SYATEM
(0008)                  0022 NUMCOM  EQU     8       ;AMOUNT OF RESIDENT COMMAND
(0008)                  0023 LENCOM  EQU     8       ;LENGTH OF RESIDENT COMMAND
                        0024 ;;;-----------------------------------------------
                        0025 ;              EQUATES                          :

                        0026 ;;;-----------------------------------------------
(0000)                  0027 WBOOT   EQU     0000H   ;WARM BOOT entry
(0004)                  0028 CDISK   EQU     0004H   ;SYSTEM CURRENT DRIVE
(0005)                  0029 BDOS    EQU     0005H   ;DOS entry
(005C)                  0030 SYSFCB  EQU     005CH   ;SYSTEM FCB
(0080)                  0031 SYSDMA  EQU     0080H   ;SYSTEM DMA add
(0081)                  0032 CMDTAL  EQU     0081H   ;SYSTEM COMMAND TAIL
(0100)                  0033 TPA     EQU     0100H   ;ADD OF TPA
(A800)                  0034 BIAS    EQU     (MSIZE-20)*1024 ;MEMORY OFFSET FOR 20K
                             SYSTEM
(D500)                  0035 RPA     EQU     2000H+BIAS      ;RESIDENT AREA
(DD00)                  0036 DOSADD  EQU     RPA+800H        ;DOS AREA
(0000)                  0037 NULL    EQU     00H     ;NULL char
(000A)                  0038 LF      EQU     0AH     ;LINE FEED char
(000D)                  0039 CR      EQU     0DH     ;CARRIAGE RETURN char
(001B)                  0040 ESC     EQU     1BH     ;ESCAPE char
                        0041 ;;;-----------------------------------------------
                        0042 ;      ENTRY POINTS TO RPA FROM I/O             :
                        0043 ;;;-----------------------------------------------
D000'                   0044         ORG     RPA
D500  C3D4D8            0045 START:  JP      AUTRUN  ;RPA entry point with AUTORUN
D503  C3D3D8            0046         JP      NOTAUT  ;RPA entry point NOT AUTORUN
                        0047 ;;;-----------------------------------------------
                        0048 ;     COMMAND BUFFER TO READ COMMAND LINE       :
                        0049 ;;;-----------------------------------------------
D506  7F                0050 MAXLEN: DB      7FH     ;Set MAX LENGTH 127 bytes
D507  00                0051 COMLEN: DB      00H     ;Length of COMMAND LINE
                        0052 ;;--- COMMAND BUFFER AREA ----------------------------
                             ---
                        0053 ;; 127 bytes for BUFFER and 1 last byte for pad with NU
```

```
                                LL
                          0054 ;;
D508  44495349            0055 COMBUF: DB      'DISK OPERATING SYSTEM'
      204F5045
      52415449
      4E472053
      59535445
      4D
D51D  20204445            0056         DB      '  DEVELOPED FROM STANDARD CP/M'
      56454C4F
      50454420
      46524F4D
      20535441
      4E444152
      44204350
      2F4D
D53B  2020464F            0057         DB      '  FOR CHULALONGKORN UNIVERSITY'
      52204348
      554C414C
      4F4E474B
      4F524E20
      554E4956
      45525349
      5459
D559  (002F)              0058         DS      47      ;DEFINE STORAGE 47 bytes
                          0059 ;;-------------------------------------------------
                               ----
D588  0BD5                0060 SCNBEG: DW      COMBUF  ;Start add to SCAN
D58A  0000                0061 STRBEG: DW      0000H   ;Add of first char that SCAN fo
                               und
                          0062 ;;;-----------------------------------------------
                          0063 ; CONIN: CONSOLE INPUT                          :
                          0064 ;    ENTRY: none                                :
                          0065 ;   ACTION: receive char from keyboard          :
                          0066 ;     EXIT: recieved char in A                  :
                          0067 ;;;-----------------------------------------------

D58C  0E01                0068 CONIN:  LD      C,01H
D58E  C30500              0069         JP      BDOS
                          0070 ;;;-----------------------------------------------
                          0071 ; GETCHR: GET CHAR FROM CONSOLE                 :
                          0072 ;    ENTRY: none                                :
                          0073 ;   ACTION: get char from console               :
                          0074 ;     EXIT: return input char in A              :
                          0075 ;;;-----------------------------------------------

D591  C5                  0076 GETCHR: PUSH    BC      ;Save REG
D592  E5                  0077         PUSH    HL
D593  CD8CD5              0078         CALL    CONIN   ;Use REG C to keep FUNC #
D596  E1                  0079         POP     HL      ;Restore REG
D597  C1                  0080         POP     BC
D598  C9                  0081         RET
                          0082 ;;;-----------------------------------------------
                          0083 ; CONOUT: CONSOLE OUTPUT                        :
                          0084 ;    ENTRY: char to write in A                  :
                          0085 ;   ACTION: write char on console               :
```

```
                        0086 ;     EXIT: none                                   :
                        0087 ;;;-------------------------------------------------

D599  5F                0088 CONOUT: LD      E,A
D59A  0E02              0089         LD      C,02H
D59C  C30500            0090         JP      DOS
                        0091 ;;;-------------------------------------------------
                        0092 ; WRTCHR: WRITE CHAR ON CONSOLE                     :
                        0093 ;     ENTRY: char to write in A                     :
                        0094 ;    ACTION: write char on console;                 :

                        0095 ;     EXIT: none                                   :
                        0096 ;;;-------------------------------------------------

D59F  C5                0097 WRTCHR: PUSH    BC      ;Save REG
D5A0  E5                0098         PUSH    HL
D5A1  CD99D5            0099         CALL    CONOUT  ;Use REG C to keep FUNC #
D5A4  E1                0100         POP     HL      ;Restore REG
D5A5  C1                0101         POP     BC
D5A6  C9                0102         RET
                        0103 ;;;-------------------------------------------------
                        0104 ; CRLF: WRITE CR&LF                                 :
                        0105 ;    ENTRY: none                                    :
                        0106 ;   ACTION: move CURSOR to new line                 :
                        0107 ;     EXIT: none                                   :
                        0108 ;;;-------------------------------------------------
D5A7  3E0D              0109 CRLF:   LD      A,CR
D5A9  CD9FD5            0110         CALL    WRTCHR  ;Write CR
D5AC  3E0A              0111         LD      A,LF
D5AE  18EF              0112         JR      WRTCHR  ;Write LF
                        0113 ;;;-------------------------------------------------
                        0114 ; WRTBLK: WRITE A BLANK ON CONSOLE                  :
                        0115 ;     ENTRY: none                                   :
                        0116 ;    ACTION: write a blank on console              :
                        0117 ;      EXIT: none                                  :
                        0118 ;;;-------------------------------------------------
D5B0  3E20              0119 WRTBLK: LD      A,' '   ;Move BLANK to write
D5B2  18E8              0120         JR      WRTCHR
                        0121 ;;;-------------------------------------------------
                        0122 ; WRTSTR: WRITE STRING                              :
                        0123 ;     ENTRY: BC points the beginning of STRING;     :
                        0124 ;            STRING terminated by NULL              :
                        0125 ;    ACTION: write each char of STRING              :
                        0126 ;            until find NULL                        :
                        0127 ;      EXIT: none                                   :
                        0128 ;;;-------------------------------------------------
D5B4  C5                0129 WRTSTR: PUSH    BC      ;BC points STRING
D5B5  CDA7D5            0130         CALL    CRLF    ;Get new line
D5B8  E1                0131         POP     HL      ;HL points STRING
D5B9  7E                0132 LPWSTR: LD      A,(HL)  ;Loop write STRING
D5BA  B7                0133         OR      A
D5BB  C8                0134         RET     Z       ;>>EXIT# STRING terminated
D5BC  23                0135         INC     HL      ;Move pointer to next char
D5BD  CD9FD5            0136         CALL    WRTCHR  ;Write a char on console
D5C0  18F7              0137         JR      LPWSTR  ;Loop until find NULL
                        0138 ;;;-------------------------------------------------
                        0139 ; RESDRV: RESET DRIVE                               :
```

```
                        0140 ;    ENTRY: none                                   :
                        0141 ;   ACTION: reset all drives to R/W;               :
                        0142 ;           logon DRIVE A; set DMA at 80H          :
                        0143 ;     EXIT: none                                   :
                        0144 ;;;-------------------------------------------------
D5C2  0E0D              0145 RESDRV: LD      C,0DH
D5C4  C30500            0146         JP      DOS
                        0147 ;;;-------------------------------------------------
                        0148 ; SELDRV: SELECT DRIVE                             :
                        0149 ;    ENTRY: code of drive to logon in A            :
                        0150 ;   ACTION: logon SELECTE DRIVE as CURRENT DRIVE:
                        0151 ;     EXIT: none                                   :
                        0152 ;;;-------------------------------------------------
D5C7  5F                0153 SELDRV: LD      E,A
D5C8  0E0E              0154         LD      C,0EH
D5CA  C30500            0155         JP      DOS
                        0156 ;;;-------------------------------------------------
                        0157 ;   REMARK: RETFLG IS USE FOR FUNCTION              :
                        0158 ;    oOPEN FILE   +    ALL OF THESE FUNCTIONS      :
                        0159 ;    oCLOSE FILE  +  RETURN DIR CODE OR FF         :
                        0160 ;    oSEARCH FIRST+  SO IF FUNCTION OK             :
                        0161 ;    oSEARCH NEXT +    -SAVE DIR CODE AND RET NZ:
                        0162 ;    oMAKE FILE   +  IF FUNCTION ERROR             :
                        0163 ;                      -SAVE FF AND RET Z          :
                        0164 ;                       (increament FF to 00)      :
                        0165 ;;;-------------------------------------------------
                        0166 ;-------------------------------------------------
                        0167 ;;;-------------------------------------------------
                        0168 ; RETFLG: RETURN DIR CODE AND SET FLAG              :
                        0169 ;    ENTRY: DISK I/O FUNC# in C; FCB add in DE     :
                        0170 ;   ACTION: call DOS                               :
                        0171 ;     EXIT: DIR CODE (0-3) in DIRCOD               :
                        0172 ;           and RET NZ if OK                       :
                        0173 ;          FF in DIRCOD and RET Z if ERROR         :
                        0174 ;;;-------------------------------------------------
D5CD  CD0500            0175 RETFLG: CALL    DOS
D5D0  32F4DC            0176         LD      (DIRCOD),A ;Save DIR CODE or FF
D5D3  3C                0177         INC     A          ;RET Z if ERROR (FF->00)
D5D4  C9                0178         RET                ;  else RET NZ if OK.
                        0179 ;;;-------------------------------------------------
                        0180 ; OPENFL: OPEN FILE                                :
                        0181 ;    ENTRY: DE points FCB add                      :
                        0182 ;   ACTION: open file                              :
                        0183 ;     EXIT: DIR CODE in DIRCOD and RET NZ if OK :
                        0184 ;          FF in DIRCOD and RET Z if NOT FOUND :
                        0185 ;;;-------------------------------------------------
D5D5  0E0F              0186 OPENFL: LD      C,0FH
D5D7  18F4              0187         JR      RETFLG
                        0188 ;;;-------------------------------------------------
                        0189 ; PREOPN: PREPARE FCB add FOR OPEN FILE             :
                        0190 ;    ENTRY: none                                   :
                        0191 ;   ACTION: set DE points TMPFCB to OPEN;          :
                        0192 ;           clear CURRENT RECORD to ZERO           :
                        0193 ;     EXIT: jump to process at OPENFL              :
                        0194 ;;;-------------------------------------------------
D5D9  AF                0195 PREOPN: XOR     A          ;Clear CURRENT RECORD
D5DA  32F3DC            0196         LD      (TCREC),A  ;  of file to be opened
```

```
05DD  11D3DC      0197        LD      DE,TMPFCB  ;Set DE points TMPFCB add
05E0  18F3        0198        JR      OPENFL     ;  then OPEN FILE
                  0199 ;;;-----------------------------------------------
                  0200 ; CLOSFL: CLOSE FILE                              :
                  0201 ;    ENTRY: DE points FCB add                     :
                  0202 ;   ACTION: close file                            :
                  0203 ;     EXIT: DIR CODE in DIRCID if OK and RET NZ   :
                  0204 ;           FF in DIRCOD if NOT FOUND and RET Z   :
                  0205 ;;;-----------------------------------------------
05E2  0E10        0206 CLOSFL: LD     C,10H
05E4  18E7        0207        JR      RETFLG
                  0208 ;;;-----------------------------------------------
                  0209 ; SFIRST: SEARCH FOR FIRST OCCURENCE               :
                  0210 ;    ENTRY: DE points FCB add                     :
                  0211 ;   ACTION: search for FIRST DIR which match FCB:
                  0212 ;     EXIT: DIR CODE in DIRCOD if OK and RET NZ   :
                  0213 ;           FF in DIRCOD if NOT FOUND and RET Z   :
                  0214 ;;;-----------------------------------------------
05E6  0E11        0215 SFIRST: LD     C,11H
05E8  18E3        0216        JR      RETFLG
                  0217 ;;;-------------------------------------------------
                  0218 ; SNEXT: SEARCH FOR NEXT OCCURENCE                  :
                  0219 ;   ENTRY: DE points FCB add                       :
                  0220 ;  ACTION: search for next DIR which match FCB     :
                  0221 ;    EXIT: DIR CODE in DIRCOD if OK and RET NZ     :
                  0222 ;          FF in DIRCOD if DIRECTORY END and RET Z:
                  0223 ;;;-------------------------------------------------
05EA  0E12        0224 SNEXT:  LD     C,12H
05EC  18DF        0225        JR      RETFLG
                  0226 ;;;-----------------------------------------------
                  0227 ; PRESF: PREPARE FCB add FOR SEARCH FOR FIRST      :
                  0228 ;   ENTRY: none                                    :
                  0229 ;  ACTION: set DE points TMPFCB                    :
                  0230 ;    EXIT: jump to process at SFIRST               :
                  0231 ;;;-----------------------------------------------
05EE  11D3DC      0232 PRESF:  LD     DE,TMPFCB  ;Set DE points TMPFCB
05F1  18F3        0233        JR      SFIRST     ;  then SEARCH FOR FIRST
                  0234 ;;;-----------------------------------------------
                  0235 ; DELFL: DELETE FILE                              :
                  0236 ;   ENTRY: DE points FCB                         :
                  0237 ;  ACTION: delete file which match FCB            :
                  0238 ;    EXIT: A=DIR CODE if OK                       :
                  0239 ;          A=FF if NOT FOUND                      :
                  0240 ;;;-----------------------------------------------
05F3  0E13        0241 DELFL:  LD     C,13H
05F5  C30500      0242        JP      BDOS
                  0243 ;;;-----------------------------------------------
                  0244 ;  REMARK: SETFLG IS USE FOR FUNCTION              :
                  0245 ;                    THESE FUNCTIONS RETURN       :
                  0246 ;    oREAD SEQ  +  00 OR NOT 00.                  :
                  0247 ;    oWRITE SEQ +  SO IF FUNCTION OK.             :
                  0248 ;                     -RET Z                      :
                  0249 ;                  IF FUNCTION ERROR              :
                  0250 ;                     -RET NZ                     :
                  0251 ; THE DIFFERENCES FROM RETFLG ARE                 :
                  0252 ;   SETFLG  -NOT SAVE DIR CODE                    :
                  0253 ;           -RET Z IF OK. (RETFLG RET NZ IF OK):
```

```
                        0254 ;;;-----------------------------------------------
                        0255 ;-----------------------------------------------
                        0256 ;;;-----------------------------------------------
                        0257 ;SETFLG: SET FLAG FOR DISK I/O OPERATION        :
                        0258 ;   ENTRY: C keeps FUNC #; DE points FCB add    :
                        0259 ;  ACTION: call DOS                             :
                        0260 ;    EXIT: A=00 and RET Z if OK.                :
                        0261 ;          A=NOT 00 and RET NZ if ERROR.        :
                        0262 ;;;-----------------------------------------------
 05F8  CD0500           0263 SETFLG: CALL    DOS
 05F8  B7               0264         OR      A       ;RET Z if OK
 05FC  C9               0265         RET             ;  else RET NZ if ERROR.
                        0266 ;;;-----------------------------------------------
                        0267 ; REDSEQ: READ SEQUENTIAL                       :
                        0268 ;    ENTRY: DE points FCB; CURRENT RECORD was set:
                        0269 ;   ACTION: read 128 bytes into DMA             :
                        0270 ;     EXIT: A=00 if READ OK and RET Z           :
                        0271 ;           else A=01 if End Of File and RET NZ :
                        0272 ;               A=02 if READ ERROR and RET NZ   :
                        0273 ;;;-----------------------------------------------
 05FD  0E14             0274 REDSEQ: LD      C,14H
 05FF  18F7             0275         JR      SETFLG
                        0276 ;;;-----------------------------------------------
                        0277 ; PREREQ: PREPARE FCB FOR READ SEQUENTIAL.       :
                        0278 ;    ENTRY: none                                 :
                        0279 ;   ACTION: set DE points TMPFCB add            :
                        0280 ;     EXIT: jump to process at REDSEQ           :
                        0281 ;;;-----------------------------------------------
 0601  11D3DC           0282 PREREQ: LD      DE,TMPFCB  ;Set DE points TMPFCB add
 0604  18F7             0283         JR      REDSEQ     ;  then READ SEQ
                        0284 ;;;-----------------------------------------------
                        0285 ; WRTSEQ: WRITE SEQUENTIAL                      :
                        0286 ;    ENTRY: DE points FCB add                   :
                        0287 ;   ACTION: write 128 bytes from DMA to file    :
                        0288 ;     EXIT: A=00 if WRITE OK and RET Z          :
                        0289 ;           else A=01 if WRITE ERROR and RET NZ :
                        0290 ;               A=02 if DISK FULL and RET NZ    :
                        0291 ;               A=FF if DIRECTORY FULL and RET NZ:
                        0292 ;;;-----------------------------------------------
 0606  0E15             0293 WRTSEQ: LD      C,15H
 0608  18EE             0294         JR      SETFLG
                        0295 ;;;-----------------------------------------------
                        0296 ; MAKEFL: MAKE FILE                             :
                        0297 ;    ENTRY: DE points FCB add                   :
                        0298 ;   ACTION: create new file                     :
                        0299 ;     EXIT: DIR CODE in DIRCOD if OK and RET Z  :
                        0300 ;           FF in DIRCOD if DIRECTORY FULL      :
                        0301 ;             and RET NZ                        :
                        0302 ;;;-----------------------------------------------
 060A  0E16             0303 MAKEFL: LD      C,16H
 060C  18BF             0304         JR      RETFLG
                        0305 ;;;-----------------------------------------------
                        0306 ; RENAME: RENAME FILENAME                       :
                        0307 ;    ENTRY: DE points FCB add;                  :
                        0308 ;           OLD NAME in FCB1/NEW NAME in FCB2   :
                        0309 ;   ACTION: change OLD NAME to NEW NAME         :
                        0310 ;     EXIT: A=DIR CODE if OK                    :
```

```
                        0311 ;           A=FF if OLD NAME is NOT FOUND      :
                        0312 ;;;-----------------------------------------------
D60E  0E17              0313 RENAME: LD      C,17H
D610  C30500            0314         JP      DOS
                        0315 ;;;-----------------------------------------------
                        0316 ; GETUSR: GET USER CODE                          :
                        0317 ;    ENTRY: none                                 :
                        0318 ;   ACTION: get CURRENT USER CODE                 :
                        0319 ;     EXIT: CURRENT USER CODE in A               :
                        0320 ;;;-----------------------------------------------
D613  1EFF              0321 GETUSR: LD      E,0FFH
                        0322 ;;;-----------------------------------------------
                        0323 ; SETUSR: SET USER CODE                          :
                        0324 ;    ENTRY: NEW USER CODE in E                   :
                        0325 ;   ACTION: set NEW USER CODE                    :
                        0326 ;     EXIT: none                                 :
                        0327 ;;;-----------------------------------------------
D615  0E20              0328 SETUSR: LD      C,20H
D617  C30500            0329         JP      DOS
                        0330 ;;;-----------------------------------------------
                        0331 ; TRNFER: TRANSFER FILE TO ANOTHER PARTITION.    :
                        0332 ;    ENTRY: B=partition#, DE points FCB          :
                        0333 ;   ACTION: transfer file from current to        :
                        0334 ;           assign partition                     :
                        0335 ;     EXIT: A=00 if OK.                          :
                        0336 ;           A=FF if NOT FOUND                    :
                        0337 ;;;-----------------------------------------------
D61A  0E26              0338 TRNFER: LD      C,26H
D61C  C30500            0339         JP      DOS
                        0340 ;;;-----------------------------------------------
                        0341 ; USRDSK: SAVE USER & CURRENT DRIVE CODE AT CDISK:
                        0342 ;         (RPA -> SYSTEM)                       :
                        0343 ;    ENTRY: new assign CURRENT DRIVE CODE was    :
                        0344 ;           saved at CURDRV                      :
                        0345 ;   ACTION: save USER & CURRENT DRIVE CODE at 4H :
                        0346 ;     EXIT: none                                 :
                        0347 ;;;-----------------------------------------------
D61F  CD13D6            0348 USRDSK: CALL    GETUSR     ;Get USER CODE in A
D622  87                0349         ADD     A,A        ;Move USER CODE to HIGH nibb
                          le
D623  87                0350         ADD     A,A        ;  ???? XXXX -+
D624  87                0351         ADD     A,A        ;             |
D625  87                0352         ADD     A,A        ;  XXXX 0000 <+
D626  21F5DC            0353         LD      HL,CURDRV
D629  B6                0354         OR      (HL)       ;Merge with new assign CUR D
                          RV
D62A  320400            0355         LD      (CDISK),A  ;  then save at CDISK.
D62D  C9                0356         RET
                        0357 ;;;-----------------------------------------------
                        0358 ; SCDISK: SAVE CURRENT DRIVE CODE AT CDISK       :
                        0359 ;         (RPA -> SYSTEM)                       :
                        0360 ;    ENTRY: new assign CURRENT DRIVE CODE was    :
                        0361 ;           saved at CURDRV                      :
                        0362 ;   ACTION: change CURRENT drive at add 4H       :
                        0363 ;           to NEW SELECT drive                  :
                        0364 ;     EXIT: none                                 :
                        0365 ;;;-----------------------------------------------
```

```
CROMEMCD CDDS Z80 ASSEMBLER version 02.15                PAGE 0008

D62E  3AF5DC      0366 SCDISK: LD      A,(CURDRV)  ;Get new assign CUR DRV COD
                       E
D631  32D400      0367         LD      (CDISK),A   ;  then save at CDISK.
D634  C9          0368         RET
                  0369 ;;;-----------------------------------------
                  0370 ; CONVUP: CONVERT TO UPPER CASE                :
                  0371 ;    ENTRY: char to convert upper case in A    :
                  0372 ;   ACTION: if char is lower case,             :
                  0373 ;             convert to upper case           :
                  0374 ;     EXIT: upper case char in A              :
                  0375 ;;;-----------------------------------------
D635  FE61        0376 CONVUP: CP      'a'
D637  D8          0377         RET     C       ;Less than 'a',not lower case
D638  FE7B        0378         CP      '{'
D63A  D0          0379         RET     NC      ;Greater than 'z',not lower cas
                       e
D63B  E65F        0380         AND     5FH     ;Convert to upper case by DFF b
                       it 5
D63D  C9          0381         RET             ;  0?X? ????,X=0 (Upper)/X=1 (L
                       ower)
                  0382 ;;;-----------------------------------------
                  0383 ; RUNBAT: RUN BATCH FILE                       :
                  0384 ;    ENTRY: enter into RPA and wait for user   :
                  0385 ;           console command                    :
                  0386 ;   ACTION: check if assign BATCH              :
                  0387 ;           if assign BATCH                    :
                  0388 ;            open BATCH FILE on DRIVE A and exec:
                  0389 ;           if not assign; goto get COMMAND LINE:
                  0390 ;     EXIT: get COMMAND from BATCH FILE        :
                  0391 ;           or~from keyboard                   :
                  0392 ;;;-----------------------------------------
D63E  3AB1DC      0393 RUNBAT: LD      A,(BATFLG)  ;Test BATFLG
D641  B7          0394         OR      A
D642  2852        0395         JR      Z,GCOMLI    ;Not assign BATCH,goto get
                       COM LINE
                  0396 ;
                  0397 ; BATCH FILE HAD BEEN ASSIGNED, OPEN FILE '$$$.SUB' ON
                       DRIVE A
                  0398 ; READ COMMAND IN LAST RECORD INTO DMA. THEN MOVE COMMA
                       ND TO
                  0399 ; COMMAND BUFFER. UPDATE FCB OF BATCH FILE TO READ A RE
                       CORD
                  0400 ; BEFORE LAST FOR NEXT EXEC AND SAVE FCB BY CLOSE FILE.
                  0401 ;   SO RPA WILL GET COMMAND LINE FROM BATCH FILE INSTEA
                       D OF
                  0402 ; GET FROM KEYBOARD. NEXT GOTO CONVERT COMMAND TO UPPER
                        CASE.
                  0403 ;   NOTE: BATFLG IS SET TO DECLEAR IN BATCH MODE.
                  0404 ;
D644  3AF5DC      0405         LD      A,(CURDRV) ;Assign BATCH
D647  B7          0406         OR      A          ;  if CURRENT DRIVE of RPA i
                       s also
D648  3E00        0407         LD      A,00H      ;   DRIVE A, not necessary t
                       o SELECT
D64A  C4C7D5      0408         CALL    NZ,SELDRV  ; else always SELECT DRIVE A
D64D  11B2DC      0409         LD      DE,BATFCB
D650  CDD5D5      0410         CALL    OPENFL     ;Open BATCH FILE
```

```
D653  2841        0411        JR      Z,GCOMLI   ;BATCH FILE NOT FOUND,get CO
                      M LINE
D655  3AC1DC      0412        LD      A,(BRECCT) ;Get RECORD COUNT of BATCH F
                      ILE
D658  3D          0413        DEC     A          ;  decrement to point last r
                      ecord
D659  32D2DC      0414        LD      (BCREC),A  ;   and use as CURRENT RECOR
                      D
D65C  1182DC      0415        LD      DE,BATFCB
D65F  CDFDD5      0416        CALL    REDSEQ     ;    then read that record
D662  2032        0417        JR      NZ,GCOMLI  ;>>EXIT# End Of File,goto ge
                      t COM LINE
D664  1107D5      0418        LD      DE,COMLEN  ;Move COM LINE assigned in B
                      ATCH FILE
D667  218000      0419        LD      HL,0080H   ;  from DMA to COMMAND BUFFE
                      R
D66A  0680        0420        LD      B,80H      ;   start from COMLEN (know
                      len of COM LINE)
D66C  CD1FD9      0421        CALL    MOVER      ;    length 128 bytes
D66F  21C0DC      0422        LD      HL,BS2
D672  3600        0423        LD      (HL),00H   ;Clear S2 with Zero
D674  23          0424        INC     HL         ;HL points RECORD COUNT
D675  35          0425        DEC     (HL)       ;Decrement RECORD COUNT
D676  1182DC      0426        LD      DE,BATFCB  ;  and save by CLOSE FILE
D679  CDE2D5      0427        CALL    CLOSFL     ;   for next exec BATCH FILE
D67C  2818        0428        JR      Z,GCOMLI   ;BATCH FILE NOT FOUND,get CO
                      M LINE
                  0429 ;
                  0430 ; GET ONE COMMAND LINE FROM BATCH FILE AND CLOSE BATCH
                      FILE,
                  0431 ; NOW ALWAYS LOGON DRIVE A. CHECK OLD CURRENT DRIVE OF
                      RPA WHICH
                  0432 ; LOGON BEFORE EXEC BATCH. IF ALSO DRIVE A NOT NECESSAR
                      Y TO SELECT
                  0433 ; ELSE SELECT BACK TO LOGON THAT DRIVE.
                  0434 ;
D67E  3AF5DC      0435        LD      A,(CURDRV) ;GET OLD CURRENT DRIVE of RP
                      A
D681  B7          0436        OR      A
D682  C4C7D5      0437        CALL    NZ,SELDRV  ;Not DRIVE A,SELECT back
D685  2103D5      0438        LD      HL,COMBUF  ;Write COMMAND LINE get from
D688  CDB9D5      0439        CALL    LPWSTR     ;  BATCH FILE on screen
D68B  CDC0D6      0440        CALL    CONSTA     ;Get CONSOLE STATUS
D68E  2817        0441        JR      Z,CVTCOM   ;No key press,goto convert u
                      pper case
                  0442 ;
                  0443 ; IF ANY KEY IS PRESSED, EXEC IS ABORTED. DELETE BATCH
                      FILE
                  0444 ; AND RESTART RPA TO GET NEXT COMMAND LINE FROM KEYBOAR
                      D.
                  0445 ;
D690  CDD3D6      0446        CALL    DELBAT     ;Delete BATCH FILE
D693  C329D8      0447        JP      RSTRPA     ;  then restart RPA
                  0448 ;;;-----------------------------------------
                  0449 ; GCOMLI: GET COMMAND LINE                    :
                  0450 ;    ENTRY: BATCH is not assigned or NOT FOUND  :
                  0451 ;   ACTION: delete BATCH FILE;                 :
```

```
                        0452 ;          get COMMAND LINE from keyboard      :
                        0453 ;----------------------------------------------
D696  CDDBD6            0454 GCOMLI: CALL    DELBAT    ;Delete BATCH FILE and logon
                             CUR DRIVE
D699  CD1FD6            0455         CALL    USRDSK    ;Save USER & CUR DRV CODE at
                             CDISK
D69C  0E0A              0456         LD      C,0AH     ;FUNCTION 10 is READ STRING
D69E  1106D5            0457         LD      DE,MAXLEN ;  set STRING BUF at COM BUF
D6A1  CD0500            0458         CALL    BDOS      ;Read COMMAND LINE into COM B
                             UF
D6A4  CD2ED6            0459         CALL    SCDISK    ;Save CURRENT DRIVE at CDISK
                        0460 ;
                        0461 ; CVTCOM: CONVERT COMMAND LINE TO UPPER CASE
                        0462 ;    ENTRY: in routine
                        0463 ;   ACTION: convert all char in COMMAND LINE to upper c
                             ase
                        0464 ;     EXIT: goto pad end of LINE with NULL
                        0465 ;
D6A7  2107D5            0466 CVTCOM: LD      HL,COMLEN
D6AA  46                0467         LD      B,(HL)    ;Get len of COMMAND LINE
D6AB  23                0468 LOPUPC: INC     HL        ;Loop convert upper case
D6AC  78                0469         LD      A,B
D6AD  B7                0470         OR      A         ;>>EXIT? End of COMMAND LINE
D6AE  2809              0471         JR      Z,PADNUL  ;  goto pad end of LINE with
                             NULL
D6B0  7E                0472         LD      A,(HL)
D6B1  CD35D6            0473         CALL    CONVUP    ;Convert each char to upper c
                             ase
D6B4  77                0474         LD      (HL),A
D6B5  05                0475         DEC     B         ;Decrement length by 1
D6B6  18F3              0476         JR      LOPUPC    ;  and loop until length and
                        0477 ;
                        0478 ; PADNUL: PAD NULL AT THE END OF COMMAND LINE
                        0479 ;    ENTRY: HL points next after end of COMMAND LINE
                        0480 ;           A=0 (length of COM LINE decrement to Zero)
                        0481 ;   ACTION: pad end of COMMAND LINE with NULL
                        0482 ;           set start add to SCAN at COMMAND BUFFER
                        0483 ;     EXIT: NULL STRING or STRING pad with NULL
                        0484 ;
D6B8  77                0485 PADNUL: LD      (HL),A       ;Pad end with NULL
D6B9  2108D5            0486         LD      HL,COMBUF    ;Set start add to SCAN
D6BC  2283D5            0487         LD      (SCNBEG),HL  ;  at COMBUF
D6BF  C9                0488         RET
                        0489 ;;;---------------------------------------------
                        0490 ; CONSTA: CONSOLE STATUS                       :
                        0491 ;    ENTRY: none                               :
                        0492 ;   ACTION: check console status               :
                        0493 ;             00 is WAIT / FF is CHAR READY    :
                        0494 ;           if CHAR READY, get in a char        :
                        0495 ;     EXIT: A=00 and RET Z if WAIT FOR CHAR     :
                        0496 ;           A=char and RET NZ if CHAR READY     :
                        0497 ;;;---------------------------------------------
D6C0  0E0B              0498 CONSTA: LD      C,0BH
D6C2  CD0500            0499         CALL    BDOS
D6C5  B7                0500         OR      A
D6C6  C8                0501         RET     Z         ;No key press,A=00 and RET Z
D6C7  3E01              0502         LD      C,01H
```

```
D6C9  CD0500      0503        CALL    DOS     ;Key press,READ CON
D6CC  B7          0504        OR      A       ;  get char in A
D6CD  C9          0505        RET             ;  and RET NZ
                  0506 ;;;-----------------------------------------------
                  0507 ; SETDRV: GET SYSTEM CURRENT DRIVE FROM CDISK   :
                  0508 ;         (SYSTEM -> RFA)                      :
                  0509 ;    ENTRY: none                               :
                  0510 ;   ACTION: to SYSTEM find CURRENT logon drive  :
                  0511 ;     EXIT: SYSTEM CDISK in A                  :
                  0512 ;;;-----------------------------------------------
D6CE  0E19        0513 GETDRV: LD      C,19H
D6D0  C30500      0514        JP      DOS
                  0515 ;;;-----------------------------------------------
                  0516 ; DMA80H: SET DMA add AT 80H                    :
                  0517 ;    ENTRY: none                               :
                  0518 ;   ACTION: set DMA add at 80H                 :
                  0519 ;     EXIT: jump to process at SETDMA          :
                  0520 ;;;-----------------------------------------------
D6D3  118000      0521 DMA80H: LD      DE,0080H  ;Set DMA at 80H
                  0522 ;;;-----------------------------------------------
                  0523 ; SETDMA: SET DMA add                           :
                  0524 ;    ENTRY: DMA add in DE                      :
                  0525 ;   ACTION: set DMA at add points by DE         :
                  0526 ;     EXIT: none                               :
                  0527 ;;;-----------------------------------------------
D6D6  0E1A        0528 SETDMA: LD      C,1AH
D6D8  C30500      0529        JP      DOS
                  0530 ;;;-----------------------------------------------
                  0531 ; DELBAT: DELETE BATCH FILE                     :
                  0532 ;    ENTRY: not assign BATCH or assign but ERROR :
                  0533 ;           (NOT FOUND,EOF)                    :
                  0534 ;   ACTION: if assign BATCH FILE                :
                  0535 ;       delete BATCH FILE on DRIVE A;          :
                  0536 ;       clear BATFLG to declear not in BATCH mode:
                  0537 ;         then SELECT back to OLD CURRENT DRIVE :
                  0538 ;     EXIT: none                               :
                  0539 ;;;-----------------------------------------------
D6D9  2131DC      0540 DELBAT: LD      HL,BATFLG
D6DC  7E          0541        LD      A,(HL)    ;Get BATFLG
D6DD  B7          0542        OR      A
D6DE  C8          0543        RET     Z         ;>>EXIT# Not assign BATCH,re
                    t
D6DF  3600        0544        LD      (HL),00H  ;Clear BATFLG
D6E1  AF          0545        XOR     A
D6E2  CDC7D5      0546        CALL    SELDRV    ;Always select DRIVE A
D6E5  1132DC      0547        LD      DE,BATFCB ;DE points FCB of BATCH FILE
D6E8  CDF3D5      0548        CALL    DELFL     ;  in DRIVE A, then delete i
                    t
D6EB  3AF5DC      0549        LD      A,(CURDRV) ;SELECT back to
D6EE  C3C7D5      0550        JP      SELDRV    ;  OLD CURRENT DRIVE
                  0551 ;;;-----------------------------------------------
                  0552 ; CHKPAT: CHECK 6 BYTES FOR COMPATIBLE WITH DOS  :
                  0553 ;    ENTRY: none                               :
                  0554 ;   ACTION: check 6 bytes in RFA with begin of DOS:
                  0555 ;     EXIT: if all compat, OK                  :
                  0556 ;           else SYSTEM FAIL                   :
                  0557 ;;;-----------------------------------------------
```

```
D6F3 11CCD7     0558 CHKPAT: LD      DE,COMPAT  ;DE points compat 6 bytes
D6F6 21B0DD     0559         LD      HL,DOSADD  ;HL points DOS add
D6F9 0606       0560         LD      B,06H      ;  length 6 bytes
D6FB 1A         0561 LOPPAT: LD      A,(DE)     ;Loop check compat
D6FC BE         0562         CP      (HL)
D6FD C280D8     0563         JP      NZ,SYSFAL  ;>>EXIT# Not compat, SYS FAI
                    L
D700 13         0564         INC     DE
D701 23         0565         INC     HL
D702 10F7       0566         DJNZ    LOPPAT  ;Loop check next byte
D704 C9         0567         RET             ;>>EXIT# All are compat,ret
                0568 ;;;-----------------------------------------------
                0569 ; DLIMIT: RTN TO FIND DELIMITOR OR TERMINATOR       :
                0570 ;    ENTRY: DE points a char in STRING             :
                0571 ;   ACTION: check if that char is terminator (NULL) :
                0572 ;           or delimitor ( SP = - . : ; < > )      :
                0573 ;     EXIT: if char is CTRL-CHAR, goto print arg err :
                0574 ;           if char is delimitor or terminator RET Z :
                0575 ;             else simply RET.                     :
                0576 ;           char return in A.                      :
                0577 ;;;-----------------------------------------------
D705 1A         0578 DLIMIT: LD      A,(DE)     ;Get that char
D706 B7         0579         OR      A
D707 C8         0580         RET     Z          ;NULL
D708 FE20       0581         CP      ' '
D70A DA00FD     0582         JP      C,ARGERR   ;CTRL-CHAR, write 'ARGUMENT
                    ERROR'
D70D C8         0583         RET     Z          ;SPACE
D70E FE3D       0584         CP      '='
D710 C8         0585         RET     Z          ;EQUAL SIGN
D711 FE5F       0586         CP      '_'
D713 C8         0587         RET     Z          ;UNDERLINE
D714 FE2E       0588         CP      '.'
D716 C8         0589         RET     Z          ;PERIOD
D717 FE3A       0590         CP      ':'
D719 C8         0591         RET     Z          ;COLON
D71A FE3B       0592         CP      ';'
D71C C8         0593         RET     Z          ;SEMI-COLON
D71D FE3C       0594         CP      '<'
D71F C8         0595         RET     Z          ;LESS THAN
D720 FE3E       0596         CP      '>'
D722 C8         0597         RET     Z          ;GREATER THAN
D723 C9         0598         RET                ;CHARACTER.
                0599 ;;;-----------------------------------------------
                0600 ; CUTBLK: RTN TO CUT OUT BLANK                      :
                0601 ;    ENTRY: POINTER is in DE                       :
                0602 ;   ACTION: move POINETR pass BLANK                 :
                0603 ;           until meet FIRST char or find TERMI    :
                0604 ;     EXIT: POINETR in DE points START             :
                0605 ;             or END of STRING (if STRING END)     :
                0606 ;           FIRST char or NULL in A                :
                0607 ;;;-----------------------------------------------
D724 1A         0608 CUTBLK: LD      A,(DE)  ;Loop cut blank
D725 B7         0609         OR      A
D726 C9         0610         RET     Z       ;>>EXIT# Found TERMI,ret
D727 FE20       0611         CP      ' '
D729 D0         0612         RET     NZ      ;>>EXIT# Found FIRST char,ret
```

```
D72A 13          0613          INC     DE      ;Found blank
D72B 18F7        0614          JR      CUTBLK  ;  loop to check next char
                 0615 ;;;-----------------------------------------------
                 0616 ; MOVSCN: RTN TO MOVE POINTER                    :
                 0617 ;    ENTRY: POINTER in HL;                       :
                 0618 ;           distance to move POINTER in A        :
                 0619 ;   ACTION: move POINTER to point new add        :
                 0620 ;     EXIT: POINTER in HL points new add        :
                 0621 ;;;-----------------------------------------------
D72D 85          0622 MOVSCN: ADD     A,L     ;ADD distance to old POINTER
D72E 6F          0623          LD      L,A
D72F D0          0624          RET     NC
D730 24          0625          INC     H       ;Has CARRY,carry to H
D731 C9          0626          RET             ;  HL=HL+A
                 0627 ;;;-----------------------------------------------
                 0628 ; INTERP: INTERPRET COMMAND & COMMAND TAIL        :
                 0629 ;    ENTRY: add of STRING to interpret in SCNBEG  :
                 0630 ;   ACTION: DE points COMBUF/ HL points TMPFCB    :
                 0631 ;           move FN&FT from SCNBEG to TMPFCB      :
                 0632 ;           first set SELNUM to default           :
                 0633 ;           expand '*' to '?'                     :
                 0634 ;           fill BLANK or ignore the exceed part  :
                 0635 ;           clear 2 RESERVE bytes and EXTENT NUM  :
                 0636 ;     EXIT: count amount of '?' in B              :
                 0637 ;           RET Z if no '?', else RET NZ          :
                 0638 ;;;-----------------------------------------------
D732 3E00        0639 INTERP: LD      A,00H   ;Set offset in TMPFCB to Zero
                 0640 ;;;-----------------------------------------------
                 0641 ; INFCB2: INTERPRET COMMAND TAIL AND PUT IN TMPFCB2  :
                 0642 ;    ENTRY: offset distance to TMPFCB2 in A         :
                 0643 ;   ACTION: as INTERP (This entry point use in TRANSI);
                 0644 ;;;-----------------------------------------------
D734 210D0C      0645 INFCB2: LD      HL,TMPFCB   ;+HL points TMPFCB for INTE
                     RP
D737 CD2DD7      0646          CALL    MOVSCN      ;+HL points TMPFCB2 for INF
                     CB2
D73A E5          0647          PUSH    HL          ;Save add of FN&FT to count '
                     ?'
D73B AF          0648          XOR     A           ;Set SELNUM to default
D73C 32F6DC      0649          LD      (SELNUM),A  ;  to use if arg not assign
                         DISK ID
D73F ED5B88D5    0650          LD      DE,(SCNBEG) ;Get start add to SCAN
D743 CD24D7      0651          CALL    CUTBLK      ;Cut out blank
D746 ED538AD5    0652          LD      (STRBEG),DE ;Save add of FIRST char fou
                         nd
D74A 1A          0653          LD      A,(DE)      ;Get FIRST char of arg
D74B B7          0654          OR      A           ;It is NULL at end of STRIN
                     G
D74C 280F        0655          JR      Z,BEGARG    ;  so goto fill FN&FT with
                     blank
D74E E5          0656          PUSH    HL          ;Save add of TMPFCB
D74F 21D8FC      0657          LD      HL,DSKID    ;First assume it is DISK ID

D752 96          0658          SUB     (HL)        ;  subtract to be DRIVE COD
                         E
D753 3C          0659          INC     A           ;    increment to be SELNUM
D754 E1          0660          POP     HL          ;Restore add of TMPFCB
```

```
D755 47          0661        LD      B,A         ;  temp save if must be us
                     ed
D756 13          0662        INC     DE
D757 1A          0663        LD      A,(DE)
D758 FE3A        0664        CP      ':'         ;   and if next char is ':
                     '
D75A 2807        0665        JR      Z,FILLDK    ;    goto fill SEL DRV NUM

D75C 1B          0666        DEC     DE          ;Move POINTER back to begin
                 0667 ;
                 0668 ; BEGARG: BEGIN TO MOVE ARG TO TMPFCB
                 0669 ;    ENTRY: arg in COMMAND LINE
                 0670 ;   ACTION: this arg is not assign DISK ID
                 0671 ;           so use CURRENT DRIVE in TMPFCB
                 0672 ;     EXIT: goto fill FILENAME
                 0673 ;
D75D 3AF5DC      0674 BEGARG: LD     A,(CURDRV) ;Not assign DISK ID
D760 77          0675        LD      (HL),A  -  ;  use CUR DRIVE in DR of TM
                     PFCB
D761 1806        0676        JR      FILLFN     ;Goto fill FILENAME
                 0677 ;
                 0678 ; FILLDK: FILL SEL DRV IN TMPFCB AND SAVE IN SELNUM
                 0679 ;    ENTRY: assign DISK ID in arg
                 0680 ;   ACTION: save SELECT DRIVE in arg into SELNUM
                 0681 ;           and fill SELNUM in DR of TMPFCB
                 0682 ;     EXIT: goto fill FILENAME
                 0683 ;
D763 78          0684 FILLDK: LD     A,B
D764 32F6DC      0685        LD      (SELNUM),A ;Save SEL DRIVE in SELNUM
D767 70          0686        LD      (HL),B     ;  and fill in DR of TMPFCB
D768 13          0687        INC     DE         ;Move POINTER to first char
                     of FN
                 0688 ;
                 0689 ; FILLFN: FILL FILENAME FROM COMMAND LINE INTO TMPFCB
                 0690 ;    ENTRY: already fill DR in TMPFCB
                 0691 ;   ACTION: move assign FILENAME to fill in TMPFCB
                 0692 ;           if assign '*' expand to '?'
                 0693 ;     EXIT: FILENAME end or already fill 8 char
                 0694 ;
D769 0608        0695 FILLFN: LD     B,08H      ;Set loop for FILENAME 8 char
D76B CD05D7      0696 LOOPFN: CALL   DLIMIT     ;Found DELI or TERMI (assign(
                     8 char)
D76E 2815        0697        JR      Z,BLKFN    ;  goto fill the rest with bl
                     ank
D770 23          0698        INC     HL         ;Move FCB SCANNER to next
D771 FE2A        0699        CP      '*'        ;Check if assign '*'
D773 2004        0700        JR      NZ,NEXPFN  ;Not assign '*'
D775 363F        0701        LD      (HL),'?'   ;Assign '*' expand to '?'
D777 1802        0702        JR      EXPFN      ;  and not move STR SCANNER
                 0703 ;
                 0704 ;;; NEXPFN: NOT EXPAND '?' IN FILENAME
D779 77          0705 NEXPFN: LD     (HL),A     ;Move char of FILENAME to TMP
                     FCB
D77A 13          0706        INC     DE         ;Move STR SCANNER to next
                 0707 ;
                 0708 ;;; EXPFN: EXPAND '?' FOR FILENAME IN TMPFCB
D77B 10EE        0709 EXPFN:  DJNZ    LOOPFN     ;Loop move FILENAME to TMPFCB
```

```
                        0710 ;
                        0711 ; IGORFN: IGNORE THE EXCEED 8 CHAR OF FILENAME
                        0712 ;    ENTRY: already move FILENAME 8 char but STRING not
                             and
                        0713 ;   ACTION: ignore the rest by move STR SCANNER
                        0714 ;           until found DELIMITOR or TERMINATOR
                        0715 ;     EXIT: goto fill FILETYPE
                        0716 ;
D770  CD05D7            0717 IGORFN: CALL    DLIMIT   ;Found DELI or TERMI
D773  2803              0718         JR      Z,FILLFT ;  goto fill FILETYPE
D775  13                0719         INC     DE
D776  18F8              0720         JR      IGORFN   ;Loop move STR SCANNER
                        0721 ;
                        0722 ; BLKFN: FILL BLANK IN FILENAME
                        0723 ;   ENTRY: assign FILENAME less than 8 char or not assi
                             gn
                        0724 ;  ACTION: fill the rest of FILENAME with blank
                        0725 ;    EXIT: goto fill FILETYPE
                        0726 ;
D778  23                0727 BLKFN:  INC     HL
D779  3620              0728         LD      (HL),' ' ;Fill TMPFCB with blank FILEN
                             AME
D77B  10FB              0729         DJNZ    BLKFN    ;Loop fill blank until fit 8
                             char
                        0730 ;
                        0731 ; FILLFT: FILL FILETYPE INTO TMPFCB
                        0732 ;    ENTRY: already fill FILENAME
                        0733 ;   ACTION: if not assign FILETYPE
                        0734 ;             goto fill FILETYPE in TMPFCB with blank
                        0735 ;           assign FILETYPE move to fill in TMPFCB
                        0736 ;             and if has '*' in assign, expand to '?'
                        0737 ;     EXIT: FILETYPE and or already fill 3 char
                        0738 ;
D77D  0603              0739 FILLFT: LD      B,03H    ;Set length of FILETYPE 3 cha
                             r
D77F  FE2E              0740         CP      '.'      ;Not assign FILETYPE
D781  201B              0741         JR      NZ,BLKFT ;  goto fill with blank
D783  13                0742         INC     DE       ;Move STR SCANNER to FIRST ch
                             ar of FT
D784  CD05D7            0743 LOOPFT: CALL    DLIMIT   ;Found DELI or TERMI (assign
                             3 char)
D787  2815              0744         JR      Z,BLKFT  ;  goto fill rest with blank
D789  23                0745         INC     HL
D78A  FE2A              0746         CP      '*'      ;Check if assign '*'
D78C  2004              0747         JR      NZ,NEXPFT ;Not assign '*'
D78E  363F              0748         LD      (HL),'?' ;Assign '*',move '?' to TMPFC
                             B
D790  1802              0749         JR      EXPFT    ;  and not move STR SACNNER
                        0750 ;
                        0751 ;;; NEXPFT: NOT EXPAND '?' IN FILETYPE
D792  77                0752 NEXPFT: LD      (HL),A   ;Move char of FILETYPE to TMPF
                             CB
D793  13                0753         INC     DE       ;Move STR SCANNER to next
                        0754 ;
                        0755 ;;; EXPFT: EXPAND '?' FOR FILETYPE
D794  10EE              0756 EXPFT:  DJNZ    LOOPFT   ;Loop move FILETYPE 3 char
                        0757 ;
```

```
                        0758 ; IGORFT: IGNORE THE EXCEED 3 CHAR OF FILETYPE
                        0759 ;    ENTRY: already fill FILETYPE 3 char but STRING not
                              end
                        0760 ;   ACTION: ignore the rest by move STR SCANNER
                        0761 ;           until found DELIMITOR or TERMINATOR
                        0762 ;     EXIT: goto FILL ZERO
                        0763 ;
D7A3  CD05D7            0764 IGORFT: CALL    DLIMIT    ;Found DELI or TERMI
D7A6  2808              0765         JR      Z,FILLZR  ; goto FILL ZERO
D7A8  13                0766         INC     DE
D7A9  18F8              0767         JR      IGORFT    ;Loop move STR SCANNER
                        0768 ;
                        0769 ; BLKFT: FILL FILETYPE WITH BLANK
                        0770 ;   ENTRY: assign FILETYPE less than 3 char or not assi
                              gn
                        0771 ;  ACTION: fill the rest of FILETYPE with blank
                        0772 ;    EXIT: goto FILL ZERO
                        0773 ;
D7AB  23                0774 BLKFT:  INC     HL
D7AC  3620              0775         LD      (HL),' '    ;Fill TMPFCB with blank FIL
                              ETYPE
D7AE  10FB              0776         DJNZ    BLKFT
                        0777 ;
                        0778 ; FILLZR: FILL ZERO FOR 2 RESERVED BYTES AND EXTENT NUM

                        0779 ;    ENTRY: already fill FN&FT in TMPFCB

                        0780 ;   ACTION: clear ZERO in RESERVED S1,S2 and EXTENT NUM

                        0781 ;     EXIT: goto count '?' in FN&FT

                        0782 ;
D7B0  0603              0783 FILLZR: LD      B,03H       ;Clear ZERO for 3 bytes
D7B2  23                0784 LOOPZR: INC     HL
D7B3  3600              0785         LD      (HL),00H
D7B5  10FB              0786         DJNZ    LOOPZR      ;Loop clear ZERO
D7B7  ED5388D5          0787         LD      (SCNBEG),DE ;Save START add for next
                              SCAN
D7BB  E1                0788         POP     HL          ;POP add of FILENAME
D7BC  010000            0789         LD      BC,0000H    ;B=count # of '?'/C=# cha
                              r of FN&FT
                        0790 ;
                        0791 ; LOPQMK: LOOP COUNT '?' IN FILENAME & FILETYPE
                        0792 ;    ENTRY: in RTN
                        0793 ;   ACTION: count amount of '?' of FN&FT in TMPFCB
                        0794 ;     EXIT: amount of '?' in B
                        0795 ;           RET Z if no '?' else RET NZ
                        0796 ;
D7BF  23                0797 LOPQMK: INC     HL
D7C0  7E                0798         LD      A,(HL)      ;Get a char in FN&FT
D7C1  FE3F              0799         CP      '?'
D7C3  2001              0800         JR      NZ,NOTQMK   ;Not '?'
D7C5  04                0801         INC     B           ;It is '?', increment counte
                              r
D7C6  0D                0802 NOTQMK: DEC     C
D7C7  20F6              0803         JR      NZ,LOPQMK   ;Loop until end of FILETYPE
D7C9  78                0804         LD      A,B         ;Ret amount of '?' in B
```

```
D7CA  B7          0805          OR      A          ; and RET Z/NZ
D7CB  C9          0806          RET
                  0807 ;
                  0808 ;;; COMPAT: COMPATIBLE 6 BYTES WITH BEGIN OF DOS ------
                       -------
D7CC  221600      0809 COMPAT: DB      22H,16H,00H
D7CF  00FB9       0810          DB      00H,0FH,0B9H
                  0811 ;;;-----------------------------------------
                  0812 ; OFFVEC: OFFSET IN VECTOR TABLE                :
                  0813 ;    ENTRY: none                                :
                  0814 ;   ACTION: check which RESIDENT COMMAND match  :
                  0815 ;           with FILENAME in TMPFCB             :
                  0816 ;     EXIT: number of RESIDENT COMMAND in A     :
                  0817 ;           (start from command number 0)       :
                  0818 ;;;-----------------------------------------
D7D2  2126FC      0819 OFFVEC: LD      HL,COMTBL  ;Set to begin of COMMAND TAB
     LE
D7D5  0E00        0820          LD      C,00H      ;C=count which RESIDENT
                  0821 ;
                  0822 ;;; BILCOM: LOOP CHECK WHICH RESIDENT COMMAND
D7D7  79          0823 BILCOM: LD      A,C
D7D8  FE08        0824          CP      NUMCOM  ;Loop all RESIDENT
D7DA  D0          0825          RET     NC      ;  but not match, it is TRANSIE
     NT
D7DB  11D4DC      0826          LD      DE,TFN1 ;DE points FIRST char of FN in
     TMPFCB
D7DE  0608        0827          LD      B,LENCOM ;Length of RESIDENT COMMAND
                  0828 ;
                  0829 ;;; BILCHR: LOOP CHECK EACH CHAR OF EACH RESIDENT COMMA
     ND
D7E0  1A          0830          LD      A,(DE)
D7E1  BE          0831          CP      (HL)       ;Check first char of COMMAND
D7E2  2016        0832          JR      NZ,SKPCOM
D7E4  13          0833          INC     DE
D7E5  23          0834          INC     HL
D7E6  05          0835          DEC     B
D7E7  1A          0836 BILCHR: LD      A,(DE)
D7E8  FE20        0837          CP      ' '        ;Check char after first is b
     lank
D7EA  2807        0838          JR      Z,THSCOM   ;Yes, partial COMMAND
D7EC  BE          0839          CP      (HL)       ;Compare each char
D7ED  2008        0840          JR      NZ,SKPCOM  ;  but not match,skip to nex
     t COMMAND
D7EF  13          0841          INC     DE
D7F0  23          0842          INC     HL         ;Get next pair to compare
D7F1  10F4        0843          DJNZ    BILCHR     ;Compare not all char,loop n
     ext char
D7F3  1A          0844 THSCOM: LD      A,(DE)     ;All char are match
D7F4  FE20        0845          CP      ' '        ;  but next char is not blan
     k
D7F6  2005        0846          JR      NZ,NXTCOM  ;  goto check next COMMAND
D7F8  79          0847          LD      A,C        ;All char match and follow w
     ith
D7F9  C9          0848          RET                ;  blank,ret number of RESID
     ENT in A
                  0849 ;
                  0850 ; SKPCOM: SKIP POINTER PASS THIS COMMAND IN COMMAND TAB
```

```
                        LE
                    0851 ;    ENTRY: found a first char in this RESIDENT which n
                        ot match
                    0852 ;             with FILENAME in TMPFCB
                    0853 ;   ACTION: skip POINTER pass this command to next comm
                        and
                    0854 ;           in COMMAND TABLE
                    0855 ;
D7FA 23             0856 SKPCOM: INC     HL         ;Skip POINTER to next COMMAN
                        D
D7FB 10FD           0857         DJNZ    SKPCOM     ;Loop skip pointer
                    0858 ;
                    0859 ;;; NXTCOM: BEGIN WITH NEXT COMMAND
D7FD 0C             0860 NXTCOM: INC     C          ;Increment number of
D7FE 18D7           0861         JR      BILCOM     ;  RESIDENT COMMAND
                    0862 ;;;-----------------------------------------
                    0863 ; NOTAUT: JUMP FROM BIOS TO RPA WITH NOT AUTORUN:
                    0864 ;    ENTRY: USER & CURRENT DRIVE CODE in C      :
                    0865 ;   ACTION: clear length of COMMAND LINE        :
                    0866 ;           from USER & DRIVE CODE              :
                    0867 ;;;-----------------------------------------
D800 AF             0868 NOTAUT: XOR     A          ;NOT AUTO,clear length
D801 3207D5         0869         LD      (COMLEN),A ;  of COMMAND LINE to Zero
D804 31AFDC         0870 AUTRUN: LD      SP,STACK   ;Set up STACK (jump from B
                        IOS)
D807 C5             0871         PUSH    BC         ;Save CDISK
D808 79             0872         LD      A,C
D809 1F             0873         RRA                ;Move HIGH nibble to LOW n
                        ibble
D80A 1F             0874         RRA                ;    XXXX ???? -+
D80B 1F             0875         RRA                ;               !
D80C 1F             0876         RRA                ;    ???? XXXX <+
D80D E60F           0877         AND     0FH        ;From USER CODE (strip out
                        CDISK)
D80F 5F             0878         LD      E,A        ;USER CODE in E
D810 CD15D6         0879         CALL    SETUSR     ;Set USER AREA
D813 CDC2D5         0880         CALL    RESDRV     ;Reset all drives
D816 32B1DC         0881         LD      (BATFLG),A ;Clear BATFLG to not BATCH
                        mode
D819 C1             0882         POP     BC         ;Get CDISK
D81A 79             0883         LD      A,C
D81B E60F           0884         AND     0FH        ;From CDISK (strip out USE
                        R CODE)
D81D 32F5DC         0885         LD      (CURDRV),A ;Use CDISK as CURRENT DRIV
                        E
D820 CDC7D5         0886         CALL    SELDRV     ;  and logon THAT drive
D823 3A07D5         0887         LD      A,(COMLEN) ;Check length of COM LINE
D826 B7             0888         OR      A
D827 201C           0889         JR      NZ,INTCOM  ;AUTO RUN, goto interpret
                        COMMAND
                    0890 ;
                    0891 ; RSTRPA: RESTART RPA
                    0892 ;    ENTRY: when finish each RESIDENT COMMAND
                    0893 ;           when finish TRANSIENT and come to this poin
                        t by 'RET'
                    0894 ;           when interpret COMMAND LINE and found arg e
                        rr
```

```
                        0895 ;          when exec command in BATCH FILE finish and
                             come
                        0896 ;            to exec next command
                        0897 ;   ACTION: if come from BATCH mode,BATFLG had been set
                        0898 ;          goto get COMMAND LINE from BATCH FILE
                        0899 ;            else get COMMAND LINE from keyboard
                        0900 ;
D829  31AFDC            0901 RSTRPA: LD     SP,STACK  ;Set up STACK (when restart R
                             PA)
D82C  0100FC            0902         LD     BC,DRVNAM
D82F  CDB4D5            0903         CALL   WRTSTR    ;Get new line and write DRIVE
                              NAME
D832  CDCED6            0904         CALL   GETDRV    ;Get CDISK
D835  2108FC            0905         LD     HL,DSKID  ;Get DISK ID.
D838  86                0906         ADD    A,(HL)    ;  change CODE to ASCII
D839  CD99D5            0907         CALL   CONOUT    ;Write DISK ID
D83C  3A09FC            0909         LD     A,(PROMPT) ;Get Prompt sign
D83F  CD99D5            0909         CALL   CONOUT    ;Write Prompt sign
D842  CDCED6            0910         CALL   RUNBAT    ;If in BATCH mode,exec BATCH
                             FILE
                        0911 ;
                        0912 ; INTCOM: INTERPRET COMMAND IN COMMAND LINE
                        0913 ;    ENTRY: not assign BATCH, get COMMAND LINE from key
                             board
                        0914 ;   ACTION: get COMMAND LINE from keyboard
                        0915 ;           move only COMMAND part to TMPFCB
                        0916 ;
D845  118000            0917 INTCOM: LD     DE,0080H
D848  CDD6D6            0918         CALL   SETDMA    ;Set DMA at B0H
D84B  CDCED6            0919         CALL   GETDRV
D84E  32F5DC            0920         LD     (CURDRV),A  ;Use CDISK as CURRENT DRIVE
D851  CD32D7            0921         CALL   INTERP    ;Get COM LINE and move COMM
                             AND part
D854  C4F0FC            0922         CALL   NZ,COMERR ;Has any '?' , write 'COMMA
                             ND ERROR'
D857  3AF6DC            0923         LD     A,(SELNUM) ;Get SELECT DRIVE in arg
D85A  B7                0924         OR     A
D85B  C298D8            0925         JP     NZ,TRANSI ;Assign DISK ID,it is TRANS
                             IENT
                        0926 ;
                        0927 ; THIS COMMAND NOT ASSIGN DISK ID, SO MAY BE RESIDENT O
                             R TRANSIENT
                        0928 ; CHECK WHICH COMMAND AND GET OFFSET, FORM OFFSET TO GE
                             T VECTOR
                        0929 ; THEN JUMP ON VECTOR
                        0930 ;
D85E  CDD2D7            0931         CALL   OFFVEC    ;Check which COMMAND to fin
                             d offset
D861  216ED8            0932         LD     HL,VECTBL ;Point at begin of VECTOR T
                             ABLE
D864  5F                0933         LD     E,A
D865  1600              0934         LD     D,00H
D867  19                0935         ADD    HL,DE
D868  19                0936         ADD    HL,DE     ;From offset
D869  7E                0937         LD     A,(HL)
D86A  23                0938         INC    HL
D86B  66                0939         LD     H,(HL)
```

```
D86C  6F          0940        LD      L,A         ;Get VECTOR
D86D  E9          0941        JP      (HL)        ;  and jump on VECTOR
                  0942 ;;;  VECTOR COMMAND TABLE.  ::::::::::::::::::::::
      (D86E)      0943 VECTBL  EQU     $
D86E  52D9        0944        DW      DIR
D870  DCD9        0945        DW      ERA
D872  E4DA        0946        DW      REN
D874  28DA        0947        DW      TYPE
D876  7DDA        0948        DW      SAVE
D878  55DB        0949        DW      USER
D87A  71DB        0950        DW      MOVE
D87C  66FC        0951        DW      SHOW
D87E  98DB        0952        DW      TRANSI
                  0953 ;;;-----------------------------------------
                  0954 ; SYSFAL: SYSTEM FAIL                      :
                  0955 ;    ENTRY: jump from RTN check compat and :
                  0956 ;           found not compat               :
                  0957 ;   ACTION: move HALT (76H) and DI (F3H) to :
                  0958 ;           begin of RPA, then jump to RPA :
                  0959 ;     EXIT: System HALT                    :
                  0960 ;;;-----------------------------------------
D880  21F376      0961 SYSFAL: LD      HL,76F3H
D883  2200D5      0962        LD      (RPA),HL
D886  2100D5      0963        LD      HL,RPA
D889  E9          0964        JP      (HL)
                  0965 ;;;-----------------------------------------
                  0966 ; REDMSG: RTN WRITE 'READ ERROR' MESSAGE   :
                  0967 ;    ENTRY: when disk read error           :
                  0968 ;   ACTION: print 'READ ERROR' on screen   :
                  0969 ;     EXIT: restart RPA                   :
                  0970 ;;;-----------------------------------------
D88A  0190D8      0971 REDMSG: LD      BC,READER  ;BC points err message
D88D  C384D5      0972        JP      WRTSTR     ;  write err message
                  0973 ;
                  0974 ;;; READER: 'READ ERROR' MESSAGE ----------------------
                       --------
D890  52656164    0975 READER: DB      'Read Error',NULL
      20457272
      6F7200
                  0976 ;;;-----------------------------------------
                  0977 ; ALLSOM: ALL FILES OR SOME FILES.         :
                  0978 ;    ENTRY: when NOT FOUND and test return code :
                  0979 ;   ACTION: if flag NOT ZERO => FOUND      :
                  0980 ;               log back OLD CUR DRV        :
                  0981 ;           if flag ZERO => NOT FOUND       :
                  0982 ;               check if '*.*' print 'NO FILE' :
                  0983 ;               else print 'FILE NOT FOUND' :
                  0984 ;     EXIT: restart RPA                     :
                  0985 ;;;-----------------------------------------
D893  CD41D9      0986 ALLSOM: CALL    LOGOLD  ;NOT FOUND ,log back OLD CUR DR
                       V
D896  0609        0987        LD      B,09H   ;Length to check '?' in FN&FT
D898  21D4DC      0988        LD      HL,TFN1 ;Check file which not found
D89B  7E          0989 LNOFND: LD      A,(HL)
D89C  FE3F        0990        CP      '?'     ;If '?'
D89E  C2F8FC      0991        JP      NZ,FNOFND  ;No, print 'FILE NOT FOUND'
D8A1  23          0992        INC     HL
```

```
D8AA  10F7      0993        DJNZ    LNOFND
                0994 ;;;-----------------------------------------
                0995 ; PNOFIL: RTN WRITE NO FILE MESSAGE          :
                0996 ;    ENTRY: when file not found             :
                0997 ;   ACTION: print 'NO FILE' on screen       :
                0998 ;     EXIT: restart RPA                     :
                0999 ;;;-----------------------------------------
D8AC  0188D8    1000 PNOFIL: LD     BC,NOFILE  ;BC point err message
D8AF  CDB4D5    1001         CALL   WRTSTR     ;  write err message
D8B2  CDA7D5    1002         CALL   CRLF
D8B5  C329D8    1003         JP     RSTRPA
                1004 ;
                1005 ;;; NOFILE: 'NO FILE' MESSAGE ----------------------
                     --------
D888  44697368  1006 NOFILE: DB     'Disk Empty',NULL
      20456D70
      747900
                1007 ;;;-----------------------------------------
                1008 ; CALVAL: CALCULATE VALUE OF ARG             :
                1009 ;    ENTRY: when current RESIDENT COMMAND want :
                1010 ;           assign value to process        :
                1011 ;   ACTION: fill arg in TMPFCB and convert  :
                1012 ;           ASC val to HEX val             :
                1013 ;     EXIT: HEX val in A                   :
                1014 ;;;-----------------------------------------
D8C3  CD3207    1015 CALVAL: CALL   INTERP     ;Fill arg in TMPFCB
D8C6  3AF6DC    1016         LD     A,(SELNUM) ;Check if addign DISK ID
D8C9  B7        1017         OR     A
D8CA  C200FD    1018         JP     NZ,AGUERR  ;Assign DISK ID, write arg e
                     rr
D8CD  21D4DC    1019         LD     HL,TFN1    ;Start from first digit
D8D0  7E        1020         LD     A,(HL)     ;Get first char of FILENAME
D8D1  FE20      1021         CP     ' '        ;  check if blank
D8D3  2010      1022         JR     NZ,NOBARG  ;Not blank, calculate value
D8D5  3AF8DC    1023         LD     A,(FLAG)   ;  else get flag
D8D8  FE53      1024         CP     'S'
D8DA  CA70FD    1025         JP     Z,PSAVUS   ;If 'S'... SAVE
D8DD  FE55      1026         CP     'U'
D8DF  CA9CFD    1027         JP     Z,PUSRUS   ;If 'U'... USER
D8E2  C386FD    1028         JP     PMOVUS     ;  else... MOVE
D8E5  010B00    1029 NOBARG: LD     BC,000BH   ;Clear B,set len 11 ASC digit
                     in C
                1030 ;
                1031 ; LOPCAL: LOOP CALCULATE
                1032 ;    ENTRY: clear B to save result
                1033 ;           set C to length of 11 ASC digit (max)
                1034 ;   ACTION: each digit is in range 0-9
                1035 ;           so calculate by change to BASE 8 in binary
                     form
                1036 ;            by R=RESULT, D=DIGIT (get each digit at a
                      time)
                1037 ;        R init Zero / D change from ASC to Binary digi
                     t (31->01)
                1038 ;            R = R*8 + 2R + D ,loop until finish all 3
                     igits
                1039 ;     EXIT: loop until digit end or overflow
                1040 ;
```

```
D8EB 7E          1041 LOPCAL: LD      A,(HL)
D8EC FE20        1042         CP      ' '
D8EE 2824        1043         JR      Z,ENDCAL  ;>>EXIT# Found blank,digit en
                      d
D8F0 23          1044         INC     HL        ;Get next digit
D8F1 D630        1045         SUB     30H       ;Change ASC to Binary val
D8F3 FE0A        1046         CP      0AH
D8F5 D200FD      1047         JP      NC,AGUERR ;>>EXIT# Exceed 9, write arg
                      err
D8F8 57          1048         LD      D,A       ;Save DIGIT in D
D8F9 78          1049         LD      A,B       ;Get RESULT from B
D8FA E6E0        1050         AND     0E0H
D8FC C200FD      1051         JP      NZ,AGUERR ;RESULT overflow,write agr er
                      r
D8FF 78          1052         LD      A,B       ;RESULT must not exceed 000X
                      XXX
D900 07          1053         RLCA              ;  for rotate left 3 times
D901 07          1054         RLCA              ;  the RESULT will not overfl
                      ow
D902 07          1055         RLCA              ;<1>This step equal R*8
D903 80          1056         ADD     A,B       ;<2>This step equal R*8+R
D904 DA00FD      1057         JP      C,AGUERR  ;  if overflow, write arg err
D907 80          1058         ADD     A,B       ;<3>This step equal R*8+2R
D908 DA00FD      1059         JP      C,AGUERR  ;  if overflow, write arg err
D90B 82          1060         ADD     A,D       ;<4>This step equal R*8+2R+D
D90C DA00FD      1061         JP      C,AGUERR  ;  if overflow, write arg err
D90F 47          1062         LD      B,A       ;Save RESULT in B for next DI
                      GIT
D910 0D          1063         DEC     C         ;Decrement length
D911 20D8        1064         JR      NZ,LOPCAL ;  but not end,goon loop
D913 C9          1065         RET               ;Calculate all 11 digits,ret
                 1066 ;
                 1067 ; ENDCAL: END OF CALCULATION
                 1068 ;    ENTRY: loop calculate digit until found blank
                 1069 ;           (digit less 11 digits)
                 1070 ;   ACTION: check all the rest must be blank
                 1071 ;     EXIT: ret HEX val in A
                 1072 ;           or write arg err if the all rest not blank
                 1073 ;
D914 7E          1074 ENDCAL: LD      A,(HL)
D915 FE20        1075         CP      ' '
D917 C200FD      1076         JP      NZ,AGUERR ;Not blank, write arg err
D91A 23          1077         INC     HL        ;Get next
D91B 0D          1078         DEC     C         ;Decrement counter
D91C 20F6        1079         JR      NZ,ENDCAL ;  but not end,goon loop
D91E 78          1080         LD      A,B       ;Loop end,ret Hex val in A
D91F C9          1081         RET
                 1082 ;;;-----------------------------------------------
                 1083 ; MVER3B: MOVER LENGTH 3 BYTES                    :
                 1084 ;    ENTRY: HL points SOURCE, DE points DESTINATION :
                 1085 ;   ACTION: Move 3 bytes from SOURCE to DES (HL->DE):
                 1086 ;     EXIT: none                                  :
                 1087 ;-----------------------------------------------
D920 0603        1088 MVER3B: LD      B,03H   ;Length 3 bytes
                 1089 ;;;-----------------------------------------------
                 1090 ; MOVER: MOVER                                    :
                 1091 ;   ENTRY: HL points SOURCE, DE points DESTINATION :
```

```
                        1092 ;          B keeps LENGTH to move                  :
                        1093 ;  ACTION: Move from SOURCE to DSE (HL->DE)        :
                        1094 ;    EXIT: none                                    :
                        1095 ;;;-------------------------------------------------
D91F 7E                 1096 MOVER:  LD      A,(HL)
D920 12                 1097         LD      (DE),A
D921 23                 1098         INC     HL
D922 13                 1099         INC     DE
D923 10FA               1100         DJNZ    MOVER
D925 C9                 1101         RET
                        1102 ;;;-----------------------------------------
                        1103 ; CINDMA: CALCULATE OFFSET IN DMA                :
                        1104 ;    ENTRY: C is offset, A is displacement      :
                        1105 ;   ACTION: get a char at add offset and disp   :
                        1106 ;           in DMA                              :
                        1107 ;     EXIT: char from DMA in A                  :
                        1108 ;;;-----------------------------------------
D926 218000   -         1109 CINDMA: LD      HL,0080H ;Begin at DMA
D929 81                 1110         ADD     A,C      ;Add offset with disp
D92A CD2DD7             1111         CALL    MOVSCN   ;Add (offset and disp) with D
                             MA
D92D 7E                 1112         LD      A,(HL)   ;Get a char at (DMA+offset+di
                             sp)
D92E C9                 1113         RET              ;  or at (HL + A + C)
                        1114 ;;;-----------------------------------------
                        1115 ; LOGSEL: LOGON SELECT DRIVE (RPA goto COMMAND) :
                        1116 ;    ENTRY: RPA current drive in CURDRV        :
                        1117 ;           assign DISK ID of arg in SELNUM    :
                        1118 ;   ACTION: set DR of TMPFCB to default        :
                        1119 ;           (for can be used with whatever     :
                        1120 ;            drive logon later)                :
                        1121 ;           if SELECT DRIVE not assign         :
                        1122 ;              use CURRENT DRIVE               :
                        1123 ;           if SELECT DRIVE assign             :
                        1124 ;              logon SELECT DRIVE              :
                        1125 ;     EXIT: always logon SELECT DRIVE          :
                        1126 ;;;-----------------------------------------
D92F AF                 1127 LOGSEL: XOR     A
D930 32D30C             1128         LD      (TDR1),A    ;Set DR to default
D933 3AF6DC             1129         LD      A,(SELNUM)  ;Get SELECT DRIVE in arg
D936 B7                 1130         OR      A
D937 C8                 1131         RET     Z           ;SELECT DRIVE is default,re
                             t
D938 3D                 1132         DEC     A           ;Change SEL NUM to DRIVE CO
                             DE
D939 21F5DC             1133         LD      HL,CURDRV   ;Get CURRENT DRIVE
D93C BE                 1134         CP      (HL)        ;SELECT DRIVE assign in arg
D93D C8                 1135         RET     Z           ;  same as CURRENT DRIVE,re
                             t
D93E C3C7D5             1136         JP      SELDRV      ; else logon SELECT DRIVE
                        1137 ;;;-----------------------------------------
                        1138 ; LOGOLD: LOGON OLD CURRENT DRIVE                :
                        1139 ;         (COMMAND back to RPA)                   :
                        1140 ;    ENTRY: CURRENT DRIVE before goto process    :
                        1141 ;           command in CURDRV, SELECT DRIVE of  :
                        1142 ;           command in SELNUM                   :
                        1143 ;   ACTION: if SELECT DRIVE is default          :
```

```
                        1144 ;            means CURRENT DRIVE not change,ret:
                        1145 ;          if SELECT DRIVE same as CURRENT DRV :
                        1146 ;            means CURRENT DRIVE not change,ret:
                        1147 ;        if SELECT DRV not same as CURRENT DRV:
                        1148 ;            means CURRENT DRIVE was changed,  :
                        1149 ;            logon back to CURRENT DRIVE       :
                        1150 ;     EXIT: logon OLD CURRENT DRIVE            :
                        1151 ;-----------------------------------------------
D941  3AF6DC            1152 LOGOLD: LD     A,(SELNUM)
D944  B7                1153         OR     A
D945  C8                1154         RET    Z           ;SELECT DRIVE is default,ret
D946  3D                1155         DEC    A           ;Change SEL NUM to DRIVE COD
                             E
D947  21F5DC            1156         LD     HL,CURDRV   ;Get CURRENT DRIVE
D94A  BE                1157         CP     (HL)
D94B  C8                1158         RET    Z           ;SELECT DRV same as CUR DRV,
                             ret
D94C  3AF5DC            1159         LD     A,(CURDRV)
D94F  C3C7D5            1160         JP     SELDRV      ;  else logon back to OLD CU
                             R DRV
                        1161 ;;;--------------------------------------------
                        1162 ;       BUILT-IN COMMAND DIR                    :
                        1163 ;;;--------------------------------------------
D952  CD32D7            1164 DIR:    CALL   INTERP      ;Move arg of DIR into TMPFCB
D955  CD2FD9            1165         CALL   LOGSEL      ;Logon SELECT DRV assign in
                             arg
D958  21D4DC            1166         LD     HL,TFN1
D95B  7E                1167         LD     A,(HL)
D95C  FE20              1168         CP     ' '         ;Get first char of arg
D95E  C268D9            1169         JP     NZ,PRTDIR   ;  Not blank,assign FILENAME
D961  0603              1170         LD     B,03H       ;Length to fill '?' in FN&FT
                        1171 ;
                        1172 ; FILDIR: FILL DIR ARG WITH '?'
                        1173 ;    ENTRY: DIR command not assign FILENAME
                        1174 ;   ACTION: fill 11 '?' in FN&FT to match all files
                        1175 ;
D963  363F              1176 FILDIR: LD     (HL),'?'
D965  23                1177         INC    HL
D966  10FB              1178         DJNZ   FILDIR      ;Loop fill '?' in FN&FT
                        1179 ;
                        1180 ; PRTDIR: PRINT DIRECTORY ENTRY
                        1181 ;    ENTRY: assign FN&FT to display in TMPFCB
                        1182 ;   ACTION: set counter of ENTRY/LINE
                        1183 ;           SEARCH FOR FIRST into DMA
                        1184 ;     EXIT: write 'NO FILE' if NOT FOUND
                        1185 ;           else loop display DIR
                        1186 ;
D968  1E00              1187 PRTDIR: LD     E,00H      ;E=count ENTRY/LINE
D96A  D5                1188         PUSH   DE         ;Save COUNTER
D96B  CDEED5            1189         CALL   PRESF      ;SEARCH FOR FIRST at TMPFCB
D96E  CA9BD8            1190         JP     Z,ALLSOM   ;NOT FOUND,goto print message
                        1191 ;
                        1192 ; LOPPRT: LOOP PRINT DIRECTORY ENTRY
                        1193 ;    ENTRY: from SEARCH FOR FIRST and SEARCH FOR NEXT
                        1194 ;   ACTION: get DIR CODE to select which DIR ENTRY (1 i
                             n 4)
                        1195 ;           and display that ENTRY on screen
```

```
                        1196 ;     EXIT: when all ENTRY are display,
                        1197 ;           SEARCH FOR NEXT will NOT FOUND
                        1198 ;
D971  CA0809            1199 LOPPRT: JP      Z,DIREND   ;>>EXIT# SEARCH NEXT NOT FOU
                             ND
D974  3AF4DC            1200         LD      A,(DIRCOD) ;SEARCH NEXT OK,get DIR CODE
D977  0F                1201         RRCA               ;DIR CODE*32          ???? ??XX
                              -+
D978  0F                1202         RRCA               ;(len of DIR ENTRY)
                               !
D979  0F                1203         RRCA               ; to form offset in    ?XX? ????
                              (+
D97A  E660              1204         AND     60H        ;  DIR BUF at 30H  [AND 0110 0000
                               ]
D97C  4F                1205         LD      C,A        ;Save offset points which ENTRY i
                             n C
D97D  3E0A              1206         LD      A,0AH      ;Disp in A, offset in C
D97F  CD26D9            1207         CALL    CINDMA     ;Get char at (80H+0AH+offset
                             ) in A
D982  17                1208         RLA                ;  it is second char of FILE
                             TYPE
D983  3847              1209         JR      C,SETCON   ;   set 'SYS',ignore this EN
                             TRY
D985  D1                1210         POP     DE         ;Get COUNTER
D986  7B                1211         LD      A,E        ;  move to CURRENT OF LINE
D987  1C                1212         INC     E          ;Increasant COUNTER
D988  D5                1213         PUSH    DE         ;  and save
D989  E603              1214         AND     03H        ;Modulo CURRENT OF LINE to 0
                              -3
D98B  2017              1215         JR      NZ,NOTET0  ;Not being display ENTRY 0
                        1216 ;
                        1217 ; BEING DISPLAY ENTRY 0 (FIRST ENTRY OF LINE)
                        1218 ; SO GET NEW LINE, WRITE DISK ID AND ':'
                        1219 ; THEN GOTO DISPLAY ENTRY
                        1220 ;
D98D  C5                1221         PUSH    BC         ;Save REG C use to keep FUNC
                              #
D98E  0100FC            1222         LD      BC,DRVNAM
D991  CDB4D5            1223         CALL    WRTSTR     ;Get new line and write DRIVE
                             NAME
D994  CDCED6            1224         CALL    GETDRV     ;Get CDISK in A
D997  C1                1225         POP     BC         ;Restore REG C
D998  2108FC            1226         LD      HL,DSKID   ;Get DISK ID.
D99B  86                1227         ADD     A,(HL)     ;  to change CODE to ASCII
D99C  CD9FD5            1228         CALL    WRTCHR     ;Write DISK ID
D99F  3E3A              1229         LD      A,':'
D9A1  CD9FD5            1230         CALL    WRTCHR     ;Write ':'
D9A4  1808              1231         JR      DIRET0     ;Goto display ENTRY
                        1232 ;
                        1233 ; NOTET0: NOT ENTRY 0
                        1234 ;    WRITE BLANK, ':' AND A BLANK
                        1235 ;
D9A6  CDB0D5            1236 NOTET0: CALL    WRTBLK     ;Write blank at end of ENTRY
D9A9  3E3A              1237         LD      A,':'
D9AB  CD9FD5            1238         CALL    WRTCHR     ;Separate by ':'
                        1239 ;
                        1240 ; DIRET0: ENTRY 0
```

```
                        1241 ;   HAD BEEN WRITE DISK ID AND ':'
                        1242 ;   SO WRITE ONLY A BLANK
                        1243 ;
D9AE  CD80D5            1244 DIRET0: CALL    WRTBLK  ;Write blank at start of ENTRY
D9B1  0601              1245         LD      B,01H   ;B=offset into FN&FT
                        1246 ;
                        1247 ; LOPDIR: LOOP PRINT EACH CHAR OF DIRECTORY ENTRY
                        1248 ;    ENTRY: offset in C (points the correct ENTRY)
                        1249 ;           disp in B (init 1 to points first char of F
                             N)
                        1250 ;   ACTION: display each char of FN&FT
                        1251 ;     EXIT: display finish,goto GETCON
                        1252 ;
D9B3  78                1253 LOPDIR: LD      A,B     ;Disp in A,offset in C
D9B4  CD26D9            1254         CALL    CINDMA  ;Get char at 80H+disp+offset
                             ) in A
D9B7  E67F              1255         AND     7FH     ;Strip out 'R/O' and 'SYS' bi
                             t
D9B9  CD9FD5            1256         CALL    WRTCHR  ;Write DIR
D9BC  04                1257         INC     B       ;Increment displacement
D9BD  78                1258         LD      A,B
D9BE  FE0C              1259         CP      0CH     ;Write TYPE finish?
D9C0  D2CCD9            1260         JP      NC,GETCON ;Yes, goto GETCON
D9C3  FE09              1261         CP      09H     ;Write NAME finish?
D9C5  20EC              1262         JR      NZ,LOPDIR ;No, continue write NAME
D9C7  CD80D5            1263         CALL    WRTBLK  ;  else separate from NAME wi
                             th a blank
D9CA  18E7              1264         JR      LOPDIR  ;  and loop write TYPE
                        1265 ;
                        1266 ; GETCON: GET CONSOLE STATUS
                        1267 ;    ENTRY: already display a ENTRY
                        1268 ;   ACTION: get console status
                        1269 ;           if key press, DIR abort
                        1270 ;           if no key press, SEARCH FOR NEXT
                        1271 ;     EXIT: display next ENTRY or goto DIREND
                        1272 ;
D9CC  CDC0D6            1273 GETCON: CALL    CONSTA  ;Get CON STA
D9CF  C2D8D9            1274         JP      NZ,DIREND  ;Key press, DIR abort goto D
                             IREND
D9D2  CDEAD3            1275         CALL    SNEXT   ;No key press, SEARCH FOR NE
                             XT
D9D5  C371D9            1276         JP      LOPPRT  ;Begin new loop for next ENT
                             RY
                        1277 ;
                        1278 ; DIREND: DIR BUILT-IN COMMAND END
                        1279 ;    ENTRY: when SEARCH FOR NEXT and NOT FOUND
                        1280 ;           or DIR abort
                        1281 ;   ACTION: select back to OLD CURRENT DRIVE
                        1282 ;     EXIT: check if has more arg,write arg arr
                        1283 ;           then restart CCP
                        1284 ;
D9D8  D1                1285 DIREND: POP     DE      ;POP COUNTER in E to clear STAC
                             K
D9D9  C37ADC            1286         JP      OLDDRV  ;Select back OLD CUR DRV
                        1287 ;;;----------------------------------------
                        1288 ;           RESIDENT COMMAND ERA          :
                        1289 ;;;----------------------------------------
```

```
D9DC  CD32D7      1290 ERA:   CALL    INTERP     ;Move arg of ERA into TMPFCB
D9DF  FE08        1291        CP      08H
D9E1  2018        1292        JR      NZ,ERASOM  ;Arg not '*.*',erase some fi
                      le
                  1293 ;
                  1294 ; ARG OF ERA COMMAND IS '*.*'. PRINT STRING 'ALL (Y/N)?
                      '
                  1295 ; AND GET ACTION. ACTION MAY COME FROM BATCH FILE OR FR
                      OM
                  1296 ; KEYBOARD, BUT ACTION IF NOT 'Y', ERA ABORT.
                  1297 ;
D9E3  0117DA      1298        LD      BC,ALLYN   ;BC points string
D9E6  CDB4D5      1299        CALL    WRTSTR     ;  write string 'ALL (Y/N)?
                      '
D9E9  CD3ED6      1300        CALL    RUNBAT     ;Get action from BATCH or k
                      eyboard
D9EC  2107D5      1301        LD      HL,COMLEN  ;Get length of COMMAND LINE
D9EF  35          1302        DEC     (HL)
D9F0  C229D8      1303        JP      NZ,RSTRPA  ;Len is not 1 (Y or N),rest
                      art RPA
D9F3  23          1304        INC     HL
D9F4  7E          1305        LD      A,(HL)     ;Get action
D9F5  FE59        1306        CP      'Y'
D9F7  C229D8      1307        JP      NZ,RSTRPA  ;Not 'Y',restart RPA
D9FA  23          1308        INC     HL
D9FB  2298D5      1309        LD      (SCNBEG),HL ;Save add of next SCAN
                  1310 ;
                  1311 ; ERASOM: ERASE SOME FILE
                  1312 ;    ENTRY: assign FILENAME in arg
                  1313 ;           or assign '*.*' and action is 'Y'
                  1314 ;   ACTION: delete file assign in TMPFCB on SELECT DRIV
                      E
                  1315 ;     EXIT: select back to OLD CURRENT DRIVE
                  1316 ;           check if assign more arg,write that arg err
                  1317 ;             then restart RPA
                  1318 ;
D9FE  21D40C      1319 ERASOM: LD     HL,TFN1
DA01  7E          1320        LD      A,(HL)     ;Get first char of FILENAME
DA02  FE20        1321        CP      ' '        ;  check if blank
DA04  CAB3FD      1322        JP      Z,PERAUS   ;   goto print usage
DA07  CD2F09      1323        CALL    LOGSEL     ;Logon SEL DRIVE assign in ar
                      g
DA0A  11D3DC      1324        LD      DE,TMPFCB
DA0D  CDF3D5      1325        CALL    DELFL      ;Delete file assign in TMPFCB
DA10  3C          1326        INC     A          ;Test return code
DA11  CA9BD8      1327        JP      Z,ALLSOM   ; NOT FOUND, goto write messa
                      ge
DA14  C37ADC      1328        JP      OLDDRV     ; FOUND, log back DRV
                  1329 ;
                  1330 ;;; ALLYN: 'ALL (Y/N)?' MESSAGE ----------------------
                      --------
DA17  416C6C20    1331 ALLYN:  DB      'All Files (Y/N)?',NULL
      46696C65
      73202859
      2F4E293F
      00
                  1332 ;;;-----------------------------------------
```

```
                        1333 ;       RESIDENT COMMAND TYPE              :
                        1334 ;;;-----------------------------------------
DA28  CD32D7            1335 TYPE:   CALL    INTERP    ;Move arg of TYPE into TMPFCB
DA2B  C200FD            1336         JP      NZ,AGUERR ;Has any '?', write 'ARGUMENT
                            ERROR'
DA2E  21D4DC            1337         LD      HL,TFN1
DA31  7E                1338         LD      A,(HL)    ;Get first char of FILENAME
DA32  FE20              1339         CP      ' '       ;  check if blank
DA34  CABEFD            1340         JP      Z,PTYPUS  ;    goto print usage
DA37  CD2FD9            1341         CALL    LOGSEL    ;Logon SEL DRIVE assign in ar
                            g
DA3A  CDD9D5            1342         CALL    FREOPN    ;Open file at TMPFCB
DA3D  2838              1343         JR      Z,ARGTYE  ; NOT FOUND,write arg err
DA3F  CDA7D5            1344         CALL    CRLF      ;Get new line
DA42  21F7DC            1345         LD      HL,CUNTER ;Use CUNTER as offset in DMA
DA45  36FF              1346         LD      (HL),0FFH ;  init offset to FF
                        1347 ;
                        1348 ; LOPTYE: LOOP TYPE CHAR
                        1349 ;    ENTRY: OPEN FILE on SELECT DRIVE assign in arg
                        1350 ;           init offset to FF
                        1351 ;   ACTION: check offset if less than 128,
                        1352 ;           type next char in sector
                        1353 ;             else read in next sector and init offset
                        1354 ;
DA47  21F7DC            1355 LOPTYE: LD      HL,CUNTER  ;Get counter
DA4A  7E                1356         LD      A,(HL)
DA4B  FE80              1357         CP      80H        ;Check offset
DA4D  3809              1358         JR      C,TYENXT   ;Offset less than 128,type n
                            ext cahr
DA4F  E5                1359         PUSH    HL         ; else type all char in this
                            sector
DA50  CD0105            1360         CALL    PREREB     ; read in next sector
DA53  E1                1361         POP     HL
DA54  201A              1362         JR      NZ,RDTYER  ;>>EXIT# READ ERR,goto check
                            why?
DA56  AF                1363         XOR     A
DA57  77                1364         LD      (HL),A     ;Init offset to 00 for incre
                            to FF
                        1365 ;
                        1366 ; TYENXT: TYPE NEXT CHAR
                        1367 ;    ENTRY: after READ A SECTOR or in loop type
                        1368 ;   ACTION: increment offset to 00 (first char) or to n
                            ext char
                        1369 ;           if char is Logical EOF (1AH)
                        1370 ;             select back to OLD CUR DRV and restart R
                            EA
                        1371 ;           if not EOF, write char on console
                        1372 ;     EXIT: goto get CON STA
                        1373 ;
DA58  34                1374 TYENXT: INC     (HL)       ;Incre offset to next char
DA59  210000            1375         LD      HL,0080H ;Begin at DMA
DA5C  CD2DD7            1376         CALL    MOVSCN     ;Get char at (DMA+offset) in
                            A
DA5F  7E                1377         LD      A,(HL)
DA60  FE1A              1378         CP      1AH        ;Check if Logical EOF
DA62  CA7ADC            1379         JP      Z,OLDDRV   ;>>EXIT# EOF,select back to O
                            LD CUR DRV
```

```
DA65  CD99D5      1380          CALL    CONOUT    ;  else write a char on conso
                      le
                  1381 ;
                  1382 ; AFTER WRITE EACH CHAR ON CONSOLE WILL GET CONSOLE STA
                      TUS
                  1383 ; IF KEY PRESS, TYPE ABORT, SELECT BACK TO OLD CURRENT
                  1384 ; DRIVE AND RESTART RPA. IF NO KEY PRESS LOOP FOR NEXT
                      CHAR.
                  1385 ;
DA68  CDC0D6      1386          CALL    CONSTA    ;Get CON STA
DA6B  C27ADC      1387          JP      NZ,OLDDRV ;>>EXIT# Key press, TYPE abor
                      t
DA6E  18D7        1388          JR      LOPTYE    ;No key press, loop for next
                      char
                  1389 ;
                  1390 ; RDTYER: READ FOR TYPE ERR
                  1391 ;    ENTRY: READ SEQ error
                  1392 ;   ACTION: check if ERROR CODE 01 =.Physical EOF
                  1393 ;           select back to OLD CURRENT DRIVE
                  1394 ;           check if assign more arg,write that arg err
                  1395 ;              then restart RPA
                  1396 ;
DA70  3D          1397 RDTYER: DEC     A
DA71  CA7ADC      1398          JP      Z,OLDDRV  ;>>EXIT#EOF,restart RPA
                  1399 ;
                  1400 ;    ENTRY: READ SEQ error and not EOF
                  1401 ;   ACTION: write STRING 'READ ERROR'
                  1402 ;
DA74  CD8AD8      1403          CALL    REDMSG    ;Write 'READ ERROR'
                  1404 ;
                  1405 ; ARGTYE: ARGUMENT TO TYPE ERROR
                  1406 ;    ENTRY: assign arg (FILENAME) to type NOT FOUND
                  1407 ;   ACTION: logon OLD CURRENT DRIVE and write arg err
                  1408 ;     EXIT: restart RPA
                  1409 ;
DA77  CD41D9      1410 ARGTYE: CALL    LOGOLD
DA7A  C3F8FC      1411          JP      FNOFND
                  1412 ;;;-----------------------------------------------
                  1413 ;          RESIDENT COMMAND SAVE                :
                  1414 ;;;-----------------------------------------------
DA7D  3E53        1415 SAVE:    LD      A,'S'
DA7F  32F8DC      1416          LD      (FLAG),A  ;Set FLAG for print usage
DA82  CDC3D8      1417          CALL    CALVAL    ;Convert PARAMETER to HEX va
                      l
DA85  F5          1418          PUSH    AF        ;Save (# of page) in A
DA86  CD32D7      1419          CALL    INTERP    ;Move FILENAME to save in TM
                      PFCB
DA89  C200FD      1420          JP      NZ,ARGERR ;Has any '?', write 'ARGUMEN
                      T ERROR'
DA8C  CD2FD9      1421          CALL    LOGSEL    ;Logon SEL DRIVE assign in F
                      ILENAME
DA8F  11D3DC      1422          LD      DE,TMPFCB
DA92  D5          1423          PUSH    DE        ;Save add of TMPFCB
DA93  CDF3D5      1424          CALL    DELFL     ;Delete OLD file (if has)
DA96  D1          1425          POP     DE        ;Get add of TMPFCB
DA97  CD0AD6      1426          CALL    MAKEFL    ;Make NEW file
DA9A  282F        1427          JR      Z,DSKFUL  ;>>EXIT# DISK FULL,write 'NO
```

```
CROMEMCO CDOS Z80 ASSEMBLER version 02.15                PAGE 0030

                        SPACE'
DA9C  AF        1428            XOR     A
DA9D  32F3DC    1429            LD      (TCREC),A  ;Clear CURRENT RECORD to Zer
                o
DAA0  F1        1430            POP     AF         ;Get (# of page)
DAA1  6F        1431            LD      L,A
DAA2  2600      1432            LD      H,00H      ;Change (# of page) to (# of
                sector)
DAA4  29        1433            ADD     HL,HL      ;  by (# of page )*2
DAA5  110001    1434            LD      DE,0100H   ;Init DMA at TPA
                1435 ;
                1436 ; LOPSAV: LOOP FOR SAVE
                1437 ;    ENTRY: MAKE new FILE and set CURRENT RECORD to Zer
                o
                1438 ;           init DMA at TPA in DE, (# of sector) in HL
                1439 ;   ACTION: save first 128 bytes from TPA in record 0
                1440 ;           then save next 128 bytes to next record by
                1441 ;           adjust DMA add and decrement (# of record)
                1442 ;           by 1, loop until (# of record) decrease to
                Zero
                1443 ;     EXIT: if err (DISK FULL), restart RPA
                1444 ;             else goto SAVE END
                1445 ;
DAA8  7C        1446 LOPSAV: LD     A,H
DAA9  B5        1447         OR     L
DAAA  2816      1448         JR     Z,SAVEND   ;>>EXIT# All sector are saved
DAAC  2B        1449         DEC    HL         ;Decrement (# of sector)
DAAD  E5        1450         PUSH   HL         ;  and save
DAAE  210080    1451         LD     HL,0080H
DAB1  19        1452         ADD    HL,DE      ;Adjust DMA to next 128 bytes
DAB2  E5        1453         PUSH   HL         ;Save new DMA (POP to next us
                e in DE)
DAB3  CDD6D6    1454         CALL   SETDMA     ;Set DMA add in DE
DAB6  11D3DC    1455         LD     DE,TMPFCB  ;Write 128 bytes from current
                DMA
DAB9  CDD6D6    1456         CALL   WRTSEQ     ;  into file assign in TMPFCB
DABC  D1        1457         POP    DE         ;POP DMA (from PUSH HL)
DABD  E1        1458         POP    HL         ;POP (# of sector)
DABE  2009      1459         JR     NZ,DSKFUL  ;>>EXIT# DISK FULL,write 'NO
                SPACE'
DAC0  18E6      1460         JR     LOPSAV     ;Goon loop save
                1461 ;
                1462 ; SAVEND: SAVE END
                1463 ;    ENTRY: loop save finish
                1464 ;   ACTION: CLOSE FILE to update DIRECTORY FCB
                1465 ;           if CLOSE ERR (NOT FOUND),write 'NO SPACE'
                1466 ;     EXIT: goto RETDMA
                1467 ;
DAC2  11D3DC    1468 SAVEND: LD     DE,TMPFCB
DAC5  CDE2D5    1469         CALL   CLOSFL     ;Close file at TMPFCB
DAC8  3C        1470         INC    A
DAC9  2009      1471         JR     NZ,RETDMA  ;>>EXIT# No ERR,ret DMA add
                1472 ;
                1473 ; DSKFUL: RTN TO WRITE 'NO SPACE'
                1474 ;    ENTRY: MAKE FILE and DIRECTORY FULL
                1475 ;           CLOSE FILE and FILE NOT FOUND
                1476 ;   ACTION: write message 'NO SPACE'
```

```
                        1477 ;     EXIT: goto RETDMA
                        1478 ;
DACB  01DADA            1479 DSKFUL: LD      BC,NOSPAC  ;BC points STRING
DACE  CDB4D5            1480         CALL    WRTSTR     ;Write STRING 'NO SPACE'
DAD1  CDA7D5            1481         CALL    CRLF
                        1482 ;
                        1483 ; RETDMA: RETURN DMA add TO SYSTEM DMA (80H)
                        1484 ;    ENTRY: end of SAVE command
                        1485 ;   ACTION: set DMA back to 80H
                        1486 ;           select back to OLD CURRENT DRIVE
                        1487 ;           if assign more arg,write arg err
                        1488 ;     EXIT: restart RPA
                        1489 ;
DAD4  CDD3D6            1490 RETDMA: CALL    DMA80H   ;Set DMA add at 80H
DAD7  C37ADC            1491         JP      OLDDRV   ;Select back OLD CUR DRV
                        1492 ;
                        1493 ;;; NOSPAC: 'NO SPACE' MESSAGE ----------------------
                             --------
DADA  44697368          1494 NOSPAC: DB      'Disk Full',NULL
      2046756C
      6C00
                        1495 ;;;-----------------------------------------
                        1496 ;          RESIDENT COMMAND REN           :
                        1497 ;;;-----------------------------------------
DAE4  CD32D7            1498 REN:    CALL    INTERP     ;Move ARG1 into TMPFCB
DAE7  C200FD            1499         JP      NZ,ASUERR  ;Has any '?',write 'ARGUM
                          ENT ERROR'
DAEA  21D4DC            1500         LD      HL,TFN1
DAED  7E                1501         LD      A,(HL)     ;Get first char of FILENA
                          ME
DAEE  FE20              1502         CP      ' '        ;  check if blank
DAF0  CAC9FD            1503         JP      Z,PRENUS   ;   goto print usage
DAF3  3AF6DC            1504         LD      A,(SELNUM)
DAF6  F5                1505         PUSH    AF         ;Save SELECT NUM of ARG1
DAF7  CD2FD9            1506         CALL    LOGSEL     ;Logon SEL DRV assign in
                          ARG1
DAFA  CDEED5            1507         CALL    PRESF      ;SEARCH FIRST for file AR
                          G1
DAFD  2050              1508         JR      NZ,REEXIS  ;>>EXIT# File FOUND,write
                          'FILE EXISTS'
DAFF  21D3DC            1509         LD      HL,TMPFCB  ;Move ARG1 (NEWNAME) from
                          TMPFCB
DB02  11E3DC            1510         LD      DE,TMP2    ;  to TMP2
DB05  0610              1511         LD      B,10H      ;    length 16 bytes
DB07  CD1FD9            1512         CALL    MOVER      ;      HL->DE
DB0A  ED5B88D5          1513         LD      DE,(SCNBEG) ;Get add of next SCAN
DB0E  CD24D7            1514         CALL    CUTBLK     ;Cut blank to find first
                          char
DB11  FE3D              1515         CP      '='
DB13  2804              1516         JR      Z,ASSGOK   ;ASSIGN SIGN must be '='
DB15  FE5F              1517         CP      '_'        ;   or '_' (Underline)
DB17  2030              1518         JR      NZ,RENERR  ; else ARG2 err
                        1519 ;
                        1520 ; ASSGOK: ASSIGN SIGN OK
                        1521 ;    ENTRY: ARG1 in TMP2; DE points start of ARG2
                        1522 ;   ACTION: fill ARG2 (OLD NAME) into TMPFCB
                        1523 ;           if SEL DRIVE of ARG2 is default; OK
```

```
                        1524 ;          but if SEL DRIVE of 2 arg not match
                        1525 ;            default ARG2 err and use SEL DRIVE of ARG
                             1
                        1526 ;            to use in LOGOLD
                        1527 ;     EXIT: goto RENMAT or write ARG2 err
                        1528 ;
DB19  13                1529 ASSGOK: INC     DE          ;Get add of first char of A
                             RG2
DB1A  ED538BD5          1530         LD      (SCNBEG),DE ;  is the start add to SCAN
DB1E  CD32D7            1531         CALL    INTERP      ;Move ARG2 into TMPFCB
DB21  2026              1532         JR      NZ,RENERR   ;Has any '?', write arg err
DB23  F1                1533         POP     AF          ;Get SEL NUM of ARG1
DB24  47                1534         LD      B,A         ;   save in B to compare
DB25  21F6DC            1535         LD      HL,SELNUM
DB28  7E                1536         LD      A,(HL)      ;Get SEL NUM of ARG2
DB29  B7                1537         OR      A
DB2A  2804              1538         JR      Z,RENMAT    ;SEL DRIVE of ARG2 is defau
                             lt,OK
DB2C  B8                1539         CP      B           ;Not default,compare with S
                             EL DRIVE of ARG1
DB2D  70                1540         LD      (HL),B      ;Save SEL NUM of ARG1 (use
                             in LOGOLD)
DB2E  2019              1541         JR      NZ,RENERR   ;Assign DISK ID not match,d
                             efault ARG2 err
                        1542 ;
                        1543 ; RENMAT: RENAME ASSIGN DISK ID MATCH
                        1544 ;    ENTRY: assign DISK ID of 2 arg are match
                        1545 ;   ACTION: save SELECT NUM of ARG1 in SELNUM
                        1546 ;           (to use in LOGOLD if any step error)
                        1547 ;           search OLD NAME if NOT FOUND write 'NO FILE
                             '
                        1548 ;             else RENAME
                        1549 ;           then select back to OLD CURRENT DRIVE
                        1550 ;           if assign more arg,write arg err
                        1551 ;     EXIT: restart RPA
                        1552 ;
DB30  70                1553 RENMAT: LD      (HL),B      ;Save SEL NUM of ARG1 (for A
                             RG2 default)
DB31  AF                1554         XOR     A
DB32  32D3DC            1555         LD      (TDRI),A    ;Clear DR because SEL DRIVE
                             is now logon
DB35  CDEED5            1556         CALL    PRESF       ;SEARCH FOR FIRST of OLDNAME
DB38  2809              1557         JR      Z,RENNOF    ;OLD NAME NOT FOUND, write '
                             NO FILE'
DB3A  11D3DC            1558         LD      DE,TMPFCB   ;OLD NAME in TFCB1
DB3D  CD0ED6            1559         CALL    RENAME      ;  NEW NAME in TFCB2, RENAME
DB40  C37ADC            1560         JP      OLDDRV      ;>>EXIT# Select back to OLD
                             CUR DRV
                        1561 ;
                        1562 ;;; RENNOF: RTN TO WRITE 'FILE NOT FOUND' IN RENAME
DB43  CD41D9            1563 RENNOF: CALL    LOGOLD  ;Log back OLD CUR DRV
DB46  C3F8FC            1564         JP      FNDFND  ;  and print 'FILE NOT FOUND'
                        1565 ;
                        1566 ;;; RENERR: RTN TO WRITE ARG2 ERROR IN RENAME
DB49  CD41D9            1567 RENERR: CALL    LOGOLD  ;Logon OLD CUR DRV
DB4C  C300FD            1568         JP      AGUERR  ;Write ARG2 err
                        1569 ;
```

```
                        1578 ;;; REEXIS: RTN TO WRITE 'FILE EXISTS' IN RENAME
DB4F  CD41D9            1571 REEXIS: CALL    LOGOLD  ;Log back OLD CUR DRV
DB52  C3FCFC            1572         JP      FEXIST  ;  and print 'FILE EXISTS'
                        1573 ;;;-----------------------------------------------
                        1574 ;          RESIDENT COMMAND USER                :
                        1575 ;;;-----------------------------------------------
DB55  3E55              1576 USER:   LD      A,'U'
DB57  32F8DC            1577         LD      (FLAG),A  ;Set FLAG for print usage
DB5A  CDC3D8            1578         CALL    CALVAL    ;Convert PARAMETER to HEX va
                             l
DB5D  FE10              1579         CP      10H       ;USER CODE not in range 0-F
DB5F  D204FD            1580         JP      NC,NINRNG ;  write 'NOT IN RANGE'
DB62  5F                1581         LD      E,A       ;NEW USER CODE in E
DB63  3AD4DC            1582         LD      A,(TFN1)
DB66  FE20              1583         CP      ' '       ;First digit is blank
DB68  CA00FD            1584         JP      Z,ARGUERR ;  write 'ARGUMENT ERROR'
DB6B  CD15D6            1585         CALL    SETUSR    ;Set NEW USER AREA
DB6E  C37DDC            1586         JP      BAKRPA    ;Restart RPA
                        1587 ;;;-----------------------------------------------
                        1588 ;          RESIDENT COMMAND MOVE.               :
                        1589 ;;;-----------------------------------------------
DB71  3E4D              1590 MOVE:   LD      A,'M'
DB73  32F8DC            1591         LD      (FLAG),A  ;Set FLAG for print usage
DB76  CDC3D8            1592         CALL    CALVAL    ;Convert PARAMETER to HEX va
                             l
DB79  FE10              1593         CP      10H       ;If partition# not in range
                             0-F
DB7B  D204FD            1594         JP      NC,NINRNG ;   goto print 'NOT IN RANGE
                             '
DB7E  32F8DC            1595         LD      (FLAG),A  ; else save partition#
DB81  CD32D7            1596         CALL    INTERP    ;Move FILE to transfer in TM
                             PFCB
DB84  CD2FD9            1597         CALL    LOGSEL    ;Logon SEL DRIVE assign in a
                             rg
DB87  3AF8DC            1598         LD      A,(FLAG)
DB8A  47                1599         LD      B,A       ;EXTRA SENT = PARTITION#
DB8B  11D3DC            1600         LD      DE,TMPFCB
DB8E  CD1AD6            1601         CALL    TRNFER    ;Transfer file assign in TMP
                             FCB
DB91  3C                1602         INC     A         ;Test return code
DB92  CA9BD8            1603         JP      Z,ALLSOM  ; NOT FOUND, goto write mess
                             age
DB95  C37ADC            1604         JP      OLDDRV    ; FOUND, log back DRV
                        1605 ;;;-----------------------------------------------
                        1606 ;          LOAD TRANSIENT FILE                  :
                        1607 ; JUMP FROM VECTOR OR ASSIGN DISK ID IN COMMAND :
                        1608 ;;;-----------------------------------------------
DB98  CDF3D6            1609 TRANSI: CALL    CHKPAT    ;Check for compat with DOS
DB9B  3AD4DC            1610         LD      A,(TFN1)
DB9E  FE20              1611         CP      ' '
DBA0  2020              1612         JR      NZ,LOADFL ;FILENAME assign,load 'COM'
                             file
                        1613 ;
                        1614 ; DISK ID. IS DEFAULT MEANS ONLY PRESS RETURN ( EX: A>
                             <RET> )
                        1615 ;
DBA2  3AF6DC            1616         LD      A,(SELNUM) ;Get assign DISK ID
```

```
DBA5  B7        1617        OR      A
DBA6  20 0D     1618        JR      NZ,INDID  ;SEL NUM not default
DBA8  3E 1B     1619        LD      A,ESC
DBAA  CD 9F D5  1620        CALL    WRTCHR
DBAD  3E 4A     1621        LD      A,'J'
DBAF  CD 9F D5  1622        CALL    WRTCHR    ; else clear screen
DBB2  C3 7D DC  1623        JP      BAKRPA    ;   and restart RPA.
                1624 ;
                1625 ; TRANSIENT WITH NOT ASSIGN FILENAME MEANS WANT TO LOGO
                    N
                1626 ; ANOTHER DRIVE ( EX: A>B: )
                1627 ;
DBB5  3D        1628 INDID:  DEC     A          ;Decrement SEL NUM to DRIVE
                    CODE
DBB6  32 F5 DC  1629         LD      (CURDRV),A ;  use as CURRENT DRIVE of R
                    PA
DBB9  CD 2E D6  1630         CALL    SCDISK     ;  and save in CDISK of SYST
                    EM
DBBC  CD C7 D5  1631         CALL    SELDRV     ;  then logon this drive to
                active
DBBF  C3 7D DC  1632         JP      BAKRPA     ;  restart RPA
                1633 ;
                1634 ; LOADFL: LOAD 'COM' FILE
                1635 ;    ENTRY: assign FILENAME in TMPFCB
                1636 ;           assign DISK ID in SELNUM
                1637 ;   ACTION: check if assign FILETYPE,write arg err
                1638 ;           logon SELECT DRIVE
                1639 ;           move 'COM' to default FILETYPE
                1640 ;           OPEN FILE on SELECT DRIVE and set DMA at TP
                    A
                1641 ;
DBC2  11 DC DC  1642 LOADFL: LD      DE,TFTI    ;Get first char of FILETYPE
DBC5  1A        1643         LD      A,(DE)
DBC6  FE 20     1644         CP      ' '        ;Compare with blank
DBC8  C2 F3 FC  1645         JP      NZ,COMERR  ;Assign FILETYPE too,write '
                    COMMAND ERROR'
DBCB  D5        1646         PUSH    DE         ;Save add of FILETYPE
DBCC  CD 2F D9  1647         CALL    LOGSEL     ;Logon SEL DRIVE assign in C
                    OMMAND
DBCF  D1        1648         POP     DE         ;Get add of FILETYPE
DBD0  21 77 DC  1649         LD      HL,COM     ;HL points STRING 'COM'
DBD3  CD 1D D9  1650         CALL    MVER38     ; move 'COM' to FILETYPE
DBD6  CD D9 D5  1651         CALL    PREOPN     ;Open 'COM' file at TMPFCB
DBD9  CA 83 DC  1652         JP      Z,NOEXTR   ;NOT FOUND,write 'COMMAND NO
                    T FOUND'
DBDC  21 00 01  1653         LD      HL,0100H   ;Init DMA start at TPA
                1654 ;
                1655 ; LOPLOD: LOOP LOAD 'COM' FILE
                1656 ;    ENTRY: open FILENAME type 'COM' at TMPFCB
                1657 ;           DMA start at TPA
                1658 ;   ACTION: READ SEQ a sector into DMA
                1659 ;           adjust DMA to next 128 bytes and check
                1660 ;           if new DMA over laid RPA,write 'BAD LOAD'
                1661 ;             else load next sector
                1662 ;     EXIT: loop until EOF or READ ERR
                1663 ;
DBDF  E5        1664 LOPLOD: PUSH    HL         ;Save DMA add
```

```
DBE0  EB          1665        EX      DE,HL
DBE1  CDD6D6      1666        CALL    SETDMA    ;Set DMA
DBE4  11D3DC      1667        LD      DE,TMPFCB
DBE7  CDFDD5      1668        CALL    REDSEQ    ;Read 'COM' file
DBEA  20I0        1669        JR      NZ,TRNEOF ;>>EXIT# READ ERR,check whic
                      h?
DBEC  E1          1670        POP     HL        ;Set OLD DMA add
DBED  I18000      1671        LD      DE,0080H
DBF0  19          1672        ADD     HL,DE     ;Adjust DMA to next 128 byte
                      s in HL
DBF1  1100D5      1673        LD      DE,RPA    ;Compare NEW DMA with RPA ad
                      d in DE
DBF4  7D          1674        LD      A,L
DBF5  93          1675        SUB     E
DBF6  7C          1676        LD      A,H
DBF7  9A          1677        SBC     A,D       ;  NEW DMA : RPA  (or HL-DE
                      )
DBF8  306C        1678        JR      NC,LDTRER ;NEW DMA laid over RPA,write
                       'BAD LOAD'
DBFA  18E3        1679        JR      LOPLOD    ;NEW DMA OK,loop load next s
                      ector
                  1680 ;
                  1681 ; TRNEOF: TRANSIENT END OF FILE
                  1682 ;    ENTRY: read 'COM' file err
                  1683 ;   ACTION: check ERROR CODE if 01=EOF
                  1684 ;             then move ARG1 into TMPFCB with SEL NUM
                  1685 ;             and move ARG2 into TMP2 with SEL NUM
                  1686 ;           clear CURRENT RECORD (byte #33 of TMPFCB)
                  1687 ;           Finally move TMPFCB 33 bytes to SYSTEM FCB
                      at 5CH
                  1688 ;     EXIT: goto ENDTAL
                  1689 ;
DBFC  E1          1690 TRNEOF: POP     HL        ;POP DMA in HL to clear STAC
                      K
DBFD  3D          1691        DEC     A
DBFE  2066        1692        JR      NZ,LDTRER ;>>EXIT# NOT EOF,write 'BAD
                       LOAD'
DC00  CD4ID7      1693        CALL    LOGOLD    ;EOF,log back to OLD CUR DRV
DC03  CD32D7      1694        CALL    INTERP    ;Move ARG1 into TMPFCB1
DC06  21F6DC      1695        LD      HL,SELNUM
DC09  E5          1696        PUSH    HL        ;PUSH add of SELNUM
DC0A  7E          1697        LD      A,(HL)
DC0B  32D3DC      1698        LD      (TDR1),A  ;Put SELNUM of ARG1 in DR of
                      ARG1
DC0E  3E10        1699        LD      A,10H     ;Set offset to TMP2
DC10  CD34D7      1700        CALL    INFCB2    ;Move ARG2 into TMPFCB2
DC13  E1          1701        POP     HL        ;POP add of SELNUM
DC14  7E          1702        LD      A,(HL)
DC15  32E3DC      1703        LD      (TDR2),A  ;Put SELNUM of ARG2 in DR of
                      ARG2
DC18  AF          1704        XOR     A
DC19  32F3DC      1705        LD      (TCRED),A ;Clear CURRENT RECORD to Zer
                      o
DC1C  115C00      1706        LD      DE,SYSFCB ;Move ARG1&ARG2 in TMPFCB
DC1F  21D3DC      1707        LD      HL,TMPFCB ;   to SYSTEM FCB (5CH)
DC22  0621        1708        LD      B,21H     ;     length 33 bytes
DC24  CD1FD9      1709        CALL    MOVER     ;       HL->DE
```

```
DC27  210805      1710        LD      HL,COMBUF  ;HL points COMMAND BUFFER
                  1711 ;
                  1712 ; FNDTAL: TO FIND COMMAND TAIL
                  1713 ;    ENTRY: POINTER in HL points begin of COMMANF BUFFE
                      R
                  1714 ;   ACTION: move POINTER pass char in COMMAND until
                  1715 ;           found NULL (at end of COMMAND LINE)
                  1716 ;           or found BLANK (end of COMMAND but not
                  1717 ;           end of LINE)
                  1718 ;     EXIT: POINTER in HL points next after COMMAND
                  1719 ;
DC2A  7E          1720 FNDTAL: LD      A,(HL)     ;Get char
DC2B  B7          1721        OR      A
DC2C  2807        1722        JR      Z,INITAL   ;>>EXIT# Found NULL,no arg
DC2E  FE20        1723        CP      ' '
DC30  2803        1724        JR      Z,INITAL   ;>>EXIT# Found BLANK,assign
                      arg
DC32  23          1725        INC     HL         ;POINTER points next
DC33  18F5        1726        JR      FNDTAL     ;Loop move POINTER
                  1727 ;
                  1728 ; INITAL: INITIALIZE COMMAND TAIL
                  1729 ;    ENTRY: HL points next after COMMAND in COMMAND BUF
                      FER
                  1730 ;   ACTION: move COMMAND TAIL from COM BUF to 81H
                  1731 ;           and count length of TAIL in B
                  1732 ;     EXIT: length of COMMAND TAIL in B
                  1733 ;
DC35  0600        1734 INITAL: LD      B,00H      ;B count length of COM TAIL
DC37  118100      1735        LD      DE,CMDTAL  ;DE points add 81H
DC3A  7E          1736 LOPTAL: LD      A,(HL)     ;HL points COM TAIL in COM B
                      UF
DC3B  12          1737        LD      (DE),A     ;Move COMMAND TAIL from COM
                      BUF
DC3C  B7          1738        OR      A          ;  to 81H
DC3D  2805        1739        JR      Z,GOTRAN   ;    until end of COM LINE
DC3F  04          1740        INC     B          ;Count length of COM LINE
DC40  23          1741        INC     HL
DC41  13          1742        INC     DE
DC42  18F6        1743        JR      LOPTAL     ;Loop move COMMAND TAIL
                  1744 ;
                  1745 ; GOTRAN: GO TO EXEC TRANSIENT
                  1746 ;    ENTRY: ARG1&ARG2 in SYSTEM FCB
                  1747 ;           COMMAND TAIL in 81H
                  1748 ;           length of COMMAND TAIL in B
                  1749 ;   ACTION: save length of COM TAIL AT 80H
                  1750 ;           set DMA at 80H (SYSTEM DMA)
                  1751 ;           set USER & CURRENT DRIVE CODE at CDI3K
                  1752 ;             then jump to EXEC at TPA
                  1753 ;     EXIT: exit to TPA by 'CALL',so from 'COM' file
                  1754 ;           can jump back to RPA by 'RET'. RPA will
                  1755 ;           set up STACK, logon OLD CURRENT DRIVE used
                  1756 ;           before exec 'COM' file and restart RPA
                  1757 ;
DC44  78          1758 GOTRAN: LD      A,B        ;Get length of COMMAND TAIL
DC45  328000      1759        LD      (SYSDMA),A ;  and save at 80H
DC48  CDA705      1760        CALL    CRLF       ;Get new line
DC4B  CDD3D6      1761        CALL    DMA80H     ;Set DMA at 80H
```

```
DC4E CD1FD6      1762        CALL    USRDSK      ;Save USER & DRIVE CODE at
                     CDISK
DC51 CD0001      1763        CALL    TPA       ;>>>EXIT# Jump to TPA by CALL
                 1764 ;
                 1765 ; COME BACK TO THIS POINT FROM TRANSIENT BY 'RET' IN 'C
                     OM' FILE
                 1766 ;
DC54 313FDC      1767        LD      SP,STACK    ;Set up STACK
DC57 CD2ED6      1768        CALL    SCDISK      ;Save OLD CUR DRV of RPA at
                      CDISK
DC5A CDC7D5      1769        CALL    SELDRV      ;  and logon back to this d
                     rive
DC5D C32900      1770        JP      RSTRPA      ;Restart RPA
                 1771 ;
                 1772 ; NOEXTR: NO EXIST TRANSIENT FILE
                 1773 ;    ENTRY: TRANSIENT 'COM' FILE, NOT FOUND
                 1774 ;   ACTION: logon OLD CURRENT DRIVE and write arg err
                 1775 ;     EXIT: restart RPA
                 1776 ;
DC60 CD41D9      1777 NOEXTR: CALL    LOGOLD
DC63 C3F4FC      1778        JP      CNOFND
                 1779 ;
                 1780 ; LDTRER: LOAD TRANSIENT FILE ERROR
                 1781 ;    ENTRY: read TRANS 'COM' FILE error
                 1782 ;           or load over laid RPA
                 1783 ;   ACTION: write STRING 'BAD LOAD'
                 1784 ;           select back to OLD CURRENT DRIVE
                 1785 ;           write arg err if assign more arg
                 1786 ;     EXIT: restart RPA
                 1787 ;
DC66 016EDC      1788 LDTRER: LD      BC,BADLOD   ;BC points STRING
DC69 CDB4D5      1789        CALL    WRTSTR      ;Write STRING 'BAD LOAD'
DC6C 1800        1790        JR      OLDDRV      ;Select OLD CUR DRV
                 1791 ;
                 1792 ;;; BADLOD: 'BAD LOAD' MESSAGE ----------------------
                      --------
DC6E 42616420    1793 BADLOD: DB      'Bad Load',NULL
     4C6F6164
     00
                 1794 ;
                 1795 ;;;; COM: 'COM' DEFAULT FILE TYPE --------------------
                      ---------
DC77 434F4D      1796 COM:    DB      'COM'
                 1797 ;;;-----------------------------------------------
                 1798 ; OLDDRV: SELECT BACK TO OLD CURRENT DRIVE      :
                 1799 ;    ENTRY: when end of each COMMAND          :
                 1800 ;   ACTION: logon OLD CURRENT DRIVE           :
                 1801 ;           (DRIVE which logon before exec COMMAND):
                 1802 ;     EXIT: do BAKRPA                        :
                 1803 ;;;-----------------------------------------------
DC7A CD41D9      1804 OLDDRV: CALL    LOGOLD
                 1805 ;;;-----------------------------------------------
                 1806 ; BAKRPA: BACK TO RPA                         :
                 1807 ;    ENTRY: when end of each COMMAND with no  :
                 1808 ;           user select drive                 :
                 1809 ;   ACTION: check if COMMAND LINE is END      :
                 1810 ;           by move arg into TMPFCB           :
```

```
                        1811 ;         Not assign DISK ID (SELNUM=00,default) ;
                        1812 ;         and Not assign FILENAME (FN=blank)     ;
                        1813 ;             else write tha EXCEED arg err      ;
                        1814 ;     EXIT: restart RPA
                        1815 ;;;-----------------------------------------------
DC7D  CD32D7            1816 BAKRPA: CALL    INTERP    ;Move arg into TMPFCB
DC80  3AD4DC            1817         LD      A,(TFN1)  ;Get first char of FILENAME
DC83  D620              1818         SUB     ' '       ;  check if BLANK
DC85  21F6DC            1819         LD      HL,SELNUM ;Get SELECT NUM
DC88  B6                1820         OR      (HL)      ;  check if NOT assign
                        1821 ;
                        1822 ; IF ASSIGN DISK ID. AND/OR FILENAME CONDITION WILL NOT
                             ZERO
                        1823 ;
DC89  C203FD            1824         JP      NZ,ARGUEXC ;Not Zero,write 'ARGUMENT EXC
                             EED'
DC8C  C32903            1825         JP      RSTRPA    ;COM LINE end,restart RPA
                        1826 ;;;-----------------------------------------------
                        1827 ;           W O R K   A R E A                   ;
                        1828 ;;;-----------------------------------------------
DC8F  (0020)            1829         DS      32        ;STACK AREA 16 ENTRY
      (DCAF)            1830 STACK   EQU     $         ;BEGIN OF STACK
DCAF  0000              1831         DW      0000H     ;BUFFER FOR STACK UNDERFLOW
                        1832 ;
DCB1  00                1833 BATFLG: DB      00H
                        1834 ;
                        1835 ;;;========== B A T C H   F. C. B. ==============
      (DCB2)            1836 BATFCB  EQU     $                 ;BATCH FCB
DCB2  00                1837 BDR:    DB      00H               ;DRIVE NUMBER
DCB3  24242420          1838 BFN:    DB      '$$$      '       ;FILE NAME
      20202020
DCBB  535542            1839 BFT:    DB      'SUB'             ;FILE TYPE
DCBE  00                1840 BEXNUM: DB      00H               ;EXTENT NUMBER
DCBF  00                1841 BS1:    DB      00H               ;RESERVED S1
DCC0  00                1842 BS2:    DB      00H               ;RESERVED S2
DCC1  00                1843 BRECCT: DB      00H               ;RECORD COUNT
DCC2  (0010)            1844 BDM:    DB      16                ;DISK MAP
DCD2  00                1845 BCREC:  DB      00H               ;CURRENT RECORD
                        1846 ;
                        1847 ;;;========== T E M P   F. C. B. ================
      (DCD3)            1848 TMPFCB  EQU     $         ;TMP FCB 1
DCD3  00                1849 TDR1:   DB      00H       ;DRIVE NUMBER
DCD4  (0008)            1850 TFN1:   DS      8         ;FILE NAME
DCDC  (0003)            1851 TFT1:   DS      3         ;FILE TYPE
DCDF  00                1852 EXNUM1: DB      00H       ;EXTENT NUMBER
DCE0  00                1853 S1T1:   DB      00H       ;RESERVED S1
DCE1  00                1854 S2T1:   DB      00H       ;RESERVED S2
DCE2  00                1855 RECCT1: DB      00H       ;RECORD COUNT
                        1856 ;
      (DCE3)            1857 TMP2    EQU     $         ;TMP FCB 2
DCE3  00                1858 TDR2:   DB      00H       ;DRIVE NUMBER
DCE4  (0008)            1859 TFN2:   DS      8         ;FILE NAME
DCEC  (0003)            1860 TFT2:   DS      3         ;FILE TYPE
DCEF  00                1861 EXNUM2: DB      00H       ;EXTENT NUMBER
DCF0  00                1862 S1T2:   DB      00H       ;RESERVED S1
DCF1  00                1863 S2T2:   DB      00H       ;RESERVED S2
DCF2  00                1864 RECCT2: DB      00H       ;RECORD COUNT
```

```
CROMEMCO CDOS Z80 ASSEMBLER version 02.15                PAGE 0037


                        1365 ;
0DF3  00                1366 TCREC:  DB      00H     ;CURRENT RECORD (BYTE #33)
                        1367 ;
                        1368 ;;; ==========================================================
                             ===
0DF4  00                1369 DIRCOD: DB      00H     ;KEEP DIR CODE (0-3) OR FF
0DF5  00                1370 CURDRV: DB      00H     ;RFA CURRENT DRIVE (0=A)
0DF6  00                1371 SELNUM: DB      00H     ;SEL DRV NUM FROM ARG (0=defaul
                             t)
0DF7  00                1372 COUNTER: DB     00H     ;COUNTER USE IN TYPE COMMAND
0DF8  00                1373 FLAG:   DB      00H     ;TEMP AREA TO SAVE FLAG AND PAR
                             AMETER
      (0DF9)            1374 FINISH  EQU     $
0DF9  (0580)            1375         END     START

errors                  0
```

```
CROMEMCO CDOS Z80 ASSEMBLER Version 02.15                    PAGE 0001

                        0001 ;;;  EXTERNAL FROM EXTRA.  ::::::::::::::::::::::::
(FC00)                  0002 DRVNAM  EQU     0FC00H
(FC08)                  0003 DSKID   EQU     0FC08H
                        0004 ;;;  EXTERNAL FROM IO.  ::::::::::::::::::::::::::::
(EFA6)                  0005 DOSMSG  EQU     0EFA6H
                        0006 ;;;  EQUATES FOR SYSTEM ENTRY POINT  ::::::::::::::
(003E)                  0007 MSIZE   EQU     62              ;MEMORY SIZE OF SYSTEM
(0000)                  0008 WARMB   EQU     0000H           ;WARM BOOT ENTRY POINT
(0003)                  0009 IOBYTE  EQU     0003H           ;ADDRESS OF IOBYTE
(0080)                  0010 SYSDMA  EQU     0080H           ;ADDRESS OF DEFAULT SYS
                             TEM DMA
(9800)                  0011 BIAS    EQU     (MSIZE-20)*1024 ;MEMORY OFFSET FOR 20K
                             SYSTEM
(DE00)                  0012 RPA     EQU     2000H+BIAS      ;RESIDENT AREA
(DD00)                  0013 DOS     EQU     RPA+0800H       ;DOS AREA
(EB00)                  0014 IO      EQU     RPA+1600H       ;I/O AREA
                        0015 ;;;  ADDRESS OF I/O DRIVER ROUTINE  ::::::::::::::::
(EB03)                  0016 WBOOT   EQU     IO+03H  ;WARM BOOT ENTRY POINT
(EB06)                  0017 CONST   EQU     IO+06H  ;CONSOLE STATUS ROUTINE
(EB09)                  0018 CONIN   EQU     IO+09H  ;CONSOLE INPUT
(EB0C)                  0019 CONOUT  EQU     IO+0CH  ;CONSOLE OUTPUT
(EB0F)                  0020 LIST    EQU     IO+0FH  ;LIST DEVICE OUTPUT
(EB12)                  0021 PUNCH   EQU     IO+12H  ;PUNCH DEVICE OUTPUT
(EB15)                  0022 READER  EQU     IO+15H  ;READER DEVICE INPUT
(EB18)                  0023 HOME    EQU     IO+18H  ;HOME DRIVE
(EB1B)                  0024 SETDRV  EQU     IO+1BH  ;SELECT DISK
(EB1E)                  0025 SETTRK  EQU     IO+1EH  ;SET TRACK
(EB21)                  0026 SETSEC  EQU     IO+21H  ;SET SECTOR
(EB24)                  0027 SETDMA  EQU     IO+24H  ;SET DMA ADDRESS
(EB27)                  0028 READ    EQU     IO+27H  ;READ THE DISK
(EB2A)                  0029 WRITE   EQU     IO+2AH  ;WRITE THE DISK
(EB2D)                  0030 SECTRN  EQU     IO+2DH  ;SECTOR TRANSLATION
                        0031 ;;;  EQUATES FOR SYMBOL USED BY DOS  ::::::::::::::
(EB03)                  0032 FUNC0   EQU     WBOOT
(EB12)                  0033 FUNC4   EQU     PUNCH
(EB0F)                  0034 FUNC5   EQU     LIST
(0008)                  0035 BS      EQU     08H
(0009)                  0036 TAB     EQU     09H
(000A)                  0037 LF      EQU     0AH
(000D)                  0038 CR      EQU     0DH
(0027)                  0039 QUOTA   EQU     27H
(007F)                  0040 DEL     EQU     7FH
0000'                   0041         ORG     DOS             ;DOS STARTING ADDRESS
DD00  221800            0042 START:  DB      22H,18H,00H     ;FIXED ADDRESS 6 BYTES
DD03  00FE7F            0043         DB      00H,0FH,089H    ;  AT BEGINNING OF DOS.
                        0044 ;;;=================================================
                             ==
                        0045 ;;; DOS ENTRY POINT GEN BY ROUTINE GOSYS IN I/O.
                                =
                        0046 ;;;  FIRST PROCESS ROUTINE IN DOS. SELECT VECTOR OF DOS
                                =
                        0047 ;;;  FUNC# AND JUMP ON VECTOR WITH PARAMETER(S) IN
                                =
                        0048 ;;;  REG DE OR REG C.
                                =
                        0049 ;;;=================================================
                             ==
```

```
DD06 ED4363E0   0050        LD      (FNUM),BC       ;SAVE EXTRA SENT AND FU
                    NC#
DD0A ED5362E0   0051        LD      (SENT2B),DE     ;SAVE SENT VALUE 2 BYTE
                    S (REG D&E)
DD0E 7B         0052        LD      A,E
DD0F 3293EA     0053        LD      (SENT1B),A      ;SAVE SENT VALUE 1 BYTE
                     (REG E)
DD12 210000     0054        LD      HL,0000H
DD15 2264E0     0055        LD      (EXTVAL),HL     ;CLEAR EXIT VALUE
DD18 39         0056        ADD     HL,SP
DD19 2229E0     0057        LD      (OLDSP),HL      ;SAVE STACK POINTER OF
                    CALLING PROGRAM
DD1C 315DE0     0058        LD      SP,STACK        ;  AND SET UP STACK OF
                    DOS
DD1F AF         0059        XOR     A
DD20 3290EA     0060        LD      (SELNUM),A      ;SET SELECT NUM TO DEFA
                    ULT
DD23 3298EA     0061        LD      (DIRCOD),A      ;CLEAR DIRECTORY CODE
DD26 2144EA     0062        LD      HL,EXTRTN   -
DD29 E5         0063        PUSH    HL              ;PUSH ADDRESS OF EXIT R
                    OUTINE
DD2A 79         0064        LD      A,C             ;GET FUNC#
DD2B FE29       0065        CP      29H             ;COMPARE WITH 41
DD2D D0         0066        RET     NC              ;  FUNC# > 40... SUDDEN
                    LY EXIT
DD2E 48         0067        LD      C,E             ;(1)SENT VALUE 1 BYTE I
                    N C
DD2F 213FDD     0068        LD      HL,FNCTBL       ;(2)SET ADDRESS OF FUNC
                     VECTOR TABLE
DD32 5F         0069        LD      E,A
DD33 1600       0070        LD      D,00H           ;(3)DE KEEP FUNC#
DD35 19         0071        ADD     HL,DE
DD36 19         0072        ADD     HL,DE           ;(4)FROM ADDRESS OF VEC
                    TOR IN HL
DD37 5E         0073        LD      E,(HL)
DD38 23         0074        INC     HL
DD39 56         0075        LD      D,(HL)          ;(5)PICK UP VECTOR IN D
                    E
DD3A 2A62E0     0076        LD      HL,(SENT2B)
DD3D EB         0077        EX      DE,HL           ;(6)SENT VALUE 2 BYTES
                    IN DE
DD3E E9         0078        JP      (HL)            ;(7) JUMP ON VECTOR
                0079 ;;;=============================================
                    ==
                0080 ;;;  FUNCTION VECTOR TABLE (FUNC# 0 THRU 40).
                         =
                0081 ;;;=============================================
                    ==
DD3F 03E9       0082 FNCTBL: DW     FUNC0
DD41 E7DF       0083        DW      FUNC1
DD43 C7DE       0084        DW      FUNC2
DD45 ECDF       0085        DW      FUNC3
DD47 1353       0086        DW      FUNC4
DD49 0FE9       0087        DW      FUNC5
DD4B F1DF       0088        DW      FUNC6
DD4D 08E9       0089        DW      FUNC7
DD4F 03E9       0090        DW      FUNC8
```

```
DD51  12E0        0091        DW      FUNC9
DD53  14DF        0092        DW      FUNC10
DD55  18E0        0093        DW      FUNC11
DD57  26E9        0094        DW      FUNC12
DD59  28E9        0095        DW      FUNC13
DD5B  EFE8        0096        DW      FUNC14
DD5D  43E9        0097        DW      FUNC15
DD5F  4CE9        0098        DW      FUNC16
DD61  52E9        0099        DW      FUNC17
DD63  6EE9        0100        DW      FUNC18
DD65  72E9        0101        DW      FUNC19
DD67  86E9        0102        DW      FUNC20
DD69  8CE9        0103        DW      FUNC21
DD6B  92E9        0104        DW      FUNC22
DD6D  98E9        0105        DW      FUNC23
DD6F  A4E9        0106        DW      FUNC24
DD71  A9E9        0107        DW      FUNC25
DD73  AFE9        0108        DW      FUNC26
DD75  B6E9        0109        DW      FUNC27
DD77  2FE2        0110        DW      FUNC28
DD79  B8E9        0111        DW      FUNC29
DD7B  C0E9        0112        DW      FUNC30
DD7D  C9E9        0113        DW      FUNC31
DD7F  D3E9        0114        DW      FUNC32
DD81  E7E9        0115        DW      FUNC33
DD83  E9E9        0116        DW      FUNC34
DD85  EFE9        0117        DW      FUNC35
DD87  B3E3        0118        DW      FUNC36
DD89  F5E9        0119        DW      FUNC37
DD8B  16EA        0120        DW      FUNC38
DD8D  1EE3        0121        DW      FUNC39
DD8F  33EA        0122        DW      FUNC40
                  0123 ;;;=================================================
                       ==
                  0124 ;;;  PRINT DOS ERROR MESSAGE AND GET USER ACTION.
                           =
                  0125 ;;;    ALL ERROR GOTO WARM BOOT EXCEPT ERROR# 1.
                           =
                  0126 ;;;=================================================
                       ==
DD91  CDC3DD      0127 PINER1: CALL    PDOSER          ;PRINT 'SYSTEM MESSAGE
                       FROM '
DD94  01A3EA      0128         LD      BC,DERMS1
DD97  CD87DF      0129         CALL    LCPWRT          ;PRINT 'R/W DISK ERROR'
DD9A  CD2CDE      0130         CALL    GETCHR          ;GET USER ACTION
DD9D  FE03        0131         CP      03H             ;IF CTRL-C
DD9F  C0          0132         RET     NZ              ;NOT CTRL-C... RETURN
DDA0  CDFDDE      0133         CALL    CRLF
DDA3  C20000      0134         JP      WARMB           ;  ELSE GOTO WARM BOOT
DDA6  CDC3DD      0135 PINER2: CALL    PDOSER
DDA9  01C3EA      0136         LD      BC,DERM82       ;PRINT 'SELECT DRIVE ER
                       ROR'
DDAC  1811        0137         JR      GOWARM          ;  AND GOTO WARM BOOT
DDAE  CDC3DD      0138 PINER3: CALL    PDOSER
DDB1  CDF4DD      0139         CALL    PINFLN          ;PRINT 'FILENAME'
DDB4  01E2EA      0140         LD      BC,DERMS3       ;PRINT 'FILE R/O'
DDB7  1806        0141         JR      GOWARM          ;  AND GOTO WARM BOOT
```

```
DDB9  CDCB0D     0142 PINER4: CALL     PDOSER
DDBC  010CEA     0143         LD       BC,DERM64       ;PRINT 'DISK R/O'
DDBF  CD07DF     0144 GOWARM: CALL     LOPWRT          ;PRINT DOS ERROR MESSAG
                      E
DDC2  CD2CDE     0145         CALL     GETCHR          ;GET USER ACTION
DDC5  CDFDDE     0146         CALL     CRLF
DDC8  C30000     0147         JP       WARMB           ;  AND ALWAYS GOTO WARM
                       BOOT
                 0148 ;;-----------------------------------------------
                      --
                 0149 ;;  COMMON ROUTINE OF PRINT DOS ERROR MESSAGE.
                                  =
                 0150 ;;    PRINT 'SYSTEM MESSAGE FROM "XXXX.XX" '
                                  =
                 0151 ;;-----------------------------------------------
                      --
DDCB  CDFDDE     0152 PDOSER: CALL     CRLF            ;GET NEW LINE
DDCE  01A6EF     0153         LD       BC,DOSMSG
DDD1  CD07DF     0154         CALL     LOPWRT          ;PRINT 'SYSTEM MESSAGE
                      FROM '
DDD4  2100FC     0155         LD       HL,DRVNAM
DDD7  7E         0156 EDRIVE: LD       A,(HL)
DDD8  B7         0157         OR       A
DDD9  2809       0158         JR       Z,EDSKID
DDDB  E5         0159         PUSH     HL
DDDC  4F         0160         LD       C,A
DDDD  CDC7DE     0161         CALL     FUNC2           ;PRINT ASSIGN DRIVE NAM
                      E
DDE0  E1         0162         POP      HL
DDE1  23         0163         INC      HL
DDE2  18F3       0164         JR       EDRIVE          ;LOOP PRINT DRIVE NAME
DDE4  3A5FE0     0165 EDSKID: LD       A,(CDRIVE)      ;GET CDRIVE CODE
DDE7  2108FC     0166         LD       HL,DSKID        ;  AND ASSIGN DISK ID.
DDEA  86         0167         ADD      A,(HL)          ;   TO CHANGE TO ASSCII
DDEB  4F         0168         LD       C,A
DDEC  CDC7DE     0169         CALL     FUNC2           ;PRINT DISK ID.
DDEF  C9         0170         RET
                 0171 ;;-----------------------------------------------
                      --
                 0172 ;;   ROUTINE TO PRINT FILENAME IN FORM
                                  :
                 0173 ;;          ': FILE "XXXX.XXX"'.
                                  :
                 0174 ;;-----------------------------------------------
                      --
DDF0  01F1EA     0175 PINFLN: LD       BC,DERMGA
DDF3  CD07DF     0176         CALL     LOPWRT          ;PRINT ' :FILE"'
DDF6  CD58E2     0177         CALL     DIRETY          ;GET HL POINT FDE
DDF9  0608       0178         LD       B,08H           ;COUNTER LENGTH OF FN 8
                        CHAR
DDFB  CD19DE     0179         CALL     LPINFL          ; LOOP PRINT FILENAME
DDFE  23         0180         INC      HL              ;SKIP TO FIRST CHAR OF
                      FT
DDFF  7E         0181         LD       A,(HL)          ;GET FIRST CHAR OF FILE
                       TYPE
DE00  2B         0182         DEC      HL              ;MOVE POINTER BACK
DE01  E67F       0183         AND      7FH             ;STRIP OUT R/O BIT
```

```
DE03 FE20      0184          CP      ' '             ;CHECK IF ASSIGN FILETY
                 PE
DE05 280C      0185          JR      Z,EPINFL        ;NO FILETYPE ...PRINT E
                 ND
DE07 E5        0186          PUSH    HL
DE08 0E2E      0187          LD      C,'.'
DE0A CDC7DE    0188          CALL    FUNC2           ;PRINT SEPARATOR OF FN&
                 FT
DE0D E1        0189          POP     HL              ;RESTORE POINTER
DE0E 0603      0190          LD      B,03H           ;COUNTER LENGTH OF FT 3
                  CHAR
DE10 CD19DE    0191          CALL    LPINFL          ; LOOP PRINT FILETYPE
DE13 0E27      0192 EPINFL:  LD      C,QUOTA
DE15 CDC7DE    0193          CALL    FUNC2           ;PRINT ''' AT END OF FI
                 LENAME
DE18 C9        0194          RET
               0195 ; LOOP PRINT CHAR OF FILENAME AND FILETYPE.
DE19 23        0196 LPINFL:  INC     HL              ;SKIP POINTER
DE1A 7E        0197          LD      A,(HL)          ;GET NEXT CHAR
DE1B E67F      0198          AND     7FH             ;STRIP OUT R/O BIT
DE1D FE20      0199          CP      ' '             ;IF BLANK
DE1F 2808      0200          JR      Z,SKLPIN        ;  NOT PRINT ON SCREEN
DE21 E5        0201          PUSH    HL              ;ELSE SAVE POINTER
DE22 C5        0202          PUSH    BC              ;  AND COUNTER
DE23 4F        0203          LD      C,A
DE24 CDC7DE    0204          CALL    FUNC2           ;PRINT ONE CHAR OF FN&F
                 T
DE27 C1        0205          POP     BC              ;RESTORE COUNTER
DE28 E1        0206          POP     HL              ;  AND POINTER
DE29 10EE      0207 SKLPIN:  DJNZ    LPINFL          ;LOOP UNTIL END
DE2B C9        0208          RET                     ;LOOP END ...RETURN.
               0209 ;;-----------------------------------------------
                 --
               0210 ;;  GET CHAR FROM KEYBOARD BY CHECK IF KEYBOARD HAD
                        =
               0211 ;;   BEEN PRESSED, WHICH KEEP CHAR IN ACTIVE CHAR, GET
                        =
               0212 ;;   IT AND CLEAR. ELSE DO CONIN TO RECIEVE CHAR.
                        =
               0213 ;;-----------------------------------------------
                 --
DE2C 2127E0    0214 GETCHR:  LD      HL,ACTCHR
DE2F 7E        0215          LD      A,(HL)          ;CHECK ACTIVE CHAR
DE30 3600      0216          LD      (HL),00H        ;  AND CLEAR FOR NEXT U
                 SE
DE32 B7        0217          OR      A               ;IF KEY HAD BEEN PRESSE
                 D ?
DE33 C0        0218          RET     NZ              ;  YES, GET THAT ACTIVE
                  CHAR
DE34 C309E3    0219          JP      CONIN           ;  ELSE WAIT FOR IT.
               0220 ;;-----------------------------------------------
                 --
               0221 ;;  ECHO RECIEVED CHAR ON SCREEN EXCEPT CTRL-CHAR
                        =
               0222 ;;   (WHICH NOT CR,LF,TAB,BS)
                        =
               0223 ;;-----------------------------------------------
```

```
                        --
DE37 CD2D3E     0224 ECHO:  CALL    GETCHR          ;GET CHAR (FROM ACTIVE
                     CHAR OR BY CONIN)
DE3A CD4B3E     0225        CALL    IFCTRL          ;IF KEYIN CTRL-CHAR
DE3D D8         0226        RET     C               ;  NOT ECHO, RETURN THA
                     T CTRL-CHAR IN A
DE3E F5         0227        PUSH    AF              ;SAVE THAT CHAR
DE3F 4F         0228        LD      C,A
DE40 CDC7DE     0229        CALL    FUNC2           ;ECHO RECIEVED CHAR ON
                     SCREEN
DE43 F1         0230        POP     AF              ;RESTORE THAT CHAR TO R
                     ETURN IN A
DE44 C9         0231        RET
                0232 ;;-------------------------------------------------------
                        --
                0233 ;;  CHECK IF RECIEVED CHAR IS CTRL-CHAR BY
                        =
                0234 ;;    CR,LF,TAB,BS => RET Z
                        =
                0235 ;;       CTRL-CHAR => RET C
                        =
                0236 ;;            CHAR => RET NC
                        =
                0237 ;;-------------------------------------------------------
                        --
DE45 FE0D       0238 IFCTRL: CP     CR
DE47 C8         0239        RET     Z       ; CR => RET Z
DE48 FE0A       0240        CP      LF
DE4A C8         0241        RET     Z       ; LF => RET Z
DE4B FE09       0242        CP      TAB
DE4D C8         0243        RET     Z       ;TAB => RET Z
DE4E FE08       0244        CP      BS
DE50 C8         0245        RET     Z       ; BS => RET Z
DE51 FE20       0246        CP      ' '
DE53 C9         0247        RET             ;CTRL-CHAR => RET C
                0248 ;;-------------------------------------------------------
                        --
                0249 ;;  FREEZE SCREEN BY PRESS CTRL-S (AND IF FOLLOW BY
                        =
                0250 ;;    CTRL-C, DO WARM BOOT ELSE ANY KEY CONTINUE). IF
                        =
                0251 ;;    PRESS OTHER KEY, SAVE AS ACTIVE CHAR.
                        =
                0252 ;;-------------------------------------------------------
                        --
DE54 3A27E0     0253 FREEZE: LD     A,(ACTCHR)      ;GET ACTIVE CHAR TO CHE
                     CK
DE57 B7         0254        OR      A               ;IF KEY HAD BEEN PRESSE
                     D BEFORE THIS
DE58 2005       0255        JR      NZ,KEYPRS       ; YES, GOTO GET FUNCII
                     (00=NO KEY PRESS)
DE5A CD06E3     0256        CALL    CONST           ;ELSE CHECK IF KEY PRES
                     S NOW
DE5D E601       0257        AND     01H
DE5F C8         0258        RET     Z               ; NO KEY PRESS, GET FUN
                     CII (01=KEY PRESS)
DE60 CD0FE3     0259        CALL    CONIN           ;KEY PRESS, GET CHAR
```

```
DE63  FE13      0260          CP      13H             ;IF CTRL-S ?
DE65  2015      0261          JR      NZ,SAVACT       ;NOT CTRL-S, SAVE AS AC
                    TIVE CHAR
DE67  CD09EB    0262          CALL    CONIN           ;CTRL-S, FREEZE SCREEN
                    BY WAIT FOR NEXT CHAR
DE6A  FE03      0263          CP      03H             ;  AND IF NEXT CHAR IS
                    CTRL-C
DE6C  CA0000    0264          JP      Z,WARMB         ;    GOTO WARM BOOT
DE6F  FE10      0265          CP      10H             ;  IF NEXT CHAR IS CTRL
                    -P
DE71  2007      0266          JR      NZ,SWITP        ;NOT CTRL-P
DE73  2126E0    0267          LD      HL,TOGLEP       ;CTRL-P, GET TOGGLE
DE76  3E01      0268          LD      A,01H           ;SET
DE78  96        0269          SUB     (HL)            ;  OR RESET TOGGLE
DE79  77        0270          LD      (HL),A          ;SAVE TOGGLE
DE7A  AF        0271 SWITP:   XOR     A               ;ELSE RETURN TO CONTINU
                    E
DE7B  C9        0272          RET
DE7C  3227E0    0273 SAVACT:  LD      (ACTCHR),A      ;SAVE ACTIVE CHAR
DE7F  3E01      0274 KEYFRS:  LD      A,01H           ;SET EXIT VALUE FUNC11
                    (01=KEY PRESS)
DE81  C9        0275          RET
                0276 ;;-----------------------------------------------
                    --
                0277 ;;  HANDLE SCREEN BY TAKE CARE OF CURSOR POSITION
                         =
                0278 ;;   DEPEND ON WHAT CHAR KEYIN.  IF CHAR IS  NOT
                         =
                0279 ;;   BACKSPACE, PRINT ON SCREEN OR MAY BE ON PRINTER
                         =
                0280 ;;   IF PRINTER TOGGLE IS SET.
                         =
                0281 ;;-----------------------------------------------
                    --
DE82  3A23E0    0282 HDLESC:  LD      A,(FLBSG)       ;SET BACKSPACE FLAG
DE85  B7        0283          OR      A
DE86  2013      0284          JR      NZ,HDLESS       ;BEING BACKSPACE, NOT D
                    O FOLLOW
DE88  C5        0285          PUSH    BC              ;SAVE CHAR
DE89  CD54DE    0286          CALL    FREEZE          ;FREEZE SCREEN IF PRESS
                     CTRL-S
DE8C  C1        0287          POP     BC              ;SET CHAR TO CONOUT
DE8D  C5        0288          PUSH    BC              ;  AND SAVE TO USE IN L
                    IST
DE8E  CD0CEB    0289          CALL    CONOUT          ;PRINT ON SCREEN
DE91  C1        0290          POP     BC              ;SET CHAR TO LIST
DE92  C5        0291          PUSH    BC              ;  AND SAVE TO SET CURS
                    OR POSITION
DE93  3A24E0    0292          LD      A,(TOGLEP)      ;CHECK PRINTER TOGGLE
DE96  B7        0293          OR      A
DE97  C43FEB    0294          CALL    NZ,LIST         ;TOGGLE SET, PRINT ON P
                    RINTER
DE9A  C1        0295          POP     BC              ;SET CHAR TO SET CURSOR
                     POSITION
                0296 ;-----------------------------------------------
                    --
                0297 ;  DEAL WITH CURSOR POSITION BY INCRE CURSOR 1 FOR
```

```
                        =
                    0298 ;   NORMAL CHAR, NOT INCRE CURPOS FOR CTRL-CHAR EXCEPT
                        =
                    0299 ;   BS DECRE CURPOS 1 (IF CAN DO) AND LF SET CURPOS TO
                        =
                    0300 ;   BEGINNING.
                        =
                    0301 ;-----------------------------------------------------
                        --
DE78 79             0302 HOLE3S: LD      A,C
DE7C 212EE0         0303         LD      HL,CURPOS      ;GET CURSOR POSITION
DE7F FE7F           0304         CP      DEL            ;IF CHAR IS DEL
DEA1 C8             0305         RET     Z              ;  NOT INCRE CURPOS
DEA2 34             0306         INC     (HL)           ;ELSE INCRE CURPOS FOR
                       LETTER
DEA3 FE20           0307         CP      ' '            ; AND IF NOT CTRL-CHAR
DEA5 D0             0308         RET     NC             ; CURPOS IS CORRECT
DEA6 35             0309         DEC     (HL)           ;CTRL-CHAR NOT INCRE CU
                       RPOS
DEA7 7E             0310         LD      A,(HL)         ;CHECK NOW CURSOR POSIT
                       ION
DEA8 B7             0311         OR      A              ;IF AT THE BEGINNING
DEA9 C8             0312         RET     Z              ;RETURN ALTHOUGH KEY BS
                       OR LF
DEAA 79             0313         LD      A,C            ;BUT IF CURPOS NOT AT B
                       EGIN
DEAB FE08           0314         CP      BS             ;  AND KEY BACKSPACE
DEAD 2002           0315         JR      NZ,HOLELF
DEAF 35             0316         DEC     (HL)           ;DECRE CURPOS BY 1
DEB0 C9             0317         RET
DEB1 FE0A           0318 HOLELF: CP      LF             ;  AND IF KEY LINEFEED
DEB3 C0             0319         RET     NZ
DEB4 7600           0320         LD      (HL),00H       ;SET CURPOS BACK TO BEG
                       INNING
DEB6 C9             0321         RET
                    0322 ;;-----------------------------------------------------
                        --
                    0323 ;;  WRITE CHAR ON SCREEN OR ^CHAR IF CTRL-CHAR.
                        =
                    0324 ;;-----------------------------------------------------
                        --
DEB7 79             0325 WCTRL:  LD      A,C            ;GET CHAR
DEB8 CD45DE         0326         CALL    IFCTRL         ;CHECK IF CTRL-CHAR
DEB9 300A           0327         JR      NC,FUNC2       ;NO, WRITE THIS CHAR ON
                       SCREEN
DEBD F5             0328         PUSH    AF             ;YES, SAVE THIS CTRL-CH
                       AR
DEBE 0E5E           0329         LD      C,'^'
DEC0 CD42DE         0330         CALL    HOLE3C         ;WRITE PRECEEDING WITH
                       '^'
DEC3 F1             0331         POP     AF             ;GET CTRL-CHAR
DEC4 F640           0332         OR      40H            ;  AND CONVERT TO ASCII
                       CHAR
DEC6 4F             0333         LD      C,A            ;LOAD IN C FOR COMPAT
                    0334 ;;;=====================================================
                        ==
                    0335 ;;;  FUNC2  WRITE CONSOLE
```

```
                        0336 ;;;==========
                             ==
DEC7  79        0337 FUNC2:  LD      A,C          ;GET CHAR TO WRITE
DEC8  FE09      0338         CP      TAB          ;CHECK IF TAB
DECA  2006      0339         JR      NZ,HDLESC    ;NO, DO WRITE CONSOLE
DECC  0E20      0340 EXPTAB: LD      C,' '        ;IT IS TAB
DECE  CD82DE    0341         CALL    HDLESC       ; EXPAND WITH BLANK
DED1  3A25E0    0342         LD      A,(CURPOS)   ;  UNTIL CURSOR POSITIO
                  N
DED4  E607      0343         AND     07H          ;   IS ON POSITION 8*N
DED6  20F4      0344         JR      NZ,EXPTAB
DED8  C9        0345         RET
                        0346 ;;----------
                             --
                        0347 ;;  DO BACKSPACE ON SCREEN BY BACKSPACE CURSOR 1 COL
                             =
                        0348 ;;   AND OVERWRITE WITH BLANK. THEN BACKSPACE CURSOR
                             =
                        0349 ;;   BACK AGAIN.
                             =
                        0350 ;;----------
                             --
DED9  CDE1DE    0351 BSCHR:  CALL    BSPOS        ;MOVE CURSOR BACK
DEDC  0E20      0352         LD      C,' '
DEDE  CD0CE3    0353         CALL    CONOUT       ;OVERWRITE WITH BLANK
DEE1  0E08      0354 BSPOS:  LD      C,BS
DEE3  C30CE3    0355         JP      CONOUT       ;  THEN MOVE CURSOR BAC
                  K AGAIN
                        0356 ;;----------
                             --
                        0357 ;;  CANCEL CURRENT LINE BY WRITE '#' AT END OF LINE AND
                             =
                        0358 ;;   START AT THE SAME CURSOR POSITION ON NEXT LINE.
                             =
                        0359 ;;   (NOT DO IN CASE OF BS)
                             =
                        0360 ;;----------
                             --
DEE6  0E23      0361 CANCEL: LD      C,'#'
DEE8  CD82DE    0362         CALL    HDLESC       ;WRITE '#'
DEEB  CDFDDE    0363         CALL    CRLF         ;GOTO NEXT LINE
                        0364 ;;
                        0365 ;; SKIP CURSOR BY GET CURPOS (WHICH LF SET TO 0 FROM GE
                             T NEW LINE)
                        0366 ;;  AND WRITE BLANK TO INCRE CURPOS UNTIL EQUAL STRPOS.
DEEE  3A25E0    0367 SKPCUR: LD      A,(CURPOS)
DEF1  2124E0    0368         LD      HL,STRPOS
DEF4  BE        0369         CP      (HL)
DEF5  D0        0370         RET     NC
DEF6  0E20      0371         LD      C,' '
DEF8  CD82DE    0372         CALL    HDLESC
DEFB  18F1      0373         JR      SKPCUR
                        0374 ;;----------
                             --
                        0375 ;;  GET NEW LINE BY WRITE CR AND LF.
                             =
```

```
CROMEMCO CDOS Z80 ASSEMBLER version 02.15                PAGE 0010

                    0376 ;;-----------------------------------------------
                         --
DEFD  0E0D          0377 CRLF:  LD      C,CR
DEFF  CD82DE        0378        CALL    HDLESC
DF02  0E0A          0379        LD      C,LF
DF04  C382DE        0380        JP      HDLESC
                    0381 ;;-----------------------------------------------
                         --
                    0382 ;;  LOOP WRITE STRING , TERMINATED BY '$'.
                             =
                    0383 ;;-----------------------------------------------
                         --
DF07  0A            0384 LOPWRT: LD     A,(BC)          ;GET ONE CHAR FROM STRI
                         NG
DF08  FE24          0385        CP      '$'             ;CHECK IF '$'
DF0A  C8            0386        RET     Z               ;YES, STRING END... RET
                         URN
DF0B  03            0387        INC     BC              ;SKIP POINTER TO NEXT
DF0C  C5            0388        PUSH    BC              ;  AND SAVE POINTER
DF0D  4F            0389        LD      C,A
DF0E  CDC7DE        0390        CALL    FUNC2           ;WRITE ONE CHAR ON SCRE
                         EN
DF11  C1            0391        POP     BC              ;USE NEXT CHAR
DF12  18F3          0392        JR      LOPWRT          ;  LOOP UNTIL STRING EN
                         D
                    0393 ;;;=================================================
                         ==
                    0394 ;;;  FUNC10 READ STRING
                             =
                    0395 ;;;     INIT: B COUNT LENGTH OF INPUT STRING
                             = -
                    0396 ;;;           C KEEP ASSIGN MAX. LENGTH
                             =
                    0397 ;;;           HL IS POINTER IN STRING BUFFER
                             =
                    0398 ;;;=================================================
                         ==                                          .
DF14  3A25E0        0399 FUNC10: LD     A,(CURPOS)      ;SAVE START CURSOR POSI
                         TION
DF17  3224E0        0400        LD      (STRPOS),A      ;  AS START POSITION
DF1A  2A62E0        0401        LD      HL,(SENT2B)     ;ADDRESS OF STRING BUFF
                         ER IN HL
DF1D  4E            0402        LD      C,(HL)          ;ASSIGN MAX. LENGTH IN
                         C
DF1E  23            0403        INC     HL              ;GET ADDRESS OF STRING
                         LENGTH
DF1F  E5            0404        PUSH    HL              ;(*) AND SAVE TO FILL W
                         HEN READ END
DF20  0600          0405        LD      B,00H           ;COUNT LENGTH OF STRING
                          IN B
                    0406 ;;
                    0407 ;; LOOP READ# 1, STRING LENGTH AND POINTER OF STRING BU
                         FFER
                    0408 ;;  SET FROM INITIALIZE.
DF22  C5            0409 LOOPR1: PUSH   BC              ;SAVE STRING LENGTH
DF23  E5            0410        PUSH    HL              ;  AND POINTER OF BUFFE
                         R
```

```
                        0411 ;;
                        0412 ;; LOOP READ# 2, STRING LENGTH AND POINTER OF STRING BU
                             FFER
                        0413 ;;  SET BY EACH LINE EDIT ROUTINE.
DF24  CD2CDE            0414 LOOPR2: CALL    GETCHR          ;GET IN CHAR
DF27  E6FF              0415         AND     0FFH
DF29  E1                0416         POP     HL              ;RESTORE LENGTH
DF2A  C1                0417         POP     BC              ;  AND POINTER OF BUFFE
                             R TO USE
DF2B  FE0D              0418         CP      CR
DF2D  CAE0DF            0419         JP      Z,F10END        ;  CR => FUNC END
DF30  FE0A              0420         CP      LF
DF32  CAE0DF            0421         JP      Z,F10END        ;  LF => FUNC END
                        0422 ;-----------------------------------------------------
                             --
                        0423 ;  BACKSPACE CALCULATE AMOUNT OF COL TO BACKSPACE (AS
                                    :
                        0424 ;   IN CASE OF BACKSPACE TAB CHAR).
                                    :
                        0425 ;-----------------------------------------------------
                             --
DF35  FE08              0426         CP      BS
DF37  200D              0427         JR      NZ,DELCHR
DF39  78                0428         LD      A,B             ;CHECK STRING LENGTH
DF3A  B7                0429         OR      A
DF3B  28E5              0430         JR      Z,LOOPR1        ;NO CHAR TO DELETE, REA
                             D NEXT
DF3D  05                0431         DEC     B               ;(%) ELSE DECRE STRING
                             LENGTH BY 1
DF3E  3A25E0            0432         LD      A,(CURPOS)      ;AND GET CURPOS TO SET
                             FLAG
DF41  3223E0            0433         LD      (FLGBS),A       ; (MAYBE OFF IF CURPOS
                             DECRE TO 0)
DF44  184D              0434         JR      FROMBS          ;GOTO CALCULATE COL TO
                             BACKSPACE
                        0435 ;-----------------------------------------------------
                             --
                        0436 ;  DELETE CHAR DECRE STRING LENGTH BY 1, MOVE POINTER
                                    =
                        0437 ;   OF BUFFER BACK 1 CHAR AND ECHO DELETED CHAR.
                                    =
                        0438 ;-----------------------------------------------------
                             --
DF46  FE7F              0439 DELCHR: CP      DEL
DF48  2009              0440         JR      NZ,CTRLE
DF4A  78                0441         LD      A,B             ;CHECK STRING LENGTH
DF4B  B7                0442         OR      A
DF4C  28D4              0443         JR      Z,LOOPR1        ;NO CHAR TO DELETED, RE
                             AD NEXT
DF4E  7E                0444         LD      A,(HL)          ;  ELSE GET LAST CHAR
DF4F  05                0445         DEC     B               ;DECRE STRING LENGTH
DF50  2B                0446         DEC     HL              ;MOVE POINTER OF BUFFER
                             BACK 1 CHAR
DF51  1876              0447         JR      ECDOLY          ;GOTO ECHO DELETED CHAR
                        0448 ;-----------------------------------------------------
                             --
                        0449 ;  CTRL-E MOVES CURSOR TO THE BEGINNING NEXT LINE AND
```

```
                                =
                    0450 ;  CONTINUE READ STRING.
                                =
                    0451 ;-----------------------------------------------------
                                --
DF53  FE05          0452 CTRLE: CP      05H
DF55  2008          0453        JR      NZ,CTRLP
DF57  C5            0454        PUSH    BC              ;SAVE STRING LENGTH
DF58  E5            0455        PUSH    HL              ;  AND POINTER OF BUFFE
                          R
DF59  CDFDDE        0456        CALL    CRLF            ;DO ONLY GOTO NEXT LINE
DF5C  AF            0457        XOR     A               ; AND CONTINUE AT FIRST
                           POSITION
DF5D  3224E0        0458        LD      (STRPOS),A      ;  OF NEXT LINE
DF60  18C2          0459        JR      LOOPR2
                    0460 ;-----------------------------------------------------
                                --
                    0461 ;  CTRL-P SET/RESET TOGGLE PRINTER.
                                =
                    0462 ;-----------------------------------------------------
                                --
DF62  FE10          0463 CTRLP: CP      10H
DF64  2008          0464        JR      NZ,CTRLX
DF66  E5            0465        PUSH    HL              ;TEMP PUSH
DF67  2126E0        0466        LD      HL,TOGLEP       ;GET TOGGLE
DF6A  3E01          0467        LD      A,01H
DF6C  96            0468        SUB     (HL)            ;  SET IF OLD=00 -> NEW
                          =01
DF6D  77            0469        LD      (HL),A          ;RESET IF OLD=01 -> NEW
                          =00
DF6E  E1            0470        POP     HL              ;POP FROM TEMP PUSH
DF6F  18B1          0471        JR      LOOPR1          ;GOTO READ NEXT
                    0472 ;-----------------------------------------------------
                                --
                    0473 ;  CTRL-X CANCEL CURRENT LINE BY MOVE CURSOR BACK TO
                                =
                    0474 ;   START POSITION, THEN REDO FUNC10 AGAIN.
                                =
                    0475 ;-----------------------------------------------------
                                --
DF71  FE18          0476 CTRLX: CP      18H
DF73  2010          0477        JR      NZ,CTRLU
DF75  E1            0478        POP     HL              ;POP TO CLEAR STACK FRO
                          M (*)
DF76  3A24E0        0479 LOOPBS: LD     A,(STRPOS)      ;COMPARE START POSITION
DF79  2125E0        0480        LD      HL,CURPOS       ;  WITH NOW CURSOR POSI
                          TION
DF7C  BE            0481        CP      (HL)
DF7D  3095          0482        JR      NC,FUNC10       ;SAME POSITION, REDO FU
                          NC10 AGAIN
DF7F  35            0483        DEC     (HL)            ; ELSE BACKSPACE 1 COL
                          EACH TIME
DF80  CDD9DE        0484        CALL    BSCHR           ;UNTIL MOVE BACK TO STA
                          RT
DF83  18F1          0485        JR      LOOPBS          ;LOOP BACKSPACE
                    0486 ;-----------------------------------------------------
                                --
```

```
                        0487 ;  CTRL-U PRINT '#' AND CANCEL CURRENT LINE. MOVE
                                =
                        0488 ;   CURSOR TO NEXT LINE.
                                =
                        0489 ;-----------------------------------------------------
                                --
DF85  FE15              0490 CTRLU:  CP      15H
DF87  2006              0491         JR      NZ,CTRLR
DF89  CDE6DE            0492         CALL    CANCEL          ;WRITE '#' AND MOVE CUR
                            SOR TO NEXT LINE
DF8C  E1                0493         POP     HL              ;POP TO CLEAR STACK FRO
                            M (*)
DF8D  1885              0494         JR      FUNC10          ;REDO FUNC10 AGAIN.
                        0495 ;-----------------------------------------------------
                                --
                        0496 ;  CTRL-R PRINT '#' AND RETYPE OLD STRING ON NEXT LINE.
                                =
                        0497 ;-----------------------------------------------------
                                --
DF8F  FE12              0498 CTRLR:  CP      12H
DF91  2033              0499         JR      NZ,SAVECO
                        0500 ;;
                        0501 ;;  THIS POINT COME FROM BS AND CTRL-R.
                        0502 ;;      IF CTRL-R WILL RETYPE CHAR IN BUFFER ON SCREEN.
                        0503 ;;      IF BS WILL NOT RETYPE CHAR ON SCREEN BUT USE ROU
                            TINE
                        0504 ;;         TO CALCULATE CURPOS OF (N-1) CHAR (BS DOES L
                            ENGTH
                        0505 ;;         OF STRING BY 1 FROM (X)).
                        0506 ;;
DF93  C5 ~              0507 FROMBS: PUSH    BC              ;TEMP PUSH
DF94  CDE6DE            0508         CALL    CANCEL          ;WRITE '#' AND GOTO NEX
                            T LINE (NOT DO IF BS)
DF97  C1                0509         POP     BC              ;POP FROM TEMP PUSH
DF98  E1                0510         POP     HL              ;GET ADDRESS OF STRING
                            LENGTH
DF99  E5                0511         PUSH    HL              ;SAVE ADDRESS OF STRING
                             LENGTH
DF9A  C5                0512         PUSH    BC              ;  AND NEW LENGTH (IF C
                            OME FROM BS)
                        0513 ;
                        0514 ; LOOP REPEAT WILL RETYPE ALL CHAR IN BUFFER ON SCREEN
                            FOR CTRL-R.
                        0515 ;  BUT IF BS WILL CALCULATE (N-1) CHAR IN BUFFER TO FIN
                            D CURPOS.
DF9B  78                0516 LOPREP: LD      A,B             ;GET LENGTH OF STRING T
                            O RETYPE
DF9C  B7                0517         OR      A
DF9D  2830              0518         JR      Z,DOBS          ;REPEAT ALL, GOTO DO BS
DF9F  23                0519         INC     HL              ;SKIP POINTER IN BUFFER
DFA0  4E                0520         LD      C,(HL)          ;GET CHAR FROM BUFFER
DFA1  05                0521         DEC     B               ;DECRE COUNTER FOR REPE
                            AT
DFA2  C5                0522         PUSH    BC
DFA3  E5                0523         PUSH    HL
DFA4  CD87DE            0524         CALL    WCTRL           ;RETYPE CHAR ON SCREEN
DFA7  E1                0525         POP     HL
```

```
DFA8  C1          0526          POP     BC
DFA9  18F0        0527          JR      LOPREP          ;LOOP REPEAT.
                  0528 ;
                  0529 ; THIS PROCESS FOR BS ONLY. TO FIND AMOUNT OF COL TO BA
                       CKSPACE
                  0530 ;  FROM CURPOS WHEN RECIEVE BS AND CURPOS OF (N-1) CHAR
                       , THEN
                  0531 ;  MOVE CURSOR BACK. (IN CASE OF BACKSPACE TAB CHAR)
DFAB  E5          0532 DOBS:    PUSH    HL              ;SAVE CURRENT POINTER O
                       F BUFFER
DFAC  3A23E0      0533          LD      A,(FLGBS)       ;CHECK IF COME FROM BS
DFAF  B7          0534          OR      A
DFB0  CA240F      0535          JP      Z,LOOPR2        ;NO, FROM CTRL-R... ONL
                       Y REPEAT
DFB3  2125E0      0536          LD      HL,CURPOS       ;FIND DISTANCE BETWEEN
DFB6  96          0537          SUB     (HL)            ;  OLD CURSOR POSITION
                       (WHEN GETIN BS)
DFB7  3223E0      0538          LD      (FLGBS),A       ;  AND CURRENT POSITION
                       (AFTER REPEAT)
DFBA  CD0F3E      0539 LOOPDO:  CALL    BSCHR           ;  THEN USE AS COUNTER
DFBD  2123E0      0540          LD      HL,FLGBS        ;  TO MOVE CURSOR BACK
DFC0  35          0541          DEC     (HL)            ;  AND CLEAR WITH BLANK
DFC1  20F7        0542          JR      NZ,LOOPDO
DFC3  C3240F      0543          JP      LOOPR2          ;THEN READ NEXT.
                  0544 ;;-------------------------------------------------
                       --
                  0545 ;; SAVE CHAR RECEIVE FROM KEYBOARD AND ECHO THIS CHAR
                            =
                  0546 ;;  ON SCREEN.
                            =
                  0547 ;;-------------------------------------------------
                       --
DFC6  23          0548 SAVECO:  INC     HL              ;SKIP POINTER OF BUFFER
                       TO NEXT EMPTY
DFC7  77          0549          LD      (HL),A          ;  AND FILL CHAR IN BUF
                       FER
DFC8  04          0550          INC     B               ;INCRE LENGTH OF STRING
                  0551 ;;-------------------------------------------------
                       --
                  0552 ;; ECHO CHAR ON SCREEN ONLY, NOT FILL IN STRING BUFFER
                            =
                  0553 ;;  (USED IN CASE OF DEL).
                            =
                  0554 ;;-------------------------------------------------
                       --
DFC9  D5          0555 ECOONLY: PUSH    DE
DFCA  E5          0556          PUSH    HL
DFCB  4F          0557          LD      C,A
DFCC  CD972E      0558          CALL    WCTRL           ;ECHO CHAR OR ^CHAR ON
                       SCREEN
DFCF  E1          0559          POP     HL
DFD0  C1          0560          POP     BC
                  0561 ;
                  0562 ; THIS PROCESS CHECK IF CHAR IS CTRL-C AND MUST BE FIRS
                       T
                  0563 ;  INPUT CHAR, GOTO WARM BOOT.
DFD1  7E          0564          LD      A,(HL)          ;GET RECIEVE CHAR FROM
```

```
                        BUFFER
DFD2  FE03      0565            CP      03H
DFD4  78        0566            LD      A,B
DFD5  2005      0567            JR      NZ,NOCTLC       ;NOT CTRL-C, OK.
DFD7  FE01      0568            CP      01H             ;IT IS CTRL-C
DFD9  CA0000    0569            JP      Z,WARMS         ;  AND LENGTH 1 (FIRST
                        CHAR), WARM BOOT
DFDC  B9        0570 NOCTLC: CP         C               ;COMPARE WITH ASSIGN MA
                        X LENGTH
DFDD  DA22DF    0571            JP      C,LOOPR1        ;BUFFER STILL AVIABLE,
                        READ NEXT
                0572 ;;;--------------------------------------------------
                        --
                0573 ;;;  END OF FUNC10 BY PRESS <CR> OR BUFFER FULL.
                                =
                0574 ;;;--------------------------------------------------
                        --
DFE0  E1        0575 F10END: POP        HL              ;GET ADDRESS OF STRING
                        LENGTH FROM (*)
DFE1  70        0576            LD      (HL),B          ;  FILL WITH LENGTH OF
                        INPUT STRING
DFE2  0E0D      0577            LD      C,CR
DFE4  C382DE    0578            JP      HDLESC          ;AND GOTO NEXT LINE.
                0579 ;;;==================================================
                        ==
                0580 ;;;  FUNC1  READ CONSOLE.
                                =
                0581 ;;;    RECIEVE CHAR FROM KEYBOARD AND ECHO ON SCREEN
                                =
                0582 ;;;    EXCEPT CTRL-CHAR (WHICH NOT CR,LF,TAB,BS).
                                =
                0583 ;;;==================================================
                        ==
DFE7  CD37DE    0584 FUNC1:  CALL       ECHO            ;GETIN CHAR AND ECHO ON
                         SCREEN
DFEA  182F      0585            JR      EXIT1B          ;SAVE INPUT CHAR AS EXI
                        T VALUE.
                0586 ;;;==================================================
                        ==
                0587 ;;;  FUNC3  READER
                                =
                0588 ;;;==================================================
                        ==
DFEC  CD15E8    0589 FUNC3:  CALL       READER          ;GET CHAR FROM READER D
                        EVICE
DFEF  182A      0590            JR      EXIT1B          ;SAVE INPUT CHAR AS EXI
                        T VALUE.
                0591 ;;;==================================================
                        ==
                0592 ;;;  FUNC6  DIRECT I/O CONSOLE
                                =
                0593 ;;;    IF SENT VALUE IS FF => INPUT MODE
                                =
                0594 ;;;       SENT VALUE NOT FF (CHAR) => OUTPUT MODE
                                =
                0595 ;;;==================================================
                        ==
```

```
DFF1  79        0596 FUNC6:  LD      A,C          ;GET SENT VALUE FROM C
DFF2  3C        0597         INC     A            ;CHECK IF FF
DFF3  2807      0598         JR      Z,INMODE     ;YES, GOTO INPUT MODE
DFF5  3C        0599         INC     A
DFF6  CA06EB    0600         JP      Z,CONST      ;CHECK CONSOLE STATUS
DFF9  C30CE3    0601         JP      CONOUT       ;SENT CHAR IN C TO CONS
                     OLE
                0602 ;; INPUT MODE.
DFFC  CD06EB    0603 INMODE: CALL    CONST        ;CHECK CONSOLE STATUS
DFFF  B7        0604         OR      A            ;IF NO KEYPRESS
E000  CA5FEA    0605         JP      Z,RSTPOS     ; NOT WAIT, RETURN TO C
                     ALLING PROGRAM
E003  CD09EB    0606         CALL    CONIN        ;KEYPRESS, GETIN CHAR
E006  1813      0607         JR      EXIT1B       ; AND SAVE AS EXIT VALU
                     E.
                0608 ;;;=============================================================
                     ==
                0609 ;;;  FUNC7  DETERMINE IOBYTE
                                =                                    -
                0610 ;;;=============================================================
                     ==
E008  3A0300    0611 FUNC7:  LD      A,(IOBYTE)   ;GET IOBYTE
E00B  180E      0612         JR      EXIT1B       ;  AND SAVE AS EXIT VAL
                     UE
                0613 ;;;=============================================================
                     ==
                0614 ;;; FUNC8  SET IOBYTE
                                =
                0615 ;;;=============================================================
                     ==
E00D  210300    0616 FUNC8:  LD      HL,IOBYTE    ;GET ADDRESS OF IOBYTE
E010  71        0617         LD      (HL),C       ;  TO FILL WITH NEW ASS
                     IGN
E011  C9        0618         RET
                0619 ;;;=============================================================
                     ==
                0620 ;;;  FUNC9  PRINT STRING
                                =
                0621 ;;;=============================================================
                     ==
E012  EB        0622 FUNC9:  EX      DE,HL
E013  4D        0623         LD      C,L
E014  44        0624         LD      B,H          ;SET BC POINT ADDRESS O
                     F STRING
E015  C3070F    0625         JP      LOPWRT       ;GOTO LOOP WRITE EACH C
                     HAR OF STRING
                0626 ;;;=============================================================
                     ==
                0627 ;;;  FUNC11  RETURN CONSOLE STATUS
                                =
                0628 ;;;     00 => NO KEYPRESS
                                =
                0629 ;;;     01 => KEYPRESS (CHAR SAVE IN ACTIVE CHAR)
                                =
                0630 ;;;=============================================================
                     ==
E018  CD54DE    0631 FUNC11: CALL    FREEZE       ;CHECK KEYPRESS OR NOT
```

```
                        BY RTN FREEZE.
                    0632 ;;;=====================================================
                         ==
                    0633 ;;; EXIT I BYTE IS COMMON ENTRY POINT OF ALL FUNCTIONS
                              =
                    0634 ;;;  TO EXIT DOS TO CALLING PROGRAM WITH RETURN VALUE
                              =
                    0635 ;;;  LENGTH ** 1 BYTE **
                              =
                    0636 ;;;=====================================================
                         ==
E018  326AE0        0637 EXIT13: LD      (EXTVAL),A      ;SAVE EXIT VALUE 1 BYTE
      (E01E)        0638 FUNC39  EQU     $
E01E  C9            0639         RET                    ;EXIT BY POP ADDRESS OF EXIT RO
                         UTINE.
                    0640 ;;-----------------------------------------------------
                         --
                    0641 ;;  ROUTINE SET EXIT VALUE NOT ZERO FOR DISK I/O.
                              =
                    0642 ;;-----------------------------------------------------
                         --
E01F  3E01          0643 NOTZED: LD      A,01H          ;SET EXIT VALUE NOT ZER
                         O (01)
E021  18F8          0644         JR      EXIT13
                    0645 ;;;=====================================================
                         ==
                    0646 ;;;    W O R K    A R E A  ::::::::::::::::::::::::::::::
                         ::
                    0647 ;;;=====================================================
                         ==
E023  00            0648 FLBSS:  DB      00H     ;FLAG OF BACKSPACE
E024  00            0649 STRPOS: DB      00H     ;START CURSOR POSITION
E025  00            0650 CURPOS: DB      00H     ;CURRENT CURSOR POSITION
E026  00            0651 TOGLEP: DB      00H     ;PRINTER TOGGLE (0/1)
E027  00            0652 ACTCHR: DB      00H     ;ACTIVE CHAR
E028  0000          0653 OLDSP:  DW      0000H   ;STACK POINTER OF CALLING PROGR
                         AM
E02A  (0032)        0654         DS      50      ;DOS STACK AREA FOR 25 ENTRIES
      (E05C)        0655 STACK   EQU     $       ;DOS STACK START HERE !!!
E05C  0000          0656         DW      0000H   ;STACK BUFFER UNDERFLOW
E05E  00            0657 USRCOD: DB      00H     ;USER CODE
E05F  00            0658 CDRIVE: DB      00H     ;CURRENT DRIVE CODE
E060  00            0659 FNUM:   DB      00H     ;CURRENT DOS FUNC#
E061  00            0660 EXSENT: DB      00H     ;EXTRA SENT PARAMETER
E062  0000          0661 SENT2B: DW      0000H   ;SENT PARAMETER 2 BYTES
E064  0000          0662 EXTVAL: DW      0000H   ;AREA TO SAVE EXIT VALUE
                    0663 ;;-----------------------------------------------------
                         --
                    0664 ;; MOVE FROM DE TO HL, LENGTH IN C (DE ==> HL).
                              =
                    0665 ;;-----------------------------------------------------
                         --
E066  0C            0666 MOVER:  INC     C       ;INCRE TO TEST START LOOP
E067  0D            0667 LOPMOV: DEC     C
E068  C8            0668         RET     Z
E069  1A            0669         LD      A,(DE)
E06A  77            0670         LD      (HL),A
```

```
E06B 13          0671          INC     DE
E06C 23          0672          INC     HL
E06D 18F8        0673          JR      LOPMOV
                 0674 ;;-------------------------------------------------
                      ---
                 0675 ;; MOVE DISK PARAMETER BLOCK.
                                =
                 0676 ;; CONDITION: REG E KEEP LOGVEC OF DRIVE IN RIGHTMOST B
                      IT=
                 0677 ;;   THIS ROUTINE MOVE DISK CHARACTERISTIC FROM I/O INT
                      O =
                 0678 ;;   DOS WHEN LOGON DRIVE.
                                =
                 0679 ;; RET Z IF ERROR/ RET NZ IF OK.
                                =
                 0680 ;;-------------------------------------------------
                      ---
E06F 3A5FE0      0681 DSKBOK: LD      A,(CDRIVE)      ;SET CDRIVE
E072 4F          0682          LD      C,A             ;SET C=DRIVE CODE, E=LO
                      GVEC OF DRIVE
E073 CD1BEB      0683          CALL    SETDRV          ;  FOR ROUTINE SETDRV I
                      N I/O
E076 7C          0684          LD      A,H             ;ON RETURN FROM I/O
E077 B5          0685          OR      L               ;  HL=ADD. OF DPH OR HL
                      =00 IF ERROR
E078 C8          0686          RET     Z               ;(*) RET Z TO DECLEAR E
                      RROR.
E079 5E          0687          LD      E,(HL)
E07A 23          0688          INC     HL
E07B 56          0689          LD      D,(HL)          ;DE KEEP ADDRESS OF SKE
                      W TABLE
E07C 23          0690          INC     HL
E07D 2270EA      0691          LD      (ADDDW1),HL     ;(1)SAVE ADDRESS OF DW1
E080 23          0692          INC     HL
E081 23          0693          INC     HL
E082 2272EA      0694          LD      (ADDDW2),HL     ;(2)SAVE, ADDRESS OF DW2
E085 23          0695          INC     HL
E086 23          0696          INC     HL
E087 2274EA      0697          LD      (ADDDW3),HL     ;(3)SAVE ADDRESS OF DW3
E08A 23          0698          INC     HL
E08B 23          0699          INC     HL
E08C EB          0700          EX      DE,HL           ;DE KEEP ADDRESS OF DIR
                      BUF
E08D 228DEA      0701          LD      (ADDSKW),HL     ;(4)SAVE ADDRESS OF SKE
                      W TABLE
E090 2176EA      0702          LD      HL,ADDDIB       ;(5)SAVE ADDRESS OF DIR
                      BUF
E093 0E0B        0703          LD      C,0BH           ;(6)ADDRESS OF DPB, (7)
                      ADDRESS OF ALV
E095 CD66E0      0704          CALL    MOVER           ;(3)ADDRESS OF CSV, LEN
                      GTH 3 BYTES.
E098 2A78EA      0705          LD      HL,(ADDDPB)
E09B EB          0706          EX      DE,HL           ;DE POINT DPB IN I/O
E09C 217EEA      0707          LD      HL,SPT          ;(9)TRANSFER DPB FROM I
                      /O
E09F 0E0F        0708          LD      C,0FH           ;     INTO DOS
E0A1 CD66E0      0709          CALL    MOVER           ;      LENGTH 15 BYTES
```

```
E0A4  2A83EA      0710          LD      HL,(DSM)        ;CHECK AMOUNT OF ALLOC
                      BLOCK
E0A7  7C          0711          LD      A,H
E0A8  219AEA      0712          LD      HL,DMLEN        ; TO SET LENGHT OF DISK
                       MAP
E0AB  36FF        0713          LD      (HL),0FFH       ;<10>SET DMLEN=FF IF #3
                     LOCK < 256
E0AD  B7          0714          OR      A
E0AE  2802        0715          JR      Z,ENDBOK
E0B0  3600        0716          LD      (HL),00H        ; OR SET DMLEN=00 IF #3
                     LOCK >= 256
E0B2  3EFF        0717 ENDBOK:  LD      A,0FFH          ;(*) SET RET NZ
E0B4  B7          0718          OR      A               ;  TO DECLEAR NO ERROR.
E0B5  C9          0719          RET
                  0720 ;;-------------------------------------------------
                      --
                  0721 ;;  INITIALIZE HOME TRACK.
                          =
                  0722 ;;    BY CLEAR TRACK COUNT (DW2) AND SECTOR COUNT (DW3)
                          =
                  0723 ;;    IN I/O AREA TO ZERO.
                          =
                  0724 ;;-------------------------------------------------
                      --
E0B6  CD18EB      0725 INTHOM:  CALL    HOME            ;SET TRACK 0
E0B9  AF          0726          XOR     A
E0BA  2A72EA      0727          LD      HL,(ADDDW2)     ;GET ADDRESS OF DW2
E0BD  77          0728          LD      (HL),A
E0BE  23          0729          INC     HL
E0BF  77          0730          LD      (HL),A          ;  TO CLEAR WITH ZERO
E0C0  2A74EA      0731          LD      HL,(ADDDW3)     ;GET ADDRESS OF DW3
E0C3  77          0732          LD      (HL),A
E0C4  23          0733          INC     HL
E0C5  77          0734          LD      (HL),A          ;  TO CLEAR WITH ZERO
E0C6  C9          0735          RET
                  0736 ;;;=================================================
                      ==
                  0737 ;;;  READ A SECTRO FROM DISK.
                          =
                  0738 ;;;=================================================
                      ==
E0C7  CD27EB      0739 RDISK:   CALL    READ    ;READ SECTOR
E0CA  1803        0740          JR      RWSTAT  ;GOTO CHECK STATUS OF READ
                  0741 ;;;=================================================
                      ==
                  0742 ;;;  WRITE A SECTOR ON DISK.
                          =
                  0743 ;;;=================================================
                      ==
E0CC  CD2AEB      0744 WDISK:   CALL    WRITE   ;WRITE SECTOR
                  0745 ;;;  CHECK STATUS OF READ/WRITE DISK ;;;;;;;;;;;;;;;;;;;
                      ;;
E0CF  B7          0746 RWSTAT:  OR      A
E0D0  C8          0747          RET     Z       ;READ/WRITE OK...RETURN
E0D1  C3910D      0748          JP      PINER1  ; ELSE PRINT 'R/W DISK ERROR'**
                     ***
                  0749 ;;;=================================================
```

```
                                ==
                        0750 ;;;  SET PHYSICAL TRACK AND SECTOR FOR DIRECTORY ENTRY
                                =
                        0751 ;;;==========================================================
                                ==
                        0752 ;<1> DIVIDE DIRECTORY ENTRY COUNT BY 4 TO FIND SECTOR O
                                F
                        0753 ;  THIS ENTRY. GET CURRENT TRACK AND SECTOR FROM TRKCNT
                        0754 ;  AND SECCNT.
E0D4  2AA7EA            0755 TRKSEC: LD      HL,(DIRCTL)     ;GET DIRCNT HIGH & LOW
                                BYTE
E0D7  0E02              0756         LD      C,02H           ;  DIVIDE BY 4 TO FIND
                                DIRSEC
E0D9  CDF0E1            0757         CALL    SFTRHT          ;  (1 SECTOR HAS 4 ENTR
                                IES).
E0DC  22A2EA            0758         LD      (DIRSC1),HL     ;AND SAVE IN
E0DF  22A9EA            0759         LD      (DIRSC2),HL     ;  AND IN
E0E2  21A2EA            0760 DATARE: LD      HL,DIRSC1
E0E5  4E                0761         LD      C,(HL)
E0E6  23                0762         INC     HL
E0E7  46                0763         LD      B,(HL)          ;MOVE DIRECTORY SECTOR
                                IN BC
E0E8  2A74EA            0764         LD      HL,(ADDDW3)
E0EB  5E                0765         LD      E,(HL)
E0EC  23                0766         INC     HL
E0ED  56                0767         LD      D,(HL)          ;MOVE SECTOR COUNT IN D
                                E
E0EE  2A72EA            0768         LD      HL,(ADDDW2)
E0F1  7E                0769         LD      A,(HL)
E0F2  23                0770         INC     HL
E0F3  66                0771         LD      H,(HL)
E0F4  6F                0772         LD      L,A             ;MOVE TRACK COUNT IN HL
                        0773 ;<2>  TO FIND TRACK BY DECREASE SECTOR COUNT BY SECTOR/
                                TRACK AND
                        0774 ;  DECREASE TRACK COUNT BY 1. SO SECCNT IS LOWER BOUND
                                OF DIRSEC.
E0F5  79                0775 FNDTRK: LD      A,C             ;COMPARE DIRSEC
E0F6  93                0776         SUB     E
E0F7  78                0777         LD      A,B
E0F8  9A                0778         SBC     A,D             ;  WITH SECCNT
E0F9  300E              0779         JR      NC,LOGTRK       ;IF DIRSEC>=SECCNT, GOT
                                O NEXT STEP.
E0FB  E5                0780         PUSH    HL              ;  ELSE (PUSH TRKCNT)
E0FC  2A7EEA            0781         LD      HL,(SPT)        ;<1>GET SPT
E0FF  7B                0782         LD      A,E
E100  95                0783         SUB     L
E101  5F                0784         LD      E,A
E102  7A                0785         LD      A,D
E103  9C                0786         SBC     A,H
E104  57                0787         LD      D,A             ;<2>DECRE SECCNT BY SPT
E105  E1                0788         POP     HL              ;GET TRKCNT
E106  2B                0789         DEC     HL              ;<3>DECRE TRKCNT BY 1
E107  18EC              0790         JR      FNDTRK          ;<4> LOOP UNTIL DIRSEC>
                                =SECCNT.
                        0791 ;<3> TO FIND TRUE LOGICAL TRACK BY TRIAL INCREASE SECCN
                                T BY SPT.
                        0792 ;  IF GREATER THAN DIRSEC, SECCNT IS TRUE LOWER BOUND O
```

```
                        F DIRSEC,
                    0793 ;  TRACK# IS CORRECT.  IF NOT, INCREASE SECCNT BY SPT,
                        INCREASE
                    0794 ;  TRKCNT BY 1, AND INCREASE UNTIL FIND TRUE LOWER BOUN
                        D.
E109  E5            0795 LOGTRK: PUSH    HL              ;(PUSH TRKCNT)
E10A  2A7EEA        0796         LD      HL,(SPT)        ;GET SPT
E10D  19            0797         ADD     HL,DE           ;TRIAL INCRE SECCNT BY
                        SPT
E10E  3803          0798         JR      C,PHYTRK        ;GREATER THAN DIRSEC, T
                        RACK IS CORRECT
E110  79            0799         LD      A,C             ;  ELSE USE THIS NEW SE
                        CCNT
E111  95            0800         SUB     L
E112  78            0801         LD      A,B
E113  9C            0802         SBC     A,H
E114  3805          0803         JR      C,PHYTRK
E116  EB            0804         EX      DE,HL
E117  E1            0805         POP     HL              ;(POP TRKCNT)
E118  23            0806         INC     HL              ;  AND INCRE TRKCNT BY
                        1
E119  18EE          0807         JR      LOGTRK          ;INCRE SECCNT TO TRUE L
                        OWER BOUND.
                    0808 ;(4) FOUND LOGICAL TRACK, SO TRKCNT AND SECCNT ARE CORR
                        ECT. SAVE IN DW2
                    0809 ;  AND DW3 FOR USE WITH NEXT ENTRY. THEN ADD TRKCNT WIT
                        H OFFSET TRACK
                    0810 ;  TO SET ABSOLUTE TRACK.
E11B  E1            0811 PHYTRK: POP     HL              ;GET TRKCNT
E11C  C5            0812         PUSH    BC              ;PUSH DIRSEC
E11D  D5            0813         PUSH    DE              ;PUSH SECCNT
E11E  E5            0814         PUSH    HL              ;PUSH TRKCNT
E11F  EB            0815         EX      DE,HL           ;DE = TRKCNT
E120  2A80EA        0816         LD      HL,(OFF)        ;HL = OFFSET TRACK
E123  19            0817         ADD     HL,DE           ;FROM ABSOLUTE TRACK
E124  44            0818         LD      B,H
E125  4D            0819         LD      C,L             ;  AND MOVE TO BC FOR
E126  CD1EEB        0820         CALL    SETTRK          ;(1)SET ABSOLUTE TRACK
E129  D1            0821         POP     DE              ;POP TRKCNT
E12A  2A72EA        0822         LD      HL,(ADDDW2)     ;GET ADDRESS OF DW2
E12D  73            0823         LD      (HL),E
E12E  23            0824         INC     HL
E12F  72            0825         LD      (HL),D          ;(2)SAVE TRKCNT IN DW2
E130  D1            0826         POP     DE              ;POP SECCNT
E131  2A74EA        0827         LD      HL,(ADDDW3)     ;GET ADDRESS OF DW3
E134  73            0828         LD      (HL),E
E135  23            0829         INC     HL
E136  72            0830         LD      (HL),D          ;(3)SAVE SECCNT IN DW3
                    0831 ;(5) TO FIND TRUE LOGICAL SECTOR BY SUBTRACT DIRSEC WIT
                        H SECCNT
                    0832 ;  (LOWER BOUND) TO SET SECTOR IN TRACK.
E137  C1            0833         POP     BC              ;POP DIRSEC
E138  79            0834         LD      A,C             ;TO FIND SECTOR# BY
E139  93            0835         SUB     E               ;  SECTOR# = DIRSEC - S
                        ECCNT
E13A  4F            0836         LD      C,A
E13B  78            0837         LD      A,B
```

```
E13C 9A          0838        SBC     A,D
E13D 47          0839        LD      B,A             ;TRUE LOGICAL SECTOR# I
                             N BC
                 0840 ;(6) TO FIND PHYSICAL SECTOR BY SET FROM SKEW TABLE.
E13E 2A8DEA      0841        LD      HL,(ADDSKW)     ;GET ADDRESS OF SKEW TA
                             BLE
E141 E3          0842        EX      DE,HL           ;SET LOG# IN BC, ADD. O
                             F SKEW IN DE
E142 CD7DE3      0843        CALL    SECTRN          ;  FOR ROUTINE SECTRN I
                             N I/O
E145 4D          0844        LD      C,L             ;RETURN PHYSICAL SECTOR
                              IN HL
E146 44          0845        LD      B,H             ;MOVE TO BC FOR
E147 C321E9      0846        JP      SETSEC          ;(4)SET PHYSICAL SECTOR
                             .
                 0847 ;;-------------------------------------------------
                             --
                 0848 ;;  TO FIND POSITION OF DM OF THIS CURRENT RECORD AND
                                  =
                 0849 ;;  EXTENT. RETURN POSITION OF DM IN A.
                                  =
                 0850 ;;-------------------------------------------------
                             --
                 0851 ;(1) GET BLOCK SHIFT FACTOR AND CURRENT RECORD.
E14A 218DEA      0852 GETDM: LD      HL,BSH          ;GET BLOCK SHIFT FACTOR
E14D 4E          0853        LD      C,(HL)          ;  IN C
E14E 3AA8EA      0854        LD      A,(CURREC)      ; AND GET CURREC IN A
                 0855 ;(2) SHIFT RIGHT (DIVIDE) CURRENT RECORD BY BSH TO FIND
                 0856 ;  SECTOR ALLOCATE BY WHICH DM.
                 0857 ;     +------------+----+----+-----/  /----+----+----+
                 0858 ;     ! FILE SPEC. !DM0 !DM1 ! ....   .... !DM15! CR !
                 0859 ;     +------------+----+----+-----/  /----+----+----+
                 0860 ;                     /    /                   /
                 0861 ;                   BOLCK BLOCK              BLOCK
E151 B7          0862 LBLOCK: OR      A              ;CLEAR CARRY
E152 1F          0863        RRA                     ;DIVIDE CURREC
E153 0D          0864        DEC     C               ;  WITH FACTOR
E154 20FB        0865        JR      NZ,LBLOCK       ;    FROM BSH
E156 47          0866        LD      B,A             ;#BLOCK IN B
                 0867 ;(3) FIND NUMBER TO SLOVE DM0 OF THIS EXTENT
                 0868 ;  (IN CASE OF 1 PHYSICAL EXTENT HAS MANY LOGICAL EXTEN
                             T).
                 0869 ; :<------  LOGICAL EXTENT# 0  ----->:<------  LOGICAL
                             EXTENT# 1  ----->:
                 0870 ; +------------+----+--/ /-+----+----+-----------+----
                             +-/ /--+----+----+
                 0871 ; ! FILE SPEC. !DM0 !      !DM15! CR ! FILE SPEC. !DM0
                             !      !DM15! CR !
                 0872 ; +------------+----+--/ /-+----+----+-----------+----
                             +-/ /--+----+----+
E157 3E08        0873        LD      A,08H           ;SET 8
E159 96          0874        SUB     (HL)            ;  SUBTRACT WITH BSH
E15A 4F          0875        LD      C,A             ;    TO FIND COUNTER
E15B 3A9FEA      0876        LD      A,(EXTENT)      ;GET CURRENT EXT#
                 0877 ;(4) SHIFT LEFT (MULTIPLY) CURRENT EXTENT TO FIND POSIT
                             ION OF
                 0878 ;  DM0 OF THIS EXTENT.
```

```
E15E  0D          0879 LEXTEN: DEC     C
E15F  2804        0880         JR      Z,FORMDM
E161  B7          0881         OR      A              ;CLEAR CARRY
E162  17          0882         RLA                    ;ROTATE LEFT (MULTIPLY)
E163  18F9        0883         JR      LEXTEN
                  0884 ;(5) ADD DM OF CURREC (OFFSET) WITH POSITION OF DM0 OF
                       THIS
                  0885 ;  EXTENT (BASE) TO GET POSITION OF DM FOR CURREC IN A.
E165  80          0886 FORMDM: ADD     A,B            ;ADD POSITION OF DM0 WI
                       TH DM.
E166  C9          0887         RET
                  0888 ;;-----------------------------------------------------
                       --
                  0889 ;;  GET BLOCK# POINT BY DISK ALLOCATION MAP IN HL.
                                =
                  0890 ;;    CONDITION: POSITION OF DM SENT IN BC.
                                =
                  0891 ;;-----------------------------------------------------
                       --
                  0892 ;(1) GET FCB ADDRESS, OFFSET TO FIRST DM THEN ADD WITH
                  0893 ;  POSITION OF DM TO GET BLOCK#.
E167  2A62E0      0894 GETBK:  LD      HL,(SENT2B)    ;GET FCB ADDRESS
E16A  111000      0895         LD      DE,0010H       ;OFFSET TO FIRST DM
E16D  19          0896         ADD     HL,DE          ;POINT AT FIRST DM
E16E  09          0897         ADD     HL,BC          ;POINT AT DM ALLOCATE C
                       URREC
E16F  3A9AEA      0898         LD      A,(DMLEN)      ;GET LENGTH OF DM
E172  B7          0899         OR      A              ;  (FF=DM 1 BYTE/ 00=DM
                        2 BYTES)
E173  2804        0900         JR      Z,FORMBK       ;DM 2 BYTES
                  0901 ;(2) DM 1 BYTE, MOVE BBLOCK# IN L AND CLEAR H (HL = 00XX
                       ).
E175  6E          0902         LD      L,(HL)         ;DM 1 BYTE
E176  2600        0903         LD      H,00H          ;SET HL KEEP BLOCK#
E178  C9          0904         RET
                  0905 ;(3) DM 2 BYTES, MOBE BLOCK# IN HL (HL = XXXX).
E179  09          0906 FORMBK: ADD     HL,BC          ;DOUBLE ADD FOR DM 2 BY
                       TES
E17A  5E          0907         LD      E,(HL)
E17B  23          0908         INC     HL
E17C  56          0909         LD      D,(HL)         ;GET BLOCK# IN DE
E17D  EB          0910         EX      DE,HL          ;SET HL KEEP BLOCK#
E17E  C9          0911         RET
                  0912 ;;-----------------------------------------------------
                       --
                  0913 ;;  SAVE BLOCK# OF CURRENT RECORD IN DIRSEC.
                                =
                  0914 ;;-----------------------------------------------------
                       --
E17F  CD4AE1      0915 SAVEBK: CALL    GETDM          ;GET DM OF CURREC IN A
E182  4F          0916         LD      C,A
E183  0600        0917         LD      B,00H          ;MOVE POSITION OF DM IN
                        BC
E185  CD67E1      0918         CALL    GETBK          ; TO GET RETURN BLOCK#
                       IN HL
E188  22A2EA      0919         LD      (DIRSC1),HL    ;  AND SAVE BOLCK# IN
                       DIRSEC.
```

```
E18B C9          0920         RET
                 0921 ;;-----------------------------------------------------
                      --
                 0922 ;; CHECK BLOCK#. IF BOLCK 0 (DM NOT USED)... RET Z.
                      =
                 0923 ;;-----------------------------------------------------
                      --
E18C 2AA2EA      0924 CHEKBK: LD      HL,(DIRSC1)     ;GET BLOCK# IN HL
E18F 7D          0925         LD      A,L
E190 B4          0926         OR      H               ;CHECK IF BLOCK 0
E191 C9          0927         RET                     ;  ... RET Z
                 0928 ;;-----------------------------------------------------
                      --
                 0929 ;;  TO FIND BLOCK# * BLOCK SIZE + SECTOR IN BLOCK.
                      =
                 0930 ;;-----------------------------------------------------
                      --
                 0931 ;(1) GET BLOCK SHIFT TO SLOVE BLOCK SIZE.
E192 3A80EA      0932 BADDSB: LD      A,(BSH)         ;GET BSH
E195 2AA2EA      0933         LD      HL,(DIRSC1)     ;GET BLOCK#
                 0934 ;(2) MULTIPLY BLOCK# WITH BLOCK SIZE.
E198 29          0935 BXBSIZ: ADD     HL,HL           ;MULTIPLY BLOCK#
E199 3D          0936         DEC     A               ;  WITH BLOCK SIZE
E19A 20FC        0937         JR      NZ,BXBSIZ       ;  (# OF LOG. SECTOR IN
                      BLOCK)
E19C 22A4EA      0938         LD      (BKXBKZ),HL     ;SAVE RESULT
                 0939 ;(3) GET BLOCK MASK TO FIND SECTOR IN BLOCK.
E19F 3A81EA      0940         LD      A,(BLM)         ;GET BLM
E1A2 4F          0941         LD      C,A             ;  IN C
E1A3 3AA0EA      0942         LD      A,(CURREC)      ;GET CURREC TO FIND SEC
                      TOR IN BLOCK
E1A6 A1          0943         AND     C               ;  BY MOD WITH BLM
                 0944 ;(4) ADD BLOCK# * BLOCK SIZE WITH SECTOR IN BLOCK.
E1A7 B5          0945         OR      L               ;THEN ADD TO
E1A8 6F          0946         LD      L,A             ;  BLOCK# * BLOCK SIZE
                      (BY OR)
E1A9 22A2EA      0947         LD      (DIRSC1),HL     ;SAVE RESULT.
E1AC C9          0948         RET
                 0949 ;;-----------------------------------------------------
                      --
                 0950 ;;  SET HL POINT EXTENT NUMBER OF FCB.
                      =
                 0951 ;;-----------------------------------------------------
                      --
E1AD 2A62E0      0952 PNTEXT: LD      HL,(SENT2B)     ;HL POINT FCB ADDRESS
E1B0 110C00      0953         LD      DE,000CH        ;OFFSET TO EXTENT#
E1B3 19          0954         ADD     HL,DE           ;HL POINT EXTENT#
E1B4 C9          0955         RET
                 0956 ;;-----------------------------------------------------
                      --
                 0957 ;;  SET DE POINT RECORD COUNT (RC)
                      =
                 0958 ;;   AND HL POINT CURRENT RECORD (CR) OF FCB.
                      =
                 0959 ;;-----------------------------------------------------
                      --
E1B5 2A62E0      0960 PNTREC: LD      HL,(SENT2B)     ;HL POINT FCB
```

```
E1B8  11@F@@      @961          LD      DE,@@@FH        ;OFFSET TO RECCNT
E1BB  19          @962          ADD     HL,DE
E1BC  EB          @963          EX      DE,HL           ;<1> SET DE POINT RECCN
                                T
E1BD  211@@@      @964          LD      HL,@@11H        ;OFFSET FROM RECCNT TO
                                CURREC
E1C@  19          @965          ADD     HL,DE           ;<2> SET HL POINT CURRE
                                C.
E1C1  C9          @966          RET
                  @967 ;;-------------------------------------------------
                  --
                  @968 ;;  SAVE RECORD COUNT, EXTENT COUNT AND CURRENT RECORD.
                  =
                  @969 ;;-------------------------------------------------
                  --
E1C2  CD85E1      @97@ SAVREC: CALL   PNTREC          ;SET DE POINT RECCNT,HL
                   POINT CURREC
E1C5  7E          @971          LD      A,(HL)          ;<1> GET CURREC
E1C6  32A@EA      @972          LD      (CURREC),A      ;      AND SAVE
E1C9  EB          @973          EX      DE,HL
E1CA  7E          @974          LD      A,(HL)          ;<2> GET RECCNT
E1CB  329EEA      @975          LD      (RECCNT),A      ;      AND SAVE
E1CE  CDADE1      @976          CALL    PNTEXT          ;SET HL POINT EXT#
E1D1  3A82EA      @977          LD      A,(EXM)         ;GET EXTENT MASK
E1D4  A6          @978          AND     (HL)            ;  MASK WITH EXT#
E1D5  329FEA      @979          LD      (EXTCNT),A      ;<3> AND SAVE EXT#.
E1D8  C9          @98@          RET
                  @981 ;;-------------------------------------------------
                  --
                  @982 ;;  UPDATE CURRENT RECORD AND RECORD COUNT.
                  =
                  @983 ;;-------------------------------------------------
                  --
                  @984 ;<1> CHECK UPDATE FACTOR (@@=RANDOM/@1=SEQ./@2=RANDOM W
                  . ZERO FILL)
                  @985 ;  IF UPDATE FACTOR IS RANDOM OR SEQ., FACTOR IS CORREC
                  T
                  @986 ;   BUT IF UPDATE FACTOR IS RANDOM WITH ZERO FILL, SET
                  FACTOR = @@.
E1D9  CD85E1      @987 UPDREC: CALL   PNTREC          ;SET DE POINT RECCNT,HL
                   POINT CURREC
E1DC  3A92EA      @988          LD      A,(UPFACT)      ;CHECK UPDATE FACTOR
E1DF  FE@2        @989          CP      @2H             ;  IF RANDOM WITH ZERO
                  FILL
E1E1  2@@1        @99@          JR      NZ,ADDFAC       ;NO, UPDATE FACTOR IS C
                  ORRECT
E1E3  AF          @991          XOR     A               ;  ELSE SET UPDATE FACT
                  OR = @@
                  @992 ;<2> UPDATE CURREC BY SKIP TO NEXT RECORD IF SEQ. OR NO
                  T SKIP
                  @993 ;  IF RANDOM, SAVE UPDATE CURREC AND RECCNT.
E1E4  4F          @994 ADDFAC: LD     C,A             ;GET UPDATE FACTOR
E1E5  3AA@EA      @995          LD      A,(CURREC)      ;GET CURREC
E1E8  81          @996          ADD     A,C             ;UPDATE CURREC
E1E9  77          @997          LD      (HL),A          ;  AND SAVE
E1EA  EB          @998          EX      DE,HL           ;HL POINT RECCNT
E1EB  3A9EEA      @999          LD      A,(RECCNT)      ;GET UPDATE RECCNT
```

```
E1EE  77        1000        LD      (HL),A          ; AND SAVE.
E1EF  C9        1001        RET
                1002 ;;-----------------------------------------------------
                    --
                1003 ;;  SHIFT RIGHT OF HL, #BIT TO SHIFT IN C.
                        =
                1004 ;;-----------------------------------------------------
                    --
E1F0  0C        1005 SFTRHT: INC    C       ;          H          L
E1F1  0D        1006 LOPRHT: DEC    C       ;  +-- XXXX XXXX  XXXX XXXX
E1F2  C8        1007        RET     Z       ;  +-> 0XXX XXXX  XXXX XXXX
E1F3  7C        1008        LD      A,H     ;GET HIGH BYTE
E1F4  B7        1009        OR      A       ;CLEAR CARRY TO ZERO     H
E1F5  1F        1010        RRA             ;ROTATE RIGHT    0-> XXXX XXXX
                    ->CY
E1F6  67        1011        LD      H,A     ;SAVE ROTATED HIGH BYTE
E1F7  7D        1012        LD      A,L     ;GET LOW BYTE            L
E1F8  1F        1013        RRA             ;ROTATE RIGHT  CY-> XXXX XXXX
E1F9  6F        1014        LD      L,A     ;SAVE ROTATED LOW BYTE
E1FA  18F5      1015        JR      LOPRHT  ;  LOOP # OF TIME IN C.
                1016 ;;;-----------------------------------------------------
                    --
                1017 ;;;  FIND CHECKSUM VECTOR OF DIRECTORY ENTRY 1 SECTOR
                        =
                1018 ;;; IN DIRECTORY BUFFER. RETURN CHECKSUM IN A.
                        =
                1019 ;;;-----------------------------------------------------
                    --
E1FC  0E80      1020 FNDCSV: LD     C,80H           ;TO FIND CHECKSUM OF 12
                    8 BYTES
E1FE  2A76EA    1021        LD      HL,(ADDDI3)     ;   IN DIRBUF
E201  AF        1022        XOR     A               ;CLEAR RESULT FOR INIT
E202  86        1023 SUMCSV: ADD    A,(HL)          ;ADD ALL BYTES IN A
E203  23        1024        INC     HL
E204  0D        1025        DEC     C
E205  20FB      1026        JR      NZ,SUMCSV
E207  C9        1027        RET                     ;  AND RETURN CHECKSUM
                    IN A.
                1028 ;;-----------------------------------------------------
                    --
                1029 ;;  SHIFT LEFT OF HL, #BIT TO SHIFT IN C.
                        =
                1030 ;;-----------------------------------------------------
                    --
E208  0C        1031 SFTLFT: INC    C       ;FIRST INCRE TO SET FLAG
E209  0D        1032 LOPLFT: DEC    C       ;LOOP SHIFT LEFT       H
                      L
E20A  C8        1033        RET     Z       ;  SHIFT LEFT BY   0000 0000  0
                    000 0001 --+
E20B  29        1034        ADD     HL,HL   ;  ADD ITSELF      0000 0000  0
                    000 0010 <-+
E20C  18F9      1035        JR      LOPLFT
                1036 ;;;-----------------------------------------------------
                    --
                1037 ;;;  MASK BIT OF CDRIVE IN R/O VECTOR OR LOGIN VECTOR.
                        =
                1038 ;;;  OLD VECTOR SENT IN HL, AND NEW VECTOR RETURN IN HL
```

```
                        .=
                    1039 ;;;-----------------------------------------------
                        --
E20E  C5            1040 MSKVEC: PUSH    BC              ;(*)SAVE SENT R/OVEC OR
                         LOGVEC
E20F  3A5FE0        1041         LD      A,(CDRIVE)      ;GET CURRENT DRIVE CODE
E212  4F            1042         LD      C,A             ;  MOVE IN C TO COUNT S
                        HIFT LEFT
E213  210100        1043         LD      HL,0001H        ;PREPARE MASK BIT
E216  CD38E2        1044         CALL    SFTLFT          ;SHIFT MASK BIT TO POSI
                        TION OF CDRIVE
E219  C1            1045         POP     BC              ;(*)RESTORE SENT VECTOR
E21A  79            1046         LD      A,C
E21B  B5            1047         OR      L               ;     TO MERGE
E21C  6F            1048         LD      L,A
E21D  78            1049         LD      A,B
E21E  B4            1050         OR      H               ;     WITH MASK BIT OF
                        CDRIVE
E21F  67            1051         LD      H,A             ;  THEN RETURN NEW VECT
                        OR IN HL.
E220  C9            1052         RET
                    1053 ;;-----------------------------------------------
                        --
                    1054 ;;  CHECK BIT OF READ/ONLY VECTOR.
                        =
                    1055 ;;-----------------------------------------------
                        --
E221  2A6AEA        1056 CHKROB: LD      HL,(ROVEC)      ;GET R/O VECTOR IN HL
E224  3A5FE0        1057         LD      A,(CDRIVE)      ;GET CURRENT DRIVE CODE
E227  4F            1058         LD      C,A             ;  IN C
E228  CDF0E1        1059         CALL    SFTRHT          ;TO SHIFT BIT OF CDRIVE
E22B  7D            1060         LD      A,L             ;  TO RIGHTMOST POSITIO
                        N
E22C  E601          1061         AND     01H             ;==>RET NZ IF SET DRIVE
                         READ/ONLY
E22E  C9            1062         RET                     ;==>RET Z IF NOT SET
                    1063 ;;;=======================================================
                        ==
                    1064 ;;;  FUNC28  WRITE PROTECT DISK.
                        =
                    1065 ;;;    MASK BIT OF CDRIVE IN R/O VECTOR,
                        =
                    1066 ;;;    SET ENTRY COUNT = DRM+1.
                        =
                    1067 ;;;=======================================================
                        ==
E22F  216AEA        1068 FUNC28: LD      HL,ROVEC        ;GET R/O VECTOR
E232  4E            1069         LD      C,(HL)
E233  23            1070         INC     HL
E234  46            1071         LD      B,(HL)          ;MOVE R/O VECTOR TO BC
E235  CD3EE2        1072         CALL    MSKVEC          ;GOTO MASK BIT OF CDRIV
                        E
E238  226AEA        1073         LD      (ROVEC),HL      ;  AND REPLACE THE OLD
E23B  2A8EEA        1074         LD      HL,(DRM)        ;GET DRM
E23E  23            1075         INC     HL              ;  INCRE DRM
E23F  EB            1076         EX      DE,HL
E240  2A73EA        1077         LD      HL,(ADDDWI)
```

```
E243 73          1078          LD      (HL),E
E244 23          1079          INC     HL
E245 72          1080          LD      (HL),D          ;  AND PUT IN ENTRY COU
                     NT (DW1)
E246 C9          1081          RET
                 1082 ;;-----------------------------------------------------
                      --
                 1083 ;;  CHECK FILE SET READ/ONLY.
                           =
                 1084 ;;-----------------------------------------------------
                      --
E247 CD58E2      1085 CHKFRO: CALL    DIRETY          ;SET HL POINT DIRECTORY
                      ENTRY
E24A 110900      1086 DATFRO: LD      DE,0009H        ;OFFSET TO BYTE T1 OF F
                     DE
E24D 19          1087          ADD     HL,DE           ;FORM POINTER TO BYTE T
                     1
E24E 7E          1088          LD      A,(HL)          ;  AND GET THAT BYTE
E24F 17          1089          RLA                     ;IF BIT T1 IS 'OFF'
E250 D0          1090          RET     NC              ;NOT SET ATTRIBUTE...RE
                     TURN
E251 C3AEDD      1091          JP      PINER3          ;  ELSE PRINT 'FILE R/O
                     '******
                 1092 ;;-----------------------------------------------------
                      --
                 1093 ;;  CHECK DRIVE SET READ/ONLY.
                           =
                 1094 ;;-----------------------------------------------------
                      --
E254 CD21E2      1095 CHKDRO: CALL    CHKRCB          ;GOTO CHECK R/OVED OF D
                     DRIVE
E257 C8          1096          RET     Z               ;RET Z...NOT SET R/O
E258 C3B9DD      1097          JP      PINER4          ;  ELSE PRINT 'R/O' ***
                     **
                 1098 ;;-----------------------------------------------------
                      --
                 1099 ;;  SET HL POINT FDE IN DIRECTORY BUFFER.
                           =
                 1100 ;;-----------------------------------------------------
                      --
E25B 2A76EA      1101 DIRETY: LD      HL,(ADDDIB)     ;ADDRESS OF DIRBUF IN H
                     L
E25E 3AA8EA      1102          LD      A,(OFFLEN)      ;LENGTH OF OFFSET IN A
                 1103 ;; COMMON POINT => SET HL POINT DM OF FDE.
                 1104 ;;    CONDITION: HL = ADDRESS OF DM0, A = POSITION OF DM
                      ,
E261 85          1105 FNTDM:  ADD     A,L             ;ADD ADDRESS OF BASE (I
                     N HL)
E262 6F          1106          LD      L,A             ;  WITH LENGTH OF OFFSE
                     T (IN A)
E263 D0          1107          RET     NC
E264 24          1108          INC     H               ;  AND SAVE POINTER IN
                     HL.
E265 C9          1109          RET
                 1110 ;;-----------------------------------------------------
                      --
                 1111 ;;  GET BYTE S2 OF FDE IN A.
```

```
                                =
                        1112 ;;-----------------------------------------------
                             --
E266  2A62E0            1113 GETS2:  LD      HL,(SENT23)     ;HL POINT FCB ADDRESS
E269  110E00            1114         LD      DE,000EH        ;OFFSET TO BYTE S2
E26C  19                1115         ADD     HL,DE           ;HL POINT S2
E26D  7E                1116         LD      A,(HL)          ;  AND RETURN S2 IN A.
E26E  C9                1117         RET
                        1118 ;;-----------------------------------------------
                             --
                        1119 ;;  CLEAR BYTE S2 OF FCB TO ZERO.
                                =
                        1120 ;;-----------------------------------------------
                             --
E26F  CD66E2            1121 CLRS2:  CALL    GETS2           ;GOTO SET HL POINT S2
E272  3600              1122         LD      (HL),00H        ;CLEAR S2 WITH ZERO
E274  C9                1123         RET
                        1124 ;;-----------------------------------------------
                             --
                        1125 ;;  SET HIGH BIT OF S2 OF FCB.
                                =
                        1126 ;;-----------------------------------------------
                             --
E275  CD66E2            1127 SETS2:  CALL    GETS2           ;GOTO SET HL POINT S2
E278  F680              1128         OR      80H             ;SET HIGH BIT TO 'ON'
E27A  77                1129         LD      (HL),A          ;  AND SAVE.
E27B  C9                1130         RET
                        1131 ;;-----------------------------------------------
                             --
                        1132 ;;  COMPARE DIRCNT WITH ETYCNT.
                                =
                        1133 ;;    RETURN BY: DIRCNT<ETYCNT ==> FLAG CARRY
                                =
                        1134 ;;             : DIRCNT>=ETYCNT ==> FLAG NOT CARRY
                                =
                        1135 ;;-----------------------------------------------
                             --
E27C  2AA7EA            1136 COMPAR: LD      HL,(DIRCTL)     ;GET DIRCNT HIGH & LOW
                             BYTE
E27F  EB                1137         EX      DE,HL
E280  2A78EA            1138         LD      HL,(ADDDW1)     ;GET ETYCNT
E283  7B                1139         LD      A,E
E284  96                1140         SUB     (HL)
E285  23                1141         INC     HL
E286  7A                1142         LD      A,D             ;COMPARE DIRCNT
E287  9E                1143         SBC     A,(HL)          ;  WITH ETYCNT.
E288  C9                1144         RET
                        1145 ;;-----------------------------------------------
                             --
                        1146 ;;  ADJUST ENTRY COUNT TO ALWAYS MORE THAN DIRECTORY
                                =
                        1147 ;;   ENTRY COUNT.
                                =
                        1148 ;;-----------------------------------------------
                             --
E289  CD7CE2            1149 ADJETY: CALL    COMPAR          ;COMPARE DIRCNT WITH ET
                             YCNT
```

```
E28C  D8          1150          RET     C               ;DIRCNT<ETYCNT...RETURN
E28D  13          1151          INC     DE              ;ELSE SET ETYCNT
E28E  72          1152          LD      (HL),D          ;  TO MORE THAN
E28F  2B          1153          DEC     HL              ;   DIRCNT BY 1
E290  73          1154          LD      (HL),E          ; (ETYCNT=DIRCNT+1)
E291  C9          1155          RET
                  1156 ;;-------------------------------------------------
                       ---
                  1157 ;; SUBTRACT DE WITH HL, RESULT KEEP IN HL(HL = DE - HL)
                       . =
                  1158 ;;-------------------------------------------------
                       ---
E292  79          1159 SUBTAC: LD      A,E     ;DE SUBTRACT
E293  95          1160          SUB     L       ;  WITH HL
E294  6F          1161          LD      L,A
E295  7A          1162          LD      A,D
E296  9C          1163          SBC     A,H
E297  67          1164          LD      H,A     ; RESULT IN HL.
E298  C9          1165          RET
                  1166 ;;;=============================================
                       ==
                  1167 ;;;  WRITE CHECKSUM VECTOR OF DIRECTORY ENTRY.
                               =
                  1168 ;;;=============================================
                       ==
E299  0EFF        1169 CSVDIR: LD      C,0FFH          ;SET WRTTYP=DIR
E29B  2AA9EA      1170 CALCSV: LD      HL,(DIRSEC)     ;GET DIRSEC (IN DE)
E29E  EB          1171          EX      DE,HL
E29F  2A89EA      1172          LD      HL,(CKS)        ; TO COMPARE WITH CKS (
                       IN HL)
E2A2  CD92E2      1173          CALL    SUBTAC
E2A5  D0          1174          RET     NC              ;DIRSEC>CKS,NOT DIR MOD
                       E...RETURN
E2A6  C5          1175          PUSH    BC              ;SAVE WRTTYP
E2A7  CDFCE1      1176          CALL    FNDCSV          ;GOTO FIND CHECKSUM IN
                       A
E2AA  2A7AEA      1177          LD      HL,(ADDCSV)     ;GET ADDRESS OF CSV
E2AD  EB          1178          EX      DE,HL
E2AE  2AA9EA      1179          LD      HL,(DIRSEC)     ;USE DIRSEC AS OFFSET
E2B1  19          1180          ADD     HL,DE           ;TO FORM POSITION OF SE
                       CTOR IN CSV
E2B2  C1          1181          POP     BC              ;RESTORE WRTTYP
E2B3  0C          1182          INC     C
E2B4  280A        1183          JR      Z,WRTCSV        ;WRTTYP=DIR,WRITE CHECK
                       SUM
                  1184 ;;;
                  1185 ;;;  IN PROGRAM MODE, CHECK CHECKSUM IN CSV WITH CURREN
                       T CHECKSUM.
E2B6  BE          1186          CP      (HL)            ;COMPARE CHECKSUM
E2B7  C8          1187          RET     Z               ;EQUAL... CAN DO WRITE
                       DISK
                  1188 ;;;
                  1189 ;;;  CHECKSUM NOT EQUAL, CHECK DIRCNT.
E2B8  CD7CE2      1190          CALL    COMPAR          ;COMPARE DIRCNT WITH ET
                       YCNT
E2BB  D0          1191          RET     NC              ;DIRCNT>=ETYCNT, OK.
E2BC  CD2FE2      1192          CALL    FUNC28          ;DIRCNT<ETYCNT, SET DRI
```

```
                        VE READ/ONLY
E2BF C9         1193            RET                     ;  AND SET ETYCNT=DRM+1
                1194 ;;;
E2C0 77         1195 WRTCSV: LD      (HL),A             ;WRITE CHECKSUM IN CSV.
E2C1 C9         1196            RET
                1197 ;;;=================================================
                        ==
                1198 ;;;  WRITE DIRECTORY ENTRY BACK ON DISK
                                =
                1199 ;;;  WRITE NEW CHECKSUM (FROM CHANGE CONTENT OF FDE)
                                =
                1200 ;;;  IN CSV TO PREVENT DRIVE SET R/O.  SET DMA AT
                                =
                1201 ;;;  DIRECTORY BUFFER AND WRITE THIS SECTOR BACK ON
                                =
                1202 ;;;  DISK (TRACK AND SECTOR ARE STILL NOT CHANGED).
                                =
                1203 ;;;=================================================
                        ==
E2C2 CD99E2     1204 DIRDSK: CALL    CSVDIR             ;WRITE NEW CHECKSUM IN
                        CSV
E2C5 CDDAE2     1205            CALL    SETDIB          ;SET DMA AT DIRECTORY B
                        UFFER
E2C8 0E01       1206            LD      C,01H           ;SET WRTTYP FOR I/O
E2CA CDCCE0     1207            CALL    WDISK           ;  WRITE THIS SECTOR BA
                        CK ON DISK
E2CD 1806       1208            JR      SETADD          ;SET DMA BACK TO 80H
                1209 ;;;=================================================
                        ==
                1210 ;;;  READ DIRECTORY ENTRY FROM DISK.
                                =
                1211 ;;;=================================================
                        ==
E2CF CDDAE2     1212 READIR: CALL    SETDIB             ;SET DMA AT DIRECTORY B
                        UFFER
E2D2 CDD7E0     1213            CALL    RDISK           ;READ FDE 1 SECTOR
                1214 ;;-------------------------------------------------
                        --
                1215 ;;  SET DMA ADDRESS AT 80H.
                                =
                1216 ;;-------------------------------------------------
                        --
E2D5 2168EA     1217 SETADD: LD      HL,DOSDMA
E2D8 1803       1218            JR      DOSET
                1219 ;;-------------------------------------------------
                        --
                1220 ;;  SET DMA ADDRESS AT DIRECTORY BUFFER.
                                =
                1221 ;;-------------------------------------------------
                        --
E2DA 2174EA     1222 SETDIB: LD      HL,ADDDIB
E2DD 4E         1223 DOSET:  LD      C,(HL)
E2DE 23         1224            INC     HL
E2DF 46         1225            LD      B,(HL)
E2E0 C324E3     1226            JP      SETDMA
                1227 ;;-------------------------------------------------
                        --
```

```
                        1228 ;;  MOVE DIRECTORY ENTRY 1 SECTOR FROM DIRECTORY BUFFER
                                 =
                        1229 ;; TO 80H.
                                 =
                        1230 ;;-----------------------------------------------------
                             --
E2E3  2A76EA            1231 MOVETY: LD      HL,(ADDDI3)
E2E6  EB                1232         EX      DE,HL
E2E7  2A6EEA            1233         LD      HL,(DDSDMA)
E2EA  0E80              1234         LD      C,80H
E2EC  C366E0            1235         JP      MOVER
                        1236 ;;;---------------------------------------------------
                             --
                        1237 ;;;  CHECK IF DIRECTORY END.
                                 =
                        1238 ;;;     RET Z => DIRECTORY END
                                 =
                        1239 ;;;     RET NZ => NOT END.
                                 =
                        1240 ;;;---------------------------------------------------
                             --
E2EF  21A7EA            1241 CHKDIR: LD      HL,DIRCTL       ;GET DIRCNT LOW BYTE
E2F2  7E                1242         LD      A,(HL)
E2F3  23                1243         INC     HL              ; COMPARE WITH DIRCNT H
                          IGH BYTE
E2F4  BE                1244         CP      (HL)
E2F5  C0                1245         RET     NZ              ;NOT FFFF, DIRECTORY NO
                          T END
E2F6  3C                1246         INC     A               ; ELSE INCRE FROM FF TO
                           00
E2F7  C9                1247         RET                     ;  FOR RET Z, DIRECTORY
                          END.
                        1248 ;;;---------------------------------------------------
                             --
                        1249 ;;;   INITIALIZE DIRECTORY ENTRY COUNT OR MARK END OF
                                 =
                        1250 ;;;  DIRECTORY BY SET DIRCNT EQUAL FFFF TO INCREMENT
                                 =
                        1251 ;;;  TO 0000.
                                 =
                        1252 ;;;---------------------------------------------------
                             --
E2F8  21FFFF            1253 INTDIR: LD      HL,0FFFFH       ;SET DIRCNT = FFFF
E2FB  22A7EA            1254         LD      (DIRCTL),HL     ;  FOR INIT DIRCNT
E2FE  C9                1255         RET                     ;   OR MARK END OF DIRE
                          CTORY.
                        1256 ;;;===================================================
                             ==
                        1257 ;;;  GET NEXT DIRECTORY ENTRY.
                                 =
                        1258 ;;;    INCREASE DIRCNT TO NEXT ENTRY AND CHECK
                                 =
                        1259 ;;;    IF DIRCNT<=DRM, OK.  ELSE DIRECTORY FULL.
                                 =
                        1260 ;;;===================================================
                             ==
E2FF  2A85EA            1261 GETDIR: LD      HL,(DRM)        ;GET DRM
```

```
E302 EB          1262        EX      DE,HL           ;  KEEP IN DE
E303 2AA7EA      1263        LD      HL,(DIRCTL)     ;INCRE DIRCNT
E306 23          1264        INC     HL              ;  KEEP IN HL
E307 22A7EA      1265        LD      (DIRCTL),HL     ;AND SAVE IF OK.
E30A CD92E2      1266        CALL    SUBTAC          ;COMPARE DRM WITH NEW D
                 IRCNT
E30D 3002        1267        JR      NC,INDRM        ;DRM>DIRCNT,DIRECTORY S
                 TILL AVAILABLE
E30F 18E7        1268        JR      INTDIR          ;  ELSE SET DIRCNT=FFFF
                  FOR DIR END.
                 1269 ;;;
                 1270 ;;;  MOD DIRCNT IN RANGE 0-3 TO FIND OFFSET LENGTH BY
                 1271 ;;;  MULTIPLE WITH 32.  THEN CHECK IF ENTRY 0, IT IS
                 1272 ;;;  FIRST ENTRY OF SECTOR, SO READ NEXT PHYSICAL SECTO
                 R
                 1273 ;;;  AND FILL CHECKSUM.
E311 3AA7EA      1274 INDRM: LD      A,(DIRCTL)      ;GET DIRCNT TO      +--
                 -+---+---+---+
E314 E603        1275        AND     03H             ; MOD IN RANGE 0-3  +--
                 -+---+---+---+
E316 0605        1276        LD      B,05H           ;                   0
                               1   2   3
E318 87          1277 FRMOFF: ADD     A,A             ;FROM OFFSET LENGTH
E319 05          1278        DEC     B               ;  OF THIS ENTRY
E31A 20FC        1279        JR      NZ,FRMOFF       ;   BY MULTIPLY WITH 32
E31C 32A6EA      1280        LD      (OFFLEN),A      ;   (LENGTH OF DIRECTOR
                 Y ENTRY)
E31F B7          1281        OR      A               ;CHECK IF ENTRY 0
E320 C0          1282        RET     NZ              ;NOT ENTRY 0... RETURN
                 1283 ;;;
                 1284 ;;;  ENTRY 0... READ A SECTOR OF FDE FROM DISK.
E321 C5          1285        PUSH    BC              ;SAVE WRTTYP IN C
E322 CD34E0      1286        CALL    TRKSEC          ;SET PHYSICAL TRACK & S
                 ECTOR
E325 CDCFE2      1287        CALL    REDDIR          ;READ A SECTOR OF FDE I
                 NTO DIRBUF
E328 C1          1288        POP     BC              ;RESTORE WRTTYP
E329 C39BE2      1289        JP      CALCSV          ;CALCULATE CHECKSUM AND
                  FILL IN CSV.
                 1290 ;;;-----------------------------------------------
                 --
                 1291 ;;;  MOVE ALV BIT OF ALLOCATION BLOCK TO RIGHTMOST BIT
                        =
                 1292 ;;;-----------------------------------------------
                 --
                 1293 ; MOD DM LOW IN RANGE 0-7 TO USE FOR ROTATE BIT IN
                 1294 ;  ALV BYTE. BUT FIRST MUST GET THAT BYTE BY DIVIDE
                 1295 ;  #BLOCK (FROM DM OF FDE) BY 8, THEN ADD WITH ADDRESS
                 1296 ;  OF ALV TO PICK UP BYTE.
E32C 79          1297 DIVBY8: LD      A,C             ;GET DM LOW
E32D E607        1298        AND     07H             ;MOD IN RANGE 0-7
E32F 3C          1299        INC     A               ;INCRE 1 BECAUSE BLOCK
                 START FROM 0
E330 5F          1300        LD      E,A             ;SAVE THIS VALUE TO ROT
                 ATE BIT (IN E)
E331 57          1301        LD      D,A             ;  AND TO ROTATE BACK (
                 IN D)
```

```
                        1302 ;<1> DIVIDE DM LOW BY 8
E332  79                1303        LD      A,C             ;GET DM LOW
E333  0F                1304        RRCA
E334  0F                1305        RRCA
E335  0F                1306        RRCA                    ;DIVIDE BY 8
E336  E61F              1307        AND     1FH             ;STRIP OUT UNWANT BIT
E338  4F                1308        LD      C,A             ;(*)DM LOW DIVIDE BY 8
                            IN C
                        1309 ;<2> ADD REMAINDER FROM DIVIDE DM HIGH TO LOW RESULT
E339  78                1310        LD      A,B             ;GET DM HIGH
E33A  87                1311        ADD     A,A
E33B  87                1312        ADD     A,A
E33C  87                1313        ADD     A,A
E33D  87                1314        ADD     A,A             ;FIND REMAINDER OF
E33E  87                1315        ADD     A,A             ;  DIVIDE DM HIGH BY 8
E33F  B1                1316        OR      C               ;(*)CARRY TO DM LOW
E340  4F                1317        LD      C,A             ;  DIVIDE BY 8 BY MERGE
                        1318 ;<3> DIVIDE DM HIGH BY 8
E341  78                1319        LD      A,B             ;GET DM HIGH
E342  0F                1320        RRCA
E343  0F                1321        RRCA
E344  0F                1322        RRCA                    ;DIVIDE BY 8
E345  E61F              1323        AND     1FH             ;STRIP OUT UNWANT BIT
E347  47                1324        LD      B,A             ;DM HIGH DIVIDE BY 8 IN
                            B
E348  2A7CEA            1325        LD      HL,(ADDRALV)    ;GET ADDRESS OF ALV
E34B  09                1326        ADD     HL,BC           ;OFFSET TO BYTE OF THIS
                            BLOCK
E34C  7E                1327        LD      A,(HL)          ;PICK UP THAT ALV BYTE
                            IN A
                        1328 ;
                        1329 ;  ROTATE ALV BIT OF BLOCK TO RIGHTMOST OF A.
E34D  07                1330 RHTMST: RLCA
E34E  1D                1331        DEC     E
E34F  20FC              1332        JR      NZ,RHTMST
E351  C9                1333        RET
                        1334 ;;;=================================================
                            ==
                        1335 ;;;  ALLOCATE/DEALLOCATE DISK SPACE.
                                =
                        1336 ;;;   CONDITION: MASK BIT IN E WITH 1 FOR ALLOCATE
                                =
                        1337 ;;;                                0 FOR DEALLOCATE.
                                =
                        1338 ;;;=================================================
                            ==
E352  D5                1339 ALVBIT: PUSH   DE              ;SAVE MASK BIT IN E
E353  CD2CE3            1340        CALL    DIVBY8          ;ALV BIT OF BLOCK IN RI
                            GHTMOST
E356  E6FE              1341        AND     0FEH            ;CLEAR ALV BIT FOR NEW
                            SET
E358  C1                1342        POP     BC              ;GET MASK BIT
E359  B1                1343        OR      C               ;  TO SET ALV BIT
                        1344 ;
                        1345 ;  ROTATE NEW SET ALV BIT BACK TO OLD POSITION AND SAVE
                            IN ALV.
E35A  0F                1346 SAVALV: RRCA
```

```
E35B 15          1347        DEC     D           ;(# OF ROTATE SET FROM
                     DIVBY8)
E35C 20FC        1348        JR      NZ,SAVALV   ;ROTATE BACK
E35E 77          1349        LD      (HL),A      ;  AND SAVE
E35F C9          1350        RET
                 1351 ;;;=================================================
                     ==
                 1352 ;;;  ALLOCATE/DEALLOCATE FILE ON DM OF FDE.
                          =
                 1353 ;;;   CONDITION: MASK BIT IN C WITH 1 FOR ALLOCATE
                          =
                 1354 ;;;                              0 FOR DEALLOCATE
                          =
                 1355 ;;;=================================================
                     ==
                 1356 ;(1) SET POINTER TO FIRST DM OF FDE, SET COUNTER FOR
                 1357 ;  16 BYTES OF DM.
E360 CD5BE2      1358 MSKALV: CALL    DIRETY      ;SET HL POINT FDE
E363 111000      1359        LD      DE,0010H    ;OFFSET TO DM OF FDE
E366 19          1360        ADD     HL,DE       ;START FROM FIRST DM
E367 C5          1361        PUSH    BC          ;SAVE MASK BIT IN C
E368 0E11        1362        LD      C,11H       ;COUNTER TO MASK ALV FO
                     R DM 16 BYTES
                 1363 ;(2) LOOP FOR 16 BYTES OF DM.
E36A D1          1364 LOPMSK: POP     DE          ;MASK BIT IN E
E36B 0D          1365        DEC     C           ;(%)DECRE COUNTER
E36C C8          1366        RET     Z           ;MASK ALV FOR ALL DM...
                         RETURN
E36D D5          1367        PUSH    DE          ;SAVE MASK BIT
E36E 3A9AEA      1368        LD      A,(DMLEN)   ;CHECK LENGTH OF DM
E371 B7          1369        OR      A
E372 2807        1370        JR      Z,DM2B      ;DM LENGTH 2 BYTES
                 1371 ;(3) DM LENGTH 1 BYTE, SET HIGH BYTE TO 00.
E374 C5          1372        PUSH    BC          ;SAVE COUNTER
E375 E5          1373        PUSH    HL          ;  AND POINTER OF DM
E376 4E          1374        LD      C,(HL)      ;DM LENGTH 1 BYTE
E377 0600        1375        LD      B,00H       ;  SO SET IN BC = 00XX
E379 1806        1376        JR      NULLDM      ;GOTO MASK ALV BIT
                 1377 ;(4) DM LENGTH 2 BYTES, USE LOW DM AND HIGH DM.
E37B 0D          1378 DM2B:   DEC     C           ;(%)DECRE COUNTER FOR H
                     IGH DM
E37C C5          1379        PUSH    BC          ;  AND SAVE COUNTER
E37D 4E          1380        LD      C,(HL)      ;GET DM LOW IN C
E37E 23          1381        INC     HL
E37F 46          1382        LD      B,(HL)      ;GET DM HIGH IN B, SO B
                     C = XXXX
E380 E5          1383        PUSH    HL          ;  AND SAVE POINTER
                 1384 ;(5) CHECK IF NULL DM (DM = 0000), NOT USE DISK SPACE,
                     NOT SET ALV.
                 1385 ;     ELSE SET ALV BIT OF THIS ALLOCATION BLOCK.
E381 79          1386 NULLDM: LD      A,C         ;IF DM LOW
E382 B0          1387        OR      B           ;  AND DM HIGH
E383 280A        1388        JR      Z,NEXTDM    ;  IS 0000, NOT SET ALV
                         .. DO NEXT DM
E385 2A83EA      1389        LD      HL,(DSM)    ;ELSE GET DSM
E388 7D          1390        LD      A,L
E389 91          1391        SUB     C
```

```
E38A 7C          1392        LD      A,H
E38B 98          1393        SBC     A,B             ;  TO CHECK WITH DM
E38C D452E3      1394        CALL    NC,ALVBIT       ;DSM>DM.. OK, MASK ALV
                     BIT
                 1395 ;(6) SKIP POINTER TO NEXT DM.
E38F E1          1396 NEXTDM: POP     HL              ;RESTORE POINTER
E390 23          1397        INC     HL              ;  SKIP TO NEXT DM
E391 C1          1398        POP     BC              ;RESTORE COUNTER
E392 1806        1399        JR      LOPMSK          ;DO NEXT DM.
                 1400 ;;;=====================================================
                     ==
                 1401 ;;;  CREATE CSV AND ALV OF ALL DIRECTORY ENTRIES.
                          =
                 1402 ;;;=====================================================
                     ==
                 1403 ;(1) GET # OF BYTE TO CLEAR ALV FROM DSM/8 + 1.
E394 2A83EA      1404 CSVALV: LD      HL,(DSM)        ;GET DSM
E397 0E03        1405        LD      C,03H           ;TO FIND # OF BYTE TO C
                     LEAR ALV
E399 CDF0E1      1406        CALL    SFTRHT          ;  BY DIVIDE DSM BY 8
E39C 23          1407        INC     HL              ; AND ADD 1 FOR REMAIND
                     ER
E39D 44          1408        LD      B,H
E39E 4D          1409        LD      C,L             ;COUNTER TO CLEAR IN BC
                 1410 ;(2) CLEAR ALL ALV WITH ZERO.
E39F 2A7DEA      1411        LD      HL,(ADDALV)     ;GET ADDRESS OF ALV
E3A2 3600        1412 CLRALV: LD      (HL),00H        ; TO CLEAR ALV WITH 00.
E3A4 23          1413        INC     HL
E3A5 0B          1414        DEC     BC
E3A6 78          1415        LD      A,B
E3A7 B1          1416        OR      C
E3A8 20F8        1417        JR      NZ,CLRALV       ;LOOP CLEAR ALL ALV.
                 1418 ;(3) SET ALV FOR RESERVED DIRECTORY AREA FROM AL0 & AL1
                          .
                 1419 ;  START AT TRACK 0, INITIALIZE TRKCNT AND SECCNT.
E3AA 2A87EA      1420        LD      HL,(AL0)        ;GET AL0 & AL1
E3AD EB          1421        EX      DE,HL           ; IN DE
E3AE 2A7DEA      1422        LD      HL,(ADDALV)     ;GET ADDRESS OF ALV
E3B1 73          1423        LD      (HL),E
E3B2 23          1424        INC     HL
E3B3 72          1425        LD      (HL),D          ;  TO SET AL0 & AL1
E3B4 CD36E0      1426        CALL    INTHCM          ;SET TRACK 0, CLEAR TRK
                     CNT & SECCNT
                 1427 ;(4) INITIALIZE DIRCNT AND SET ETYCNT EQUAL 3.
E3B7 2A79EA      1428        LD      HL,(ADDDIM1)    ;GET ADDRESS OF DM1
E3BA 3603        1429        LD      (HL),03H
E3BC 23          1430        INC     HL
E3BD 3600        1431        LD      (HL),00H        ;  TO ETYCNT = 3
E3BF CDFBE2      1432        CALL    INTDIR          ;INIT DIRCNT = FFFF FOR
                     START
                 1433 ;(5) READ 1 SECTOR OF FDE, FIND CHECKSUM TO FILL IN CSV
                      (IF ENTRY 0).
E3C2 0EFF        1434 LOPETY: LD      C,0FFH          ;SET WRITE2=DIR
E3C4 CDFFE2      1435        CALL    GETDIR          ;READ 1 SECTOR OF FDE
E3C7 CDEFE2      1436        CALL    CHKDIR          ;CHECK IF DIRECTORY END
E3CA C8          1437        RET     Z               ;DIR END, ALL DONE.. RE
                     TURN
```

```
E3CB  CD5BE2      1438          CALL    DIRETY          ;SET HL POINT DIRECTORY
                                ENTRY
                  1439 ;(6) IF DELETED FDE, NOT SET ALLOCATE DISK SPACE
E3CE  3EE5        1440          LD      A,0E5H          ;CHECK IF FILE DELETED
E3D0  BE          1441          CP      (HL)
E3D1  28EF        1442          JR      Z,LOPETY        ;YES, FILE DELETE.. NOT
                                SET ALV
E3D3  3A5EE0      1443          LD      A,(USRCOD)      ;CHECK IF CURRENT USER
E3D6  BE          1444          CP      (HL)
E3D7  200A        1445          JR      NZ,WRTALV       ;NO, GOTO WRITE ALV
E3D9  23          1446          INC     HL              ;CURRENT USER
E3DA  7E          1447          LD      A,(HL)          ;  GET FIRST FILENAME
E3DB  D624        1448          SUB     '$'             ;  TO CHECK IF TEMP FIL
                                E
E3DD  2004        1449          JR      NZ,WRTALV       ;NOT TEMP FILE, GOTO WR
                                ITE ALV
E3DF  3D          1450          DEC     A               ;TEMP FILE
E3E0  3264E0      1451          LD      (EXTVAL),A      ;SET EXIT VALUE
                  1452 ;(7) SET ALLOCATE DISK SPACE FOR DM IN UNDELETED FDE.
E3E3  0E01        1453 WRTALV:  LD      C,01H           ;SET MASK BIT FOR ALLOC
                                ATE
E3E5  CD60E3      1454          CALL    MSKALV          ;MASK ALV BIT OF THIS F
                                DE
E3E8  CD89E2      1455          CALL    ADJETY          ;GOTO ADJUST ETYENT
E3EB  18D5        1456          JR      LOPETY          ;  LOOP FOR NEXT FDE.
                  1457 ;;-----------------------------------------------
                                --
                  1458 ;;  SET EXIT VALUE FOR SEARCH DIRECTORY ENTRY.
                                =
                  1459 ;;     MARKER = FF MEANS SEARCH FIRST AND NOT FOUND
                                =
                  1460 ;;     MARKER = 00 MEANS SEARCH FIRST FOUND (BUT SEARCH
                                =
                  1461 ;;               NEXT MAY BE FOUND OR NOT).
                                =
                  1462 ;;-----------------------------------------------
                                --
E3ED  3A91EA      1463 FFOR00:  LD      A,(MARKER)      ;SET EXIT VALUE FF=NOT
                                FOUND
E3F0  C319E3      1464          JP      EXIT13          ;  OR 00=OK.
                  1465 ;;-----------------------------------------------
                                --
                  1466 ; CHECK EXTENT NUMBER OF FDE.
                                =
                  1467 ;;-----------------------------------------------
                                --
E3F3  C5          1468 EXTMSK:  PUSH    BC
E3F4  F5          1469          PUSH    AF
E3F5  3A82EA      1470          LD      A,(EXM)
E3F8  2F          1471          CPL
E3F9  47          1472          LD      B,A
E3FA  79          1473          LD      A,C
E3FB  A0          1474          AND     B
E3FC  4F          1475          LD      C,A
E3FD  F1          1476          POP     AF
E3FE  A3          1477          AND     B
E3FF  91          1478          SUB     C
```

```
E400  E61F        1479          AND     1FH
E402  C1          1480          POP     BC
E403  C9          1481          RET
                  1482 ;;;=============================================
                       ==
                  1483 ;;;  SEARCH FIRST.
                            =
                  1484 ;;;     CONDITION: LENGTH OF CHAR TO COMPARE IN C.
                            =
                  1485 ;;;    SEARCH FIRST SEARCH FOR FIRST FDE WHICH MATCH
                            =
                  1486 ;;;    ASSIGN FCB.
                            =
                  1487 ;;;=============================================
                       ==
E404  3EFF        1488 SERFIR: LD       A,0FFH          ;SET MARKER = FF
E406  3291EA      1489          LD      (MARKER),A      ;  TO DECLEAR BEING SEA
                       RCH FIRST
E409  2195EA      1490          LD      HL,LENGER
E40C  71          1491          LD      (HL),C          ;SAVE LENGTH OF COMPARE
E40D  2A62E3      1492          LD      HL,(SENT2B)     ;  AND ADDRESS OF FCB
E410  2296EA      1493          LD      (FCBADD),HL     ;  TO USE AGAIN IN SEAR
                       CH NEXT.
                  1494 ;(1) SET TRACK 0, INITIALIZE DIRCNT, CLEAR TRKCNT AND S
                       ECCNT
                  1495 ;  TO SEARCH FROM FIRST FDE.
E413  CDF3E2      1496          CALL    INTDIR          ;INIT DIRCNT=FFFF FOR S
                       TART
E416  CD36E3      1497          CALL    INTHDM          ;TRACK 0, CLEAR TRKCNT
                       & SECCNT
                  1498 ;;;=============================================
                       ==
                  1499 ;;;  SEARCH NEXT.
                            =
                  1500 ;;;    IS IN LOOP OF SEARCH FIRST BY USE FCB ADDRESS
                            =
                  1501 ;;;    AND LENGTH OF COMPARE AS ASSIGN IN SEARCH FIRST.
                            =
                  1502 ;;;=============================================
                       ==
                  1503 ;(2) READ FDE 1 SECTOR (IF BEING ENTRY 0), IF DIRECTORY
                  1504 ;  END... GOTO SEARCH NOT FOUND.
E419  3E00        1505 SERNXT: LD       C,00H           ;SET C=0 (WRTTYP=PRO0)
E41B  CDFFE2      1506          CALL    GETDIR          ;READ FDE 1 SECTOR
E41E  CDEFE2      1507          CALL    CHKDIR          ;CHECK IF READ DIR END
E421  2852        1508          JR      Z,SERNFD        ;DIR END,SET DIRCOD=FF
                       (NOT FOUND)
E423  2A96EA      1509          LD      HL,(FCBADD)     ;GET FCB ADDRESS
E426  EB          1510          EX      DE,HL           ;  IN DE
E427  1A          1511          LD      A,(DE)          ;CHECK IF ASSIGN DR = E
                       5
E428  FEE5        1512          CP      0E5H
E42A  2807        1513          JR      Z,INTSER        ;DR = E5, SEARCH ALL US
                       ER
E42C  D5          1514          PUSH    DE
E42D  CD7CE2      1515          CALL    COMPAR          ;COMPARE DIRCNT WITH ET
                       YCNT
```

```
E430 D1          1516        POP     DE
E431 3042        1517        JR      NC,SERNFD       ;DIR END,SET DIRCOD=FF
                 1518 ;(3) INITIALIZE LOOP OF COMPARE.
                 1519 ;  COMPARE EACH CHAR IN FCB WITH FDE, IF MATCH SKIP TO
                      COMPARE
                 1520 ;  NEXT PAIR ACCORDING ASSIGN LENGTH. IF ALL CHAR MATCH
                      , GOTO
                 1521 ;  SEACH FOUND.  BUT IF FOUND FIRST UNMATCH, IGNORE THI
                      S ENTRY,
                 1522 ;  REDO WITH NEXT ENTRY.
E433 CD5BE2      1523 INTSER: CALL   DIRETY          ;SET HL POINT DIRECTORY
                       ENTRY
E436 3A95EA      1524        LD      A,(LENSER)
E439 4F          1525        LD      C,A             ;C IS LENGTH TO COMPARE
E43A 0600        1526        LD      B,00H           ;B IS POINTER OF ENTRY
E43C 79          1527 LOPSER: LD     A,C             ;LOOP COMPARE FCB
E43D B7          1528        OR      A               ;  WITH FDE
E43E 2824        1529        JR      Z,SERFND        ;ASSIGN FCB MATCH, SEAR
                       CH FOUND
E440 1A          1530        LD      A,(DE)          ;GET ONE CHAR FROM FCB
E441 FE3F        1531        CP      '?'             ;IF WILD CARD ?
E443 2819        1532        JR      Z,SKPSER        ; SKIP TO NEXT CHAR
E445 78          1533        LD      A,B             ;GET POINTER OF ENTRY
E446 FE0D        1534        CP      0DH             ;IF BYTE S1
E448 2814        1535        JR      Z,SKPSER        ; SKIP, NOT COMPARE
E44A FE0C        1536        CP      0CH             ;IF EXTENT NUMBER
E44C 1A          1537        LD      A,(DE)
E44D 2007        1538        JR      Z,EXTBYT        ; GOTO CHECK EXTENT
E44F 96          1539        SUB     (HL)            ;ELSE SUBTRACT CHAR IN
                       FCB
E450 E67F        1540        AND     7FH             ;  WITH CHAR IN FDE TO
                       COMPARE
E452 20C5        1541        JR      NZ,SERNXT       ;NOT EQUAL, REDO WITH N
                       EXT ENTRY
E454 1808        1542        JR      SKPSER          ;IF EQUAL, SKIP TO COMP
                       ARE NEXT PAIR
                 1543 ;
E456 C5          1544 EXTBYT: PUSH   BC              ;SAVE POINTER AND COUNT
                       ER
E457 4E          1545        LD      C,(HL)          ;GET FDE EXTENT#
E458 CDF3E3      1546        CALL    EXTMSK          ;GOTO CHECK EXTENT#
E45B C1          1547        POP     BC              ;RESTORE POINTER AND CO
                       UNTER
E45C 20B9        1548        JR      NZ,SERNXT       ;NOT EXTENT# 0, SEARCH
                       NEXT
E45E 13          1549 SKPSER: INC    DE              ;SKIP POINTER OF FCB
E45F 23          1550        INC     HL              ;SKIP POINTER OF FDE
E460 04          1551        INC     B               ;SKIP POINTER OF ENTRY
E461 0D          1552        DEC     C               ;DECRE COUNTER (LENGTH
                       TO COMPARE)
E462 18D8        1553        JR      LOPSER          ; LOOP COMPARE NEXT CHA
                       R.
                 1554 ;(4) SEARCH FOUND. SET DIRCOD IN EXIT VALUE, SET MARKER
                       TO 00 TO
                 1555 ;  DECLEAR SEARCH FIRST FOUND.
E464 3AA7EA      1556 SERFND: LD     A,(DIRCTL)      ;GET DIRCNT LOW BYTE
E467 E603        1557        AND     03H             ;MOD IN RANGE 0-3
```

```
CROMEMOD CODE Z80 ASSEMBLER Version 02.15                    PAGE 0040

E469 3264E0     1558        LD      (EXTVAL),A      ;  TO USE AS DIRCOD FOR
                     EXIT VALUE
E46C 2191EA     1559        LD      HL,MARKER       ;HL POINT MARKER
E46F 7E         1560        LD      A,(HL)          ;GET MARKER
E470 17         1561        RLA
E471 D0         1562        RET     NC              ;MARKER=00, BEING SEARC
                     H NEXT
E472 AF         1563        XOR     A               ;MARKER=FF, BEING SEARC
                     H FIRST
E473 77         1564        LD      (HL),A          ;  SET MARKER=00
E474 C9         1565        RET                     ;    FOR SEARCH FIRST F
                     OUND.
                1566 ;(5) SEARCH NOT FOUND. SEARCH ALL FDE BUT NOT FOUND (MA
                     RKER STILL FF
                1567 ;  IF SEING SEARCH FIRST), SET DIRCOD=FF IN EXIT VALUE
                     FOR NOT FOUND.
E475 CDF8E2     1568 SERNFD: CALL    INTDIR          ;SET DIRCNT=FFFF TO MAR
                     K DIR END
E478 3EFF       1569        LD      A,0FFH  -       ;SET DIRCOD=FF
E47A C318E0     1570        JP      EXIT1B          ;  TO SEND FOR EXIT VAL
                     UE
                1571 ;;;=============================================
                     ==.
                1572 ;;;  DELETE FILE.
                          =
                1573 ;;;    CHECK IF DRIVE SET R/O, WILL PRINT 'R/O' AND
                          =
                1574 ;;;   PROCESS END, ELSE SEARCH FDE BY COMPARE ONLY USER
                          =
                1575 ;;;   FILENAME AND FILETYPE (NOT EXTENT#). WHEN FOUND
                          =
                1576 ;;;   CHECK IF FILE SET R/O, WILL PRINT 'File R/O' AND
                          =
                1577 ;;;   PROCESS END, ELSE FILL ENTRY TYPE WITH E5,
                          =
                1578 ;;;   DEALLOCATE BLOCK POINT BY DM, CALCULATE NEW
                          =
                1579 ;;;   CHECKSUM TO REPLACE IN CSV. FINALLY WRITE THIS
                          =
                1580 ;;;   FDE BACK ON DISK.
                          =
                1581 ;;;=============================================
                     ==
E47D CD54E2     1582 DELFIL: CALL    CHKDRO          ;CHECK IF DRIVE SET R/O
E480 0E3C       1583        LD      C,3CH           ;SET LENGTH FOR USER,FI
                     LENAME & FILETYPE
E482 CD84E4     1584        CALL    SERFIR          ;  TO COMPARE IN SEARCH
                     FDE
E485 CDEFE2     1585 NXTDEL: CALL    CHKDIR          ;CHECK RESULT OF SEARCH
E488 C8         1586        RET     Z               ;DIR END, ALL DONE...RE
                     TURN
E489 CD47E2     1587        CALL    CHKFRO          ;CHECK IF FILE SET R/O
E48C CD33E2     1588        CALL    DIRELY          ;SET HL POINT DIRECTORY
                     ENTRY
E48F 36E5       1589        LD      (HL),0E5H       ;PUT E5 TO DELETE FILE
E491 0E00       1590        LD      C,00H           ;SET MASK BIT IN C
E493 CD60E3     1591        CALL    MSKALV          ;  TO DEALLOCATE BLOCK
```

```
E496  CDD2E2      1592        CALL    DIRDSK          ;WRITE THIS SECTOR OF F
                      DE ON DISK
E499  CD19E4      1593        CALL    SERNXT          ;SEARCH THIS FILE BUT N
                      EXT EXTENT
E49C  18E7        1594        JR      NXTDEL          ;  TO DELETE ALL FILE.
                  1595 ;;-----------------------------------------------------
                      ----
                  1596 ;;  SEARCH EMPTY BLOCK TO WRITE DATA. RETURN BLOCK# IN
                      HL =
                  1597 ;;-----------------------------------------------------
                      ----
                  1598 ;<1> SET START BLOCK TO SEARCH BACKWARD AND FORWARD.
E49E  50          1599 EMTYBK: LD      D,B             ;MOVE START TO SEARCH
E49F  59          1600        LD      E,C             ; IN BC (BACKWARD) AND
                      DE (FORWARD)
                  1601 ;<2> CHECK IF SEARCH UNTIL BLOCK# Ø, GOTO CHECK SEARCH
                      FORWARD
                  1602 ;  ELSE CHECK THIS BLOCK.
E4A0  79          1603 BAKWRD: LD      A,C             ;CHECK IF SEARCH BACKWA
                      RD
E4A1  B0          1604        OR      B               ;  UNTIL BOLCK# Ø
E4A2  2803        1605        JR      Z,FORWRD        ;YES, NO MORE BACKWARD.
                      . FORWARD ONLY
E4A4  0B          1606        DEC     BC              ;DECRE BLOCK TO SEARCH
                      (BACKWARD)
E4A5  D5          1607        PUSH    DE
E4A6  C5          1608        PUSH    BC
                  1609 ;<3> CHECK BLOCK SEARCH BACKWARD. IF BOLCK EMPTY GOTO G
                      ET ALLOCATE
                  1610 ;  ELSE SEARCH FORWARD.
E4A7  CD2DE3      1611        CALL    DIVBY8          ;MOVE ALV OF BLOCK TO R
                      IGHTMOST
E4AA  1F          1612        RRA                     ;TEST ALV BIT
E4AB  301A        1613        JR      NC,BKEMTY       ;NOT SET... FOUND EMPTY
                       BLOCK
E4AD  C1          1614        POP     BC
E4AE  D1          1615        POP     DE
                  1616 ;<4> CHECK IF SEARCH FORWARD UNTIL EXCEED EXIST BLOCK,
                      GOTO CHECK
                  1617 ;  SEARCH BACKWARD, ELSE CHECK THIS BLOCK
E4AF  2A83EA      1618 FORWRD: LD      HL,(DSM)        ;GET DSM
E4B2  78          1619        LD      A,E             ;TO CHECK
E4B3  95          1620        SUB     L               ;  IF SEARCH FORWARD
E4B4  7A          1621        LD      A,D             ;    UNTIL EXCEED DSM
E4B5  9C          1622        SBC     A,H             ;      (BLOCK# > DSM)
E4B6  3017        1623        JR      NC,BKFULL       ;YES, NO MORE FORWARD..
                       BACKWARD ONLY
E4B8  13          1624        INC     DE              ;INCRE BLOCK TO SEARCH
                      (FORWARD)
E4B9  C5          1625        PUSH    BC
E4BA  D5          1626        PUSH    DE
E4BB  42          1627        LD      B,D
E4BC  4B          1628        LD      C,E             ;MOVE TO BC FOR SHIFT A
                      LV BIT
E4BD  CD2DE3      1629        CALL    DIVBY8          ;MOVE ALV BIT OF BLOCK
                      TO RIGHTMOST
E4C0  1F          1630        RRA                     ;TEST ALV BIT
```

```
E4C1  3004        1631        JR      NC,BKEMTY       ;NOT SET... FOUND EMPTY
                              BLOCK
E4C3  D1          1632        POP     DE
E4C4  C1          1633        POP     BC
E4C5  1809        1634        JR      BAKWRD          ;LOOP SEARCH EMPTY BLOC
                              K.
                  1635 ;<6> FOUND EMPTY BLOCK. SET ALLOCATE AND RETURN BLOCK#
                              IN HL.
E4C7  17          1636 BKEMTY: RLA                    ;ROTATE BACK FROM TEST
                              BIT
E4C8  3C          1637        INC     A               ;SET 1 FOR ALLOCATE
E4C9  CD5AE3      1638        CALL    SAVALV          ;SAVE BACK TO ALV
E4CC  E1          1639        POP     HL              ;POP BLOCK# THAT SEARCH
                              FOUND
E4CD  D1          1640        POP     DE              ;POP TO CLEAR STACK
E4CE  C9          1641        RET
                  1642 ;<7> COME TO THIS POINT WHEN SEARCH FORWARD END.
                  1643 ;      IF SEARCH BACKWARD NOT END, SEARCH BACKWARD ONLY
                  1644 ;      BUT IF SEARCH BACKWARD END TOO, NO MORE SEARCH
                  1645 ;      => NOT FOUND EMPTY BLOCK... RETURN 0000 IN HL.
E4CF  79          1646 BKFULL: LD      A,C             ;GET BLOCK# OF SEARCH B
                              ACKWARD
E4D0  B0          1647        OR      B               ;TO CHECK IF BLOCK# 0
E4D1  20CD        1648        JR      NZ,BAKWRD       ;NO, SEARCH BACKWARD
E4D3  210000      1649        LD      HL,0000H        ;YES, SEARCH END... NOT
                              FOUND
E4D6  C9          1650        RET
                  1651 ;;-------------------------------------------------
                  --
                  1652 ;;  WRITE MAKE DIRECTORY TO DISK (USE IN MAKE FILE).
                           =
                  1653 ;;-------------------------------------------------
                  --
E4D7  0E00        1654 WRTMAK: LD      C,00H           ;C = OFFSET FROM FCB AD
                              DRESS
E4D9  1E20        1655        LD      E,20H           ;E = LENGTH TO MOVE FDE
                  1656 ;;-------------------------------------------------
                  --
                  1657 ;;   WRITE CHANGED FDE TO DISK (USE IN RENAME AND SET
                           =
                  1658 ;;  FILE ATTRIBUTE).
                           =
                  1659 ;;-------------------------------------------------
                  --
E4DB  D5          1660 WRTCHG: PUSH    DE              ;SAVE LENGTH TO MOVE (I
                              N E)
E4DC  0600        1661        LD      B,00H           ;CLEAR B
E4DE  2A62E0      1662        LD      HL,(SENT2B)     ;HL POINT FCB ADDRESS
E4E1  09          1663        ADD     HL,BC           ;  ADD WITH OFFSET
E4E2  EB          1664        EX      DE,HL           ;DE POINT SOURCE
E4E3  CD5BE2      1665        CALL    DIRETY          ;HL POINT DIRECTORY ENT
                              RY
E4E6  C1          1666        POP     BC              ;POP LENGTH TO MOVE IN
                              C
E4E7  CD66E0      1667        CALL    MOVER           ;MOVE FCB TO FDE IN DIR
                              BUF
E4EA  CD34E0      1668 WRTFDE: CALL    TRKSEC          ;SLOVE PHY. TRACK & SEC
```

```
                        TOR OF THIS ENTRY
E4ED  C3C2E2    1669        JP      DIRDSK          ;WRITE NEW CSV, WRITE F
                        DE TO DISK.
                1670 ;;;----------------------------------------------------
                        --
                1671 ;;;  RENAME FILE.
                                =
                1672 ;;;----------------------------------------------------
                        --
                1673 ;(1) CHECK IF DRIVE SET R/O. IF NOT, SEARCH OLD NAME.
E4F0  CD54E2    1674 RENAM:  CALL    CHKDRO          ;CHECK IF DRIVE SET R/O
E4F3  0E0C      1675        LD      C,0CH           ;SET LENGTH TO COMPARE
                        USER, FN&FT
E4F5  CD04E4    1676        CALL    SERFIR          ;  TO SEARCH FIRST
                1677 ;(2) WHEN SEARCH FOUND, CHECK IF FILE SET R/O. IF NOT G
                        ET USER
                1678 ;  OF OLD FILE (OR OF FCB1) FILL IN USER OF NEW NAME (O
                        R OF
                1679 ;  FCB2).
E4F8  2A62E0    1680        LD      HL,(SENT23)     ;HL POINT OR OF FCB1
E4FB  7E        1681        LD      A,(HL)          ;GET USER OF OLD NAME
E4FC  111000    1682        LD      DE,0010H        ;OFFSET TO FCB2
E4FF  19        1683        ADD     HL,DE           ;HL POINT OR OF FCB2
E500  77        1684        LD      (HL),A          ; FILL USER IN OR OF FC
                        B2
E501  CDEFE2    1685 LRENAM: CALL    CHKDIR          ;CHECK IF SEARCH FOUND
E504  C8        1686        RET     Z               ;DIR END... NOT FOUND,R
                        ETURN
E505  CD47E2    1687        CALL    CHKFRO          ;CHECK IF FILE SET R/O
                1688 ;(3) MOVE NEW NAME (PORTION USER, FILENAME AND FILE TYP
                        E ONLY)
                1689 ;  TO OVERWRITE FDE. THEN WRITE FDE BACK TO DISK.
E508  0E10      1690        LD      C,10H           ;OFFSET TO FCB2
E50A  1E0C      1691        LD      E,0CH           ;LENGTH TO OVERWRITE US
                        ER, FN&FT
E50C  CD02E4    1692        CALL    WRTCHG          ;OVERWRITE FDE AND WRIT
                        E TO DISK
                1693 ;(4) SEARCH NEXT TO RENAME ALL EXTENT OF FILE.
E50F  CD19E4    1694        CALL    SERNXT          ;SEARCH NEXT EXTENT
E512  18ED      1695        JR      LRENAM          ;  TO RENAME ALL FILE
                1696 ;----------------------------------------------------
                        --
                1697 ;  SET FILE ATTRIBUTE.
                                =
                1698 ;----------------------------------------------------
                        --
                1699 ;(1) SEARCH FILE
E514  0E0C      1700 SETATI: LD      C,0CH           ;SET LENGTH TO COMPARE
                        USER, FN&FT
E516  CD04E4    1701        CALL    SERFIR          ;  TO SEARCH FIRST
                1702 ;(2) MOVE NEW FCB TO OVERWRITE FDE AND WRITE TO DISK.
E519  CDEFE2    1703 LOPATI: CALL    CHKDIR          ;CHECK IF SEARCH FOUND
E51C  C8        1704        RET     Z               ;DIR END... NOT FOUND,R
                        ETURN
E51D  0E00      1705        LD      C,00H           ;NO OFFSET, START FROM
                        FCB
E51F  1E0C      1706        LD      E,0CH           ;LENGTH TO MOVE USER, F
```

```
                        N&FT
E521  CDD8E4    1707            CALL    WRTCH3          ;MOVE FCB TO FDE AND WR
                        ITE TO DISK
                1708 ;(3) SEARCH NEXT TO SET ALL EXTENT OF FILE.
E524  CD19E4    1709            CALL    SERNXT          ;SEARCH NEXT EXTENT
E527  18F0      1710            JR      LOPATI          ;  TO SET ALL FILE.
                1711 ;;;=================================================
                        ==
                1712 ;;;  OPEN DIRECTORY ENTRY.
                                =
                1713 ;;;    SEARCH FDE OF FILE, WHEN FOUND, MOVE THAT FDE
                                =
                1714 ;;;    TO FCB.
                                =
                1715 ;;;=================================================
                        ==
                1716 ;(1) SEARCH FDE OF FILE
E529  0E0F      1717 OPNETY: LD         C,0FH           ;SET LENGTH FOR USER, F
                        N&FT AND EXT#
E52B  CD04E4    1718            CALL    SERFIR          ;  TO COMPARE IN SEARCH
                         FIRST
E52E  CDEFE2    1719            CALL    CHKDIR          ;CHECK IF DIR END
E531  C8        1720            RET     Z               ;DIR END.. NOT FOUND,RE
                        TURN
                1721 ;(2) MOVE FDE 1 ENTRY (32 BYTES) TO FCB.
E532  CDADE1    1722 OPNOLD: CALL       FNTEXT          ;SET HL POINT EXT# OF F
                        CB
E535  7E        1723            LD      A,(HL)          ;SET ASSIGN EXT# OF FCB
E536  F5        1724            PUSH    AF              ; (*) SAVE EXT# OF FCB
E537  E5        1725            PUSH    HL              ;       AND POINTER EXT#
                         OF FCB
E538  CD59E2    1726            CALL    DIRETY          ;SET HL POINT FDE
E53B  EB        1727            EX      DE,HL           ;DE POINT FDE
E53C  2A62E0    1728            LD      HL,(SENT23)     ;HL POINT FCB ADDRESS
E53F  0E20      1729            LD      C,20H           ;TRANSFER 32 BYTES
E541  D5        1730            PUSH    DE              ;(TEMP SAVE)
E542  CD66E0    1731            CALL    MOVER           ;  FROM FDE TO FCB (FDE
                         => FCB)
                1732 ;(3) SET HIGH BIT OF BYTE 32, FILL ASSIGN EXT# OF FCB (
                        SAVE BEFORE
                1733 ;  MOVE) IN EXT# OF FCB.
E545  CD75E2    1734            CALL    SETS2           ;SET HIGH BIT OF S2 IN
                        FCB
E548  D1        1735            POP     DE              ;DE POINT FCB
E549  210C00    1736            LD      HL,000CH        ;OFFSET TO EXT#
E54C  19        1737            ADD     HL,DE           ;HL POINT EXT# OF FDE
E54D  4E        1738            LD      C,(HL)          ;(1)GET EXT# OF FDE IN
                        C
E54E  210F00    1739            LD      HL,000FH        ;OFFSET TO RC
E551  19        1740            ADD     HL,DE           ;HL POINT RC OF FDE
E552  46        1741            LD      B,(HL)          ;(2)GET RC OF FDE IN B
E553  E1        1742            POP     HL              ;(*) GET POINTER EXT#
E554  F1        1743            POP     AF              ;       AND EXT# ASSIGN
                        IN FCB
E555  77        1744            LD      (HL),A          ;COMPARE ASSIGN EXT#
E556  79        1745            LD      A,C
                1746 ;(4) COMPARE ASSIGN EXT# OF FCB WITH EXT# IN FDE TO FIN
```

```
                        D RECORD COUNT
                    1747 ;  BY ASSIGN IN FCB = IN FDE => RC USED FROM FDE.
                    1748 ;      .....,...,...> ..,... => RC = 00 (NO DATA).
                    1749 ;      .....,...,... < ..,... => RC = 80H (DATA FULL).
E557  BE            1750        CP      (HL)            ;  WITH EXT# OF FDE
E558  78            1751        LD      A,B             ;IF ASSIGN = IN FDE
E559  2806          1752        JR      Z,FILLRC        ;  USE RC SAME AS IN FD
                        E
E55B  3E00          1753        LD      A,00H           ;IF ASSIGN > IN FDE
E55D  3802          1754        JR      C,FILLRC        ;  NEW ENTRY, SET RC =
                        00
E55F  3E80          1755        LD      A,80H           ;IF ASSIGN < IN FDE, SE
                        T RC = 80H
E561  2A62E0        1756 FILLRC: LD     HL,(SENT2B)     ;GET ADDRESS OF FCB
E564  110F00        1757        LD      DE,000FH        ;OFFSET TO RC
E567  19            1758        ADD     HL,DE
E568  77            1759        LD      (HL),A          ;FILL RC OF FCB WITH
E569  C9            1760        RET                     ; TRUE SIZE, NO DATA OR
                         DATA FULL.. FUNC END.
                    1761 ;-----------------------------------------------------
                        --
                    1762 ;  FILL NULL DM OF FCB WITH DM OF FDE,
                                =
                    1763 ;   OR FILL NULL DM OF FDE WITH DM OF FCB.
                                =
                    1764 ;-----------------------------------------------------
                        --
E56A  7E            1765 FILLDM: LD     A,(HL)          ;GET DM FIRST BYTE
E56B  23            1766        INC     HL              ;GET DM SECOND BYTE
E56C  B6            1767        OR      (HL)            ; CHECK IF NULL DM
E56D  2B            1768        DEC     HL              ;(MOVE POINTER BACK)
E56E  C0            1769        RET     NZ              ;NOT NULL DM... NOT FIL
                        L
E56F  1A            1770        LD      A,(DE)          ;NULL DM... FILL WITH A
                        NOTHER DM
E570  77            1771        LD      (HL),A          ;FILL DM FIRST BYTE
E571  13            1772        INC     DE
E572  23            1773        INC     HL
E573  1A            1774        LD      A,(DE)
E574  77            1775        LD      (HL),A          ;FILL DM SECOND BYTE
E575  1B            1776        DEC     DE              ;(MOVE POINTER BACK)
E576  2B            1777        DEC     HL
E577  C9            1778        RET
                    1779 ;-----------------------------------------------------
                        --
                    1780 ;  WRITE FULL FCB TO DISK AND SET FCB STATUS AVAILABLE.
                                =
                    1781 ;-----------------------------------------------------
                        --
                    1782 ;(1) SET EXIT VALUE = 00 (TO DECREASE TO FF IF ERROR).
                    1783 ;  SET DIRCNT = 00, CHECK IF DRIVE SET R/O OR BYTE 32 O
                        F
                    1784 ;  FCB NOT SET... RETURN.
E578  AF            1785 WRTFCB: XOR    A
E579  32A4E8        1786        LD      (EXITVAL),A     ;SET EXIT VALUE = 00
E57C  32A7EA        1787        LD      (DIRCTL),A
E57F  32A8EA        1788        LD      (DIRCTH),A      ;SET DIRCNT = 00
```

```
E582  CD21E2      1789        CALL    CHKRO8          ;CHECK IF DRIVE SET R/O
E585  C0          1790        RET     NZ              ;DRIVE SET R/O... RETUR
                                N
E586  CD66E2      1791        CALL    GETS2           ;GET BYTE S2
E589  E680        1792        AND     80H             ;CHEK HIGH BIT OF S2
E58B  C0          1793        RET     NZ              ;NOT SET... RETURN
                  1794 ;<2> SEARCH FDE WHICH MATCH FCB (FROM OPEN FILE).
                  1795 ;  IF NOT FOUND... RETURN.
E58C  0E3F        1796        LD      C,0FH           ;GET LENGTH TO COMPARE
                                USER, FN&FT AND EXT#
E58E  CD04E4      1797        CALL    SERFIR          ;  TO SEARCH FIRST
E591  CDEFE2      1798        CALL    CHKDIR          ;CHECK IF SEARCH FOUND
E594  C8          1799        RET     Z               ;DIR END... NOT FOUND,
                                RETURN
E595  011000      1800        LD      BC,0010H        ;OFFSET TO DM
E598  CD58E2      1801        CALL    DIRETY          ;HL POINT FDE
E59B  09          1802        ADD     HL,BC
E59C  EB          1803        EX      DE,HL           ;<%> SET DE POINT DM OF
                                FDE
E59D  2A62E0      1804        LD      HL,(SENT2B)     ;HL POINT FCB
E5A0  09          1805        ADD     HL,BC           ;<%> SET HL POINT DM OF
                                FCB
                  1806 ;<3> CHECK ALL DM OF FCB AND FDE (FROM SEARCH FOUND)
E5A1  0E10        1807        LD      C,10H           ;COUNTER FOR DM 16 BYTE
                                S
E5A3  3A9AEA      1808 LCHKDM: LD     A,(DMLEN)       ;GET LENGTH OF DM
E5A6  B7          1809        OR      A               ; (FF=DM 1 BYTE/ 00=DM
                                2 BYTES)
E5A7  2810        1810        JR      Z,CHKDM2        ;DM 2 BYTES
                  1811 ;<3.1> IF DM OF FCB IS NULL
E5A9  7E          1812        LD      A,(HL)          ;GET DM OF FCB
E5AA  B7          1813        OR      A               ;CHECK IF DM OF FCB IS
                                NULL
E5AB  1A          1814        LD      A,(DE)          ;(PREPARE DM OF FDE TO
                                CHECK)
E5AC  2001        1815        JR      NZ,DMFDE        ;DM OF FCB NOT NULL
                  1816 ;<3.2> FILL DM OF FCB WITH DM OF FDE.
E5AE  77          1817        LD      (HL),A          ;DM OF FCB IS NULL, FIL
                                L WITH DM OF FDE
                  1818 ;<3.3> ON OTHER HAND, IF DM OF FDE IS NULL
E5AF  B7          1819 DMFDE: OR      A               ;CHECK IF DM OF FDE IS
                                NULL
E5B0  2002        1820        JR      NZ,BOTHDM       ;DM OF FDE NOT NULL
                  1821 ;<3.4> FILL DM OF FDE WITH DM OF FCB.
E5B2  7E          1822        LD      A,(HL)          ;DM OF FDE IS NULL, FIL
                                L WITH DM OF FCB
                  1823 ;<3.5> IF BOTH DM NOT NULL, COMPARE
E5B3  12          1824        LD      (DE),A          ;BOTH DM NOT NULL, GET
                                DM OF FCB
E5B4  BE          1825 BOTHDM: CP     (HL)            ; TO COMPARE WITH DM OF
                                 FDE
                  1826 ;<3.6> IF NOT EQUAL... FILE OPERATION ERROR, ELSE COMPA
                                RE NEXT PAIR.
E5B5  2005        1827        JR      NZ,NOTMAT       ;NOT EQUAL... ERROR
E5B7  1813        1828        JR      NXTPIR          ; CHECK NEXT PAIR OF DM
                  1829 ;<4> DO THE SMAE WITH DM 2 BYTES.
E5B9  CD6AE5      1830 CHKDM2: CALL   FILLDM  ;IF DM OF FCB IS NULL, FILL WIT
```

```
                        H DM OF FDE
E5BC  EB        1831    EX      DE,HL   ;EXCHANGE DM TO CHECK
E5BD  CD6AE5    1832    CALL    FILLDM  ;IF DM OF FDE IS NULL, FILL WIT
                        H DM OF FCB
E5C0  EB        1833    EX      DE,HL   ;EXCHANGE TO OLD POSITION
E5C1  1A        1834    LD      A,(DE)  ;COMPARE DM FIRST BYTE
E5C2  BE        1835    CP      (HL)    ;  OF BOTH DM
E5C3  2027      1836    JR      NZ,NOTMAT       ;NOT EQUAL... ERROR
E5C5  13        1837    INC     DE
E5C6  23        1838    INC     HL
E5C7  1A        1839    LD      A,(DE)  ;COMPARE DM SECOND BYTE
E5C8  BE        1840    CP      (HL)    ;  OF BOTH DM
E5C9  2021      1841    JR      NZ,NOTMAT       ;NOT EQUAL... ERROR
E5CB  0D        1842    DEC     C       ;DECREASE COUNTER FOR SECOND BY
                        TE
                1843 ;<5> SKIP POINTER OF DM IN FCB AND FDE, DECREASEMENT CO
                        UNTER
                1844 ;  LOOP ALL DM.
E5CC  13        1845 NXTPIR: INC    DE
E5CD  23        1846    INC     HL              ;SKIP POINTER TO NEXT D
                        M
E5CE  0D        1847    DEC     C               ;DERES COUNTER
E5CF  2002      1848    JR      NZ,LCHKDM       ;  LOOP ALL 16 BYTES OF
                         DM
                1849 ;<6> COMPARE EXTENT# IN FCB AND IN FDE.
E5D1  01E0FF    1850    LD      BC,-20          ;MOVE BACK FROM LAST DM
E5D4  09        1851    ADD     HL,BC           ;  TO EXTENT#
E5D5  EB        1852    EX      DE,HL           ;DE POINT EXT# OF FCB
E5D6  09        1853    ADD     HL,BC           ;HL POINT EXT# IF FDE
E5D7  1A        1854    LD      A,(DE)          ;COMPARE EXT# OF FCB
E5D8  BE        1855    CP      (HL)            ;  WITH EXT# OF FDE
                1856 ;<6.1> IF EXT# OF FCB < EXT# OF FDE, OK.
E5D9  3809      1857    JR      C,DMMAT         ;EXT# FCB < EXT# FDE..
                        OK
                1858 ;<6.2> ELSE FILL EXT# AND RECORD COUNT OF FCB IN FDE.
E5DB  77        1859    LD      (HL),A          ;PUT EXT# OF FCB IN FDE
E5DC  01F0FF    1860    LD      BC,0FFF0H       ;OFFSET FROM EXT# TO RC
E5DF  09        1861    ADD     HL,BC           ;POINT RC OF FDE
E5E0  EB        1862    EX      DE,HL           ;DE POINT RC OF FDE
E5E1  09        1863    ADD     HL,BC           ;HL POINT RC OF FCB
E5E2  7E        1864    LD      A,(HL)          ;MOVE RC OF FCB
E5E3  12        1865    LD      (DE),A          ;  INTO RC OF FDE
                1866 ;<7> SET WRITE TYPE AND WRITE FDE TO DISK. SET FCB STAT
                        US
                1867 ;  TO AVAILABLE (THIS FCB HAS BEEN ALREADY WRITTEN TO D
                        ISK).
E5E4  3EFF      1868 DMMAT:  LD      A,0FFH          ;SET WRTTYP = DIR
E5E6  32BFEA    1869    LD      (FCBSTA),A      ;SET FCB STATUS TO AVAI
                        LABLE
E5E9  CDEAE4    1870    CP      WRTFDE          ;WRITE FDE TO DISK.
                1871 ;<8> DM OF FCB AND FDE NOT MATCH. SET EXIT VALUE TO DEC
                        LEAR ERROR.
E5EC  216AE0    1872 NOTMAT: LD     HL,EXTVAL       ;SET EXIT VALUE WHICH S
                        ET = 00
E5EF  35        1873    DEC     (HL)            ;DECRE TO FF
E5F0  C9        1874    RET
                1875 ;;;=================================================
```

```
                        ==
                    1876 ;;;  MAKE NEW DIRECTORY ENTRY.
                        =
                    1877 ;;;=================================================
                        ==
                    1878 ;(1) CHECK DRIVE NOT SET R/O, SEARCH DELETED ENTRY BY G
                        ET
                    1879 ;  LENGTH TO COMPARE 1 BYTE.
E5F1  CD54E2        1880 MAKETY: CALL    CHKDRO          ;CHECK IF DRIVE SET R/O
E5F4  2A62E0        1881         LD      HL,(SENT2B)     ;SET FCB ADDRESS
E5F7  E5            1882         PUSH    HL              ;  TO SAVE
E5F8  2197EA        1883         LD      HL,E5FCB        ;TEMPORARY SET FCB ADDR
                        ESS
E5FB  2262E0        1884         LD      (SENT2B),HL     ;  AT E5 BYTES
E5FE  0E01          1885         LD      C,01H           ;SET LENGTH TO SEARCH
E600  CD04E4        1886         CALL    SERFIR          ; ONLY ENTRY TYPE
E603  CDEFE2        1887         CALL    CHKDIR          ;CHECK IF DIRECTORY FUL
                        L
E606  E1            1888         POP     HL              ;POP FCB ADDRESS
E607  2262E0        1889         LD      (SENT2B),HL     ;  TO GET BACK FROM TEM
                        P CHANGE
E60A  C8            1890         RET     Z               ;NOT FOUND E5 ENTRY...D
                        IR FULL
                    1891 ;(2) WHEN SEARCH FOUND, CLEAR S1, RC & DM OF FCB TO ZER
                        O.
E60B  E3            1892         EX      DE,HL           ;DE KEEP FCB ADDRESS
E60C  210F00        1893         LD      HL,000FH        ;OFFSET TO RC
E60F  19            1894         ADD     HL,DE           ;START TO CLEAR FROM RC
E610  0E11          1895         LD      C,11H           ;CLEAR RC AND DM 16 BYT
                        ES
E612  AF            1896         XOR     A               ;  (TOTAL 17 BYTES) WIT
                        H ZERO
E613  77            1897 CLRDM:  LD      (HL),A
E614  23            1898         INC     HL
E615  0D            1899         DEC     C
E616  20F9          1900         JR      NZ,CLRDM        ;LOOP CLEAR RC & DM OF
                        FCB
E618  210000        1901         LD      HL,0000H        ;OFFSET TO BYTE S1 OF F
                        CB
E61B  19            1902         ADD     HL,DE           ;HL POINT BYTE S1
E61C  77            1903         LD      (HL),A          ;CLEAR S1 WITH ZERO
                    1904 ;(3) MOVE THIS PREPARED FCB OVERWRITE FDE IN DIRECTORY
                        BUFFER
                    1905 ;  AND WRITE CREATED FDE TO DISK.
E61D  CD89E2        1906         CALL    ADJETY          ;GOTO ADJUST ETYCNT
E620  CD07E4        1907         CALL    WRTMAK          ;MOVE FCB OVERWRITE FDE
                        & WRITE TO DISK
E623  C375E2        1908         JP      SETS2           ;SET HIGH BIT OF S2...
                        FUNC END
                    1909 ;-----------------------------------------------------
                        -------
                    1910 ; WRITE FULL FCB OF THIS EXTENT TO DISK AND OPEN NEXT E
                        XTENT =
                    1911 ;-----------------------------------------------------
                        -------
                    1912 ;(1) SET FCB STATUS TO FULL, ATTEMP TO WRITE FULL FCB T
                        O DISK.
```

```
                        1913 ;      IF WRITE OK, SET FCB STATUS TO AVAILABLE.
                        1914 ;      IF WRITE ERROR,DIRCNT SET = 0000.
E626  AF                1915 NXTEXT: XOR    A
E627  328FEA            1916         LD     (FCBSTA),A      ;SET FCB STATUS TO FULL
E62A  CD78E5            1917         CALL   WRTFCB          ;WRITE FULL FCB TO DISK
E62D  CDEFE2            1918         CALL   CHKDIR          ;CHECK IF WRITE SUCCESS
E630  C8                1919         RET    Z               ;ERROR (DIRCNT SET 0000
                             )... RETURN
                        1920 ;<2> INCREASE EXTENT NUMBER TO NEXT. CHECK IF EXTENT BY
                             TE IS
                        1921 ;  OVERFLOW (MAX = X XXXX) CARRY TO BYTE S2.
E631  2A62E0            1922         LD     HL,(SENT2B)     ;GET FCB ADDRESS
E634  010C00            1923         LD     BC,000CH        ;OFFSET TO EXT#
E637  09                1924         ADD    HL,BC           ;HL POINT EXT#
E638  7E                1925         LD     A,(HL)          ;GET EXT#
E639  3C                1926         INC    A               ;INCRE TO NEXT EXTENT
E63A  E61F              1927         AND    1FH             ;CHECK IF EXTENT BYTE >
                              X XXXX
E63C  77                1928         LD     (HL),A          ;(SAVE EXT# IN EXTENT B
                             YTE
E63D  2807              1929         JR     Z,CARYS2        ;OVERFLOW... CARRY TO S
                             2
                        1930 ;<3> CHECK IF THIS PHYSICAL EXTENT HAS ANOTHER LOGICAL
                             EXTENT
                        1931 ;  AND FCB STATUS IS AVAILABLE.
E63F  47                1932         LD     B,A             ;EXTENT BYTE NOT OVERFL
                             OW
E640  3A82EA            1933         LD     A,(EXM)         ;GET EXTENT MASK
E643  A0                1934         AND    B               ;  TO CHECK IF HAS ANOT
                             HER LOG. FCB
E644  218FEA            1935         LD     HL,FCBSTA       ;AND GET FCB STATUS
E647  A6                1936         AND    (HL)            ;  TO CHECK IF FCB IS A
                             VAILABLE
E648  280C              1937         JR     Z,USENEW        ;FCB IS FULL, SEARCH NE
                             XT FCB
E64A  1824              1938         JR     USEOLD          ;FCB IS AVAILABLE, USE
                             OLD FCB
                        1939 ;<4> EXTENT BYTE IS OVERFLOW, CARRY TO BYTE S2. CHECK I
                             F BYTE S2
                        1940 ;  IS OVERFLOW TOO (MAX = XXXX), GOTO SET ERROR.
E64C  010200            1941 CARYS2: LD     BC,0002H        ;OFFSET FROM EXT# TO S2
E64F  09                1942         ADD    HL,BC           ;HL POINT S2
E650  34                1943         INC    (HL)            ;INCRE S2 FROM CARRY
E651  7E                1944         LD     A,(HL)          ;  AND SAVE
E652  E60F              1945         AND    0FH             ;CHECK IF S2 > XXXX
E654  2824              1946         JR     Z,OPFAIL        ;YES, EXTENT OVERFLOW..
                             . ERROR
                        1947 ;<5> FCB IS FULL. GET NEXT EXTENT AND MOVE TO FCB
                        1948 ;      IF NO MORE EXTENT OF THIS FILE
                        1949 ;      BEING READ => ATTEMP TO AFETR EOF... READ ERROR
                        1950 ;      BEING WRITE => MAKE NEW EXTENT TO WRITE DATA.
E656  0E3F              1951 USENEW: LD     C,0FH           ;SET LENGTH TO COMPARE
                             USER, FN&FT AND EXT#
E658  CD04E4            1952         CALL   SERFIR          ;  TO SEARCH FIRST
E65B  CDEFE2            1953         CALL   CHKDIR          ;CHECK IF SEARCH FOUND
E65E  2810              1954         JR     NZ,USEOLD       ;SEARCH FOUND... GOTO O
                             PEN ENTRY
```

```
E660 3A90EA     1955        LD      A,(ACCTYP)      ;NOT FOUND, CHECK ACCES
                    S TYPE
E663 3C         1956        INC     A               ;CHECK IF BEING READ
E664 2814       1957        JR      Z,OPFAIL        ;YES,READ END ... SET E
                    RROR
E666 CDF1E5     1958        CALL    MAKETY          ;BEING WRITE, MAKE NEW
                    ENTRY
E669 CDEFE2     1959        CALL    CHKDIR          ;CHECK IF MAKE OK.
E66C 280C       1960        JR      Z,OPFAIL        ;NO SPACE... SET ERROR
E66E 1803       1961        JR      SAVPAR          ;MAKE OK, SAVE PARAMETE
                    R.
                1962 ;<6> FCB IS AVAILABLE. USE OLD ENTRY, NO NEED TO OPEN N
                    EW ONE.
E670 CD32E5     1963 USEOLD: CALL   OPNOLD          ;MOVE OLD FDE INTO FCB
E673 CDC2E1     1964 SAVPAR: CALL   SAVREC          ;SAVE RECCNT, EXTCNT AN
                    D CURREC
E676 AF         1965        XOR     A               ;RETURN READ SUCCESSFUL
E677 C313E0     1966        JP      EXITIB          ;  BY SET EXIT VALUE ZE
                    RO.
                1967 ;<7> OPERATION NOT SUCCESS. SET EXIT VALUE NOT ZERO, SE
                    T HIGH
                1968 ;  BIT OF BYTE S2.
E67A CD1FE0     1969 OPFAIL: CALL   NOTZEO          ;SET EXIT VALUE NOT ZER
                    O
E67D C375E2     1970        JP      SETS2           ;SET HIGH BIT OF S2.
                1971 ;;;=================================================
                    ==
                1972 ;;;  READ SEQUENTIAL.
                                =
                1973 ;;;=================================================
                    ==
                1974 ;<1> SET UPDATE FACTOR=01 (SKIP TO NEXT RECORD)
E680 3E01       1975 REDSEQ: LD     A,01H
E682 3292EA     1976        LD      (UPFACT),A      ;SET UPDATE FACTOR=01 (
                    SKIP)
                1977 ;;;=================================================
                    ==
                1978 ;;;   COMMON POINT FOR READ RANDOM.
                                =
                1979 ;;;=================================================
                    ==
                1980 ;<2> SET ACCESS TYPE TO READ
E685 3EFF       1981 COMONR: LD     A,0FFH
E687 3290EA     1982        LD      (ACCTYP),A      ;SET ACCESS TYPE TO REA
                    D
E68A CDC2E1     1983        CALL    SAVREC          ;SAVE RECCNT,EXTCNT AND
                     CURREC
                1984 ;<3> IF CURREC<RECCNT, EXTENT IS CORRECT. GOTO READ 1 S
                    ECTOR.
E68D 3AA0EA     1985        LD      A,(CURREC)      ;GET CURREC
E690 219EEA     1986        LD      HL,RECCNT       ;  TO COMPARE
E693 BE         1987        CP      (HL)            ;    WITH RECCNT
E694 3811       1988        JR      C,SAMEXT        ;CURREC<RECCNT... OK.
                1989 ;<4> IF CURREC>RECCNT, IT MUST 80H (RECORD AFTER LAST R
                    ECORD
                1990 ;  IN EXTENT). IF NOT 80H SET ERROR, ELSE WRITE THIS EX
                    TENT
```

```
                        1991 ;  TO DISK AND OPEN NEXT EXTENT.
E696  FE80              1992        CP      80H             ;CURREC MUST BE ONE AFT
                            ER LAST
E698  2021              1993        JR      NZ,REDEOF       ;NO, SO EOF.
E69A  CD26E6            1994        CALL    NXTEXT          ;THIS FCB IS FULL, WRIT
                            E FCB TO DISK
E69D  AF                1995        XOR     A               ;SET CURREC=0
E69E  32A0EA            1996        LD      (CURREC),A      ;  FOR NEXT EXTENT
E6A1  3A64E0            1997        LD      A,(EXTVAL)      ;CHECK RESULT OF WRITE
                            FCB TO DISK
E6A4  B7                1998        OR      A               ; (EXTVAL=FFFF IF BOTH
                            DM NOT MATCH)
E6A5  2014              1999        JR      NZ,REDEOF       ;WRITE ERROR... RETURN
                        2000 ;(5) FIND LOGICAL TRACK & SECTOR OF THIS RECORD BY
                        2001 ;     LOGICAL SECTOR# = BLOCK# * BLOCK SIZE + SECTOR IN
                             BLOCK
                        2002 ;  THEN USE ROUTINE OF DIR TO FIND PHYSICAL TRACK & SEC
                            TOR.
E6A7  CD7FE1            2003 SAMEXT: CALL   SAVEBK          ;SAVE BLOCK# OF CURREC
                            IN DIRSEC
E6AA  CD8CE1            2004        CALL    CHEKBK          ;CHECK IF BLOCK#
E6AD  280C              2005        JR      Z,REDEOF        ;BLOCK 0, NO MORE DATA.
                            .. EOF
E6AF  CD92E1            2006        CALL    BADDSB          ;FIND BK# * BK SIZE + S
                            ECTOR IN BK
E6B2  CDE2E0            2007        CALL    DATARE          ;FIND PHY. TRACK & SECT
                            OR OF THIS SECTOR
                        2008 ;(6) READ 1 SECTOR AND UPDATE CURREC AND RECCNT.
E6B5  CDC7E0            2009        CALL    RDISK           ;READ DATA 1 SECTOR
E6B8  C3D9E1            2010        JP      UPDREC          ;UPDATE CURREC & RECCNT
                            ... FUNC END
                        2011 ;(7) IF ERROR,SET EXIT VALUE NOT VALUE.
E6BB  C31FE0            2012 REDEOF: JP     NOTZED          ;SET EXIT VALUE NOT ZER
                            0 FOR EOF.
                        2013 ;;;=============================================
                            ==
                        2014 ;;;  WRITE SEQUENTIAL.
                                 =
                        2015 ;;;=============================================
                            ==
                        2016 ;(1) SET UPDATE FACTOR=01 (SKIP TO NEXT RECORD)
E6BE  3E01              2017 WRTSEQ: LD     A,01H
E6C0  3292EA            2018        LD      (UPFACT),A      ;SET UPDATE FACTOR=01 (
                            SKIP)
                        2019 ;;;=============================================
                            ==
                        2020 ;;;  COMMON POINT FOR WRITE RANDOM.
                                 =
                        2021 ;;;=============================================
                            ==
                        2022 ;(2) SET ACCESS TYPE TO WRITE.
E6C3  3E00              2023 COMONW: LD     A,00H
E6C5  3290EA            2024        LD      (ACCTYP),A      ;SET ACCESS TYPE TO WRI
                            TE
                        2025 ;(3) CHECK IF DRIVE SET R/O OR FILE SET R/O
E6C8  CD54E2            2026        CALL    CHKDRO          ;CHECK IF DRIVE SET R/O
E6CB  2A62E0            2027        LD      HL,(SENT23)     ;HL POINT FCB (FROM OPE
```

```
                        N FILE)
E6CE CD4AE2     2028    CALL    DATFRO          ;CHECK IF FILE SET R/O
                2029 ;(4) SAVE RECCNT, EXTCNT AND CURREC. CHECK CURREC MUST
                        BE
                2030 ;  A RECORD IN EXTENT.
E6D1 CDC2E1     2031    CALL    SAVREC          ;SAVE RECCNT, EXTCNT &
                        CURREC
E6D4 3AA8EA     2032    LD      A,(CURREC)      ;GET CURREC
E6D7 FE80       2033    CP      80H             ;  TO CHECK IF WITHIN E
                        XTENT (<80H)
E6D9 D21FE8     2034    JP      NC,NOTIED       ;CURREC NOT < 80H, RETU
                        RN ERROR
                2035 ;(5) CHECK BLOCK# POINTED BY DM OF CURREC.
E6DC CD7FE1     2036    CALL    SAVE3K          ;SAVE BLOCK# OF CURREC
E6DF CD8CE1     2037    CALL    CHEKBK          ;CHECK IF BLOCK 0
E6E2 0E00       2038    LD      C,00H           ;(+) SET FLAG=00 IN C F
                        OR NOT EMPTY BLOCK
E6E4 2043       2039    JR      NZ,BEGWRT       ;NOT BLOCK 0
                2040 ;(6) BLOCK OF CURREC IS BLOCK 0, SEARCH EMPTY BLOCK TO
                        USE.
E6E6 CD4AE1     2041    CALL    GETDM           ;BLOCK 0, GET POSITION
                        OF DM
E6E9 3294EA     2042    LD      (DMPOS),A       ;SAVE POSITION OF DM OF
                         CURREC
                2043 ;(6.1) IF DM0, SET TO SEARCH FROM BLOCK# 0.
E6EC 010000     2044    LD      BC,0000H        ;SET TO SEARCH EMPTY BL
                        OCK FROM BLOCK# 0
E6EF B7         2045    OR      A               ;CHECK IF DM0
E6F0 2007       2046    JR      Z,SEMT3K        ;YES, SEARCH FROM BLOCK
                        # 0
E6F2 4F         2047    LD      C,A             ;NOT DM0
E6F3 0B         2048    DEC     BC              ;DECRE 1 FOR DM(N-1)
E6F4 CD67E1     2049    CALL    GET3K           ;GET BLOCK# POINTED BY
                        DM(N-1) IN HL
E6F7 44         2050    LD      B,H
E6F8 4D         2051    LD      C,L             ;SEARCH FROM BLOCK POIN
                        TED BY DM(N-1)
                2052 ;(6.2) IF DM(N), SET START TO SEARCH FROM BLOCK POINTED
                         BY DM(N-1).
E6F9 CD9EE4     2053 SEMT3K: CALL   EMTY3K          ;SEARCH EMPTY BLOCK
                2054 ;(7) IF SEARCH FOUND EMPTY BLOCK, HL=# OF BLOCK.
                2055 ;  BUT IF NOT FOUND, HL=0000. SET ERROR CODE=02 FOR DIS
                        K FULL.
E6FC 7D         2056    LD      A,L             ;CHECK # OF EMPTY BLOCK
E6FD B4         2057    OR      H               ;  WHICH SEARCH FOUND
E6FE 2005       2058    JR      NZ,FEMT3K       ;YES, SEARCH FOUND EMPT
                        Y BLOCK
E700 3E02       2059    LD      A,02H           ;NO, SET ERROR CODE=02
E702 C31BE8     2060    JP      EXIT1B          ;  FOR DISK FULL.
                2061 ;(8) FOUND EMPTY BLOCK. SAVE BLOCK# AND PUT INTO DM
                2062 ;       DM = 00XX FOR DM 1 BYTE
                2063 ;       DM = XXXX FOR DM 2 BYTES.
E705 22A2EA     2064 FEMT3K: LD      (DIRSC1),HL    ;SAVE # OF EMPTY BLOCK
E708 EB         2065    EX      DE,HL           ;AND SAVE IN DE
E709 2A62E8     2066    LD      HL,(SENTCB)     ;GET ADDRESS OF FCB
E70C 011000     2067    LD      BC,0010H        ;OFFSET TO DM0
E70F 09         2068    ADD     HL,BC           ;HL POINT DM0
```

```
E710 3A9AEA     2069        LD      A,(DMLEN)       ;GET LENGTH OF DM
E713 B7         2070        OR      A
E714 3A94EA     2071        LD      A,(DMPOS)       ;(GET POSITION OF DM)
E717 2806       2072        JR      Z,FEMT32        ;DM 2 BYTES
E719 CD61E2     2073        CALL    PNTDM           ;SET HL POINT DM
E71C 73         2074        LD      (HL),E          ;PUT EMPTY BLOCK# IN DM
                        (DM=00XX)
E71D 1803       2075        JR      FBKNUM
                2076 ;
E71F 4F         2077 FEMT32: LD     C,A             ;GET POSITION OF DM IN
                        C
E720 0600       2078        LD      B,00H           ;  AND CLEAR B, SO OFFS
                        ET=00XX
E722 09         2079        ADD     HL,BC           ;SET HL POINT DM BY
E723 09         2080        ADD     HL,BC           ;  DOUBLE ADD WITH OFFS
                        ET
E724 73         2081        LD      (HL),E          ;PUT BLOCK# FIRST BYTE
E725 23         2082        INC     HL
E726 72         2083        LD      (HL),D          ;PUT BLOCK# SECOND BYTE
E727 0E02       2084 FBKNUM: LD     C,02H           ;(+) SET FLAG=02 FOR EM
                        PTY BLOCK
                2085 ;(9) CHECK IF DM POINT EMPTY BLOCK OR NOT EMPTY BLOCK.
                2086 ;  IF NOT EMPTY BLOCK, GOTO WRITE DISK.
E729 3A64E3     2087 BEGWRT: LD     A,(EXTVAL)      ;CHECK EXIT VALUE
E72C B7         2088        OR      A               ;  (00 IF OK/ FF IF ERRO
                        R)
E72D C0         2089        RET     NZ              ;RETURN ERROR.
E72E C5         2090        PUSH    BC              ;SAVE FLAG
E72F CD92E1     2091        CALL    BADDSB          ;FIND LOGICAL SECTOR OF
                         THIS RECORD
E732 3A92EA     2092        LD      A,(UPFACT)      ;GET UPDATE FACTOR
E735 3D         2093        DEC     A
E736 3D         2094        DEC     A
E737 200A       2095        JR      NZ,WERCOD       ;NOT FUNC40, GOTO WRITE
                         RECORD
E739 C1         2096        POP     BC              ;RESTORE FLAG
E73A C5         2097        PUSH    BC
E73B 79         2098        LD      A,C
E73C 3D         2099        DEC     A               ;FLAG=00 FOR NOT EMPTY
                        BLOCK
E73D 3D         2100        DEC     A               ;FLAG=02 FOR EMPTY BLOC
                        K
E73E 2003       2101        JR      NZ,WERCOD       ;NOT EMPTY BLOC, GOTO W
                        RITE RECORD
                2102 ;(10) DM POINT EMPTY BLOCK, CLEAR ALL RECORD IN THIS BL
                        OCK.
E740 E5         2103        PUSH    HL              ;SAVE ADDRESS OF DM
                2104 ;(10.1) GET ADDRESS OF DIRBUF AND WITH ZERO LENGTH 128
                        BYTES.
E741 2A76EA     2105        LD      HL,(ADDDIB)     ;HL POINT DIRBUF
E744 57         2106        LD      D,A             ;D=COUNTER START FROM 0
                        0
E745 77         2107 PREZRO: LD     (HL),A          ;CLEAR DIRBUF
E746 23         2108        INC     HL              ;  WITH ZERO
E747 14         2109        INC     D               ;    LENGTH 128 BYTES
E748 F245E7     2110        JP      P,PREZRO        ;    (COUNTER 00=>7F)
                2111 ;(10.2) SET DMA AT DIRBUF, GET FIRST LOGICAL SECTOR IN
```

```
                        BLOCK
                    2112 ;   FROM BLOCK# * BLOCK SIZE. THEN SLOVE PHYSICAL TRACK
                         & SECTOR
                    2113 ;   AND OVERWRITE WITH ZERO 128 BYTES FROM DIRBUF. LOOP
                         WRITE
                    2114 ;   UNTIL LAST SECTOR IN BLOCK. THEN SET DMA ADDRESS BA
                        CK TO 80H.
E748 CDDAE2         2115        CALL    SETDIR          ;SET DMA AT DIRBUF
E74E 2AA4EA         2116        LD      HL,(BKXBKZ)     ;HL=BLOCK# * BLOCK SIZE
E751 0E02           2117        LD      C,02H           ;SET WRTTYP
E753 22A2EA         2118 CLRREC: LD     (DIRSC1),HL     ;SET LOG. SECTOR AT BK#
                         * BK SIZE
E756 C5             2119        PUSH    BC              ;SAVE WRTTYP
E757 CDE2E0         2120        CALL    DATARE          ;SET PHYSICAL TRACK & S
                        ECTOR
E75A C1             2121        POP     BC              ;GET WRTTYP
E75B CDCCE0         2122        CALL    WDISK           ;WRITE DIRBUF ON DISK
E75E 2AA2EA         2123        LD      HL,(DIRSC1)     ;GET BK# * BK SIZE
E761 0E00           2124        LD      C,00H           ;CHANGE WRTTYP
E763 3A81EA         2125        LD      A,(BLM)         ;GET BLM
E766 47             2126        LD      B,A
E767 A5             2127        AND     L               ;GET RECORD# WITH BLM
E768 B8             2128        CP      B               ;TO CHECK IF LAST RECOR
                        D IN BLOCK
E769 23             2129        INC     HL              ;SKIP TO NEXT RECORD IN
                         BLOCK
E76A 20E7           2130        JR      NZ,CLRREC       ;NOT LAST RECORD, LOOP
E76C E1             2131        POP     HL              ;CLEAR ALL, RESTORE ADD
                        RESS OF DM
E76D 22A2EA         2132        LD      (DIRSC1),HL     ;SAVE AS # OF RECORD TO
                         WRITE
E770 CD03E2         2133        CALL    SETADD          ;SET DMA ADDRESS BACK T
                        O 80H
                    2134 ;(11) SLOVE PHYSICAL TRACK & SECTOR OF CURREC, THEN WRI
                        TE TO DISK.
E773 CDE2E0         2135 WERCOD: CALL   DATARE          ;SET PHYSICAL TRACK & S
                        ECTOR
E776 C1             2136        POP     BC              ;GET WRTTYP IN C
E777 C5             2137        PUSH    BC
E778 CDCCE0         2138        CALL    WDISK           ;WRITE RECORD
                    2139 ;(11.1) IF CURREC (WHICH WRITE TO DISK)>RECCNT, UPDATE
                        RECCNT = CURREC+1.
E77B C1             2140        POP     BC
E77C 3AA0EA         2141        LD      A,(CURREC)      ;GET CURREC
E77F 219EEA         2142        LD      HL,RECCNT       ;  TO COMPARE
E782 BE             2143        CP      (HL)            ;    WITH RECCNT
E783 3804           2144        JR      C,WDCSS2        ;CURREC<RECCNT, RECCNT
                        CORRECT.
E785 77             2145        LD      (HL),A          ;  ELSE PUT CURREC+1
E786 34             2146        INC     (HL)            ;    INTO RECCNT.
E787 0E02           2147        LD      C,02H
                    2148 ;(12) SAVE BYTE 52 AT RPA. CHECK IF FDB FULL (RECCNT=7F
                        ) OR NOT.
E789 00             2149 WDCSS2: NOP
E78A 00             2150        NOP
E78B 2100D5         2151        LD      HL,RPA
E78E F5             2152        PUSH    AF              ;SAVE RECCNT IN A
```

```
E78F CD66E2     2153        CALL    GETS2           ;GET BYTE S2
E792 E67F       2154        AND     7FH             ;STRIP OUT HIGH BIT
E794 77         2155        LD      (HL),A          ;  AND SAVE IN DOS
E795 F1         2156        POP     AF              ;RESTORE RECCNT
E796 FE7F       2157        CP      7FH             ;  TO CHECK IF FCB FULL
                2158 ;<12.1> IF NOT FULL, GOTO UPDATE CURREC & RECCNT.
E798 201A       2159        JR      NZ,WRTEND       ;NOT FULL, UPDATE CURRE
                    C & RECCNT.
E79A 3A92EA     2160        LD      A,(UPFACT)      ;SET UPDATE FACTOR
E79D FE01       2161        CP      01H             ;CHECK IF SEQ.
                2162 ;<12.2> IF FULL BUT RANDOM ACCESS, GOTO UPDATE CURREC &
                     RECCNT.
E79F 2013       2163        JR      NZ,WRTEND       ;RANDOM, UPDATE CURREC
                    & RECCNT.
                2164 ;<12.3>IF FULL AND SEQ. ACCESS, WRITE THIS FCB TO DISK
                    AND OPEN
                2165 ;   NEXT EXTENT. THEN UPDATE CURREC & REECNT.
E7A1 CDD9E1     2166        CALL    UPDREC          ;SEQ, UPDATE CURREC & R
                    ECCNT
E7A4 CD26E6     2167        CALL    NXTEXT          ;WRITE FCB ON DISK AND
                    OPEN NEXT EXTENT
E7A7 2164E0     2168        LD      HL,EXTVAL       ;GET SET EXIT VALUE
E7AA 7E         2169        LD      A,(HL)          ;  TO CHECK IF
E7AB B7         2170        OR      A               ;    HAS ANOTHER FCB
E7AC 2004       2171        JR      NZ,WCOMPL       ;FCB FULL, USE NEXT FCB
E7AE 3D         2172        DEC     A               ;FCB AVAILABLE, DECRE C
                    URREC
E7AF 32A0EA     2173        LD      (CURREC),A      ;  AND SAVE
                2174 ;<13> SET EXIT VALUE ZERO FOR OPERATION SUCCESS. UPDATE
                     CURREC AND
                2175 ;   RECCNT FOR NEXT OPERATION.
E7B2 3600       2176 WCOMPL: LD     (HL),00H        ;SET EXIT VALUE ZERO
E7B4 C3D9E1     2177 WRTEND: JP     UPDREC          ;GOTO UPDATE CURREC & R
                    ECCNT.
                2178 ;;;-----------------------------------------------
                    --
                2179 ;;;  CONVERT RANDOM RECORE.
                                =
                2180 ;;;-----------------------------------------------
                    --
                2181 ;<1> SET UPDATE FACTOR=00 FOR NOT SKIP CURREC AND RECCN
                    T.
E7B7 AF         2182 CVRREC: XOR    A
E7B8 3292EA     2183        LD      (UPFACT),A      ;SET UPDATE FACTOR=00
                2184 ;<2> FROM ASSIGN RANDOM RECORD, FIND RECORD IN EXTENT B
                    Y
                2185 ;  MOD R0 WITH 7F.
E7BB C5         2186 CVRF80: PUSH   BC
E7BC 2A62E0     2187        LD      HL,(SENT2B)     ;GET FCB ADDRESS
E7BF EB         2188        EX      DE,HL           ;DE POINT FCB ADDRESS
E7C0 212100     2189        LD      HL,0021H        ;OFFSET TO R0
E7C3 19         2190        ADD     HL,DE           ;HL POINT R0
E7C4 7E         2191        LD      A,(HL)          ;GET R0
E7C5 E67F       2192        AND     7FH             ;MOD WITH 7F TO FIND
E7C7 F5         2193        PUSH    AF              ;<+> RECORD IN EXTENT I
                    N A
                2194 ;<3> FIND EXT# FOR EXTENT BYTE BY DIVIDE R0&R1 BY 128.
```

```
                        2195 ;  DIVIDE DO BY GET BIT 7 OF R0 AND BIT 1,2,3 & 4 OF R1
                        2196 ;  TO C (BASE 128).
E7C8  7E                2197        LD      A,(HL)          ;GET R0
E7C9  17                2198        RLA                     ;ROTATE BIT 7 OF R0 TO
                            CY
E7CA  23                2199        INC     HL              ;MOVE TO R1
E7CB  7E                2200        LD      A,(HL)          ;GET R1
E7CC  17                2201        RLA                     ;ROTATE BIT 1,2,3 & 4 O
                            F R1
E7CD  E61F              2202        AND     1FH             ;  AND GET BIT 7 OF R0
                            FROM CY
E7CF  4F                2203        LD      C,A             ;(+) EXT# FOR EXTENT BY
                            TE IN C
                        2204 ;(4) FIND EXT# FOR BYTE S2 BY DIVIDE R1 BY 16 (BYTE S2
                            IS BASE 16
                        2205 ;  OF EXTENT BYTE).
E7D0  7E                2206        LD      A,(HL)          ;GET R1
E7D1  1F                2207        RRA
E7D2  1F                2208        RRA
E7D3  1F                2209        RRA
E7D4  1F                2210        RRA                     ;DIVIDE BY 16
E7D5  E60F              2211        AND     0FH             ;STRIP OUT UNWANT BIT (
                            S2 USE ONLY 4 BITS)
E7D7  47                2212        LD      B,A             ;(+) EXT# FOR BYTE S2 I
                            N B
E7D8  F1                2213        POP     AF              ;(RESTORE RECORD IN EXT
                            ENT)
E7D9  23                2214        INC     HL              ;MOVE TO R2
                        2215 ;(5) AFTER CONVERT RANDOM RECORD WHICH DETERMINE ONLY I
                            N POSSIBLE
                        2216 ;  RANGE. NOW WILL CHECK IF ASSIGN RANDOM RECORD OVERFL
                            OW (IF R2
                        2217 ;  NOT ZERO MEANS OVERFLOW FROM R0&R1... ASSIGN > 65535
                            ), SET
                        2218 ;  ERROR CODE 6 FOR ATTEMP TO READ/WRITE BEYOND END OF
                            DISK.
E7DA  6E                2219        LD      L,(HL)          ;GET R2
E7DB  2C                2220        INC     L               ;INCRE
E7DC  2D                2221        DEC     L               ;  AND DECRE TO SET FLA
                            G
E7DD  2E06              2222        LD      L,06H           ;SET ERROR CODE 6
E7DF  2057              2223        JR      NZ,CVTER2       ;R2 NOT ZERO... ASSIGN
                            OVERFLOW
                        2224 ;(6) ASSIGN RANDOM RECORD OK. FILL CALCULATED RECORD IN
                             EXTENT IN CR.
E7E1  112000            2225        LD      HL,0020H        ;OFFSET TO CR
E7E4  19                2226        ADD     HL,DE           ;HL POINT CR
E7E5  77                2227        LD      (HL),A          ;(+)  FILL WITH RECORD
                            IN EXTENT
                        2228 ;(7) CHECK EXT# OF OPENED FILE WITH CALCULATED EXT# (BO
                            TH IN
                        2229 ;  EXTENT BYTE AND BYTE S2). IF NOT MATCH WRITE THIS FC
                            B
                        2230 ;  ON DISK AND SEARCH NEXT EXTENT UNTIL MATCH.
E7E6  210C00            2231        LD      HL,000CH        ;OFFSET TO EXT#
E7E9  19                2232        ADD     HL,DE           ;HL POINT EXT#
E7EA  79                2233        LD      A,C             ;(+) C KEEP EXT# FOR EX
```

```
                        TENT BYTE
E7E8  96        2234            SUB     (HL)            ;COMPARE WITH EXT# OF F
                        CB
E7EC  200A      2235            JR      NZ,EXTNUM       ;NOT EQUAL... WRITE FCB
                         ON DISK
                        2236 ;(8) EXTENT BYTE MATCH. CHECK BYTE S2. IF NOT MATCH WRI
                        TE FCB TO
                        2237 ;  DISK AND SEARCH NEXT EXTENT.
E7EE  210E00    2238            LD      HL,000EH        ;OFFSET TO BYTE S2
E7F1  19        2239            ADD     HL,DE           ;HL POINT BYTE S2
E7F2  78        2240            LD      A,B             ;(+) B KEEP EXT# FOR BY
                        TE S2
E7F3  96        2241            SUB     (HL)            ;COMPARE WITH S2 OF FCB
E7F4  E67F      2242            AND     7FH
E7F6  2834      2243            JR      Z,OVTOK         ;EQUAL.. EXTENT CORRECT
                        , DO READ/WRITE.
                        2244 ;(9) EXT# OF OPENED FILE NOT MATCH WITH CALCULATE EXT#
                        OF RANDOM
                        2245 ;  RECORD. FIRST WRITE THIS FCB TO DISK.
E7F8  C5        2246 EXTNUM: PUSH   BC              ;SAVE CALCULATED EXT#
E7F9  D5        2247            PUSH    DE              ;SAVE FCB ADDRESS
E7FA  CD78E5    2248            CALL    WRTFCB          ;WRITE FCB ON DISK
E7FD  D1        2249            POP     DE
E7FE  C1        2250            POP     BC
                        2251 ;(9.1) IF WRITE ERROR, SET ERROR CODE 3 FOR CANNOT CLOS
                        E CURRENT EXTENT.
E7FF  2E03      2252            LD      L,03H           ;SET ERROR CODE 3
E801  3A64E8    2253            LD      A,(EXTVAL)      ;GET EXIT VALUE
E804  3C        2254            INC     A               ;  TO CHECK IF ERROR OC
                        CURE
E805  282A      2255            JR      Z,OVTERR        ;ERROR, CAN'T CLOSE EXT
                        ENT.
                        2256 ;(10) WRITE FCB OF CURRENT EXTENT OK. FILL CALCULATED E
                        XT# INTO
                        2257 ;  FCB AND OPEN NEXT ENTRY.
E807  210C00    2258            LD      HL,000CH        ;OFFSET TO EXT#
E80A  19        2259            ADD     HL,DE           ;HL POINT EXT#
E80B  71        2260            LD      (HL),C          ;  FILL WITH EXT# OF RA
                        NDOM RECORD
E80C  210E00    2261            LD      HL,000EH        ;OFFSET TO S2
E80F  19        2262            ADD     HL,DE           ;HL POINT S2
E810  70        2263            LD      (HL),B          ;  FILL WITH EXT# OF RA
                        NDOM RECORD
E811  CD29E5    2264            CALL    OPNETY          ;OPEN NEXT ENTRY
                        2265 ;(10.1) ENTRY WHICH TRY TO OPEN IS NONEXIST EXTENT AND
                        BEING READ,
                        2266 ;  SET ERROR CODE 4 FOR ATTEMP TO READ UNWRITTEN EXTEN
                        T.
E814  3A64E8    2267            LD      A,(EXTVAL)      ;GET EXIT VALUE
E817  3C        2268            INC     A               ;  TO CHECK IF ERROR OC
                        CURS
E818  2812      2269            JR      NZ,OVTOK        ;OPEN OK.... DO READ/WR
                        ITE
E81A  D1        2270            POP     BC              ;POP WRITYP IN C
E81B  C5        2271            PUSH    BC
E81C  2E04      2272            LD      L,04H           ;SET ERROR CODE 4
E81E  0C        2273            INC     C
```

```
E81F 2810          2274          JR      Z,CVTERR        ;EXTENT TO OPEN NONEXIS
                                 T... ERROR
                   2275 ;<10.2> TRY TO OPEN NONEXIST EXTENT BUT BEING WRITE...
                                 NOT ERROR,
                   2276 ;   MAKE NEW ENTRY TO WRITE. BUT IF MAKE ENTRY ERROR SE
                                 T ERROR
                   2277 ;   CODE 5 FOR DIRECTORY FULL.
E821 CDF1E5        2278          CALL    MAKETY          ;MAKE NEW ENTRY
E824 2E05          2279          LD      L,05H           ;SET ERROR CODE 5
E826 3A64E0        2280          LD      A,(EXTVAL)      ;GET EXIT VALUE
E829 3C            2281          INC     A               ;  TO CHECK IF ERROR OC
                                 CURE
E82A 2805          2282          JR      Z,CVTERR        ;DIRECTORY FULL... ERRO
                                 R
                   2283 ;<11> CONVERT OK, SET EXIT VALUE ZERO.
E82C C1            2284 CVTOK:   POP     BC              ;NO ERROR... CLEAR STAC
                                 K
E82D AF            2285          XOR     A               ;SET EXIT VALUE ZERO
E82E C315E0        2286          JP      EXIT1B
                   2287 ;<12> CONVERT ERROR, SET EXIT VALUE WITH ERROR CODE.
E831 E5            2288 CVTERR:  PUSH    HL              ;SAVE ERROR CODE IN L (
                                 03,04,05)
E832 CD66E2        2289          CALL    GETS2           ;GET BYTE S2
E835 36C0          2290          LD      (HL),0C0H       ;  TO SET BIT
E837 E1            2291          POP     HL              ;RESTORE ERROR CODE
E838 C1            2292 CVTER2:  POP     BC              ;CLEAR STACK
E839 7D            2293          LD      A,L             ;GET ERROR CODE (06 IF
                                 CVTER2)
E83A 3264E0        2294          LD      (EXTVAL),A      ;  TO RETURN AS EXIT VA
                                 LUE
E83D C375E2        2295          JP      SETS2           ;SET HIGH BIT OF S2.
                   2296 ;;;-------------------------------------------------
                                 --
                   2297 ;;;  READ RANDOM.
                                   =
                   2298 ;;;-------------------------------------------------
                                 --
E840 0EFF          2299 RRADOM:  LD      C,0FFH          ;SET ACCESS TYPE TO REA
                                 D IN C
E842 CD87E7        2300          CALL    CVRREC          ;CONVERT RANDOM RECORD
E845 CC85E6        2301          CALL    Z,COMONR        ;GOTO READ RANDOM.
E848 C9            2302          RET
                   2303 ;;;-------------------------------------------------
                                 --
                   2304 ;;;   WRITE RANDOM.
                                   =
                   2305 ;;;-------------------------------------------------
                                 --
E849 0E00          2306 WRADOM:  LD      C,00H           ;SET ACCESS TYPE TO WRI
                                 TE IN C
E84B CD87E7        2307          CALL    CVRREC          ;CONVERT RANDOM RECORD
E84E CCC3E6        2308          CALL    Z,COMONW        ;GOTO WRITE RANDOM.
E851 C9            2309          RET
                   2310 ;;;=================================================
                                 ==
                   2311 ;;;    FIND RANDOM RECORD NUMBER OF CR/RC AND FILL IN
                                   =
```

```
                        2312 ;;;  RANDOM FILED (R0,R1 & R2).
                                  =
                        2313 ;;;    CONDITION: DE IS OFFSET TO CR OR RC.
                                  =
                        2314 ;;;   FIND THE SUM OF  S2*32*128 + EXT#*128 + (CR/RC)
                                  =
                        2315 ;;;  AND PUT IN R0,R1 AND R2 (R0 LEAST, R1 HIGH AND
                                  =
                        2316 ;;;  R2=01 IS OVERFLOW, MORE THAN 65536).
                                  =
                        2317 ;;;==================================================
                             ==
                        2318 ;<1> GET CR/RC
E352  EB                2319 RANREC: EX      DE,HL          ;DE POINT FCB ADDRESS
E353  19                2320         ADD     HL,DE          ;HL POINT CR/RC
E354  4E                2321         LD      C,(HL)
E355  0600              2322         LD      B,00H          ;GET CR/RC IN BC (BC =
                             00XX)
E357  210C00            2323         LD      HL,000CH       ;OFFSET TO EXT#
E35A  19                2324         ADD     HL,DE          ;HL POINT EXT#
                        2325 ;<2> ADD EXT#*128 WITH CR/RC
                        2326 ;  NOTE: EXT# HAS 5 BITS (BIT 0-4)
                        2327 ;        BIT 0*128 BY SHIFT TO BIT 7 (EQUAL MULTIPLY WI
                             TH 128)
                        2328 ;        BIT 4,3,2 & 1*128 BY INSTEAD OF MULTIPLY WITH
                             128, THEY
                        2329 ;          ARE DIVIDED BY 2 FOR CHANGE TO BASE 256.
E35B  7E                2330         LD      A,(HL)         ;GET EXT#
E35C  0F                2331         RRCA                   ;ROTATE BIT 0 TO BIT 7
E35D  E680              2332         AND     80H            ; THIS BIT ONLY
E35F  81                2333         ADD     A,C            ; (EQUAL BIT 0*128)
E360  4F                2334         LD      C,A
E361  3E00              2335         LD      A,00H
E363  88                2336         ADC     A,B
E364  47                2337         LD      B,A            ;ADD WITH CR/RC IN BC
E365  7E                2338         LD      A,(HL)         ;GET EXT#
E366  0F                2339         RRCA                   ;ROTATE RIGHT (DIVIDE B
                             Y 2)
E367  E60F              2340         AND     0FH            ;  OF BIT 4,3,2 & 1 (CH
                             ANGE TO BASE 256)
E369  80                2341         ADD     A,B            ;ADD WITH BASE 256 OF R
                             ESULT
E36A  47                2342         LD      B,A
                        2343 ;<3> ADD S2*32*128 WITH EXT#*128 + CR/RC
                        2344 ;  NOTE: FIND S2*32*128 BY MULTIPLY S2 WITH 16 FOR IN B
                             ASE 256.
E36B  213E00            2345         LD      HL,003EH       ;OFFSET TO BYTE S2
E36E  19                2346         ADD     HL,DE          ;HL POINT BYTE S2
E36F  7E                2347         LD      A,(HL)         ;GET BYTE S2
E370  87                2348         ADD     A,A
E371  87                2349         ADD     A,A
E372  87                2350         ADD     A,A            ;(FIND S2*32*128 BY)
E373  87                2351         ADD     A,A            ;S2*16 BASE 256
E374  F5                2352         PUSH    AF
E375  80                2353         ADD     A,B            ;ADD WITH BASE 256 OF R
                             ESULT
E376  47                2354         LD      B,A
```

```
E877 F5          2355        PUSH    AF
E878 E1          2356        POP     HL
E879 7D          2357        LD      A,L
E87A E1          2358        POP     HL
E87B B5          2359        OR      L
E87C E601        2360        AND     01H             ;A=01 IF OVERFLOW.
E87E C9          2361        RET
                 2362 ;;;=====================================================
                      ==
                 2363 ;;;   CALCULATE FILE SIZE. RETURN MAX RANDOM RECORD
                           =
                 2364 ;;; NUMBER IN RANDOM FIELD.
                           =
                 2365 ;;;=====================================================
                      ==
                 2366 ;(1) SEARCH FDE ALL EXTENT. EACH EXTENT AT A TIME.
E87F 0E0C        2367 CALSIZ: LD      C,0CH           ;SET LENGTH TO COMPARE
                      USER, FN&FT
E881 CD04E4      2368        CALL    SERFIR          ;  TO SEARCH FIRST
                 2369 ;(2) CLEAR RANDOM FIELD FOR RETURN FILE SIZE. CHECK RES
                      ULT OF
                 2370 ;  SEARCH. IF NOT FOUND AT THE FIRST TIME, RANDOM FILED
                       SET
                 2371 ;  RETURN ZERO.
E884 2A62E0      2372        LD      HL,(SENT2B)     ;GET FCB ADDRESS
E887 112100      2373        LD      DE,0021H        ;OFFSET TO R0
E88A 19          2374        ADD     HL,DE           ;HL POINT R0
E88B E5          2375        PUSH    HL              ;SAVE ADDRESS OF R0
E88C 72          2376        LD      (HL),D          ;CLEAR R0
E88D 23          2377        INC     HL
E88E 72          2378        LD      (HL),D          ;CLEAR R1
E88F 23          2379        INC     HL
E890 72          2380        LD      (HL),D          ;CLEAR R2
E891 CDEFE2      2381 LOSIZE: CALL    CHKDIR          ;CHECK RESULT OF SEARCH
E894 2820        2382        JR      Z,FNDSIZ        ;DIR END.. NOT FOUND, R
                      ETURN SIZE=00
                 2383 ;(3) SEARCH FOUND. GET RECORD COUNT TO FIND RANDOM NUMB
                      ER OF
                 2384 ;  THIS EXTENT.
E896 CD5BE2      2385        CALL    DIRETY          ;SET HL POINT FDE
E899 110F00      2386        LD      DE,000FH        ;OFFSET TO RC
E89C CD52E8      2387        CALL    RANREC          ;CONVERT RC TO RANDOM N
                      UMBER
E89F E1          2388        POP     HL              ;(RESTORE ADDRESS OF R0
                      )
E8A0 E5          2389        PUSH    HL
                 2390 ;(4) COMPARE RANDOM NUMBER OF PREVIOUS EXTENT WITH RAND
                      OM NUMBER
                 2391 ;  OF THIS EXTENT (RTN RANREC RETURN RANDOM NUMBER IN A
                      , B & C).
E8A1 5F          2392        LD      E,A             ;SAVE CALCULATE R2 IN E
E8A2 79          2393        LD      A,C             ;GET CALCULATE R0 FROM
                      C
E8A3 96          2394        SUB     (HL)            ;COMPARE WITH PREVIOUS
                      R0
E8A4 23          2395        INC     HL              ;SKIP TO R1
E8A5 78          2396        LD      A,B             ;GET CALCULATE R1 FROM
```

```
                                B
E8A5  9E          2397          SBC     A,(HL)          ;COMPARE WITH PREVIOUS
                                R1
E8A7  23          2398          INC     HL              ;SKIP TO R2
E8A8  7B          2399          LD      A,E             ;GET CALCULATE R2
E8A9  9E          2400          SBC     A,(HL)          ;COMPARE WITH PREVIOUS
                                R2
E8AA  3805        2401          JR      C,NOREPA        ;NEW < OLD, SEARCH NEXT
                                 EXTENT
                  2402 ;<5> IF NEW < OLD, GOTO SEARCH NEXT EXTENT TO COMPARE.
                  2403 ;  BUT IF NEW > OLD, REPLACE THE OLD WITH NEW BEFORE SE
                                ARCH NEXT.
E8AC  73          2404          LD      (HL),E          ;REPLACE R2
E8AD  2B          2405          DEC     HL              ;(MOVE BACK TO R1)
E8AE  70          2406          LD      (HL),B          ;REPLACE R1
E8AF  2B          2407          DEC     HL              ;(MOVE BACK TO R0)
E8B0  71          2408          LD      (HL),C          ;REPLACE R0
                  2409 ;<6> SEARCH NEXT EXTENT TO COMPARE (AND REPLACE).
E8B1  CD19E4      2410 NOREPA: CALL     SERNXT          ;SEARCH NEXT FDE
E8B4  1808        2411          JR      LOSIZE          ; AND GET NEW TO COMPAR
                                E (AND REPLACE).
                  2412 ;<7> SO WHEN SEARCH END, RANDOM FIELD WILL KEEP THE MAX
                                 RANDOM
                  2413 ;  RECORD NUMBER OF FILE.
E8B6  E1          2414 FNDSIZ: POP      HL              ;POP TO CLEAR STACK
E8B7  C9          2415          RET                     ;RETURN WITH FILE SIZE IN RANDO
                                M FIELD.. FUNC END
                  2416 ;;;=============================================================
                                ==
                  2417 ;;;  FUNC36 GET RANDOM RECORD.
                                =
                  2418 ;;;=============================================================
                                ==
E8B8  2A42E8      2419 FUNC36: LD       HL,(SENT23)     ;HL POINT FCB
E8BB  112100      2420          LD      DE,0021H        ;DE KEEP OFFSET TO CR (
                                SENT TO RTN RANREC)
E8BE  CD31E8      2421          CALL    RANREC          ;CALCULATE RANDOM NUMBE
                                R OF CR
E8C1  212100      2422          LD      HL,0021H        ;OFFSET TO BYTE R1
E8C4  19          2423          ADD     HL,DE           ;HL POINT R0
E8C5  71          2424          LD      (HL),C          ;FILL RANDOM NUMBER IN
                                R0
E8C6  23          2425          INC     HL
E8C7  70          2426          LD      (HL),B          ;FILL IN R1
E8C8  23          2427          INC     HL
E8C9  77          2428          LD      (HL),A          ;  AND FILL IN R2
E8CA  C9          2429          RET                     ;RETURN RANDOM RECORD NUMBER IN
                                 RANDOM FIELD.. FUNC END
                  2430 ;;;-------------------------------------------------------------
                                --
                  2431 ;;;  LOGON DRIVE ASSIGNED IN FCB.
                                =
                  2432 ;;;    CONDITION: ASSIGN DRIVE HAS ALREADY CHANGED TO
                                =
                  2433 ;;;                BE CURRENT DRIVE.
                                =
                  2434 ;;;-------------------------------------------------------------
```

```
                        2435 ;<1> LOGON ASSIGN DRIVE AND MOVE DISK PARAMETER BLOCK
                        2436 ;  FROM I/O INTO DOS AREA.
E8C8  2A8CEA            2437 LOGASG: LD      HL,(LOGVEC)     ;GET LOGIN VECTOR (0000
                              IF FROM RESET DRIVE)
E8CE  3A5FE0            2438         LD      A,(CDRIVE)      ;GET CDRIVE CODE
E8D1  4F                2439         LD      C,A             ;  IN C
E8D2  CDF0E1            2440         CALL    SFTRHT          ; TO SHIFT LOGVEC BIT O
                           F CDRIVE
E8D5  E5                2441         PUSH    HL              ;SAVE RETURN SHIFT LOGV
                           EC
E8D6  E3                2442         EX      DE,HL           ;GET LOGVEC BIT OF CDRI
                           VE IN E
E8D7  CD6FE0            2443         CALL    DSK3CK          ;  TO MOVE DISK PARAMET
                           ER BLOCK
E8DA  E1                2444         POP     HL              ;RESTORE SHIFT LOGVEC
E8DB  CCA6DD            2445         CALL    Z,PINER2        ;SELECT ERROR, PRINT 'S
                           ELECT NON-EXIST'*****
                        2446 ;<2> CHECK IF THIS DRIVE HAD BEEN LOGON BEFORE, IF NOT
                           MASK LOGVEC BIT
                        2447 ;  AND CREATE CSV AND ALV OF THIS DRIVE.
E8DE  7D                2448         LD      A,L             ;GET LOGVEC BIT OF CDRI
                           VE (IN L)
E8DF  1F                2449         RRA                     ;  TO CHECK IF LOGON BE
                           FORE
E8E0  D8                2450         RET     C               ;YES,RETURN... FUNC END
E8E1  2A8CEA            2451         LD      HL,(LOGVEC)     ; ELSE GET LOGIN VECTOR
E8E4  4D                2452         LD      C,L
E8E5  44                2453         LD      B,H             ;  MOVE TO BC
E8E6  CDDEE2            2454         CALL    MSKVEC          ;  FOR MASK LOGVEC BIT
                           OF CDRIVE
E8E9  228CEA            2455         LD      (LOGVEC),HL     ;AND REPLACE THE OLD
E8EC  C3F4E3            2456         JP      CSVALV          ;GOTO CREATE CSV & ALV
                           OF THIS DRIVE.
                        2457 ;;;=================================================
                           ==
                        2458 ;;;  FUNC14 SELECT DRIVE.
                                 =
                        2459 ;;;=================================================
                           ==
                        2460 ;<1> CHECK IF ASSIGN DRIVE IS CURRENT DRIVE... RETURN
E8EF  3A9CEA            2461 FUNC14: LD      A,(SENT1B)      ;GET ASSIGN DRIVE CODE
E8F2  215FE0            2462         LD      HL,CDRIVE       ;GET CURRENT DRIVE
E8F5  BE                2463         CP      (HL)            ;  COMPARE
E8F6  C8                2464         RET     Z               ; EQUAL,SAME DRIVE...RE
                           TURN
                        2465 ;<2> ELSE USE ASSIGN DRIVE AS CURRENT DRIVE AND GOTO LO
                           GON.
E8F7  77                2466         LD      (HL),A          ; ELSE USE ASSIGN DRIVE
                              AS CDRIVE
E8F8  18D1              2467         JR      LOGASG          ;  AND GOTO LOGON
                        2468 ;;;=================================================
                           ==
                        2469 ;;;  LOGON SELECTED DRIVE OF FC8.
                                 =
                        2470 ;;;=================================================
                           ==
```

```
                        2471 ;<1> CHECK DR BYTES IF ASSIGN '?' MEANS GET ALL USER
E8FA 3EFF               2472 LOGSEL: LD     A,0FFH          ;SET DIRCOD = FF
E8FC 329BEA             2473         LD     (DIRCOD),A      ;  TO USE IF FUNC ERROR
E8FF 2A62E8             2474         LD     HL,(SENT2B)     ;HL POINT FCB ADDRESS
E902 7E                 2475         LD     A,(HL)          ;GET SELNUM FROM DR
E903 E61F               2476         AND    1FH             ;STRIP OUT BIT NOT USE
                             (SELNUM=X XXXX)
E905 3D                 2477         DEC    A               ;DECRE SELNUM TO BE COD
                             E
E906 3293EA             2478         LD     (SENT1B),A      ;  AND SAVE FOR SELECT
                             DRIVE (FUNC14)
E909 FE1E               2479         CP     1EH             ;CHECK IF DR = '?' (1E
                             COME FROM 3F)
E90B 3010               2480         JR     NC,MERUSR       ;YES, DR = '?'... GET A
                             LL USER
                        2481 ;<2> DR NOT '?', SAVE CDRIVE FOR LOGON BACK WHEN THIS F
                             UNC END,
                        2482 ;  SAVE SELNUM FOR LATER USE. CLEAR DR BYTE TO FILL WIT
                             H USER CODE.
                        2483 ;  LOGON ASSIGN DRIVE FROM SELNUM.
E90D 3A5FE0             2484         LD     A,(CDRIVE)      ;GET CDRIVE TO SAVE
E910 329CEA             2485         LD     (CDVBAK),A      ;  FOR SELECT BACK WHEN
                              FUNC END
E913 7E                 2486         LD     A,(HL)          ;GET ASSIGN SELNUM
E914 329DEA             2487         LD     (SELNUM),A      ;  AND SAVE FOR LATER U
                             SE
E917 E6E0               2488         AND    0E0H            ;CLEAR SELNUM
E919 77                 2489         LD     (HL),A          ;  OF FCB
E91A CDEFE8             2490         CALL   FUNC14          ;   TO SELECT ASSIGN DR
                             IVE (BY FUNC14)
                        2491 ;<3> FILL DR WITH USER CODE EXCEPT ASSIGN DR = '?' WILL
                              GET ALL USER
                        2492 ;  BECAUSE 3F TAKE ALL BIT OF USER CODE.
E91D 3A5EE0             2493 MERUSR: LD     A,(USRCOD)      ;GET CURRENT USER
E920 2A62E0             2494         LD     HL,(SENT2B)     ;HL POINT DR OF FCB
E923 B6                 2495         OR     (HL)            ;MERGE WITH USER CODE
E924 77                 2496         LD     (HL),A          ;  AND FILL BACK IN DR.
E925 C9                 2497         RET
                        2498 ;;;=====================================================
                             ==
                        2499 ;;;  FUNC12 RETURN VERSION NUMBER.
                                  =
                        2500 ;;;=====================================================
                             ==
E926 3E22               2501 FUNC12: LD     A,22H           ;LOAD VERSION# 2.2
E928 C31BE0             2502         JP     EXIT1B          ;  AND SAVE AS EXIT VAL
                             UE.
                        2503 ;;;=====================================================
                             ==
                        2504 ;;;  FUNC13  RESET DRIVE.
                                  =
                        2505 ;;;=====================================================
                             ==
                        2506 ;<1> RELEASE R/O VECTOR AND LOGIN VECTOR OF ALL DRIVES.
E92B 210000             2507 FUNC13: LD     HL,0000H
E92E 226AEA             2508         LD     (ROVEC),HL      ;RELEASE R/OVEC OF ALL
                             DRIVES
```

```
E931 226CEA     2509        LD      (LOGVEC),HL     ;RELEASE LOGVCE OF ALL
                            DRIVES
                2510 ;<2> LOGON ONLY DRIVE A, SET DMA ADDRESS AT 80H.
E934 AF         2511        XOR     A
E935 325FE0     2512        LD      (CDRIVE),A      ;SELECT DRIVE A
E938 218000     2513        LD      HL,SYSDMA
E93B 226EEA     2514        LD      (DOSDMA),HL     ;SET DMA ADDRESS BACK T
                            O 80H
E93E CD03E2     2515        CALL    SETADD
E941 1888       2516        JR      LOGASG          ;LOGON DRIVE A.
                2517 ;;;=================================================
                            ==
                2518 ;;;  FUNC15  OPEN FILE.
                                =
                2519 ;;;=================================================
                            ==
E943 CD6FE2     2520 FUNC15: CALL   CLR32           ;CLEAR BYTE 32 OF FCB
E946 CDFAE3     2521        CALL    LOGSEL          ;LOGON DRIVE ASSIGN IN
                            FCB
E949 C329E5     2522        JP      OPNETY          ;OPEN DIRECTORY ENTRY O
                            F FILE.
                2523 ;;;=================================================
                            ==
                2524 ;;;  FUNC16  CLOSE FILE.
                                =
                2525 ;;;=================================================
                            ==
E94C CDFAE3     2526 FUNC16: CALL   LOGSEL          ;LOGON SELECT DRIVE
E94F C37BE5     2527        JP      WRTFCB          ;WRITE FCB TO DISK.
                2528 ;;;=================================================
                            ==
                2529 ;;;  FUNC17 SEARCH FOR FIRST.
                                =
                2530 ;;;=================================================
                            ==
                2531 ;<1> CHECK IF ASSIGN DR IN FCB IS '?', SEARCH ALL FILES
                2532 ;  BY SET LENGTH TO COMPARE IS 0.
E952 0E00       2533 FUNC17: LD     C,00H           ;PREPARE LENGTH OF COMP
                            ARE TO 0
E954 EB         2534        EX      DE,HL           ;HL POINT FCB ADDRESS
E955 7E         2535        LD      A,(HL)          ;GET DR ASSIGN IN FCB
E956 FE3F       2536        CP      '?'             ;CHECK IF '?'
E958 2B0E       2537        JR      Z,SERALL        ;ASSIGN DR = '?', SEARC
                            H ALL FILES
                2538 ;<2> DR NOT '?', LOGON SELECT DRIVE ASSIGN IN DR.
E95A CDADE1     2539        CALL    PNTEXT          ;SET HL POINT EXT# OF F
                            CB
E95D 7E         2540        LD      A,(HL)          ;GET EXT# ASSIGN IN FCB
E95E FE3F       2541        CP      '?'             ;CHECK IF '?'
E960 C46FE2     2542        CALL    NZ,CLR32        ;EXT# NOT '?', CLEAR HI
                            GH BIT OF S2
E963 CDFAE3     2543        CALL    LOGSEL          ;LOGON SELECT DRIVE
E966 0E0F       2544        LD      C,0FH           ;SET LENGTH TO COMPARE
                            USER, FN&FT AND EXT#
                2545 ;<3> SEARCH FOR BY LENGTH OF COMPARE IS 0 IF DR ='?'
                2546 ;  OR LENGTH IS 15 IF DR = SELNUM.
E968 CD04E4     2547 SERALL: CALL   SERFIR          ;SEARCH
```

```
E968  CDE3E2      2548          JP      MOVETY          ;MOVE FDE IN DIRBUF TO
                                80H, 128 BYTES.
                  2549 ;;;=================================================
                                ==
                  2550 ;;;  FUNC13 SEARCH NEXT.
                                =
                  2551 ;;;=================================================
                                ==
E96E  2A96EA      2552 FUNC13: LD       HL,(FCBADD)     ;GET FCB ADDRESS SAVE B
                                Y SEARCH FIRST
E971  2262E0      2553          LD      (SENT2B),HL     ;AND SAVE AS SENT VALUE
                                 OF THIS FUNC
E974  CDFAE3      2554          CALL    LOGSEL          ;LOGON SELECT DRIVE IN
                                FDE
E977  CD19E4      2555          CALL    SERNXT          ;SEARCH
E97A  CDE3E2      2556          JP      MOVETY          ;MOVE FDE IN DIRBUF TO
                                80H.
                  2557 ;;;=================================================
                                ==
                  2558 ;;;  FUNC19 DELETE FILE.
                                =
                  2559 ;;;=================================================
                                ==
E97D  CDFAE3      2560 FUNC19: CALL    LOGSEL          ;LOGON SELECT DRIVE
E980  CD72E4      2561          CALL    DELFIL          ;DELETE FILE
E983  CDEDE3      2562          JP      FFOR00          ;SET EXIT VALUE FF OR 0
                                0.
                  2563 ;;;=================================================
                                ==
                  2564 ;;;  FUNC20  READ SEQUENTIAL.
                                =
                  2565 ;;;=================================================
                                ==
E986  CDF9E3      2566 FUNC20: CALL    LOGSEL          ;LOGON SELECT DRIVE
E989  C3B3E6      2567          JP      RDSEQ           ;GOTO READ SEQ.
                  2568 ;;;=================================================
                                ==
                  2569 ;;;  FUNC21  WRITE SEQUENTIAL.
                                =
                  2570 ;;;=================================================
                                ==
E98C  CDFAE3      2571 FUNC21: CALL    LOGSEL          ;LOGON SELECT DRIVE
E98F  C3BEE6      2572          JP      WRTSEQ          ;GOTO WRITE SEQ.
                  2573 ;;;=================================================
                                ==
                  2574 ;;;  FUNC22  MAKE FILE.
                                =
                  2575 ;;;=================================================
                                ==
E992  CD6FE2      2576 FUNC22: CALL    CLR32           ;CLEAR BYTE 32 OF FCB
E995  CDFAE3      2577          CALL    LOGSEL          ;LOGON SELECT DRIVE
E998  C3F1E3      2578          JP      MAKETY          ;MAKE NEW DIRECTORY ENT
                                RY.
                  2579 ;;;=================================================
                                ==
                  2580 ;;;  FUNC23  RENAME FILE.
                                =
```

```
                        2581 ;;;=================================================
                             ==
E998  CDFAE3            2582 FUNC23: CALL    LOGSEL          ;LOGON SELECT DRIVE
E99E  CDF0E4            2583         CALL    RENAM           ;GOTO RENAME
E9A1  C3E0E3            2584         JP      FFOR00          ;SET EXIT VALUE FF OR 0
                             0.
                        2585 ;;;=================================================
                             ==
                        2586 ;;;  FUNC24  GET LOGIN VECTOR.
                                  =
                        2587 ;;;=================================================
                             ==
E9A4  2A6CEA            2588 FUNC24: LD      HL,(LOGVEC)     ;GET LOGIN VECTOR
E9A7  1823              2589         JR      EXIT2B          ;  AND SAVE AS EXIT VAL
                             UE.
                        2590 ;;;=================================================
                             ==
                        2591 ;;;  FUNC25  FIND CURRENT DRIVE.
                                  =
                        2592 ;;;=================================================
                             ==
E9A9  3A5FE0            2593 FUNC25: LD      A,(CDRIVE)      ;GET CURRENT DRIVE CODE
E9AC  C312E0            2594         JP      EXIT1B          ;  AND SAVE AS EXIT VAL
                             UE.
                        2595 ;;;=================================================
                             ==
                        2596 ;;;  FUNC26  SET DMA ADDRESS.
                                  =
                        2597 ;;;=================================================
                             ==
E9AF  EB                2598 FUNC26: EX      DE,HL           ;GET ASSIGN DMA
E9B0  226EEA            2599         LD      (DOSDMA),HL     ;SAVE IN DMA OF DOS
E9B3  C305E2            2600         JP      SETADD          ;  TO SET DMA ADDRESS.
                        2601 ;;;=================================================
                             ==
                        2602 ;;;  FUNC27  GET ADDRESS OF ALLOCATION VECTOR.
                                  =
                        2603 ;;;=================================================
                             ==
E9B6  2A7DEA            2604 FUNC27: LD      HL,(ADDALV)     ;GET ADDRESS OF ALV
E9B9  1811              2605         JR      EXIT2B          ;  AND SAVE AS EXIT VAL
                             UE.
                        2606 ;;;=================================================
                             ==
                        2607 ;;;  FUNC29  GET READ/ONLY VECTOR.
                                  =
                        2608 ;;;=================================================
                             ==
E9BB  2A6AEA            2609 FUNC29: LD      HL,(ROVEC)      ;GET READ/ONLY VECTOR
E9BE  180C              2610         JR      EXIT2B          ;  AND SAVE AS EXIT VAL
                             UE.
                        2611 ;;;=================================================
                             ==
                        2612 ;;;  FUNC30  SET FILE ATTRIBUTES.
                                  =
                        2613 ;;;=================================================
                             ==
```

```
E9C0  CDFAE3      2614 FUNC30: CALL    LOGSEL          ;LOGON SELECT DRIVE
E9C3  CD14E5      2615         CALL    SETATI          ;GOTO SET ATTRIBUTES
E9C6  C3EDE3      2616         JP      FFOR00          ;SET EXIT VALUE FF OR 0
                       0.
                  2617 ;;;=================================================
                       ==
                  2618 ;;;  FUNC31  GET ADDRESS OF DISK PARAMETER BLOCK.
                            =
                  2619 ;;;=================================================
                       ==
E9C9  2A78EA      2620 FUNC31: LD      HL,(ADDPB)      ;GET ADDRESS OF DPB
                  2621 ;;;=================================================
                       ==
                  2622 ;;;  EXIT 2 BYTES IS COMMON ROUTINE OF ALL FUNCTIONS
                            =
                  2623 ;;; WHICH RETURN VALUE LENGTH 2 BYTES TO CALLING PROG.
                            =
                  2624 ;;;=================================================
                       ==
E9CC  2264E8      2625 EXIT2B: LD      (EXTVAL),HL     ;  AND SAVE AS EXIT VAL
                       UE.
E9CF  C9          2626         RET             ;RETURN TO EXIT ROUTINE.
                  2627 ;;;=================================================
                       ==
                  2628 ;;;  FUNC32  GET/SET USER.
                            =
                  2629 ;;;=================================================
                       ==
                  2630 ;<1> IF SENT VALUE IS FF, DECLEAR TO GET USER
E9D0  3A7CEA      2631 FUNC32: LD      A,(SENT1B)      ;GET SENT VALUE
E9D3  FEFF        2632         CP      0FFH            ;CHECK IF FF
E9D5  2006        2633         JR      NZ,SETUSR       ;NOT FF, GOTO SET USER
E9D7  3A5EE8      2634         LD      A,(USRCOD)      ;  ELSE GET USER CODE
E9DA  C31BE8      2635         JP      EXIT1B          ;  AND SAVE AS EXIT VAL
                       UE
                  2636 ;<2> SENT VALUE NOT FF, IT IS USER TO SET
E9DD  E61F        2637 SETUSR: AND     1FH             ;MOD CODE IN RANGE 0-31
E9DF  325EE8      2638         LD      (USRCOD),A      ;  AND USE AS CURRENT U
                       SER.
E9E2  C9          2639         RET
                  2640 ;;;=================================================
                       ==
                  2641 ;;;  FUNC33  READ RANDOM.
                            =
                  2642 ;;;=================================================
                       ==
E9E3  CDFAE3      2643 FUNC33: CALL    LOGSEL          ;LOGON SELECT DRIVE
E9E6  C340E3      2644         JP      RRADOM          ;GOTO READ RANDOM.
                  2645 ;;;=================================================
                       ==
                  2646 ;;;  FUNC34 WRITE RANDOM.
                            =
                  2647 ;;;=================================================
                       ==
E9E9  CDFAE3      2648 FUNC34: CALL    LOGSEL          ;LOGON SELECT DRIVE
E9EC  C349E3      2649         JP      WRADOM          ;GOTO WRITE RANDOM.
                  2650 ;;;=================================================
```

```
                        ==
                        2651 ;;;  FUNC35  GET FILE SIZE.
                                  =
                        2652 ;;;=============================================================
                             ==
E9EF  CDFAE3            2653 FUNC35: CALL    LOGSEL          ;LOGON SELECT DRIVE
E9F2  C37FEB            2654         JP      CALSIZ          ;GOTO CALCULATE FILE SI
                             ZE.
                        2655 ;;;=============================================================
                             ==
                        2656 ;;;  FUNC37  RESET ASSIGN DRIVE.
                                  =
                        2657 ;;;=============================================================
                             ==
                        2658 ;<1> COMPLEMENT ASSIGN BIT MAP.
E9F5  2A62E9            2659 FUNC37: LD      HL,(SENT2B)     ;HL KEEP ASSIGN BIT MAP
                              (0=SET/ 1=RESET)
E9F8  7D                2660         LD      A,L             ; (ASSIGN BIT MAP IS 0=
                             SET
E9F9  2F                2661         CPL                     ;   AND 1=RESET. SO MUS
                             T BE
E9FA  5F                2662         LD      E,A             ;   COMPLEMENTED TO THE
                              SAME
E9FB  7C                2663         LD      A,H             ;   STANDARD AS LOGON V
                             ECTOR).
E9FC  2F                2664         CPL                     ;COMPLEMENT ASSIGN BIT
                             MAP IN AE
                        2665 ;<2> AND WITH OLD LOGIN VECTOR TO RESET DRIVE.
E9FD  2A6CEA            2666         LD      HL,(LOGVEC)     ;HL KEEP LOGIN VECTOR
EA00  A4                2667         AND     H
EA01  57                2668         LD      D,A
EA02  7D                2669         LD      A,L
EA03  A3                2670         AND     E               ; AND WITH COMPLEMENT B
                             IT MAP
EA04  5F                2671         LD      E,A             ;  KEEP IN DE
                        2672 ;<3> AND NEW LOGIN VECTOR WITH R/O VECTOR TO SET RESET
                              DRIVE BACK
                        2673 ;       TO R/W (AND CREATE CSV & ALV WHEN LOGON).
EA05  2A6AEA            2674         LD      HL,(ROVEC)      ;GET R/O VECTOR
EA08  E3                2675         EX      DE,HL           ;DE=R/OVEC, HL=NEW LOGI
                             EC
EA09  226CEA            2676         LD      (LOGVEC),HL     ;SAVE NEW LOGIN VECTOR
EA0C  7D                2677         LD      A,L
EA0D  A3                2678         AND     E
EA0E  6F                2679         LD      L,A
EA0F  7C                2680         LD      A,H
EA10  A2                2681         AND     D               ;  AND R/O VECTOR
EA11  67                2682         LD      H,A             ; IN DE
EA12  226AEA            2683         LD      (ROVEC),HL      ;SAVE NEW R/O VECTOR.
EA15  C9                2684         RET
                        2685 ;;;=============================================================
                             ==
                        2686 ;;;  FUNC38  MOVE FILE.
                                  =
                        2687 ;;;=============================================================
                             ==
                        2688 ;<1> LOGON SELECT DRIVE AND SEARCH FILE TO MOVE.
```

```
EA16  CDFAE8    2688 FUNC38: CALL    LOGSEL          ;LOGON SELECT DRIVE
EA19  0E0C      2690         LD      C,0CH           ;SET LENGTH TO COMPARE
                    USER, FN&FT
EA1B  CD04E4    2691         CALL    SERFIR          ;  TO SEARCH FIRST
EA1E  CD5FE2    2692 LMVEFL: CALL    CHKDIR          ;CHECK RESULT OF SEARCH
EA21  CAEDE3    2693         JP      Z,FFOR00        ;DIR END...ALL DONE, FU
                    NC END
                2694 ;<2> WHEN SEARCH FOUND ..FILL OLD USER WITH NEW USER.
EA24  CD5BE2    2695         CALL    DIRETY          ;SET HL POINT FDE
EA27  3A61E0    2696         LD      A,(EXSENT)      ;SET NEW USER
EA2A  77        2697         LD      (HL),A          ;  TO REPLACE THE OLD
                2698 ;<3> WRITE NEW USER BACK TO DISK AND SEARCH NEXT.
EA2B  CD02E2    2699         CALL    DIRDSK          ;WRITE FDE TO DISK
EA2E  CD19E4    2700         CALL    SERNXT          ;AND SEARCH NEXT
EA31  18E3      2701         JR      LMVEFL
                2702 ;;;=============================================
                    ==
                2703 ;;;  FUNC40  WRITE RANDOM WITH ZERO FILL.
                                =
                2704 ;;;=============================================
                    ==
EA33  CDFAE8    2705 FUNC40: CALL    LOGSEL          ;LOGON SELECT DRIVE
EA36  3E02      2706         LD      A,02H
EA38  3292EA    2707         LD      (UFFACT),A      ;SET UPDATE FACTOR=02 (
                    NOT SKIP)
EA3B  0E00      2708         LD      C,00H           ;SET WRTTYP
EA3D  CD83E7    2709         CALL    CVRF40          ;GOTO CONVERT RANDOM RE
                    CORD
EA40  CDC3E6    2710         CALL    Z,COMNW         ;GOTO WRITE RANDOM.
EA43  C9        2711         RET
                2712 ;;;=============================================
                    ==
                2713 ;;;  EXIT ROUTINE IS LAST ROUTINE IN BDOS.
                                =
                2714 ;;;    ALL FUNCTION (EXCEPT PRINT BDOS ERROR MESSAGE)
                                =
                2715 ;;;    MUST COME TO THIS POINT (AFTER SAVE EXIT VALUE
                                =
                2716 ;;;    IF FUNCTION HAS RETURN VALUE) BY 'RET' WHICH
                                =
                2717 ;;;    BDOS PUSH ADDRESS OF THIS ROUTINE IN FIRST ENTRY
                                =
                2718 ;;;    OF BDOS STACK.
                                =
                2719 ;;;=============================================
                    ==
                2720 ;<1> CHECK IF FUNC OK... RETURN TO CALLING PROG
EA44  3A98EA    2721 EXTRTN: LD      A,(DIRCOD)      ;GET DIRCOD
EA47  B7        2722         OR      A               ;CHECK IF FF
EA48  2815      2723         JR      Z,RSTFCB        ;NO,FUNC OK...RETURN TO
                    PROG.
                2724 ;<2> FUNC ERROR... FUNC ABORT,SO MUST LOGON BACK TO OLD
                    CURRENT DRIVE.
EA4A  2A62E0    2725         LD      HL,(SENT23)     ;GET FCB ADDRESS
EA4D  3600      2726         LD      (HL),00H        ;CLEAR DR
EA4F  3A70EA    2727         LD      A,(SELNUM)      ;GET SELNUM OF THIS FCB
EA52  B7        2728         OR      A
```

```
EA53 280A       2729        JR      Z,RSTPOG        ;IT IS DEFAULT... DRIVE
                                NOT CHANGE,RETURN
EA55 77         2730        LD      (HL),A          ;  ELSE PUT SELNUM BACK
                                IN DR
EA56 3A9CEA     2731        LD      A,(CDVBAK)      ;GET OLD CDRIVE TO LOG
                                BACK
EA59 3293EA     2732        LD      (SENT13),A      ;  BY SAVE AS SENT VALUE
EA5C CDEFE8     2733        CALL    FUNC14          ;  OF FUNC14 (SELECT DR
                                IVE).
                2734 ;<3> RETURN TO CALLING PROG BY STACK POINTER WHEN CALL
                                DOS.
                2735 ;  VALUE RETURN IN HL OR A.
EA5F 2A28E0     2736 RSTPOG: LD     HL,(OLDSP)      ;GET STACK POINTER OF C
                                ALLING PROGRAM
EA62 F9         2737        LD      SP,HL           ;RESTORE OLD SP
EA63 2A64E0     2738        LD      HL,(EXTVAL)     ;GET EXIT VALUE
EA66 7D         2739        LD      A,L             ;  TO RETURN IN REG HL
EA67 44         2740        LD      B,H             ;  OR IN REG A
EA68 C9         2741        RET             ;RETURN TO CALLING PROGRAM.###
                2742 ;;;=============================================
                                ==
                2743 ;;;  W O R K   A R E A ::::::::::::::::::::::::::::
                                ::
                2744 ;;;=============================================
                                ==
EA69 E5         2745 ESFCB:  DB     0E5H    ;SIMULATE FCB TO SEARCH DELETED
                                FDE.
EA6A 0000       2746 ROVEC:  DW     0000H   ;READ/ONLY VECTOR
EA6C 0000       2747 LOGVEC: DW     0000H   ;LOGIN VECTOR
EA6E B000       2748 DOSDMA: DW     SYSDMA  ;DEFAULT SYSTEM DMA AT B0H
EA70 0000       2749 ADDDW1: DW     0000H   ;ADDRESS OF ENTRY COUNT
EA72 0000       2750 ADDDW2: DW     0000H   ;ADDRESS OF TRACK COUNT
EA74 0000       2751 ADDDW3: DW     0000H   ;ADDRESS OF SECTOR COUNT
EA76 0000       2752 ADDDIB: DW     0000H   ;ADDRESS OF DIRBUF
EA78 0000       2753 ADDDPB: DW     0000H   ;ADDRESS OF DPB (CURRENT LOGON
                                DRIVE)
EA7A 0000       2754 ADDCSV: DW     0000H   ;ADDRESS OF CSV (CURRENT LOGON
                                DRIVE)
EA7C 0000       2755 ADDALV: DW     0000H   ;ADDRESS OF ALV (CURRENT LOGON
                                DRIVE)
                2756 ;;;  DATA OF DISK PARAMETER BLOCK ::::::::::::::::::::
                                ::
EA7E 0000       2757 SPT:    DW     0000H
EA80 00         2758 BSH:    DB     00H
EA81 00         2759 BLM:    DB     00H
EA82 00         2760 EXM:    DB     00H
EA83 0000       2761 DSM:    DW     0000H
EA85 0000       2762 DRM:    DW     0000H
EA87 00         2763 AL0:    DB     00H
EA88 00         2764 AL1:    DB     00H
EA89 0000       2765 CKS:    DW     0000H
EA8B 0000       2766 OFF:    DW     0000H
EA8D 0000       2767 ADDSKW: DW     0000H   ;ADDRESS OF SKEW TABLE
EA8F 00         2768 FCBSTA: DB     00H     ;FCB STATUS (00=FULL/ FF=AVAILA
                                BLE)
EA90 00         2769 ACCTYP: DB     00H     ;ACCESS TYPE (00=WRITE/ FF=READ
                                )
```

```
EA91  00          2770 MARKER: DB      00H     ;MARKER OF SEARCH FIRST (FF=SEA
                       RCH FIRST NOT FOUND)
EA92  00          2771 UPFACT: DB      00H     ;UPDATE FACTOR (00=RANDOM/ 01=S
                       EQ./ 02=FUNC40)
EA93  00          2772 SENTLB: DB      00H     ;SENT PARAMETER 1 BYTE.
EA94  00          2773 DMPOS:  DB      00H     ;POSITION OF DM IN FDE
EA95  00          2774 LENGER: DB      00H     ;LENGTH TO COMPARE WHEN SEARCH
                       FIRST AND NEXT
EA96  0000        2775 FCBADD: DW      0000H   ;FCB ADDRESS SAVE BY SEARCH FIR
                       ST FOR SEARCH NEXT
EA98  0000        2776         DW      0000H
EA9A  00          2777 DMLEN:  DB      00H     ;LENGTH OF DM (00=DM 2 BYTES/ F
                       F=DM 1 BYTE)
EA9B  00          2778 DIRCOD: DB      00H     ;DIRECTORY CODE 0-3
EA9C  00          2779 CDVBAK: DB      00H     ;CDRIVE TO LOGON BACK WHEN EACH
                        FUNC END
EA9D  00          2780 SELNUM: DB      00H     ;SELECT DRIVE NUMBER (00=DEFAUL
                       T/ 01=DRIVE 1.... )
                  2781 ;;;  VALUE USED FOR FILE OPERATION.  ;;;;;;;;;;T;;;;;;;
                       ;;
EA9E  00          2782 RECCNT: DB      00H     ;RECORD COUNT (RC OF FDE WHEN R
                       EAD/WRITE FILE)
EA9F  00          2783 EXTCNT: DB      00H     ;EXTENT COUNT (EX OF FDE WHEN R
                       EAD/WRITE FILE)
EAA0  00          2784 CURREC: DB      00H     ;CURRENT RECORD (CR OF FDE TO R
                       EAD/WRITE FILE)
EAA1  00          2785         DB      00H
EAA2  0000        2786 DIRSC1: DW      0000H   ;LOGICAL SECTOR TO SLOVE PHYSIC
                       AL TRACK & SECTOR
EAA4  0000        2787 BKXBKZ: DW      0000H   ;BLOCK# * BLOCK SIZE
                  2788 ;;;  VALUE USED FOR DIRECTORY ENTRY.  ;;;;;;;;;;;;;;;;;;
                       ;;
EAA6  00          2789 OFFLEN: DB      00H     ;OFFSET LENGTH IN DIRBUF
EAA7  00          2790 DIRCTL: DB      00H     ;DIRECTORY ENTRY COUNT (LOW BYT
                       E)
EAA8  00          2791 DIRCTH: DB      00H     ;DIRECTORY ENTRY COUNT (HIGH BY
                       TE)
EAA9  0000        2792 DIRSC2: DW      0000H   ;DIRECTORY ENTRY COUNT IN SECTO
                       R
                  2793 ;;;  STRING AREA FOR DOS ERROR MESSAGE.  ;;;
EAAB  3A205265    2794 DERM01: DB      ': Read/Write Disk Error$'
      61642F57
      72697465
      20446973
      6B204572
      726F7224
EAC3  3A2053656   2795 DERM02: DB      ': Select Non-Exist Drive$'
      6C656374
      204E6F6E
      2D457869
      73742044
      72697665
      24
EADC  3A20446973   2796 DERM04: DB      ': Disk'
      6B
EAE2  20577269    2797 DERM03: DB      ' Set Read/Only$'
      74652F4F
      6E6C7924
```

```
CROMEMCO CDOS Z80 ASSEMBLER version 02.15                PAGE 0072

            642F4F6E
            6C7924
EAF1  3A204669  2798 DERMSA: DB      ': File ',QUOTA,'$'
            6D652027
            24
            (EAFA)      2799 FINISH  EQU     $
EAFA  (DD00)        2800         END     START

Errors              0
```

```
                        0001 ;;; EXTERNAL REFERENCE FROM EXTRA. ::::::::::::::
(FC0A)                  0002 AUTFLG  EQU     0FC0AH          ;FLAG OF AUTORUN MODE
(FC0B)                  0003 AUTLEN  EQU     AUTFLG+1        ;LENGTH OF AUTO COMMAND
(FC0C)                  0004 AUTRUN  EQU     AUTFLG+2        ;AUTO COMMAND
(FC25)                  0005 LOGOFG  EQU     0FC25H          ;LOGO FLAG
                        0006 ;;; EQUATES WHICH MUST CHANGEED FROM ONE TO ONE VERSION
                             . :::::::::
(003E)                  0007 MSIZE   EQU     62              ;MEMORY SIZE OF SYSTEM
(0002)                  0008 MAXDSK  EQU     2               ;MAXIMUM # OF DISK DRIV
                             ES
                        0009 ;;;  INTERNAL EQUATES.  ::::::::::::::::::::::::::::::::
                             :::::
(0080)                  0010 INTIOB  EQU     80H             ;INITIAL IOBYTE
(0000)                  0011 WBOT    EQU     0               ;WARM BOOT JUMP ADDRESS
(0003)                  0012 IOBYTE  EQU     3               ;IOBYTE LOCATION
(0004)                  0013 CDISK   EQU     4               ;ADDRESS OF LAST LOGGED
                             DISK
(0005)                  0014 ENTRY   EQU     5               ;DOS ENTRY JUMP ADDRESS
(0080)                  0015 BUFF    EQU     80H             ;DEFAULT BUFFER ADDRESS
(0100)                  0016 TPA     EQU     100H            ;TRANSIENT PROGRAM AREA
(A800)                  0017 BIAS    EQU     (MSIZE-20)*1024 ;MEMORY OFFSET FROM 20K
                             SYSTEM
(D500)                  0018 RPA     EQU     2000H+BIAS      ;RESIDENT PROGRAM AREA
(D806)                  0019 DOS     EQU     RPA+506H        ;DOS ADDRESS
(E300)                  0020 IO      EQU     RPA+1500H       ;I/O ADDRESS
(0008)                  0021 DBLSID  EQU     8               ;SIDE BIT FROM CONTROLL
                             ER
(0004)                  0022 RETRY   EQU     10              ;MAX RETRIES ON DISK I/
                             O BEFORE ERROR
(0000)                  0023 NULL    EQU     00H             ;NULL CHAR
(0009)                  0024 TAB     EQU     09H             ;TAB CHAR
(000A)                  0025 LF      EQU     0AH             ;LINE FEED CHAR
(000D)                  0026 CR      EQU     0DH             ;CARRIAGE RETURN CHAR
(001B)                  0027 ESC     EQU     1BH             ;ESC CHAR
(0027)                  0028 QUOTA   EQU     27H             ;QUOTATION MARK
(00FF)                  0029 SPC     EQU     0FFH            ;SPECIAL CHAR
                        0030 ;;;  FIRMWARE EQUATES FOR ADDRESS OF I/O ROUTINE.  ::::
                             ::::::::::
(F800)                  0031 FIRM    EQU     0F800H
(F803)                  0032 DJCIN   EQU     FIRM+3H         ;CHARACTER INPUT ROUTIN
                             E
(F806)                  0033 DJCOUT  EQU     FIRM+6H         ;CHARACTER OUTPUT ROUTI
                             NE
(F809)                  0034 DJHOME  EQU     FIRM+9H         ;TRACK ZERO SEEK
(F80C)                  0035 DJTRK   EQU     FIRM+0CH        ;TRACK SEEK ROUTINE
(F80F)                  0036 DJSEC   EQU     FIRM+0FH        ;SET SECTOR ROUTINE
(F812)                  0037 DJDMA   EQU     FIRM+012H       ;SET DMA ADDRESS
(F815)                  0038 DJREAD  EQU     FIRM+15H        ;DISK READ ROUTINE
(F818)                  0039 DJWRT   EQU     FIRM+18H        ;DISK WRITE ROUTINE
(F81B)                  0040 DJSEL   EQU     FIRM+1BH        ;SELECT DRIVE ROUTINE
(F821)                  0041 DJTSTA  EQU     FIRM+21H        ;TERMINAL STATUS ROUTIN
                             E
(F827)                  0042 DJSTAT  EQU     FIRM+27H        ;DISK STATUS ROUTINE
(F82A)                  0043 DJERR   EQU     FIRM+2AH        ;DISK ERROR, FLASH LED
(F82D)                  0044 DJDEN   EQU     FIRM+2DH        ;SET DENSITY ROUTINE
(F830)                  0045 DJSIDE  EQU     FIRM+30H        ;SET SIDE ROUTINE
(F833)                  0046 LSTOUT  EQU     FIRM+33H        ;PRINT ON LINE PRINTER
```

```
     (F836)       0047 LSTSTA  EQU    FIRM+36H        ;LINE PRINTER STATUS
                  0048 ;;;-----------------------------------------------------
                       ----------
                  0049 ;;; THE JUMP TABLE BELOW MUST REMAIN IN THE SAME ORDER,
                        THE    :
                  0050 ;;; ROUTINES MAY BE CHANGED, BUT THE FUNCTION EXECUTED
                       MUST BE :
                  0051 ;;; THE SAME.
                                 :
                  0052 ;;;-----------------------------------------------------
                       ----------
0000'             0053         ORG    IO              ;I/O STARTING ADDRESS
E800  C36CF1      0054 START:  JP     CBOOT           ;COLD BOOT ENTRY POINT
E803  C336EB      0055         JP     WBOOT           ;WARM BOOT ENTRY POINT
E806  C321EC      0056         JP     CONST           ;CONSOLE STATUS ROUTINE
E809  C325EC      0057         JP     CONIN           ;CONSOLE INPUT
E80C  C32EEC      0058         JP     CONOUT          ;CONSOLE OUTPUT
E80F  C337EC      0059         JP     LIST            ;LIST DEVICE OUTPUT
E812  C311EC      0060         JP     PUNCH           ;PUNCH DEVICE OUTPUT
E815  C313EC      0061         JP     READER          ;READER DEVICE INPUT
E818  C32EEC      0062         JP     HOME            ;HOME DRIVE
E81B  C30DEC      0063         JP     SETDRV          ;SELECT DISK
E81E  C330EC      0064         JP     SETTRK          ;SET TRACK
E821  C335EC      0065         JP     SETSEC          ;SET SECTOR
E824  C339AED     0066         JP     SETDMA          ;SET DMA ADDRESS
E827  C336AED     0067         JP     READ            ;READ THE DISK
E82A  C343ED      0068         JP     WRITE           ;WRITE THE DISK
E82D  C302EC      0069         JP     LISTST          ;LIST DEVICE STATUS
E830  CCA0ED      0070         JP     SECTRN          ;SECTOR TRANSLATION
E833  C313F8      0071         JP     DJSEL           ;HOOK FOR SINGLE.COM PR
                       OGRAM
                  0072 ;;;-----------------------------------------------------
                       ----------
                  0073 ;;; WBOOT LOADS IN ALL OF SYSTEM EXCEPT THE I/O,THEN IN
                       ITIALIZES:
                  0074 ;;; SYSTEM PARAMETERS AS IN COLD BOOT. SEE THE COLD BOO
                       T LOADER        :
                  0075 ;;; LISTING FOR EXACTLY WHAT HAPPENS DURING WARM AND CO
                       LD BOOTS.:
                  0076 ;;;-----------------------------------------------------
                       ----------
E836  310001      0077 WBOOT:  LD     SP,TPA          ;SET UP STACK POINTER
E839  3E01        0078         LD     A,1
     (E83A)       0079 WFLG   EQU    $-1             ;TEST IF BEGIN (WFLG=1)
                       OR
E83B  A7          0080         AND    A               ;  END (WFLG=0) OF WARM
                       BOOT
E83C  3E01        0081         LD     A,1
E83E  32DAEB      0082         LD     (WFLG),A
E841  32B3E3      0083         LD     (CWFLG),A       ;SET C/WBOOT FLAG TO WA
                       RM BOOT
E844  2876        0084         JR     Z,BDSYS         ;END OF WARM BOOT, DO B
                       DSYS
E846  AF          0085         XOR    A               ;NOT END OF WARM BOOT
E847  32DAEB      0086         LD     (WFLG),A        ;  SET END OF WARM BOOT
                       FOR NEXT PASS
E84A  4F          0087         LD     C,A             ;SELECT DRIVE A
```

```
E848  CD15F3      0088          CALL    DJSEL
E84B  0E00        0089          LD      C,0             ;SELECT SINGLE DENSITY
E84D  CD20F3      0090          CALL    DJDEN
E850  0E00        0091          LD      C,0             ;SELECT SIDE 0
E852  CD30F3      0092          CALL    DJSIDE
E855  3E0F        0093          LD      A,15            ;INITIALIZE THE SECTOR
                        TO READ
E857  327EE8      0094          LD      (NEWSEC),A
E85A  2100D4      0095          LD      HL,RPA-100H     ;AND THE DMA ADDRESS
E85D  227AE8      0096          LD      (NEWDMA),HL
E860  CD77E8      0097          CALL    WARMLD          ;READ IN ALL SYSTEM ON
                        TRACK 0
E863  0100DA      0098          LD      BC,RPA+500H     ;SET DMA ADDRESS
E866  CD12F3      0099          CALL    DJDMA           ;  FOR SLOAD AT RPA+500
                        H
E869  0E09        0100          LD      C,8             ;SET TO READ SLOAD
E86B  CD0FF3      0101          CALL    DJSEC
E86E  CD47E8      0102          CALL    WARMRD          ;READ SLOAD IN RPA+5000H
E871  C303DA      0103          JP      RPA+203H        ;JUMP TO DOWARM IN SLOA
                        D
E874  3E0F        0104 WARMLD:  LD      A,15            ;START SECTOR TO READ
      (E875)      0105 NEWSEC   EQU     $-1             ;PREVIOUS SECTOR #
E876  3C          0106          INC     A               ;UPDATE THE PREVIOUS SE
                        CTOR
E877  3C          0107          INC     A               ;  WITH SKEW FACTOR 2:1
E878  FE13        0108          CP      27              ;WAS IT THE LAST ?
E87A  380F        0109          JR      C,NOWRAP
E87C  0609        0110          LD      B,9             ;YES
E87E  FE13        0111          CP      19
E880  C8          0112          RET     Z
E881  2A7AE8      0113          LD      HL,(NEWDMA)
E884  1180FF      0114          LD      DE,-480H
E887  19          0115          ADD     HL,DE
E888  227AE8      0116          LD      (NEWDMA),HL
E88B  3278E8      0117 NOWRAP:  LD      (NEWSEC),A      ;SAVE THE NEW SECTOR TO
                        READ
E88E  4F          0118          LD      C,A
E88F  CD0FF3      0119          CALL    DJSEC
E892  2100D4      0120          LD      HL,RPA-100H     ;GET THE PREVIOUS DMA A
                        DDRESS
      (E893)      0121 NEWDMA   EQU     $-2
E895  110001      0122          LD      DE,100H         ;UPDATE THE DMA ADDRESS
E898  19          0123          ADD     HL,DE
E899  227AE8      0124          LD      (NEWDMA),HL     ;SAVE THE DMA ADDRESS
E89C  44          0125          LD      B,H
E89D  4D          0126          LD      C,L
E89E  CD12F3      0127          CALL    DJDMA           ;SET THE DMA ADDRESS
E8A1  CD47E8      0128          CALL    WARMRD
E8A4  18CE        0129          JR      WARMLD
E8A6  0110D4      0130 WARMRD:  LD      BC,RETRY*100H+0 ;B=MAX# OF ATTEMPS, C=T
                        RACK 0
E8A9  C5          0131 WREAD:   PUSH    BC              ;SAVE ERROR COUNT
E8AA  CD0CF3      0132          CALL    DJTRK           ;SET THE TRACK
E8AD  CD15F3      0133          CALL    DJREAD          ;READ THE SECTOR
E8B0  C1          0134          POP     BC              ;RESTORE THE ERROR COUN
                        T
E8B1  C9          0135          RET     NC              ;RETURN IF SUCCESSFUL
```

```
E885  05        0136        DEC     B
E886  20F4      0137        JR      NZ,WREAD        ;KEEP TRYING
E888  C32AF8    0138        JP      DJERR           ;TOO MUCH ERROR, FLASH
                      LED
                0139 ;;;----------------------------------------------
                     ----------
                0140 ;;; GOSYS IS THE ENTRY POINT FROM COLD BOOTS, AND WARM
                      BOOTS.  :
                0141 ;;; IT INITIALIZES SOME OF THE LOCATIONS IN PAGE 0, AND
                       SETS UP :
                0142 ;;; THE INITIAL DMA ADDRESS (80H).
                                :
                0143 ;;;----------------------------------------------
                     ----------
E88B  00        0144 CWFLG:  DB      0               ;COLD/WARM BOOT FLAG
E88C  018000    0145 GOSYS:  LD      BC,BUFF         ;SET UP INITIAL DMA ADD
                      RESS
E88F  CD9AEC    0146         CALL    SETDMA
E892  3EC3      0147         LD      A,0C3H          ;INITIALIZE JUMP TO WAR
                      M BOOT
E894  320000    0148         LD      (WBOT),A
E897  32E500    0149         LD      (ENTRY),A       ;INITIALIZE JUMP TO DOS
E89A  2103E6    0150         LD      HL,START+3      ;STORE WARM BOOT ENTRY
                      POINT
E89D  220100    0151         LD      (WBOT+1),HL
E8A0  2106D0    0152         LD      HL,DOS+6        ;STORE CALL DOS ENTRY P
                      OINT
E8A3  220600    0153         LD      (ENTRY+1),HL
E8A6  AF        0154         XOR     A               ;A <- 0
E8A7  32F3F1    0155         LD      (BUFSEC),A      ;DISK BUFFER EMPTY
E8AA  32D1ED    0156         LD      (BUFWRT),A      ;FLAG BUFFER NOT DIRTY
                      (SELF MODIFYING)
E8AD  3A0400    0157         LD      A,(CDISK)       ;JUMP TO RPA WITH
E8B0  4F        0158         LD      C,A             ;  CURRENTLY SELSECTD D
                      ISK IN C
E8B1  11CCFC    0159         LD      DE,AUTRUN       ;BEGINNING OF AUTO RUN
                      COMMAND
E8B4  210803    0160         LD      HL,RPA+8        ;COMMAND BUFFER OF RPA
E8B7  3A06FC    0161         LD      A,(AUTLEN)      ;LENGTH OF COMMAND
E8BA  320705    0162         LD      (RPA+7),A
E8BD  47        0163         LD      B,A
E8BE  CD31EE    0164         CALL    MOVLOP          ;MOVE AUTO RUN TO COMMA
                      ND BUFFER
E8F1  3ABBE8    0165         LD      A,(CWFLG)       ;CWFLG=0 -> COLD BOOT
E8F4  A7        0166         AND     A               ;     =1 -> WARM BOOT
E8F5  3A09FC    0167         LD      A,(AUTFLG)      ;AUTFLG=0 -> NOT AUTO R
                      UN
E8F8  2801      0168         JR      Z,OLDBOT        ;     =1 -> AUTO RUN I
                      N COLD BOOT
E8FA  1F        0169         RRA                     ;     =2 -> AUTO RUN I
                      N WARM BOOT
E8FB  1F        0170 OLDBOT: RRA                     ;     =3 -> BOTH
E8FC  DA0005    0171         JP      C,RPA           ;EXECUTE AUTO RUN
E8FF  C30305    0172         JP      RPA+3           ;DISPLAY PROMPT SIGN
                0173 ;;;----------------------------------------------
                     ----------
                0174 ;;; LISTST: GET THE STATUS OF THE CURRENTLY ASSIGNED LI
```

```
                        ST DEVICE:
                    0175 ;;;----------
                        ----------
EC02  2145EC        0176 LISTST: LD     HL,LSTBLE       ;BEGINNING OF LIST STAT
                        US TABLE
EC05  1803          0177         JR     LIST1
                    0178 ;;;----------
                        ----------
                    0179 ;;; LIST: SELECT A LIST DEVICE BASED ON BITS 6&7 OF IOB
                        YTE    :
                    0180 ;;;----------
                        ----------
EC07  214DEC        0181 LIST:   LD     HL,LTBLE        ;BEGINNING OF LIST DEVI
                        CE TABLE
EC0A  3A0300        0182 LIST1:  LD     A,(IOBYTE)
EC0D  1F            0183         RRA
EC0E  1F            0184         RRA
EC0F  1806          0185         JR     PNCH1
                    0186-;;;----------
                        ----------
                    0187 ;;; PUNCH: SELECT CORRECT PUNCH DEVICE FROM BITS 4&5 OF
                         IOBYTE.       :
                    0188 ;;;----------
                        ----------
EC11  2155EC        0189 PUNCH:  LD     HL,PTBLE        ;BEGINNING OF PUNCH TAB
                        LE
EC14  3A0300        0190         LD     A,(IOBYTE)
                    0191 ;
                    0192 ; ENTRY AT PNCH1 ROTATES BITS A LITTLE MORE IN PREP FOR
                    0193 ; SELDEV, USED BY LIST.
                    0194 ;
EC17  1F            0195 PNCH1:  RRA
EC18  1F            0196         RRA
EC19  1803          0197         JR     READR1
                    0198 ;;;----------
                        ----------
                    0199 ;;; READER: SELECT CORRECT READER DEVICE FROM BITS 2&3
                        OF IOBYTE:
                    0200 ;;;----------
                        ----------
EC1B  215DEC        0201 READER: LD     HL,RTBLE        ;BEGINNING OF READER IN
                        PUT TABLE
                    0202 ;
                    0203 ; ENTRY AT READER1 WILL SHIFT THE BITS INTO POSITION, U
                        SED
                    0204 ; BY LIST AND PUNCH.
                    0205 ;
EC1E  1F            0206 READR1: RRA
EC1F  1819          0207         JR     SELDEV
                    0208 ;;;----------
                        ----------
                    0209 ;;; CONST: GET STATUS OF CURRENTLY ASSIGNED CONSOLE DEV
                        ICE.    :
                    0210 ;;;----------
                        ----------
EC21  2165EC        0211 CONST:  LD     HL,CSTBLE       ;BEGINNING OF CONSOLE S
                        TATUS TABLE
```

```
EC24 1810       0212        JR      CONOT1          ;SELECT CORRECT JUMP
                0213 ;;;----------
                     ----------
                0214 ;;; CONIN: SELECT CORRECT CONSOLE DEVICE FOR THE CONSOL
                     E INPUT      :
                0215 ;;;      ROUTINE FROM BITS 0&1 OF IOBYTE.
                                  :
                0216 ;;;----------
                     ----------
EC26 CDD3ED     0217 CONIN:  CALL    FLUSH           ;FLUSH THE DISK BUFFER
EC29 216DEC     0218         LD      HL,CITBLE       ;BEGINNING OF CONSOLE I
                     NPUT TABLE
EC2C 1808       0219         JR      CONOT1          ;DO THE DECODE
                0220 ;;;----------
                     ----------
                0221 ;;; CONOUT: SELECT CORRECT CONSOLE DEVICE FOR THE CONSO
                     LE OUTPUT:
                0222 ;;;      ROUTINE FROM BITS 0&1 OF IOBYTE.
                                  :
                0223 ;;;----------
                     ----------
EC2E C5         0224 CONOUT: PUSH    BC              ;SAVE THE CHARACTER
EC2F CDD3ED     0225         CALL    FLUSH           ;FLUSH THE DISK BUFFER
EC32 C1         0226         POP     BC              ;RESTORE THE CHARACTER
EC33 2175EC     0227         LD      HL,COTBLE       ;BEGINNING OF CONSOLE O
                     UTPUT TABLE
                0228 ;
                0229 ; ENTRY AT CONOT1 WILL DECODE BITS 0&1 OF IOBYTE. THIS
                     IS USED
                0230 ; BY CONIN,CONOUT, AND CONST.
                0231 ;
EC36 3A0300     0232 CONOT1: LD      A,(IOBYTE)
EC39 17         0233         RLA
                0234 ;;;----------
                     ----------
                0235 ; ENTRY AT SELDEV WILL FORM AN OFFSET INTO THE TABLE PO
                     INTED   :
                0236 ; TO BY H&L AND THEN PICK UP THE ADDRESS AND JUMP THERE
                     .       :
                0237 ;;;----------
                     ----------
EC3A E606       0238 SELDEV: AND     06H             ;STRIP OFF UNWANTED BIT
                     S
EC3C 1600       0239         LD      D,0             ;FORM OFFSET
EC3E 5F         0240         LD      E,A
EC3F 19         0241         ADD     HL,DE           ;ADD OFFSET
EC40 7E         0242         LD      A,(HL)          ;PICK UP HIGH BYTE
EC41 23         0243         INC     HL
EC42 66         0244         LD      H,(HL)          ;PICK UP LOW BYTE
EC43 6F         0245         LD      L,A             ;FORM ADDRESS
EC44 E9         0246         JP      (HL)            ;GO THERE !
                0247 ;;
                0248 ;; LIST STATUS TABLE.
EC45 87EC       0249 LSTBLE: DW      READY           ;CONSOLE ALWAYS READY
EC47 70EC       0250         DW      LSLPT           ;GET LIST STATUS
EC49 71EC       0251         DW      LSLPT
EC4B 70EC       0252         DW      LSLPT
```

```
                        0253 ;;
                        0254 ;; LIST DEVICE TABLE.
EC4D  06F8              0255 LTBLE:  DW      DJCOUT
EC4F  33F8              0256         DW      LSTOUT
EC51  33F8              0257         DW      LSTOUT
EC53  33F8              0258         DW      LSTOUT
                        0259 ;;
                        0260 ;; PUNCH DEVICE TABLE.
EC55  06F8              0261 PTBLE:  DW      DJCOUT
EC57  06F8              0262         DW      DJCOUT
EC59  06F8              0263         DW      DJCOUT
EC5B  06F8              0264         DW      DJCOUT
                        0265 ;;
                        0266 ;; READER DEVICE INPUT TABLE.
EC5D  07F8              0267 RTBLE:  DW      DJCIN
EC5F  07F8              0268         DW      DJCIN
EC61  07F8              0269         DW      DJCIN
EC63  07F8              0270         DW      DJCIN
                        0271 ;;
                        0272 ;; CONSOLE STATUS TABLE.
EC65  86EC              0273 CSTBLE: DW      CSTTY
EC67  86EC              0274         DW      CSTTY
EC69  86EC              0275         DW      CSTTY
EC6B  86EC              0276         DW      CSTTY
                        0277 ;;
                        0278 ;; CONSOLE INPUT TABLE.
EC6D  07F8              0279 CITBLE: DW      DJCIN
EC6F  07F8              0280         DW      DJCIN
EC71  07F8              0281         DW      DJCIN
EC73  07F8              0282         DW      DJCIN
                        0283 ;;
                        0284 ;; CONSOLE OUTPUT TABLE.
EC75  06F8              0285 COTBLE: DW      DJCOUT
EC77  06F8              0286         DW      DJCOUT
EC79  06F8              0287         DW      DJCOUT
EC7B  06F8              0288         DW      DJCOUT
                        0289 ;;;----------------------------------------------
                             ----------
                        0290 ; LIST DEVICE STATUS ROUTINES.
                                      :
                        0291 ;;;----------------------------------------------
                             ----------
EC7D  CD36F8            0292 LSLPT:  CALL    LSTSTA          ;PRINTER STATUS
EC80  3E00              0293         LD      A,0             ;PREP FOR ZERO RETURN
EC82  C0                0294         RET     NZ              ;BUSY, RETURN WITH 00H
EC83  3EFF              0295 READY:  LD      A,0FFH
EC85  C9                0296         RET                     ;READY, RETURN WITH FFH
                        0297 ;;;----------------------------------------------
                             ----------
                        0298 ; CONSOLE STATUS ROUTINES, TEST IF A CHARACTER HAS ARRI
                             VED.    :
                        0299 ;;;----------------------------------------------
                             ----------
EC86  CD21F8            0300 CSTTY:  CALL    DJTSTA          ;CONSOLE STATUS
EC89  3E00              0301         LD      A,0             ;PREP FOR ZERO RETURN
EC8B  C0                0302         RET     NZ              ;NOTHING FOUND, RETURN
                             WITH 00H
```

```
EC8C  3D          0303          DEC     A
EC8D  C9          0304          RET                     ;CHAR READY, RETURN WIT
                                H FFH
                  0305 ;;;----------------------------------------------------
                                ----------
                  0306 ;;; HOME IS TRANSLATED INTO A SEEK TO TRACK ZERO.
                                       :
                  0307 ;;;----------------------------------------------------
                                ----------
EC8E  0E00        0308 HOME:    LD      C,0             ;TRACK TO SEEK TO
                  0309 ;;;----------------------------------------------------
                                ----------
                  0310 ;;; SETTRK SAVES THE TRACK # TO SEEK TO. NOTHING IS DON
                                E AT THIS:
                  0311 ;;; POINT, EVERYTHING IS DEFFERED UNTIL A READ OR WRITE
                                .       :
                  0312 ;;;----------------------------------------------------
                                ----------
EC90  79          0313 SETTRK:  LD      A,C             ;A <- TRACK #
EC91  32DFF1      0314          LD      (SYSTRK),A      ;SYSTEM TRACK #
EC94  C9          0315          RET
                  0316 ;;;----------------------------------------------------
                                ----------
                  0317 ;;; SETSEC JUST SAVES THE DESIRED SECTOR TO SEEK TO UNT
                                IL AN   :
                  0318 ;;; ACTUAL READ OR WRITE IS ATTEMPTED.
                                       :
                  0319 ;;;----------------------------------------------------
                                ----------
EC95  79          0320 SETSEC:  LD      A,C             ;SAVE THE SECTOR NUMBER
EC96  32E0F1      0321          LD      (SYSSEC),A      ;  IN SYSTEM SECTOR #
EC99  C9          0322          RET
                  0323 ;;;----------------------------------------------------
                                ----------
                  0324 ;;; SETDMA SAVES THE DMA ADDRESS FOR THE DATA TRANSFER.
                                       :
                  0325 ;;;----------------------------------------------------
                                ----------
EC9A  60          0326 SETDMA:  LD      H,B             ;HL <- BC
EC9B  69          0327          LD      L,C
EC9C  22B2ED      0328          LD      (SYSDMA),HL     ;SYSTEM DMA ADDRESS
EC9F  C9          0329          RET                     ;  (SELF MODIFYING)
                  0330 ;;;----------------------------------------------------
                                ----------
                  0331 ;;; SECTRAN TRANSLATES LOGICAL SECTOR # INTO PHYSICAL S
                                ECTOR #.        :
                  0332 ;;;----------------------------------------------------
                                ----------
ECA0  03          0333 SECTRN:  INC     BC              ;INCRE LOGICAL SECTOR #
                                 BY 1
ECA1  D5          0334          PUSH    DE              ;SAVE ADDRESS OF SKEW T
                                ABLE
ECA2  C5          0335          PUSH    BC              ;SAVE LOGICAL SECTOR #
ECA3  CD4AED      0336          CALL    GETDPB          ;GET DPB ADDRESS INTO H
                                L
ECA6  7E          0337          LD      A,(HL)          ;GET # OF SYSTEM SECTOR
                                S/TRACK
```

```
ECA7  CB3F      0338        SRL   A           ;DIVIDE BY TWO
ECA9  91        0339        SUB   C           ;A = (SYS SECTOR/TRACK)
                      /2 - LOG SECTOR #
ECAA  F5        0340        PUSH  AF          ;SAVE ADJUSTED SECTOR
ECAB  FAB7EC    0341        JP    M,SIDE2     ;CHECK IF SECTOR ON SID
                      E TWO
ECAE  F1        0342 SIDEA: POP   AF          ;DISCARD ADJUSTED SECTO
                      R
ECAF  C1        0343        POP   BC          ;RESTORE SECTOR REQUEST
                      ED
ECB0  D1        0344        POP   DE          ;RESTOR ADDRESS OF XLT
                      TABLE
ECB1  EB        0345 SIDE1: EX    DE,HL       ;HL KEEP ADDRESS OF SKE
                      W TABLE
ECB2  09        0346        ADD   HL,BC       ;BC = OFFSET INTO TABLE
ECB3  6E        0347        LD    L,(HL)      ;HL <- PHYSICAL SECTOR
ECB4  2600      0348        LD    H,0
ECB6  C9        0349        RET
ECB7  011F00    0350 SIDE2: LD    BC,15       ;OFFSET TO SIDE BIT
ECBA  09        0351        ADD   HL,BC       ;  IN DPB
ECBB  7E        0352        LD    A,(HL)
ECBC  E608      0353        AND   08H         ;TEST FOR DOUBLE SIDED
ECBE  28EE      0354        JR    Z,SIDEA     ;MEDIA IS ONLY SINGLE S
                      IDED
ECC0  F1        0355        POP   AF          ;RETRIEVE ADJUSTED SECT
                      OR
ECC1  C1        0356        POP   BC          ;DISCARD OLD LOGICAL SE
                      CTOR #
ECC2  ED44      0357        NEG               ;MAKE SECTOR REQUEST (O
                      N SIDE TWO) POSITIVE
ECC4  4F        0358        LD    C,A         ;MAKE NEW SECTOR THE RE
                      QUESTED SECTOR
ECC5  D1        0359        POP   DE          ;RESTORE ADDRESS OF SKE
                      W TABLE
ECC6  CD81EC    0360        CALL  SIDE1       ;GET PHYSICAL SECTOR OF
                       SIDE TWO
ECC9  CBFD      0361        SET   7,L         ;SET SIDE BIT FOR
ECCB  C9        0362        RET               ;  SECTOR ON SIDE TWO
                0363 ;;;-------------------------------------------------
                     ----------
                0364 ;;; SETDRV SELECTS THE NEXT DRIVE TO BE USED IN READ/WR
                     ITE    :
                0365 ;;; OPERATIONS. IF THE DRIVE HAS NEVER BEEN SELECTED BE
                     FORE, A :
                0366 ;;; PARAMETER TABLE IS CREATED WHICH CORRECTLY DESCRIBE
                     S THE  :
                0367 ;;; DISKETTE CURRENTLY IN THE DRIVE.
                                :
                0368 ;;;-------------------------------------------------
                     ----------
ECCC  79        0369 SETDRV: LD   A,C           ;SAVE THE DRIVE #
ECCD  32BEF1    0370        LD    (SYSDRV),A
ECD0  FE02      0371        CP    MAXDSK      ;CHECK FOR A VALID DRIV
                      E #
ECD2  306A      0372        JR    NC,ZRET     ;ILLEGAL DRIVE #
ECD4  CB43      0373        BIT   0,E         ;TEST IF DRIVE EVER LOG
                      GED IN BEFORE
```

```
ECD6  204D        0374        JR      NZ,SETDR1       ;IF BIT 0 OF E=1 -> SEL
                      ECTED BEFORE
ECD8  3E01        0375        LD      A,1             ;SELECT SECTOR 1 OF TRA
                      CK 1
ECDA  3260F1      0376        LD      (TRUSEC),A
ECDD  325FF1      0377        LD      (SYSTRK),A
ECE0  CD1BEE      0378        CALL    FILL            ;FLUSH BUFFER AND REFIL
                      L
ECE3  3859        0379        JR      C,ZRET          ;TEST ERROR RETURN FROM
                       FILL OR FLUSH
ECE5  CD27F8      0380        CALL    DJSTAT          ;GET STATUS ON CURRENT
                      DRIVE
ECE8  E60C        0381        AND     0CH             ;STRIP OFF UNWANTED BIT
                      S
ECEA  F5          0382        PUSH    AF              ;USED TO SELECT A DPB
ECEB  1F          0383        RRA
ECEC  2142ED      0384        LD      HL,XLTS         ;TABLE OF XLT ADDRESSES
ECEF  5F          0385        LD      E,A
ECF0  1600        0386        LD      D,0             ;DE ARE OFFSET INTO TAB
                      LE OF XLT
ECF2  19          0387        ADD     HL,DE           ;HL KEEP ADDRESS OF COR
                      RECT XLT
ECF3  E5          0388        PUSH    HL              ;SAVE POINTER TO PROPER
                       XLT
ECF4  CD4AED      0389        CALL    GETDPB          ;GET DPH POINTER INTO D
                      E
ECF7  EB          0390        EX      DE,HL           ;HL KEEP ADDRESS OF DPH
ECF8  D1          0391        POP     DE              ;DE KEEP ADDRESS OF XLT
ECF9  0602        0392        LD      B,2             ;NUMBER OF BYTES TO MOV
                      E
ECFB  CD31EE      0393        CALL    MOVLOP          ;FILL ADDRESS OF XLT IN
                       DPH
ECFE  113000      0394        LD      DE,30           ;OFFSET TO DPB POINTER
ED01  19          0395        ADD     HL,DE           ;HL KEEP ADDRESS TO SAV
                      E
ED02  E5          0395        PUSH    HL              ;  ADDRESS OF DPB IN DP
                      H
ED03  2A07F8      0397        LD      HL,(FIRM+7)     ;GET ADDRESS OF DJ TERM
                      INAL OUT ROUTINE
ED06  23          0398        INC     HL              ;BUMP TO LOOK AT ADDRES
                      S OF
                  0399                                ;       UART STATUS LOC
                      ATION
ED07  7E          0400        LD      A,(HL)
ED08  EE03        0401        XOR     03H             ;ADJUST FOR PROPER REV
                      DJ
ED0A  6F          0402        LD      L,A
ED0B  26FB        0403        LD      H,0FBH          ;LOAD H WITH (FIRM+300H
                      )/100H
ED0D  7E          0404        LD      A,(HL)
ED0E  E608        0405        AND     DSLSID          ;CHECK DOUBLE SIDED BIT
ED10  1126EF      0406        LD      DE,DB128S       ;BASE FOR SINGLE SIDED
                      DPB'S
ED13  2003        0407        JR      NZ,SIDEOK
ED15  1166EF      0408        LD      DE,DB128D       ;BASE OF DOUBLE SIDED D
                      PB'S
ED18  EB          0409 SIDEOK: EX     DE,HL           ;HL <- DPB BASE, DE <-
```

```
                        &DPH.DPB
ED19  D1          0410          POP     DE              ;RESTORE DE (POINTER IN
                        TO DPH)
ED1A  F1          0411          POP     AF              ;OFFSET TO CORRECT DPB
ED1B  17          0412          RLA
ED1C  17          0413          RLA
ED1D  4F          0414          LD      C,A
ED1E  0600        0415          LD      B,0
ED20  09          0416          ADD     HL,BC           ;FORM ADDRESS OF CORREC
                        T DPB
ED21  EB          0417          EX      DE,HL           ;PUT DPB ADDRESS IN DPH
ED22  73          0418          LD      (HL),E
ED23  23          0419          INC     HL
ED24  72          0420          LD      (HL),D
ED25  CD4AED      0421 SETDR1:  CALL    GETDPB          ;GET ADDRESS OF DPB IN
                        HL
ED28  010F00      0422          LD      BC,15           ;OFFSET TO SECTOR SIZE
ED2B  09          0423          ADD     HL,BC           ;  IN LAST BYTE OF DPB
ED2C  7E          0424          LD      A,(HL)          ;GET SECTOR SIZE
ED2D  E607        0425          AND     07H             ;  IN LOW NIBBLE VALUE
                        YXXX
ED2F  326FED      0426          LD      (SECSIZ),A      ;  Y=0 (SINGLE SIDE)/
                        =1 (DOUBLE SIDE)
ED32  7E          0427          LD      A,(HL)          ;  XXX VALUE 1,2,3 OR
                        4 (SELF MODIFYING)
ED33  1F          0428          RRA
ED34  1F          0429          RRA
ED35  1F          0430          RRA
ED36  1F          0431          RRA                     ;GET SECTOR SIZE
ED37  E60F        0432          AND     0FH             ;  IN HIGH NIBBLE
ED39  32A1ED      0433          LD      (CHKSEC),A      ;  VALUE 0,1,3 OR 7 (S
                        ELF MODIFYING)
ED3C  EB          0434          EX      DE,HL           ;HL <- DPH
ED3D  C9          0435          RET
ED3E  210000      0436 ZRET:    LD      HL,0            ;SELDRV ERROR EXIT
ED41  C9          0437          RET
                  0438 ;-------------------------------------------------
                        ----------
                  0439 ; XLTS IS A TABLE OF ADDRESS THAT POINT TO EACH OF THE
                        XLT     :
                  0440 ; TABLES FOR EACH SECTOR SIZE.
                                :
                  0441 ;-------------------------------------------------
                        ----------
ED42  3BEE        0442 XLTS:    DW      XLT128          ;XLT FOR 128 BYTE SECTO
                        RS
ED44  53EE        0443          DW      XLT256          ;XLT FOR 256 BYTE SECTO
                        RS
ED46  6BEE        0444          DW      XLT512          ;XLT FOR 512 BYTE SECTO
                        RS
ED48  C3EE        0445          DW      XLT1024         ;XLT FOR 1024 BYTE SECT
                        ORS
                  0446 ;;;-------------------------------------------------
                        ----------
                  0447 ; GETDPB RETURNS HL POINTING TO THE DPB OF THE CURRENTL
                        Y       :
                  0448 ; SELECTED DRIVE, DE POINTING TO DPH.
```

```
                                        :
                        0449 ;;;--------------------------------------------------
                             ----------
ED4A  3A5EF1            0450 GETDPB: LD      A,(SYSDRV)      ;GET DRIVE #
ED4D  6F                0451         LD      L,A             ;FORM OFFSET
ED4E  2600              0452         LD      H,0
ED50  29                0453         ADD     HL,HL
ED51  29                0454         ADD     HL,HL
ED52  29                0455         ADD     HL,HL
ED53  29                0456         ADD     HL,HL
ED54  11D6EF            0457         LD      DE,DPZERO       ;BASE OF DPH'S
ED57  19                0458         ADD     HL,DE
ED58  E5                0459         PUSH    HL              ;SAVE ADDRESS OF DPH
ED59  110A00            0460         LD      DE,10           ;OFFSET TO DPB
ED5C  19                0461         ADD     HL,DE
ED5D  7E                0462         LD      A,(HL)          ;GET LOW BYTE OF DPB AD
                             DRESS
ED5E  23                0463         INC     HL
ED5F  66                0464         LD      H,(HL)          ;GET LOW BYTE OF DPB
ED60  6F                0465         LD      L,A
ED61  D1                0466         POP     DE
ED62  C9                0467         RET
                        0468 ;;;--------------------------------------------------
                             ----------
                        0469 ;;; WRITE ROUTINE MOVES DATA FROM MEMORY INTO THE BUFFE
                             R. IF THE:
                        0470 ;;; DESIRED SYSTEM SECTOR IS NOT CONTAINED IN THE DISK
                             BUFFER,  :
                        0471 ;;; THE BUFFER IS FIRST FLUSHED TO THE DISK IF IT HAS E
                             VER BEEN        :
                        0472 ;;; WRITTEN INTO, THEN A READ IS PERFORMED INTO THE BUF
                             FER TO  :
                        0473 ;;; GET THE DESIRED SECTOR. ONCE THE CORRECT SECTOR IS
                             IN      :
                        0474 ;;; MEMORY, THE BUFFER WRITTEN INDICATOR IS SET, SO THE
                              BUFFER         :
                        0475 ;;; WILL BE FLUSHED, THEN DATA IS TRANSFERRED INTO THE
                             BUFFER.         :
                        0476 ;;;--------------------------------------------------
                             ----------
ED63  79                0477 WRITE:  LD      A,C             ;SAVE WRITE COMMAND TYP
                             E
ED64  32D3ED            0478         LD      (WRTTYP),A      ;  (SELF MODIFYING)
ED67  3E01              0479         LD      A,1             ;SET WRITE COMMAND
ED69  06                0480         DB      06H             ;THIS "LD B" INSTRUCTIO
                             N  CAUSES
                        0481                                 ;  THE FOLLOWING "XOR A
                             " TO
                        0482                                 ;    BE SKIPPED OVER.
                        0483 ;;;--------------------------------------------------
                             ----------
                        0484 ;;; READ ROUTINE TO BUFFER DATA FROM THE DISK. IF THE S
                             ECTOR   :
                        0485 ;;; REQUESTED FROM SYSTEM IS IN THE BUFFER, THEN DATA I
                             S SIMPLY        :
                        0486 ;;; TRANSFERRED FROM THE BUFFER TO THE DESIRED DMA ADDR
                             ESS. IF :
```

```
                        0487 ;;; THE BUFFER DOES NOT CONTAIN THE DESIRED SECTOR, THE
                             BUFFER      :
                        0488 ;;; IS FLUSHED TO THE DISK IF IT HAS EVER BEEN WRITTEN
                             INTO,   :
                        0489 ;;; THEN FILLED WITH THE SECTOR FROM THE DISK THAT CONT
                             AINS THE    :
                        0490 ;;; DESIRED SYSTEM SECTOR.
                                         :
                        0491 ;;;-------------------------------------------------
                             ---------
ED6A  AF                0492 READ:   XOR     A              ;SET THE COMMAND TYPE T
                             O READ
ED6B  3295ED            0493         LD      (RDWR),A       ;SAVE COMMAND TYPE 0=RE
                             AD/ 1=WRITE
                        0494                                ;  (SELF MODIFYING)
                        0495 ;-------------------------------------------------
                             ---------
                        0496 ; REDWRT CALCULATES THE PHYSICAL SECTOR ON THE DISK THA
                             T       :
                        0497 ; CONTAINS THE DESIRED SYSTEM SECTOR, THEN CHECKS IF IT
                              IS THE :
                        0498 ; SECTOR CURRENTLY IN THE BUFFER. IF NO MATCH IS MADE,
                             THE     :
                        0499 ; BUFFER IS FLUSHED IF NECESSARY AND THE CORRECT SECTOR
                              READ   :
                        0500 ; FROM THE DISK.
                                         :
                        0501 ;-------------------------------------------------
                             ---------
ED6E  0600              0502 REDWRT: LD      B,0            ;THE 0 IS MODIFIED TO C
                             ONTAIN 1,2,3 OR 4
      (ED6F)            0503 SECSIZ  EQU     $-1            ;  FOR SECTOR SIZE 128,
                             256,512 OR 1024
                        0504                                ;    (FILLED BY SETDRV)
ED70  3A60F1            0505         LD      A,(SYSSEC)     ;GET THE DESIRED SYSTEM
                              SECTOR #
ED73  F5                0506         PUSH    AF             ;TEMPORARY SAVE
ED74  E680              0507         AND     80H            ;SAVE ONLY THE SIDE BIT
ED76  4F                0508         LD      C,A            ;REMEMBER THE SIDE
ED77  F1                0509         POP     AF             ;GET THE SECTOR BACK
ED78  E67F              0510         AND     7FH            ;FORGET THE SIDE BIT
ED7A  3D                0511         DEC     A              ;TEMPORARY ADJUSTMENT
ED7B  05                0512 DIVLOP: DEC     B              ;UPDATE REPEAT COUNT
ED7C  2804              0513         JR      Z,DIVDON
ED7E  B7                0514         OR      A              ;CLEAR THE CARY FLAG
ED7F  1F                0515         RRA                    ;DIVIDE THE SYSTEM SECT
                             OR # BY THE SIZE
                        0516                                ;  OF THE PHYSICAL SECT
                             ORS
ED80  18F9              0517         JR      DIVLOP         ;
ED82  3C                0518 DIVDON: INC     A
ED83  81                0519         ADD     C              ;RESTORE THE SIDE BIT
ED84  3268F1            0520         LD      (TRKSEC),A     ;SAVE THE PHYSICAL SECT
                             OR NUMBER
ED87  216EF1            0521         LD      HL,SYSDRV      ;POINTER TO DESIRED DRI
                             VE,TRACK, AND SECTOR
ED8A  1161F1            0522         LD      DE,BUFDRV      ;POINTER TO BUFFER DRIV
```

```
                        E,TRACK, AND SECTOR
ED8D  0604      0523            LD      B,4             ;COUNT LOOP
ED8F  05        0524 DTSLOP:    DEC     B               ;TEST IF DONE WITH COMP
                        ARE
ED90  280A      0525            JR      Z,MOVE          ;YES, MATCH. GO MOVE TH
                        E DATA
ED92  1A        0526            LD      A,(DE)          ;GET A BYTE TO COMPARE
ED93  BE        0527            CP      (HL)            ;TEST FOR MATCH
ED94  23        0528            INC     HL              ;BUMP POINTERS TO NEXT
                        DATA ITEM
ED95  13        0529            INC     DE
ED96  28F7      0530            JR      Z,DTSLOP        ;MATCH, CONTINUE TESTIN
                        G
                0531 ;

                0532 ; DRIVE, TRACK, AND SECTOR DON'T MATCH, FLUSH THE BUFFE
                        R IF
                0533 ; NECESSARY AND THEN REFILL.

                0534 ;

ED98  CD19EE    0535            CALL    FILL            ;FILL THE BUFFER WITH C
                        ORRECT PHYSICAL SECTOR
ED9B  D8        0536            RET     C               ;NO GOOD, RETURN WITH E
                        RROR INDICATION
                0537 ;----------------------------------------------------
                        ----------
                0538 ; MOVE HAS BEEN MODIFIED TO CAUSE EITHER A TRANSFER INT
                        O OR OUT        :
                0539 ; THE BUFFER.
                                        :
                0540 ;----------------------------------------------------
                        ----------
ED9C  3A60F1    0541 MOVE:      LD      A,(SYSSEC)      ;GET THE SYSTEM SECTOR
                        TO TRANSFER
ED9F  3D        0542            DEC     A               ;ADJUST TO PROPER SECTO
                        R IN BUFFER
EDA0  E603      0543            AND     03H             ;STRIP OFF HIGH ORDERED
                         BITS
      (EDA1)    0544 CHKSEC     EQU     $-1             ;THE 0 IS MODIFIED TO C
                        ONTAIN 0,1,3 OR 7
                0545                                    ;  DEPEND ON SECTOR SIZ
                        E (FILLED BY SETDRV)
EDA2  6F        0546            LD      L,A             ;PUT INTO HL
EDA3  2600      0547            LD      H,0
EDA5  29        0548            ADD     HL,HL           ;FORM OFFSET INTO BUFFE
                        R
EDA6  29        0549            ADD     HL,HL           ;  BY 4*128 (SYSTEM SEC
                        TOR SIZE)
EDA7  29        0550            ADD     HL,HL
EDA8  29        0551            ADD     HL,HL
EDA9  29        0552            ADD     HL,HL
EDAA  29        0553            ADD     HL,HL
EDAB  29        0554            ADD     HL,HL
EDAC  1164F1    0555            LD      DE,BUFFER       ;BEGINNING ADDRESS OF B
                        UFFER
EDAF  19        0556            ADD     HL,DE           ;FORM BEGINNING ADDRESS
```

```
                                        OF SECTOR TO TRANSFER
ED80  EB            0557          EX      DE,HL           ;DE = ADDRESS IN BUFFER
ED81  210000        0558          LD      HL,0            ;GET DMA ADDRESS, THE 0
                                        IS MODIFIED TO
                    0559                                  ;   CONTAIN THE DMA ADD
                                        RESS
      (ED82)        0560 SYSDMA   EQU     $-2
ED84  3E00          0561          LD      A,0             ;THE ZERO GETS MODIFIED
                                        TO CONTAIN
                    0562                                  ;   A ZERO IF A READ, O
                                        R A 1 IF WRITE
      (ED85)        0563 RDWR     EQU     $-1
ED86  A7            0564          AND     A               ;TEST WHICH KIND OF OPE
                                        RATION
ED87  2005          0565          JR      NZ,INTO         ;TRANSFER DATA INTO THE
                                         BUFFER
ED89  CD2FEE        0566 OUTOF:   CALL    MOVER
ED8C  AF            0567          XOR     A
ED8D  C9            0568          RET
ED8E  EB            0569 INTO:    EX      DE,HL           ;
ED8F  CD2FEE        0570          CALL    MOVER           ;MOVE THE DATA, HL = DE
                                        STINATION
                    0571                                  ;       DE = SOURCE
EDC2  3E01          0572          LD      A,1
EDC4  3201ED        0573          LD      (BUFWRT),A      ;SET BUFFER WRITTEN INT
                                        O FLAG
EDC7  3E00          0574          LD      A,0             ;CHECK FOR DIRECTORY WR
                                        ITE
      (EDC8)        0575 WRTTYP   EQU     $-1
EDC9  3D            0576          DEC     A
EDCA  3E00          0577          LD      A,0
EDCC  3208ED        0578          LD      (WRTTYP),A      ;SET NO DIRECTORY WRITE
                                         (SELF MODIFYING)
EDCF  C0            0579          RET     NZ              ;NO ERROR EXIT
                    0580 ;-----------------------------------------------------
                                        ----------
                    0581 ; FLUSH WRITES THE CONTENTS OF THE BUFFER OUT TO THE DI
                                        SK IF    :
                    0582 ; IT HAS EVER BEEN WRITTEN INTO.
                                                          :
                    0583 ;-----------------------------------------------------
                                        ----------
ED00  3E00          0584 FLUSH:   LD      A,0             ;THE 0 IS MODIFIED TO R
                                        EFLECT IF
                    0585                                  ;  THE BUFFER HAS BEEN
                                        WRITTEN INTO
      (ED01)        0586 BUFWRT   EQU     $-1
ED02  A7            0587          AND     A               ;TEST IF WRITTEN INTO
ED03  C8            0588          RET     Z               ;NOT WRITTEN, ALL DONE
ED04  211EF8        0589          LD      HL,DOWRT        ;WRITE OPERATION
                    0590 ;-----------------------------------------------------
                                        ----------
                    0591 ; PREP PREPARES TO READ/WRITE THE DISK. RETRIES ARE ATT
                                        EMPTED. :
                    0592 ; UPON ENTRY, H&L MUST CONTAIN THE READ OR WRITE OPERAT
                                        ION     :
                    0593 ; ADDRESS.
```

```
                        0594 ;-----------------------------------------------------
                             ----------
EDD7  AF                0595 PREP:  XOR     A               ;RESET BUFFER WRITTEN F
                             LAG
EDD8  32D1ED            0596        LD      (BUFWRT),A      ;FLAG BUFFER NOT DIRTY
                             (SELF MODIFYING)
EDDB  22DEEE            0597        LD      (RETYOP),HL     ;SET READ/WRITE OPERATI
                             ON (SELF MODIFYING)
EDDE  060A              0598        LD      B,RETRY         ;MAXIMUM NUMBER OF RETR
                             IES
EDE0  C5                0599 RETYLP: PUSH   BC              ;SAVE THE RETRY COUNT
EDE1  3A61F1            0600        LD      A,(BUFDRV)      ;GET DRIVE NUMBER INVOL
                             VED IN THE OPERATION
EDE4  4F                0601        LD      C,A
EDE5  CD18F8            0602        CALL    DOSEL           ;SELECT THE DRIVE
EDE8  3A62F1            0603        LD      A,(BUFTRK)
EDEB  A7                0604        AND     A               ;TEST FOR TRACK ZERO
EDEC  4F                0605        LD      C,A
EDED  C5                0606        PUSH    BC
EDEE  CC39F8            0607        CALL    Z,DOHOME        ;HOME THE DRIVE IF TRAC
                             K 0
EDF1  C1                0608        POP     BC              ;RESTORE TRACK #
EDF2  CD0CF8            0609        CALL    DOTRK           ;SEEK TO PROPER TRACK
EDF5  3A63F1            0610        LD      A,(BUFSEC)      ;GET SECTOR INVOLVED IN
                              OPERATION
EDF8  F5                0611        PUSH    AF              ;SAVE THE SECTOR #
EDF9  07                0612        RLCA                    ;BIT 0 OF A EQUALS SIDE
                             #
EDFA  E601              0613        AND     01H             ;STRIP OFF UNNECESSARY
                             BITS
EDFC  4F                0614        LD      C,A             ;C <- SIDE #
EDFD  CD33F8            0615        CALL    DOSIDE          ;SELECT THE SIDE
EE00  F1                0616        POP     AF              ;A <- SECTOR #
EE01  E67F              0617        AND     7FH             ;STRIP OFF SIDE BIT
EE03  4F                0618        LD      C,A             ;C <- SECTOR #
EE04  CD3FF8            0619        CALL    DOSEC           ;SET THE SECTOR TO TRAN
                             SFER
EE07  0164F1            0620        LD      BC,BUFFER       ;SET THE DMA ADDRESS
EE0A  CD1CF8            0621        CALL    DODMA
EE0D  CD13F8            0622        CALL    DOREAD          ;THE READ OPERATION IS
                             MODIFIED TO WRITE
      (EE0E)            0623 RETYOP EQU     $-2
EE10  C1                0624        POP     BC              ;RESTORE THE RETRY COUN
                             TER
EE11  3E00              0625        LD      A,0             ;NO ERROR EXIT STATUS
EE13  D0                0626        RET     NC              ;RETURN NO ERROR
EE14  05                0627        DEC     B               ;UPDATE THE RETRY COUNT
                             ER
EE15  37                0628        SCF                     ;ASSUME RETRY COUNT EXP
                             IRED
EE16  3EFF              0629        LD      A,0FFH          ;ERROR RETURN
EE18  C8                0630        RET     Z
EE19  18C5              0631        JR      RETYLP          ;TRY AGAIN
                        0632 ;-----------------------------------------------------
                             ----------
                        0633 ; FILL FILLS THE BUFFER WITH A NEW SECTOR FROM THE DISK
```

```
                         .           :
                          ;-----------------------------------------------------
                          ----------
EE1B  CD00ED              FILL:   CALL    FLUSH           ;FLUSH BUFFER FIRST
EE1E  D8                          RET     C               ;CHECK FOR ERROR
EE1F  115EF1                      LD      DE,SYSDRV       ;UPDATE THE DRIVE, TRAC
                          K, AND SECTOR
EE22  2161F1                      LD      HL,BUFDRV
EE25  0603                        LD      B,3             ;NUMBER OF BYTES TO MOV
                          E
EE27  CD31EE                      CALL    MOVLOP          ;COPY THE DATA
EE2A  211DF3                      LD      HL,DJREAD
EE2D  18A8                        JR      PREP            ;SELECT DRIVE, TRACK, A
                          ND SECTOR.
                                                          ;  THEN READ THE BUFFER
                          ;-----------------------------------------------------
                          ----------
                          ; MOVER MOVES 128 BYTES OF DATA. SOURCE POINTER IN DE,
                          DEST    :
                          ; POINTER IN HL.
                                                :
                          ;-----------------------------------------------------
                          ----------
EE2F  0680                MOVER:  LD      B,128           ;LENGTH OF TRANSFER
EE31  1A                  MOVLOP: LD      A,(DE)          ;GET A BTE OF SOURCE
EE32  77                          LD      (HL),A          ;MOVE IT
EE33  13                          INC     DE              ;BUMP POINTERS
EE34  23                          INC     HL
EE35  10FA                        DJNZ    MOVLOP          ;CONTINUE MOVING UNTIL
                          DONE
EE37  C9                          RET
                          ;;;---------------------------------------------------
                          ----------
                          ; XLT TABLES (SECTOR SKEW TABLES) DEFINE THE SECTOR TRA
                          NSLATION :
                          ; THAT OCCURS WHEN MAPPING SYSTEM SECTORS TO PHYSICAL S
                          ECTORS  :
                          ; ON THE DISK. THERE IS ONE SKEW TABLE FOR EACH OF THE
                          POSSIBLE :
                          ; SECTOR SIZES.
                                                :
                          ;;;---------------------------------------------------
                          ----------
EE38  00                  XLT128: DB      0
EE39  01070D13                    DB      1,7,13,19,25
      19
EE3E  050B1117                    DB      5,11,17,23
EE42  03090F15                    DB      3,9,15,21
EE46  02080E14                    DB      2,8,14,20,26
      1A
EE4B  060C1218                    DB      6,12,18,24
EE4F  040A1016                    DB      4,10,16,22
EE53  00                  XLT256: DB      0
EE54  01021314                    DB      1,2,19,20,37,38
      2526
EE5A  03041516                    DB      3,4,21,22,39,40
      2728
```

```
EE60  05061718    0671         DB      5,6,23,24,41,42
      292A
EE66  0708191A    0672         DB      7,8,25,26,43,44
      2B2C
EE6C  090A1B1C    0673         DB      9,10,27,28,45,46
      2D2E
EE72  0B0C1D1E    0674         DB      11,12,29,30,47,48
      2F30
EE78  0D0E1F20    0675         DB      13,14,31,32,49,50
      3132
EE7E  0F102122    0676         DB      15,16,33,34,51,52
      3334
EE84  11122324    0677         DB      17,18,35,36
EE88  00          0678 XLT512: DB      0
EE89  01020304    0679         DB      1,2,3,4,17,18,19,20
      11121314
EE91  21222324    0680         DB      33,34,35,36,49,50,51,52
      31323334
EE99  05060708    0681         DB      5,6,7,8,21,22,23,24
      15161718
EEA1  25262728    0682         DB      37,38,39,40,53,54,55,56
      35363738
EEA9  090A0B0C    0683         DB      9,10,11,12,25,26,27,28
      191A1B1C
EEB1  292A2B2C    0684         DB      41,42,43,44,57,58,59,60
      393A3B3C
EEB9  0D0E0F10    0685         DB      13,14,15,16,29,30,31,32
      1D1E1F20
EEC1  2D2E2F30    0686         DB      45,46,47,48
EEC5  00          0687 XLT124: DB      0
EEC6  01020304    0688         DB      1,2,3,4,5,6,7,8
      05060708
EECE  191A1B1C    0689         DB      25,26,27,28,29,30,31,32
      1D1E1F20
EED6  31323334    0690         DB      49,50,51,52,53,54,55,56
      35363738
EEDE  090A0B0C    0691         DB      9,10,11,12,13,14,15,16
      0D0E0F10
EEE6  21222324    0692         DB      33,34,35,36,37,38,39,40
      25262728
EEEE  393A3B3C    0693         DB      57,58,59,60,61,62,63,64
      3D3E3F40
EEF6  11121314    0694         DB      17,18,19,20,21,22,23,24
      15161718
EEFE  292A2B2C    0695         DB      41,42,43,44,45,46,47,48
      2D2E2F30
                  0696 ;;;-----------------------------------------------------
                       ----------
                  0697 ; DISK PARAMETER HEADER DESCRIBES A DISKETTE WITH THE S
                       PECIFIED :
                  0698 ; CHARACTERISTICS.
                  0699 ;;;-----------------------------------------------------
                       ----------
EF06  0000        0700 DHZERO: DW     0                ;ADDRESS OF TRANSLATION
                        TABLE
                  0701                                 ;  (FILLED IN BY SETDRV
```

```
EF0C 0000 0702 DW 0,0,0 ;USED BY DOS
     0000
     0000
EF0E 7AF6 0703 DW DIRBUF ;ADDRESS OF DIRECTORY B
                UFFER
EF10 0000 0704 DW 0 ;ADDRESS OF DPB (FILLED
                IN BY SETDRV)
EF12 FAF5 0705 DW CSV0 ;DIRECTORY CHECK VECTOR
EF14 64F5 0706 DW ALV0 ;ALLOCATION VECTOR
EF16 0000 0707 DHOME: DW 0
EF18 0000 0708 DW 0,0,0
     0000
     0000
EF1E 7AF6 0709 DW DIRBUF
EF20 0000 0710 DW 0
EF22 3AF6 0711 DW CSV1
EF24 AFF5 0712 DW ALV1
          0713 ;;;-----------------------------------------------
               ----------
          0714 ; DISK PARAMETER BLOCK DEFINES A DISKETTE FOR EACH SEC
               TOR SIZE :
          0715 ; SINGLE SIDED AND DOUBLE SIDED. :
          0716 ;;;-----------------------------------------------
               ----------
EF26 1A00 0717 DS128S: DW 26 ;SYSTEM SECTORS/TRACK
EF28 03 0718 DB 3 ;BSH
EF29 07 0719 DB 7 ;BLM
EF2A 00 0720 DB 0 ;EXM
EF2B F200 0721 DW 242 ;DSM
EF2D 7F00 0722 DW 63 ;DRM
EF2F C0 0723 DB 0C0H ;AL0
EF30 00 0724 DB 0 ;AL1
EF31 1000 0725 DW 16 ;CKS
EF33 0200 0726 DW 2 ;OFF
EF35 01 0727 DB 1H ;16*((#SYS SECTORS/PHYS
               ICAL SECTOR) -1) +
          0728 ;LOG2(#BYTES PER SECTOR
               /128) + 1 +
          0729 ;0 IF DOUBLE SIDED.
EF36 3400 0730 DS256S: DW 52 ;SYSTEM SECTORS/TRACK
EF38 04 0731 DB 4 ;BSH
EF39 0F 0732 DB 15 ;BLM
EF3A 00 0733 DB 0 ;EXM
EF3B F200 0734 DW 242 ;DSM
EF3D 7F00 0735 DW 127 ;DRM
EF3F C0 0736 DB 0C0H ;AL0
EF40 00 0737 DB 0 ;AL1
EF41 1000 0738 DW 32 ;CKS
EF43 0200 0739 DW 2 ;OFF
EF45 12 0740 DB 12H ;16*((#SYS SECTORS/PHYS
               ICAL SECTOR) -1) +
          0741 ;LOG2(#BYTES PER SECTOR
               /128) + 1 +
          0742 ;0 IF DOUBLE SIDED.
EF46 3C00 0743 DS512S: DW 60 ;SYSTEM SECTORS/TRACK
```

```
EF48 04          0744          DB      4               ;BSH
EF49 0F          0745          DB      15              ;BLM
EF4A 00          0746          DB      0               ;EXM
EF4B 1801        0747          DW      280             ;DSM
EF4D 7F00        0748          DW      127             ;DRM
EF4F C0          0749          DB      0C0H            ;AL0
EF50 00          0750          DB      0               ;AL1
EF51 2000        0751          DW      32              ;CKS
EF53 0200        0752          DW      2               ;OFF
EF55 33          0753          DB      33H             ;16*((#SYS SECTORS/PHYS
                                   ICAL SECTOR) -1) +
                 0754                                  ;LOG2(#BYTES PER SECTOR
                                   /128) + 1 +
                 0755                                  ;0 IF DOUBLE SIDED.
EF56 4000        0756 DB124S:  DW      64              ;SYSTEM SECTORS/TRACK
EF58 04          0757          DB      4               ;BSH
EF59 0F          0758          DB      15              ;BLM
EF5A 00          0759          DB      0               ;EXM
EF5B 2301        0760          DW      299             ;DSM
EF5D 7F00        0761          DW      127             ;DRM
EF5F C0          0762          DB      0C0H            ;AL0
EF60 00          0763          DB      0               ;AL1
EF61 2000        0764          DW      32              ;CKS
EF63 0200        0765          DW      2               ;OFF
EF65 74          0766          DB      74H             ;16*((#SYS SECTORS/PHYS
                                   ICAL SECTOR) -1) +
                 0767                                  ;LOG2(#BYTES PER SECTOR
                                   /128) + 1 +
                 0768                                  ;0 IF DOUBLE SIDED.
EF66 3400        0769 DB128D:  DW      52              ;SYSTEM SECTORS/TRACK
EF68 04          0770          DB      4               ;BSH
EF69 0F          0771          DB      15              ;BLM
EF6A 01          0772          DB      1               ;EXM
EF6B F200        0773          DW      242             ;DSM
EF6D 7F00        0774          DW      127             ;DRM
EF6F C0          0775          DB      0C0H            ;AL0
EF70 00          0776          DB      0               ;AL1
EF71 2000        0777          DW      32              ;CKS
EF73 0200        0778          DW      2               ;OFF
EF75 09          0779          DB      9H
EF76 6800        0780 DB256D:  DW      104             ;SYSTEM SECTORS/TRACK
EF78 04          0781          DB      4               ;BSH
EF79 0F          0782          DB      15              ;BLM
EF7A 00          0783          DB      0               ;EXM
EF7B E601        0784          DW      486             ;DSM
EF7D FF00        0785          DW      255             ;DRM
EF7F F0          0786          DB      0F0H            ;AL0
EF80 00          0787          DB      0               ;AL1
EF81 4000        0788          DW      64              ;CKS
EF83 0200        0789          DW      2               ;OFF
EF85 1A          0790          DB      1AH
EF86 7600        0791 DB512D:  DW      120             ;SYSTEM SECTORS/TRACK
EF88 04          0792          DB      4               ;BSH
EF89 0F          0793          DB      15              ;BLM
EF8A 00          0794          DB      0               ;EXM
EF8B 3102        0795          DW      561             ;DSM
EF8D FF00        0796          DW      255             ;DRM
```

```
EF8F  F0          0797        DB      0F0H            ;AL0
EF90  00          0798        DB      0               ;AL1
EF91  4000        0799        DW      64              ;CKS
EF93  0200        0800        DW      2               ;OFF
EF95  3B          0801        DB      3BH
EF96  8000        0802 DB124D: DW      128             ;SYSTEM SECTORS/TRACK
EF98  04          0803        DB      4               ;BSH
EF99  0F          0804        DB      15              ;BLM
EF9A  00          0805        DB      0               ;EXM
EF9B  5702        0806        DW      599             ;DSM
EF9D  FF00        0807        DW      255             ;DRM
EF9F  F0          0808        DB      0F0H            ;AL0
EFA0  00          0809        DB      0               ;AL1
EFA1  4000        0810        DW      64              ;CKS
EFA3  0200        0811        DW      2               ;OFF
EFA5  7C          0812        DB      7CH
                  0813 ;;;  STRING AREA FOR DOS ERROR MESSAGE.  :::
EFA6  4D657373    0814 DOSMSG: DB      'Message From $'
      61676520
      46726F6D
      2024

                  0815 ;;;  STRING AREA FOR PRINT USAGE  :::::::::
EFB4  55736167    0816 PSAVMG: DB      'Usage: Save contents of memory in a fi
                       le.',NULL
      653A2053
      61766520
      636F6E74
      656E7473
      206F6620
      6D656D6F
      72792069
      6E206120
      66696C65
      2E00
EFDE  55736167    0817 PMOVMG: DB      'Usage: Transfer file(s) to another par
                       tition.',NULL
      653A2054
      72616E73
      66657220
      66696C65
      28732920
      746F2061
      6E6F7468
      65722070
      61727469
      74696F6E
      2E00
F00C  55736167    0818 PUSRMG: DB      'Usage: Move to another partition.',NUL
                       L
      653A204D
      6F766520
      746F2061
      6E6F7468
      65722070
      61727469
      74696F6E
      2E00
```

```
F02E  55736167     0819 PERAMS: DB        'Usage: Delete the specified file(s).',
                        NULL
      653A0944
      656C6574
      65207468
      65207370
      65636966
      69656420
      66696C65
      2873292E
      00
F053  55736167     0820 PTYPMS: DB        'Usage: Display contents of file on con
                        sole.',NULL
      653A2044
      6973706C
      61792063
      6F6E7465
      6E747320
      6F662066
      696C6520
      6F6E2063
      6F6E736F
      6C652E00
F07F  55736167     0821 PRENMS: DB        'Usage: Change name of specified file.'
                        ,NULL
      653A2043
      68616E67
      65206E61
      6D65206F
      66207370
      65636966
      69656420
      66696C65
      2E00
F0A5  207B643A     0822 OFILMS: DB        ' {d:}filename.typ',NULL
      7D66696C
      656E616D
      652E7479
      7000
F0B7  207B643A     0823 PRENM2: DB        ' {d:}newname.typ=oldname.typ',NULL
      7D6E6577
      6E616D65
      2E747970
      3D6F6C64
      6E616D65
      2E747970
      00
F0D4  20202020     0824 DESCPP: DB        '    ',QUOTA,'p',QUOTA,' is number of p
                        age to save',NULL
      27702720
      6973206E
      756D6265
      72206F66
      20706167
      6520746F
      20736176
      6500
```

```
FØF6  2Ø2Ø2Ø2Ø      Ø825 DESCPN: DB       '    ',QUOTA,'n',QUOTA,' is destinate p
                         artition number',NULL
      276E272Ø
      69732Ø64
      65737469
      6E617465
      2Ø7Ø6172
      74697469
      6F6E2Ø6E
      756D6265
      72ØØ
F11C  2Ø6F6E6C      Ø826 TYPE1:  DB       ' only single file can be used.',NULL
      792Ø7369
      6E676C65
      2Ø66696C
      652Ø6361
      6E2Ø6265
      2Ø757365
      642EØØ
F13B  2Ø63616E      Ø827 TYPE2:  DB       ' can use * or ? in name and type.',NUL
                         L
      2Ø757365
      2Ø2A2Ø6F
      722Ø3F2Ø
      696E2Ø6E
      616D652Ø
      616E642Ø
      74797Ø65
      2EØØ
                    Ø828 ;;;---------------------------------------------------
                         ----------
                    Ø829 ; I/O RAM LOCATIONS THAT DON'T NEED INITIALIZATION.
                                          ;
                    Ø83Ø ;---------------------------------------------------
                         ----------
      (F15D)        Ø831 SYSSEC  EQU      $                ;SYSTEM SECTOR #
      (F15E)        Ø832 SYSDRV  EQU      SYSSEC+1         ;SYSTEM DRIVE #
      (F15F)        Ø833 SYSTRK  EQU      SYSDRV+1         ;SYSTEM TRACK #
      (F16Ø)        Ø834 TRUSEC  EQU      SYSTRK+1         ;SECTOR THAT CONTAINS S
                         YSTEM SECTOR
      (F161)        Ø835 BUFDRV  EQU      TRUSEC+1         ;DRIVE THAT BUFFER BELO
                         NGS TO
      (F162)        Ø836 BUFTRK  EQU      BUFDRV+1         ;TRACK THAT BUFFER BELO
                         NGS TO
      (F163)        Ø837 BUFSEC  EQU      BUFTRK+1         ;SECTOR THAT BUFFER BEL
                         ONGS TO
      (F164)        Ø838 BUFFER  EQU      BUFSEC+1         ;MAXIMUM SIZE BUFFER FO
                         R 1K SECTORS
      (F564)        Ø839 ALVØ    EQU      BUFFER+1Ø24      ;ALLOCATION VECTOR FOR
                         DRIVE A
      (F5AF)        Ø84Ø ALV1    EQU      ALVØ+75          ;ALLOCATION VECTOR FOR
                         DRIVE B
      (F5FA)        Ø841 CSVØ    EQU      ALV1+75          ;DIRECTORY CHECK VECTOR
                          FOR DRIVE A
      (F63A)        Ø842 CSV1    EQU      CSVØ+64          ;DIRECTORY CHECK VECTOR
                          FOR DRIVE B
      (F67A)        Ø843 DIRBUF  EQU      CSV1+64          ;DIRECTORY BUFFER
```

```
                (F6FA)      0844 ETABLE  EQU     DIRBUF+128      ;END OF TABLE.
                            0845 ;;;-----------------------------------------------------
                                 --
                            0846 ; CLEAR: CLEAR LOGO IN I/O WORK AREA
                                       :
                            0847 ;;;-----------------------------------------------------
                                 --
F15D  018206                0848 CLEAR:  LD      BC,FIRM-CLEAR
F160  21DDF1                0849         LD      HL,CLEAR
F163  3600                  0850 CLRLP:  LD      (HL),0
F165  23                    0851         INC     HL
F166  0B                    0852         DEC     BC
F167  CA8CE8                0853         JP      Z,GOSYS
F16A  18F7                  0854         JR      CLRLP
                            0855 ;
                            0856 ;;;-----------------------------------------------------
                                 --
                            0857 ;;; CBOOT: ALL OF SYSTEM HAS BEEN LOADED WHEN CONTROL
                                       :
                            0858 ;;; IS PASSED HERE. SET UP STACK, IOBYTE, PRINT LOGO,
                                       :
                            0859 ;;; SELECT DRIVE A THEN JUMP TO EXECUTE AUTO RUN OR
                                       :
                            0860 ;;; DISPLAY PROMPT SIGN.
                                       :
                            0861 ;;;-----------------------------------------------------
                                 --
F160  310001                0862 CBOOT:  LD      SP,TPA
F16F  3A03F0                0863         LD      A,(INTICB)
F172  320300                0864         LD      (IOBYTE),A
F175  3A25FC                0865         LD      A,(LOGOF3)
F178  B7                    0866         OR      A
F179  2825                  0867         JR      Z,ENLOGO
F17B  0E1B                  0868         LD      C,ESC
F17D  CD2EEC                0869         CALL    CONOUT
F180  0E4A                  0870         LD      C,'J'
F182  CD2EEC                0871         CALL    CONOUT
F185  21DBF1                0872         LD      HL,LOGO
F188  7E                    0873 PLOGO:  LD      A,(HL)
F189  23                    0874         INC     HL
F18A  A7                    0875         AND     A
F18B  2810                  0876         JR      Z,CLICK
F18D  FEFF                  0877         CP      0FFH
F18F  2808                  0878         JR      Z,NXTLIN
F191  FE20                  0879         CP      ' '
F193  3805                  0880         JR      C,LOGOF3
F195  E5                    0881         PUSH    HL
F196  4F                    0882         LD      C,A
F197  CD2EEC                0883         CALL    CONOUT
F19A  E1                    0884         POP     HL
F19B  18E9                  0885         JR      PLOGO
F19D  CD1BEC                0886 CLICK:  CALL    CONIN
F1A0  0E1B                  0887 ENLOGO: LD      C,ESC
F1A2  CD2EEC                0888         CALL    CONOUT
F1A5  0E4A                  0889         LD      C,'J'
F1A7  CD2EEC                0890         CALL    CONOUT
F1AA  AF                    0891         XOR     A
```

```
CROKEMOD CODS Z80 ASSEMBLER version 02.15                PAGE 0023

F1A3  320400      0892          LD      (CDISK),A
                  0893 ;
                  0894 ;; MOVE CLEAR ROUTINE TO BUFFER THEN JUMP TO CLEAR ROUT
                       INE
F1A6  215DF1      0895          LD      HL,CLEAR
F1A9  110080      0896          LD      DE,BUFF
F1AC  010F00      0897          LD      BC,CBOOT-CLEAR
F1AF  EDB0        0898          LDIR
F1B1  C38000      0899          JP      BUFF
                  0900 ;
                  0901 ;; SET NEW LINE BY PRINT CR AND LF.
F1B4  E5          0902 NXTLIN: PUSH    HL
F1B5  0E0D        0903          LD      C,CR
F1B7  CD1EEC      0904          CALL    CONOUT
F1BA  0E0A        0905          LD      C,LF
F1BC  CD1EEC      0906          CALL    CONOUT
F1BF  E1          0907          POP     HL
F1C0  1B3E        0908          JR      PLOOP
                  0909 ;
                  0910 ;; REPEAT CHAR.
F1C2  47          0911 LOOPLP: LD      B,A
F1C3  4E          0912          LD      C,(HL)
F1C4  23          0913          INC     HL
F1C5  E5          0914          PUSH    HL
F1C6  C5          0915 LPLP:    PUSH    BC
F1C7  CD1EEC      0916          CALL    CONOUT
F1CA  C1          0917          POP     BC
F1CB  10F9        0918          DJNZ    LPLP
F1CD  E1          0919          POP     HL
F1CE  1B30        0920          JR      PLOOP
                  0921 ;
                  0922 ;;; LOGO: CALL OF PICTURE END WITH 00H (NOT EXCEED ADDR
                       ESS FCFFH)
F1D0  444F5020    0923 LOGO:    DB      'DOS VERSION 1.3',0FC ;
      56455253
      494F4E20
      312E3300
F1E0  FF          0924          DB      0FC
F1E1  FF          0925          DB      0FC
F1E2  141300540   0926          DB      20,' ',12,'O',7,' ',5,'U',13,' ',5,'U',
                       0FC
      07201555
      20200555
      FF
F1F7  12201740    0927          DB      18,' ',16,'O',5,' ',5,'U',13,' ',5,'U',
                       0FC
      05200555
      00200555
      FF
F20A  10200540    0928          DB      16,' ',5,'O',3,' ',5,'O',3,' ',5,'U',13
                  ,' ',5,'U',0FC
      05200540
      05200555
      00200555
      FF
F21B  10200540    0929          DB      16,' ',5,'O',13,' ',5,'O',3,' ',5,'U',1
                  3,' ',5,'U',0FC
```

```
     0A200543
     03200555
     0D200555
     FF
F226 10200543     0930          DB     16,' ',5,'C',18,' ',5,'U',13,' ',5,'U',
                         SPC
     12200555
     0D200555
     FF
F233 10200543     0931          DB     16,' ',5,'C',18,' ',5,'U',13,' ',5,'U',
                         SPC
     12200555
     0D200555
     FF
F240 10200543     0932          DB     16,' ',5,'C',7,' ',23,'=',6,' ',5,'U',S
                         PC
     0720173D
     06200555
     FF
F24D 10200543     0933          DB     16,' ',5,'C',7,' ','COMPUTER SERVICE CE
                         NTER',6,' ',5,'U',SPC
     0720434F
     4D505554
     45522053
     45525649
     43452043
     454E5445
     52062055
     55FF
F25F 10200543     0934          DB     16,' ',5,'C',7,' ',23,'=',5,' ',5,'U',S
                         PC
     0720173D
     05200555
     FF
F270 10200543     0935          DB     16,' ',5,'C',13,' ',5,'U',13,' ',5,'U',
                         SPC
     12200555
     0D200555
     FF
F283 10200543     0936          DB     16,' ',5,'C',13,' ',5,'U',13,' ',5,'U',
                         SPC
     12200555
     0D200555
     FF
F296 10200543     0937          DB     16,' ',5,'C',13,' ',5,'U',13,' ',5,'U',
                         SPC
     12200555
     0D200555
     FF
F2A3 10200543     0938          DB     16,' ',5,'C',10,' ',5,'U',3,' ',5,'U',1
                         3,' ',5,'U',SPC
     0A200543
     03200555
     0D200555
     FF
F2B4 10200543     0939          DB     16,' ',6,'C',8,' ',6,'C',3,' ',5,'U',11
                         ,' ',6,'U',SPC
```

```
          0B200643
          05200655
          0B200655
          FF
F2C5      12201043        0940          DB      18,' ',16,'C',7,' ',19,'U',SPC
          07201355
          FF
F2CE      14200C43        0941          DB      20,' ',12,'C',11,' ',15,'U',SPC
          0B200F55
          FF
F2D7      FF              0942          DB      SPC
F2D8      FF              0943          DB      SPC
F2D9      1E201720        0944          DB      30,' ',23,' ','SYSTEM STARTUP COMPLETE'
                                ,SPC
          53595354
          454D2053
          54415254
          55502043
          4F4D504C
          455445FF
F2F5      1E201E20        0945          DB      30,' ',30,' ','....PRESS ANY KEY '
          2E2E2E2E
          50524553
          5320414E
          59204B45
          5920
F309      00              0946          DB      NULL
          (F30C)          0947 FINISH   EQU     $
F30C      (E300)          0948          END     START

Errors                    0
```

ประวัติผู้เขียน

นาย เจริญศักดิ์ ฮั่นตระกูล เกิดวันที่ 19 กันยายน 2500 สำเร็จการศึกษาวิศวกรรมศาสตร์บัณฑิต (เครื่องกล) จากคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ปีการศึกษา 2524 เข้าศึกษาระดับปริญญามหาบัณฑิต สาขาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมศาสตร์ ในปี พ.ศ. 2525