

JK82 FDC II

TECHNISCHE BESCHREIBUNG

Bestellnummern:

HKM-Z-1360: JK82 FDC II unbestückt
HKM-Z-1362: JK82 FDC II bestückt 4MHz
HKM-Z-1363: JK82 FDC II bestückt 6MHz
HKM-Z-1368: Promsatz für JK82 FDC II

Ihr autorisierter Händler: *****
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *

© 1984 by Janich & Klass Wuppertal

Inhaltsverzeichnis

1.	Übersicht	3
2.	allgemeine Beschreibung	4
3.	technische Daten	5
-	Businterface	5
-	Floppyteil	7
-	Winchester-(SASI)-teil	9
4.	I/O-Adressen	10
5.	Prominhalte	11
-	Bussteuerprom	11
-	Memoryprom	11
-	I/O-Dekoderprom	11
6.	Bestückung	12
-	Stückliste	12
-	Bestückungsplan	13
-	Aufbauhinweise	13
7.	Jumper-Einstellungen	14
8.	Schaltplan	15
9.	Programmbeispiele	18

1. Übersicht

Die M.K.C. FDC II Platine ist ein universelles Interface zwischen einem ECB-280-Computer und Massenspeichern. Anschließbar sind bis zu 7 Floppy-Disk-Laufwerke für 5,25" und 8" Disketten und über die SASI (SCSI)-Schnittstelle bis zu 2 Winchester-Laufwerke an einem XEBEC S1410(A)-Controller.

Ferner verfügt die FDC II Karte über 64 K-Byte Speicher und eine akkugepufferte Uhr.

Somit kann mit nur 2 Platinen (der FDC II und der CPU II) ein vollständiges banked CP/M Plus System aufgebaut werden. Dieses System enthält 128 K Speicher, Anschlußmöglichkeiten für die üblichen Massenspeicher, zwei V24- und eine CENTRONICS-Schnittstelle. Es ist ausbaufähig bis zu 1 MByte Speicherkapazität.

Die M.K.C. FDC II Karte enthält den NEC uPD 765 als Floppy-Controller. Als Floppy-Support Chip wird der SMC FDC 9229B eingesetzt. Der FDC 9229 enthält einen monolithischen Datenseparator, die Takterzeugung und die Write-Precompensations-Logik. 8 Precompensationszeiten sind softwaremäßig einstellbar. Der Floppyteil läuft im Vector-Interrupt-Mode. Die M.K.C FDC II Karte muß an höchster Priorität stecken. Niedrigere Interrupts werden bei Datentransfers über die IEI/IEO-Daisy-Chain gesperrt. Auch 8" MFM (Double Density) ist in einem System mit 4 MHz Systemtakt problemlos realisierbar! Ein Datentransfer darf nicht durch Interrupts oder DMA-Zugriffe gestört werden. Ein Transfer in RAM-Karten, die WAIT erzeugen, ist problematisch. Die synchrone Taktumschaltung für 5,25" und 8" ermöglicht gemischten Betrieb von 5,25" und 8" Laufwerken.

Das SASI-Interface ist interruptfähig (über die PIO). Der Datentransfer ist zeitunkritisch, d.h. hier können auch Interrupts bzw. DMA-Zugriffe oder WAITs während eines Datentransfers zugelassen werden.

Als Uhr wird das MEM E050-16 Uhrenchip verwendet. Mit dem Trimmer T ist ein genauer Abgleich der Uhr möglich. Der Akku puffert die Uhr für mindestens 500 Stunden. Somit muß die Uhr nicht nach jedem Einschalten neu gestellt werden.

Der RAMteil enthält 64K * 1 dynamische RAMs, der REFRESH erfolgt durch die 280 CPU. Dekodiert werden 20 bit Adressbus. Die Seite (zu 64KByte) ist einstellbar.

2. allgemeine Beschreibung

- Floppyteil

Vorgesehen ist der Anschluß von bis zu drei 5,25" und vier 8" Laufwerken. Als Aufzeichnungsverfahren wird MFM oder FM (double oder single density) mit IBM-kompatiblen Formaten verwendet. (uPD765 als Floppy-Controller).

Der Anschluß von ein- oder zweiseitigen Laufwerken mit Shugart-kompatiblen Bus ist möglich. (Wir empfehlen TEAC FD55A bis F bzw. NEC1165 Laufwerke!)

Der Floppy-Controller wird im Interrupt (IM2 über CTC bzw. PIO) betrieben. Damit ist bei einem Systemtakt von mindestens 4 MHz auch 8" MFM problemlos realisierbar.

Andere Interrupts können (müssen) während des Datentransfers blockiert werden.

Der CTC wird weiterhin zur Motorsteuerung der Laufwerke (falls möglich) verwendet. Der Motor wird ca. 20 sec nach dem letzten Zugriff auf das Laufwerk abgestellt. Durch Auslesen der noch verbleibenden Zeit aus dem CTC könnten auch Laufwerke ohne READY-Signal verwendet werden. Dies wird allerdings von unserer Software nicht unterstützt. Über die PIO kann die Write-Precompensation-Time per Software eingestellt werden.

- Winchesterteil

Zum Anschluß von Winchester-Laufwerken ist auf der FDC II ein SASI (SCSI)-Interface vorhanden. Für den SASI-Bus gibt es von mehreren Herstellern Controller, an die i.a. bis zu 2 Winchester-Laufwerke angeschlossen werden können. Die hier vorhandene SASI-Schnittstelle ist nur für den Anschluß von Winchester-Controllern vorgesehen, an sie kann keinesfalls ein zweiter MASTER (Rechner) angeschlossen werden. Die im weiteren angegebenen Routinen sind für den XEBEC S1410(A) Controller geschrieben. Sie müssen für andere Controller adaptiert werden. Unser CP/M PLUS unterstützt nur den XEBEC-Winchester-Controller!

Die SASI-Schnittstelle ist voll interrupt-fähig (IM2 über die PIO); dies wird von unserer Software jedoch nicht benötigt.

Benutzt wird die OPEN-COLLECTOR-Version der SASI-Norm. Die optionelle PARITY-Prüfung ist nicht implementiert.

- RAMteil

Die M.K.C. FDC II Platine enthält 64 K-Byte dynamisches RAM. Dekodiert werden 20 Bit Adressen. Die obersten 4 Bit (Speicherseite) sind über Jumper einstellbar. (Normal-einstellung: 10000H..1FFFFH) Ein DESELECT-Eingang ist nicht vorhanden. Der REFRESH wird von der Z80-CPU gesteuert.

- die Uhr

Als Uhr wird das MEM-IC E050-16 verwendet. Die Uhr ist akkugepuffert. Setzen und Lesen erfolgt über die PIO.

3. technische Daten

3.0 Businterface

Das Businterface der FDC II Karte ist kompatibel zum ECB-Bus. Abweichend vom ECB-Bus sind nur die Adressen A16 bis A20. Sie belegen die Pins 10c, 12c, 13c und 14a der VG-Leiste.

Achtung!

Bei Betrieb von Mini-Floppy-Laufwerken kann der uPD 765 nicht ohne Wait mit einer höheren Systemtakt rate als 4 MHz betrieben werden. Bei 6 MHz-Systemen ist der Wait-Jumper unbedingt zu setzen, wenn mit 5,25"-Laufwerken gearbeitet werden soll. (Sonst erfolgt die Fehlermeldung OVERRUN)

Versorgung

Die FDC-II-Karte benötigt voll bestückt bei 5 Volt ca. 1A bei 4 MHz Systemtakt. Andere Spannungen werden nicht benötigt.

fan-in/fan-out

Signal	LS-fan-in	LS-fan-out
D0 bis D7	1	60
A0 bis A1	8	
A2 bis A7	7	
A8 bis A15	6	
A16 bis A19	5	
IEI	2	
IEO		20
INT		4
WAIT		25
CLK	4	
MRQ, IORQ, M1, RD	3	
WR, PWRCLR	1	

Steckerbelegung VG 64 (ECB-kompatibel) S1

	a		c	
+5V	o	1	o	+5V
D5	o	2	o	D0
D6	o	3	o	D7
D3	o	4	o	D2
D4	o	5	o	A0
A2	o	6	o	A3
A4	o	7	o	A1
A5	o	8	o	A8
A6	o	9	o	A7
<u>WAIT</u>	o	10	o	A16
BUSRQ	o	11	o	IEI
	o	12	o	A17
	o	13	o	A18
A19	o	14	o	D1
	o	15	o	
	o	16	o	IE0
	o	17	o	A11
A14	o	18	o	A10
	o	19	o	
<u>M1</u>	o	20	o	
	o	21	o	<u>INT</u>
	o	22	o	WR
	o	23	o	
	o	24	o	<u>RD</u>
	o	25	o	
	o	26	o	<u>PWRCLR</u>
<u>IORQ</u>	o	27	o	A12
	o	28	o	A15
A13	o	29	o	<u>CLOCK</u>
<u>A9</u>	o	30	o	MRQ
BUSAK	o	31	o	
GND	o	32	o	GND
	a		c	

nicht bezeichnete Pins des Steckers sind nicht belegt

3.1 Floppyteil

Das Floppyinterface basiert auf dem Floppy-Controller uPD765 und dem FDC-Support-Chip SMC FDC 9229B. Als Ausgangstreiber zu den Laufwerken werden OPEN-COLLECTOR ICs 7406, 7407 und 7445 verwendet. Im Eingang sind LS-TTL-Bausteine mit 220/330 Ohm Pull-up/Pull-Down angeordnet. Zwei kaskadierte Kanäle des CTC werden zur Motorsteuerung benutzt. Dies ermöglicht es, Laufwerke ohne READY-Signal zu verwenden.

Der uPD765 wird im DMA-Mode betrieben. Die für einen höheren Systemtakt als 4 MHz benötigte Wait-Logik kann mittels Jumper aktiviert werden. Die DRQ-Leitung (Anforderung) wird über die PIO geführt. DACK (Quittung) erfolgt über das I/O-Dekoderprom. Damit kann auch bei nur 4 MHz Systemtakt 8" MFM gelesen und geschrieben werden. Am Ende der Transfer-Phase erzeugt der uPD765 über den CTC einen Interrupt; hierdurch wird die Ein- oder Ausgabeschleife verlassen. Über einen weiteren Kanal des CTC kann die IEI/IEO-Daisy-Chain gesetzt werden. Dies verhindert, daß andere Peripheriebausteine während der kritischen Transfer-Phase eine Unterbrechung veranlassen können. Aus diesem Grund sollte der Floppycontroller auf dem Steckplatz mit der höchsten Priorität stecken. Es ist sicherzustellen, daß evtl. im System vorhandene DMAs während eines Datentransfers inaktiv bleiben.

Die Eingänge P0 bis P2 und MINI des FDC 9229 werden über die PIO gesteuert. Dies ermöglicht das Umschalten zwischen 5,25" und 8" und die Einstellung der Write-Precompensation-Time.

Write-Precompensation-Zeiten

P2	P1	P0	PIO Port B	5"	8"	
B1	B2	B3				
0	0	0	..m.000.	0	0	ns
0	0	1	..m.100.	125	62.5	ns
0	1	0	..m.010.	250	125	ns
0	1	1	..m.110.	375	187.5	ns
1	0	0	..m.001.	500	250	ns
1	0	1	..m.101.	500	250	ns
1	1	0	..m.011.	625	312.5	ns
1	1	1	..m.111.	625	312.5	ns

Die Punkte im Port B der PIO stehen für Bits, die für andere Funktionen benutzt werden, sie dürfen keinesfalls geändert werden. m ist das MINI/MAXI-Bit; eine 1 bedeutet MINI, eine 0 MAXI.

Belegung der Floppy-Steckers

pin	8" SHUGART	5,25" SHUGART	pin
	S3	S2	
2	LOW CURRENT	HEAD LOAD	2
4			4
6		READY ++++++	6
8		INDEX	8
10	TWO SIDED	SELECT 1	10
12		SELECT 2	12
14	SIDE SELECT	SELECT 3	14
16		MOTOR ON	16
18	HEAD LOAD	DIRECTION	18
20	INDEX	STEP	20
22	READY	WRITE DATA	22
24		WRITE GATE	24
26	SELECT 1	TRACK 00	26
28	SELECT 2	WRITE PROTECT	28
30	SELECT 3	READ DATA	30
32	SELECT 4	SIDE SELECT	32
34	DIRECTION	READY ++++++	34
36	STEP	-	36
38	WRITE DATA	-	38
40	WRITE GATE	-	40
42	TRACK 00	-	42
44	WRITE PROTECT	-	44
46	READ DATA	-	46
48		-	48
50		-	50

alle Signale von und zur Floppy sind aktiv LOW

nicht bezeichnete Pins sind nicht belegt. Bei 5,25" Laufwerken gibt es zwei Möglichkeiten für das READY-Signal: pin 34 z.B. bei TEAC oder pin 6 z.B. bei BASF. Dies wird mit den Jumpers J4 bzw. J5 eingestellt. Somit ist für TEAC-Laufwerke (READY = pin 34) J5 und für BASF-Laufwerke (READY = pin 6) J4 zu setzen.

Alle ungeraden Pins liegen auf Masse. Bei Verwendung eines gemeinsamen Netzteiles für die Laufwerke und den Rechner entsteht hierdurch eine Brummschleife. Dies kann zu Störungen auf den inneren Spuren der Diskette führen. Notfalls muß Ground von den ungeraden Pins abgetrennt werden. Weiterhin ist der maximal zulässige Störpegel der Versorgungsspannungen für die Laufwerke zu beachten (meist 50 millivolt Spitze-Spitze).

3.2 Winchester-(SASI)-teil

Das SASI-Interface ist in diskreter Logik aufgebaut. Es benutzt das Port A der PIO für die Control-Signale und TTL-Treiber/register für den Datenbus. Als Ausgangstreiber werden 7406 und 74LS642 benutzt. Im Eingang sind LS-TTL oder die PIO mit 220/330 Ohm Pull-up/Pull-down angeordnet. Das SASI-Interface ist voll interruptfähig, es wird jedoch von der Beispiels- und System-Software im Polling betrieben.

Da der Winchestercontroller XEBEC 1410 einen Sektorpuffer besitzt, ist die Datenübertragung nicht zeitkritisch. Es dürfen also fremde Interrupts auch beim Lesen oder Schreiben eines Blocks zugelassen sein!

Das SASI-Interface ist die OPEN-Collector-Version ohne Parity-Prüfung. ARBITRATION und RESELECTION sind nicht implementiert. Es darf also nur einen INITIATOR (Bus-Master) geben.

Steckerbelegung S4

pin	Signal	
2	DB(0)	DATA 0 NOT
4	DB(1)	DATA 1 NOT
6	DB(2)	DATA 2 NOT
8	DB(3)	DATA 3 NOT
10	DB(4)	DATA 4 NOT
12	DB(5)	DATA 5 NOT
14	DB(6)	DATA 6 NOT
16	DB(7)	DATA 7 NOT
18	DB(P)	DATA PARITY NOT (nicht implementiert)
20		
22		
24		
26		
28		
30		
32	ATN	ATTENTION (nicht implementiert)
34		
36	BSY	BUSY NOT
38	ACK	ACKNOWLEDGE NOT
40	RST	RESET
42	MSG	MESSAGE NOT
44	SEL	SELECT NOT
46	C/D	CONTROL NOT/DATA
48	REQ	REQUEST NOT
50	I/O	INPUT NOT/OUTPUT

Alle ungeraden pins liegen auf Masse-Potential, nicht bezeichnete pins sind nicht belegt.

4. I/O-Adressen

Die M.K.C. FDC II belegt insgesamt 14 I/O-Adressen von 0F0h bis 0FDh. (Die Adresseinstellung erfolgt im I/O-PROM und ist prinzipiell auch für andere Adressen möglich; es erfolgt hierfür aber keine Softwareunterstützung durch uns.)

0F0	uPD765	Statusregister
0F1	uPD765	Datenregister
0F2		sperrt über den CTC die IEI/IEO-Kette
0F3	uPD765	Data-Acknowledge
0F4	Z80-CTC	Kanal 0 (uPD765 Interrupt-Controller)
0F5	Z80-CTC	Kanal 1 (Motorlaufzeit-Vorteiler)
0F6	Z80-CTC	Kanal 2 (Motorlaufzeit-Teiler)
0F7	Z80-CTC	Kanal 3 (sperrt die IEI/IEO-Kette)
0F8	Z80-PIO	Port A Datenregister (SASI und Uhr)
0F9	Z80-PIO	Port A Kommandoregister
0FA	Z80-PIO	Port B Datenregister (Floppy und Uhr)
0FB	Z80-PIO	Port B Kommandoregister
0FC	SASI	Daten lesen
0FD	SASI	Daten schreiben

Belegung der PIO-Ports:

A0	INPUT	SASI-REQUEST
A1	INPUT	SASI-BUSY
A2	INPUT	SASI-IO
A3	INPUT	SASI-CD
A4	INPUT	SASI-MSG
A5	OUTPUT	SASI-SEL
A6	OUTPUT	SASI-RESET
A7	OUTPUT	MEM E050 Chip-Select
B0	INPUT	uPD765 DATA-REQUEST
B1	OUTPUT	FDC9229B P2 Precompensation
B2	OUTPUT	FDC9229B P1 Precompensation
B3	OUTPUT	FDC9229B P0 Precompensation
B4	OUTPUT	Motor triggern
B5	OUTPUT	5,25" / 8" Umschaltung
B6	OUTPUT	MEM E050 Clock
B7	IN/OUTPUT	MEM E050 Data In/Output

5. Prominhalte5.1 Bussteuer-Prom (IC 10) TBP24SA10

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
bis																
70	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
80	F	F	F	F	F	F	E	F	F	F	F	F	F	F	F	F
bis																
B0	F	F	F	F	F	F	E	F	F	F	F	F	F	F	F	F
C0	F	E	F	E	F	F	F	F	F	F	F	F	F	F	F	F
D0	F	E	F	E	E	F	F	F	F	F	F	F	E	F	F	F
E0	F	E	F	E	F	F	F	F	F	F	F	F	F	F	F	F
F0	F	E	F	E	F	F	F	F	F	F	F	F	F	F	F	F

5.2 Memory-Prom (IC 11) TBP24SA10

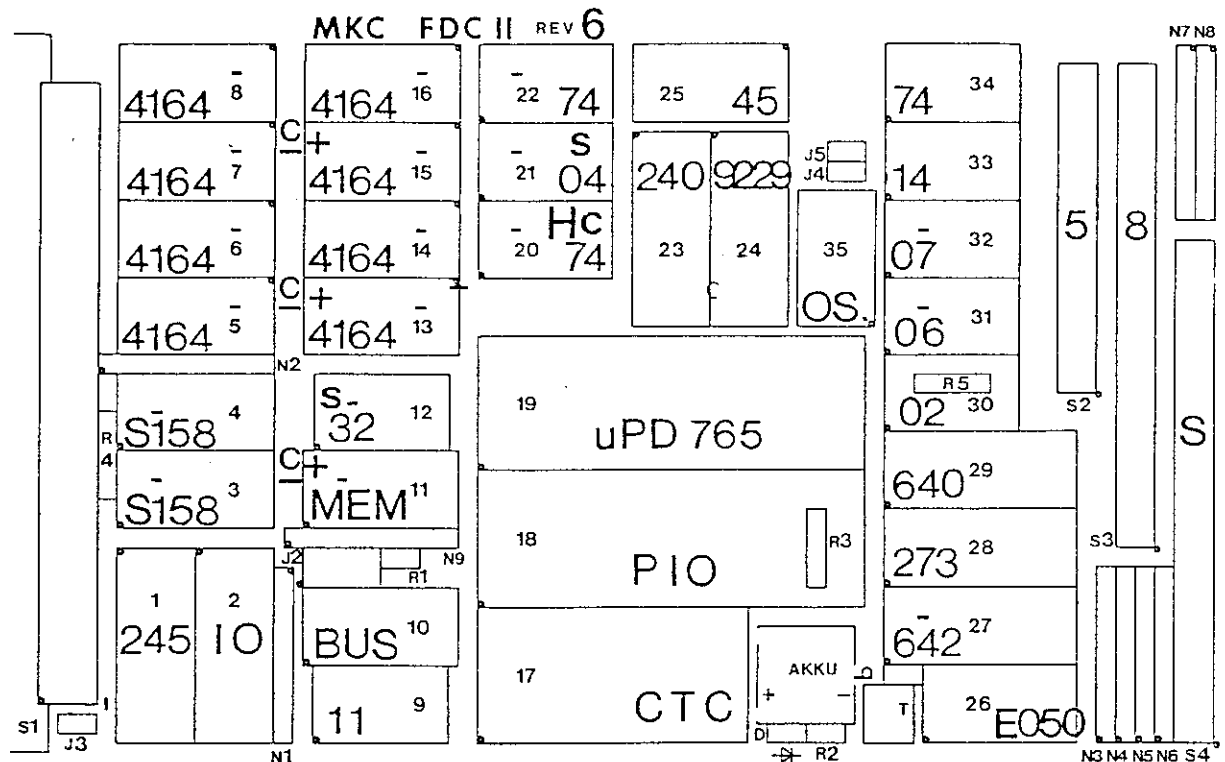
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	0	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
10	F	0	F	F	F	F	F	F	F	F	F	F	F	F	F	F
20	F	F	0	F	F	F	F	F	F	F	F	F	F	F	F	F
30	F	F	F	0	F	F	F	F	F	F	F	F	F	F	F	F
40	F	F	F	F	0	F	F	F	F	F	F	F	F	F	F	F
50	F	F	F	F	F	0	F	F	F	F	F	F	F	F	F	F
60	F	F	F	F	F	F	0	F	F	F	F	F	F	F	F	F
70	F	F	F	F	F	F	F	0	F	F	F	F	F	F	F	F
80	F	F	F	F	F	F	F	F	0	F	F	F	F	F	F	F
90	F	F	F	F	F	F	F	F	F	0	F	F	F	F	F	F
A0	F	F	F	F	F	F	F	F	F	F	0	F	F	F	F	F
B0	F	F	F	F	F	F	F	F	F	F	F	0	F	F	F	F
C0	F	F	F	F	F	F	F	F	F	F	F	F	0	F	F	F
D0	F	F	F	F	F	F	F	F	F	F	F	F	F	0	F	F
E0	F	F	F	F	F	F	F	F	F	F	F	F	F	F	0	F
F0	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	0

5.3 I/O-Dekoder-Prom (IC 2) TBP28S42

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
bis																
150	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
160	FF	FF	FF	E7	FF	FF	FF	DF	FF	FF	FF	F6	FF	FF	FF	F6
170	FF	FF	FF	E7	FF	FF	FF	B7	FF	FF	FF	F6	FF	FF	FF	F6
180	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
bis																
1D0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
1E0	FF	FF	FF	77	FF	FF	FF	77	FF	FF	FF	F5	FF	FF	FF	FF
1F0	FF	FF	FF	77	FF	FF	FF	77	FF	FF	FF	F3	FF	FF	FF	FF

6.1 Stückliste (die fett gedruckten Teile sind in MKC-Z-3048)

IC 1	74 LS 245 * == selekt. SCHOTTKY ==
IC 2	TBP 28 S 42 prog. (I/O-Prom)
IC 3, 4	74 S 158 o. 74 S 157 == SCHOTTKY ==
IC 5..8, 13..16	NEC 4164 o.ä. (64 K * 1 dyn. Ram, 128-REFRESH-Zyklen, 4Mhz 150ns / 5 oder 6MHz 120ns)
IC 9	74 LS 11
IC 10	TBP 24 SA 10 prog. (Bus-Prom)
IC 11	TBP 24 SA 10 prog. (MEM-Prom)
IC 12	74 S 32 * == selekt. SCHOTTKY ==
IC 17	280 A(B) CTC
IC 18	280 A(B) PIO
IC 19	uPD 765
IC 20	74 HC 74 == CMOS ==
IC 21	74 S 04 == SCHOTTKY ==
IC 22, 34	74 LS 74
IC 23	74 LS 240
IC 24	SMC FDC 9229B
IC 25	7445 == Achtung: LAGE ==
IC 26	MEM E050-16 Uhr
IC 27	74 LS 642
IC 28	74 LS 273
IC 29	74 LS 640
IC 30	74 LS 02
IC 31	7406 oder 7416
IC 32	7407 oder 7417
IC 33	74 LS 14
IC 35	TTL - Oszillator 16 MHz
Q	Quarz 32,768 KHz
D 1	1N4148 o.ä.
R 1	680 R
R 2, 3, 4	2K2
R 5, 6	470 R
N 1	SIL 8 * 680 R, pin 1 = common
N 2	SIL 8 * 1K2, pin 1 = common
N 3, 5, 8	SIL 8 * 220 R, pin 1 = common
N 4, 6, 7	SIL 8 * 330 R, pin 1 = common
N 9	SIL 8 * 2K2, pin 1 = common
T	3 bis 18 pF Trimmer
AKKU	2,4 Volt
C 1 bis 18	100 nF Vielschicht-C RM 2,54 mm
C 19..21	10 uF/ 16 V Tantal
S 1	VG 64 a,c bestückt
S 2	34 pin Pfostenleiste
S 3, 4	50 pin Pfostenleiste
J 1, 3, 4, 5	2 pin Pfostenleiste
J 2	2 * 4 pin Pfostenleiste
14 Fassungen DIL-16	offen
10 Fassungen DIL-14	offen
7 Fassungen DIL-20	offen
2 Fassungen DIL-40	offen
1 Fassung DIL-28	offen
6 zweifach-Kurzschlußstecker	

6.2 Bestückungsplan6.3 Aufbauhinweise

Die Kondensatoren C1 bis C18 liegen fast alle unter den ICs. Daher sind offene Fassungen zu verwenden. Kondensatoren und Fassungen sollten gemeinsam bestückt werden. Bei den Widerstandsnetzwerken markiert der Punkt den gemeinsamen Anschluß (pin 1). Bei der Montage des Quarzoszillators ist darauf zu achten, daß das Metallgehäuse die danebenliegenden Pins der IC-Fassungen nicht berühren darf. Der AKKU sollte erst nach Fertigstellung der Karte eingelötet werden. Die Kondensatoren C19 bis C21 fehlen leider im Bestückungsplan auf der Platine, in obigem Bestückungsplan sind sie richtig eingezeichnet. Falls ein CMOS-Quarzoszillator verwendet wird, muß in den pin 11 des FDC92298 ein Widerstand von 1K5 eingebaut werden. Ein TTL-Oszillator ist jedoch wesentlich besser.

Das Uhren-IC MEM E050-16 darf erst nach Einbau des AKKUS eingesetzt werden.

Die Anschlußkabel zu den Laufwerken werden sehr eng nebeneinander gesteckt. Dies ist relativ einfach, wenn alle Kabel vom vorderen Rand der Karte kommen und man die Reihenfolge SAS1, 8" und erst dann 5,25" einhält. Es sollten nur Quetschstecker ohne Polarisierung und Zugentlastung benutzt werden.

7. Jumper-Einstellungen

Auf der M.K.C. FDC II rev. 6 gibt es folgende Jumper:

J2	8 pin	binäre Einstellung der Pageadressen für den 64KByte Speicher
	1-2	A19 (geschlossen: = 0)
	3-4	A18 (geschlossen: = 0)
	5-6	A17 (geschlossen: = 0)
	7-8	A16 (geschlossen: = 0)
J3	2 pin	geschlossen: WAIT wird bei 6 MHz Systemtakt und 5,25" Floppylaufwerken benötigt.
J4	2 pin	geschlossen: READY = pin 6 MiniFloppy (für z.B. BASF)
J5	2 pin	geschlossen: READY = pin 34 Minifloppy (für z.B. TEAC)
S2	Anschlußstecker für 5,25" Floppy	
S3	Anschlußstecker für 8" Floppy	
S4	Anschlußstecker für SASI-Controller	

Auf dem XEBEC-Controller S1410 ist 512 Byte Sektorlänge zu jumpern (W3 nach 5 stecken). Beim Adress-Jumper muß S-0 verbunden sein (Voreinstellung). Das Winchester-Laufwerk muß Unit 0 sein! Der Terminator muß im XEBEC-Controller sein.

Beim XEBEC S1410A ist W1 nach 5 zu stecken (Sektorgröße 512 Byte). Achtung: J5 muß in Stellung 0 stehen!

Bei Anschluss von 5,25" und 8" Laufwerken muß der Terminator in dem jeweils letzten Laufwerk sein!

Achtung:

Es dürfen nur maximal 2 Widerstandsnetzwerke in den Floppy-Laufwerken gesteckt sein

Auslieferungszustand:

J2:	1-2 gesteckt	
	3-4 gesteckt	
	5-6 gesteckt	
	7-8 offen	=> Adresse: 10000 bis 1FFFF
J3:	bei 4 MHz offen	
	bei 6 MHz gesteckt	=> Kein WAIT bei 4 MHz
J4:	offen	
J5:	gesteckt	=> TEAC - Laufwerke

```

                                TITLE  FDC II VERSION 1.2 (25.09.84 L.K.)
                                .Z80
                                .PHASE 100H          ; ORG 100H
                                ;
                                ;   Testroutinen fuer den M.K.C. FDC II
                                ;
00F0      HLT      EQU      240      ; HEADLOADTIME IN MS (8 MHZ CLOCK)
000E      SRT      EQU      14      ; STEPRAETETIME IN MS (8 MHZ CLOCK)
00F0      HUT      EQU      240      ; HEADUNLOADTIME IN MS (8 MHZ CLOCK)
                                ;
00EE      HLTS     EQU      (<HLT-1>)/2)*2
000F      HUTS     EQU      (<HUT+15>)/16
0002      SRTS     EQU      16-SRT
                                ;
                                ;   Mini-Disketten (400 OR 800 KBYTE)
                                ;
                                ;   DOUBLE DENSITY
                                ;   40 or 80 TRACKS / DISK (each side)
                                ;   10 SECTORS / TRACK
                                ;   512 BYTE / SECTOR
                                ;   DOUBLE SIDED
                                ;
000A      MAXSEC   EQU      10      ; 10 SECTORS
001E      FMTGAP   EQU      1EH     ; END GAP LENGTH
                                ;
                                ;   RETRY COUNTER (NORM. 10 TIMES)
                                ;
000A      RETRY    EQU      10
                                ;
                                ;   I/O PORTS
                                ;
00F3      DACK     EQU      0F3H     ; DMA ACKNOWLEDGE
00F0      FDC      EQU      0F0H     ; STATUS PORT
00F1      FDD      EQU      FDC+1    ; DATA PORT
00F2      FTC      EQU      FDD+1    ; SECTOR TERMINATION
00F4      CTC0     EQU      0F4H     ; uPD 765 interrupt
00F6      TIMER    EQU      0F6H     ; MOTOR-TIMER
00F8      P10A     EQU      0F8H     ; SASI AND CLOCK
00FA      P10B     EQU      0FAH     ; FLOPPY TIMING AND CLOCK
00F5      PRESC    EQU      0F5H
0047      CTCMOD   EQU      01000111B
                                ;
0100 025B 0000  vector: defw  fdint,0,0,0
0104 0000 0000
0108 0000 0000      defw  0,0,0,0
010C 0000 0000

```

```

;
;      Command Table for uPD765
;
0110 00      CMDTAB::DEFB      0      ; COMMAND BYTE
0111 01      UNIT::  DEFB      1      ; UNIT (0...3)
0112 1E      TRACK:: DEFB      30     ; TRACK (0...39/77)
0113 00      HEAD::  DEFB      0      ; HEAD (0 for SIDE 0, 1 for SIDE 1)
; single sided is SIDE 0 only!
0114 01      SECTOR::DEFB      1      ; first SECTOR to read (1...10)
0115 02              DEFB      2      ; N = 512 Bytes/Sector
0116 0A      LSTSEC::DEFB      MAXSEC ; last SECTOR to read
0117 10              DEFB      10H    ; READ/WRITE-GAP LENGTH
0118 80              DEFB      128    ; DATA-LENGTH (128 BYTE SECTORS only!)
;
;      RESULT-TABLE FOR READ, WRITE AND READ ID
;
0119 00      REST:  DEFB      0      ; STATUS 0 (01x00xxx ist kein Fehler)
; D7 & D6 = 00 normal Termination
;          = 01 started, not succesful
;          = 10 Invalid command
;          = 11 Ready-Line Changes
; D5       = 1 Seek completed
; D4       = 1 Fault from Floppy or
;           Track 0 not found
; D3       = 1 Floppy not ready
; D2       = Head Address at Interrupt
; D1 & D0 = Unit-Number at Interrupt
011A 00              DEFB      0      ; STATUS 1
; D7       = 1 Cylinderend (no error!)
; D6       = 0 not used
; D5       = 1 CRC-Error
; D4       = 1 Overrun, not serviced
;           (READ, WRITE or FORMAT)
; D3       = 0 not used
; D2       = 1 ID not found
; D1       = 1 write protected
; D0       = 1 Missing Address Mark
011B 00              DEFB      0      ; STATUS 2
; D7       = 0 not used
; D6       = 1 deleted Address-Mark
; D5       = 1 CRC-Error in Data-Field
; D4       = 1 Wrong Cylinder
; D3       = 1 Scan equal hit
; D2       = 1 Scan not satisfied
; D1       = 1 Bad cylinder
; D0       = 1 Missing Address-Mark
011C 00              DEFB      0      ; TRACK
011D 00              DEFB      0      ; HEAD
011E 00              DEFB      0      ; SECTOR
011F 00              DEFB      0      ; N (SECTOR-LENGTH)
0120 00      errflg::defb      0      ; 0 if no error

```

```

;
;      INITIALIZE UPD 765 C and ZILOG-Chips, set IM2
;      ( SOFT RESET )
;*****
0121      INIFDC::      ; NO PARAMETERS
;*****
0121 F3      di
0122 21 0100      ld      hl,vector
0125 7C      ld      a,h
0126 ED 47      ld      i,a
0128 ED 5E      im      2
012A 7D      ld      a,1
012B D3 F4      out     (ctc0),a      ; interrupt vector for ctc
012D 3E D7      ld      a,11010111b  ; counter to generate interrupt
012F D3 F4      out     (ctc0),a
0131 3E 01      ld      a,1
0133 D3 F4      out     (ctc0),a
0135 3E 27      LD      A,00100111b  ; SET MOTOR-PRESCALER-CHANNEL
0137 D3 F5      OUT     (PRESC),A
0139 3E 00      LD      A,00
013B D3 F5      OUT     (PRESC),A
013D 3E CF      LD      A,0CFH      ; SET PIO
013F D3 F9      OUT     (PIOA+1),A
0141 3E 1F      LD      A,1FH
0143 D3 F9      OUT     (PIOA+1),A
0145 3E CF      LD      A,0CFH
0147 D3 FB      OUT     (PIOB+1),A
0149 3E 01      LD      A,1
014B D3 FB      OUT     (PIOB+1),A
014D 3E 38      LD      A,38H      ; 125 ns 5 1/4"
;      LD      A,18H      ; 125 ns 8"
014F D3 FA      OUT     (PIOB),A
0151 FB      ei
0152 06 0A      ini2:  ld      b,10
0154 10 FE      LOOP:  DJNZ    LOOP
0156 DB F0      IN      A,(FDC)
0158 FE 80      CP      80H      ; REQUEST FOR MASTER
015A 28 04      JR      Z,INI1    ; OK
015C DB F1      IN      A,(FDD)   ; ELSE FETCH DATA
015E 18 F2      JR      INI2
0160 C9      INI1:  RET      ; DONE
;
;      SPECIFY DISC PARAMETERS pseudo-DMA-MODE !!
;      head-load-time, step-rate-time and head-unload-time
;****
0161      SPEC::      ; NO PARAMETERS
;****
0161 2A 0111      LD      HL,(UNIT)
0164 E5      PUSH    HL
0165 21 EE2F      LD      HL,HLTS*256+SRTS*16+HUTS      ; dma-mode
0168 01 0303      LD      BC,0303H
016B 22 0111      LD      (UNIT),HL
016E CD 023E      CALL    CMFD
0171 E1      POP     HL
0172 22 0111      LD      (UNIT),HL
0175 C9      SPE2:  RET

```

```

;
;   move head to track 0, Track-Register is 0 after RESET
;   necessary one time for each Unit after RESET
;*****
0176 RECAL:: ; PARAMETERS: UNIT
;*****
0176 3E FF          ld      a,0ffh          ; track not reached
0178 32 0289        ld      (skdone),a
017B 01 0207        LD      BC,0207H      ; RECALIBRATE
017E CD 021E        CALL     MOTO          ; WAIT FOR DISK
0181 3A 0289        reclop: ld      a,(skdone)
0184 3C             inc      a
0185 28 FA          jr      z,reclop        ; wait until track reached
0187 C9             REC1:  RET             ; a = 20h ok, if not: aborted
;
;   SEEK TRACK
;   move the Head to a specified Track
;****
0188 SEEK:: ; PARAMETERS: UNIT, TRACK
;****
0188 3E FF          ld      a,0ffh
018A 32 0289        ld      (skdone),a
018D 01 030F        LD      BC,030FH      ; SEEK TRACK
0190 CD 021E        CALL     MOTO
0193 3A 0289        seklop: ld      a,(skdone)
0196 3C             inc      a
0197 28 FA          jr      z,seklop
0199 C9             SK1:   RET             ; YES => ALL DONE
;
;   READ ID-FIELD
;   to find the actual Track, read double density
;*****
019A RDIDF:: ; PARAMETERS: UNIT
;*****
019A 11 01A6        ld      de,rdid1
019D D5             push    de
019E 01 024A        LD      BC,024AH      ; READ ID
01A1 CD 021E        CALL     MOTO
01A4 18 FE          RDIDL: JR      RDIDL    ; WAIT FOR RESULTS
01A6 C9             RDID1: RET
;
;   SENSE DRIVE STATUS
;
;****
01A7 SDRS:: ; PARAMETERS: UNIT
;****
01A7 01 0204        LD      BC,0204H      ; SENSE DRIVE
01AA CD 023E        CALL     CMFD
01AD CD 0253        CALL     NEXT          ; A=STATUS 3
01B0 C9             SD1:   RET             ; D7 = Fault
;                                     ; D6 = Write Protected
;                                     ; D5 = Ready
;                                     ; D4 = Track 0
;                                     ; D3 = two sided
;                                     ; D2 = Head-Address
;                                     ; D1,D0 = Unit

```

```

;
; WRITE 512 SECTOR TO DISK DMA MODE !!
;*****
01B1 WR512: ; PARAMETERS: UNIT, TRACK, first SECTOR, last SECTOR
;*****
01B1 06 0A LD B,RETRY ; RETRY COUNTER
01B3 C5 WR5: PUSH BC ; SAVE COUNTER
01B4 01 0945 LD BC,0945H ; WRITE
01B7 CD 021E CALL MOTO
01BA 21 1000 LD HL,DBUF ; .BUFFER
01BD 0E F3 LD C,dack ; 256 BYTES, DATA-dma PORT
01BF 11 01CD LD DE,WR99 ; EXEC-END-ADDRESS
01C2 D5 PUSH DE ; ON STACK
01C3 DB FA WR6: IN A,(piob) ; REQUEST FOR MASTER ?
01C5 0F rrca
01C6 30 FB jr nc,wr6
01C8 ED A3 OUTI ; YES: TRANSFER BYTE
01CA C3 01C3 JP WR6
01CD C1 WR99: POP BC ; UNSAVE COUNTER
01CE 3A 0120 ld a,(errflg)
01D1 B7 or a
01D2 28 02 JR Z,WR8 ; NO ERRORS
01D4 10 DD DJNZ WR5 ; ERROR: TRY AGAIN
; FATAL ERROR
; RETURNS WITH A = 0 IF NO ERROR
; AND WITH A .NE. 0 IF ERRORS OCCURED
01D6 C9 WR8: RET
;
; READ 512 SECTOR DMA MODE !
;*****
01D7 RD512: ; PARAMETERS: UNIT, TRACK, first SECTOR, last SECTOR
;*****
01D7 06 0A LD B,RETRY ; COMPARE WR512
01D9 C5 RD4: PUSH BC
01DA 01 0946 LD BC,0946H ; READ
01DD CD 021E CALL MOTO
01E0 21 1000 LD HL,DBUF
01E3 11 01F3 ld de,rd99
01E6 D5 push de
01E7 0E F3 LD C,dack
01E9 DB FA RD5: IN A,(piob)
01EB 0F RRCA
01EC 30 FB JR NC,RD5
01EE ED A2 INI
01F0 C3 01E9 JP RD5
01F3 C1 rd99: POP BC
01F4 3A 0120 ld a,(errflg) ; error?
01F7 B7 or a
01F8 28 02 JR Z,RD7
01FA 10 DD DJNZ RD4
; RETURNS WITH A = 0 IF NO ERROR
; AND WITH A .NE. 0 IF ERRORS OCCURED
01FC C9 RD7: RET
;

```

```

01FD      RESULT: ; FETCH RESULTS OF READ/WRITE OPERATION, store in REST
           ; AND CHECK THEM FOR R/W-ERRORS
           ; DESTROYES: AF,BC,HL
           ; RETURNS WITH: Z & A=0, IF NO ERRORS
           ; AND WITH: NZ & A=? IF ANY ERROR
           ;
01FD 06 07      LD      B,7           ; 7 RESULTS
01FF 21 0119    LD      HL,REST       ; RESULT-TABLE
0202 CD 0253    RESLOP: CALL  NEXT
0205 77         LD      (HL),A        ; STORE RESULT
0206 23         INC      HL
0207 10 F9      DJNZ    RESLOP
0209 3A 0119    ld      a,(rest)
020C E6 98      and     10011000b
020E 4F         ld      c,a          ; save
020F 3A 011A    ld      a,(rest+1)
0212 E6 37      and     00110111b
0214 B1         or      c
0215 4F         ld      c,a
0216 3A 011B    ld      a,(rest+2)
0219 E6 7F      and     01111111b
021B B1         or      c          ; a = 0 if no error
021C 77         ld      (hl),a      ; store error-flag
021D C9         RET
           ;
021E      MOTO:  ; WAITS UNTIL DISK READY AND THEN
           ; TRANSMITS COMMAND TO FD-CONTROLLER
           ; DESTROYES: AF,BC,HL
           ;
021E 3E 47      LD      A,CTCMOD      ; start timer
0220 D3 F6      OUT     (TIMER),A
0222 AF         XOR     A
0223 D3 F6      OUT     (TIMER),A
0225 DB FA      in      a,(piob)      ; start motor
0227 CB A7      res     4,a
0229 D3 FA      out     (piob),a
022B CB E7      set     4,a
022D D3 FA      out     (piob),a
022F C5         PUSH    BC           ; SAVE COMMAND
0230 01 0204    MOTO1: LD      BC,0204H ; SENSE DRIVE
0233 CD 023E    CALL    CMFD
0236 CD 0253    CALL    NEXT         ; FETCH RESULT
0239 CB 6F      BIT     5,A          ; DISK READY ?
023B 28 F3      JR      Z,MOTO1      ; NO => WAIT
023D C1         POP     BC           ; YES => FETCH COMMAND
           ;

```

```

023E          CMFD:  ; TRANSMITS COMMAND TO FD-CONTROLLER
                  ; DESTROYES: AF,BC,HL
                  ;
023E 21 0110      LD      HL,CMDTAB      ; POINT TO COMMANDTABLE
0241 71           LD      (hl),C        ;
0242 0E F1        ld      c,fdd         ; data port
0244 3E 05        ld      a,5
0246 3D          cmdlop: dec      a
0247 20 FD        jr      nz,cmdlop
0249 DB F0        in      a,(fdc)       ; uPD ready for command
024B 07           rlca
024C 30 F8        jr      nc,cmdlop
024E ED A3        outi
0250 20 F4        jr      NZ,CMDLOP     ; SEND NEXT BYTE
0252 C9          RET                   ; ALL BYTES TRANSFERRED
;
0253          NEXT:  ; READ NEXT RESULT-BYTE FROM FD-CONTROLLER
                  ; DESTROYES: AF
                  ;
0253 DB F0        next1: IN      A,(FDC)      ; REQUEST FOR MASTER &
0255 07           rlca
0256 30 FB        JR      nc,NEXT1        ; NO => WAIT
0258 DB F1        IN      A,(FDD)        ; YES => FETCH RESULT
025A C9          RET
;
;          ***** interrupt service routine *****
;
025B 08          fdint: ex      af,af'
025C D9          exx
025D DB F0        fdint1: in      a,(fdc)      ; read status
025F CB 7F        bit      7,a            ; request for master
0261 28 FA        jr      z,fdint1        ; no: wait
0263 CB 77        bit      6,a
0265 28 06        jr      z,sense        ; not result phase
;          ***** result phase *****
0267 CD 01FD      call      result
026A E1           pop      hl            ; don't return into loop
026B 18 17        jr      sensend
026D          sense: ;***** sense interrupt status *****
026D 01 0108      ld      bc,0108h
0270 CD 023E      call      cmfd
0273 CD 0253      call      next
0276 F5           push     af
0277 CD 0253      call      next
027A F1           pop      af
027B E6 E0        and      0e0h
027D FE C0        cp      0c0h
027F 28 03        jr      z,sensend
;          ***** seek or recal done *****
0281 32 0289      ld      (skdone),a
0284          sensend:
0284 08          sense1: ex      af,af'
0285 D9          exx
0286 FB          ei
0287 ED 4D        reti
;

```

```
0289 FF      SKDONE: DEFB      0FFH      ; NOT DONE, 20H = DONE
              ;
              ;      SECTOR BUFFER
              ;
1000      DBUF      EQU      1000H      ; SECTOR BUFFER
              ;
              .DEPHASE
              ;
              END
```

```

                                title    winchester-driver v.1.6
;
;      aufrufbare Routinen:
;          INIT:   initialisieren der Hardware
;          HDINIT: reset XEBEC 1410
;          HDREAD: einen Block lesen
;          HDWRIT: einen Block schreiben
;
;*****
;*      Non-interrupt SASI-driver      *
;*****
;      .z80
;*****
;*      i/o-ports                      *
;*****
00F8      fpioa    equ    0f8h          ; pio a-port
00FC      sasiin   equ    0fch          ; sasi read
00FD      sasiout  equ    0fdh          ; sasi write
;*****
;*      SASI-Status bits              *
;*****
0003      cd      equ    3             ; command/data
0002      io      equ    2             ; input/output
0000      req     equ    0             ; request
0001      busy    equ    1             ; busy
;*****
;*      initialize FDC II SASI-part   *
;*****
0000' 3E CF      init:  ld      a,0cfh
0002' D3 F8          out      (fpioa),a
0004' 3E 1F          ld      a,1fh
0006' D3 F8          out      (fpioa),a
0008' C9            ret
;*****
;*      reset hard-disk-controller    *
;*****
0009' DB F8      hdinit: in      a,(fpioa)      ; reset SASI-controller
000B' CB F7          set      6,a              ; reset on
000D' D3 F8          out      (fpioa),a
000F' CB B7          res      6,a
0011' D3 F8          out      (fpioa),a      ; reset off

```

```

;*****
;*      read sector from hard-disk      *
;*****
0013' CD 005F' hread: call   hdrw           ; compute block-number
0016' CD 008A' hrdrl: call   select        ; select controller
0019' 3E 08          ld     a,8           ; read-command
001B' CD 00A2'          call   cmdout       ; send command
001E' 2A 00EA'          ld     hl,(Sdma)    ; buffer
0021' 0E FC          ld     c,sasiin
0023' CD 0083' hrdrlp: call   rqwait       ; byte ready?
0026' CB 5F          bit     cd,a
0028' 28 05          jr      z,hrdone      ; end of sector
002A' ED A2          ini      ; fetch byte
002C' C3 0023'          jp     hrdrlp      ; next byte
002F' CD 00AF' hrdone: call   getstat      ; fetch status
0032' C8          ret     z
0033' CD 00BE'          call   herror      ; hard-disk-error
0036' 20 DE          jr      nz,hrdrl      ; try again
0038' C9          ret

;*****
;*      write sector to hard-disk      *
;*****
0039' CD 005F' hwrit: call   hdrw           ; compute block-number
003C' CD 008A' hwrl: call   select        ; select controller
003F' 3E 0A          ld     a,10
0041' CD 00A2'          call   cmdout       ; send command
0044' 2A 00EA'          ld     hl,(Sdma)    ; buffer
0047' 0E FD          ld     c,sasiout
0049' CD 0083' hwrlp: call   rqwait       ; byte ready?
004C' CB 5F          bit     cd,a
004E' 28 05          jr      z,hwrone      ; end of sector
0050' ED A3          outi      ; send byte
0052' C3 0049'          jp     hwrlp      ; next byte
0055' CD 00AF' hwrone: call   getstat      ; fetch status
0058' C8          ret     z
0059' CD 00BE'          call   herror      ; hard-disk-error
005C' 20 DE          jr      nz,hwrl      ; try again
005E' C9          ret

;*****
;*      internal subroutine hdrw      *
;*****
005F' CD 0077' hdrw: call   seldrv
0062' 3A 00E9'          ld     a,(Ssect)
0065' 32 00DF'          ld     (la),a      ; lowest block-address (8 bit)
0068' 2A 00E7'          ld     hl,(Strk)
006B' 7D          ld     a,l
006C' 32 00DE'          ld     (ma),a      ; middle block-address
006F' 3A 00DD'          ld     a,(drive)
0072' B4          or      h
0073' 32 00DD'          ld     (drive),a   ; highest block-address
0076' C9          ret

```

```

;*****
;*      internal subroutine seldrv      *
;*****
0077' 3A 00E6' seldrv: ld      a,(&rdrv)      ; relativ drive
007A' 06 05      ld      b,5
007C' 87      hdrwl: add     a,a
007D' 10 FD      djnz     hdrwl      ; compute unit
007F' 32 00DD'      ld      (drive),a
0082' C9      ret

;*****
;*      internal subroutine wait for request  *
;*****
0083' DB F8      rqwait: in      a,(fpioa)
0085' CB 47      bit      req,a
0087' 20 FA      jr      nz,rqwait
0089' C9      ret

;*****
;*      internal subroutine select      *
;*****
008A' DB F8      select: in      a,(fpioa)
008C' CB 4F      bit      busy,a
008E' 28 FA      jr      z,select
0090' CB EF      set      5,a      ; select
0092' 0E FD      ld      c,sasiout
0094' 06 01      ld      b,1
0096' ED 41      out      (c),b      ; controller-address
0098' D3 F8      out      (fpioa),a      ; select on
009A' CB AF      res      5,a      ; select off
009C' D3 F8      out      (fpioa),a
009E' 05      dec      b      ; b = 0
009F' 10 FE      sloop: djnz     sloop
00A1' C9      ret

;*****
;*      internal subroutine command-output  *
;*****
00A2' 01 06FD      cmdout: ld      bc,600h+sasiout ; 6 bytes command
00A5' 21 00DC'      ld      hl,task      ; command buffer
00A8' 77      ld      (hl),a      ; store command
00A9' CD 0083'      call     rqwait      ; wait for request
00AC' ED B3      otir      ; send command-bytes
00AE' C9      ret

;*****
;*      internal subroutine getstat      *
;*****
00AF'      getstat:
00AF' CD 0083'      call     rqwait      ; wait for request
00B2' DB FC      in      a,(sasiin)      ; fetch result
00B4' F5      push     af
00B5' CD 0083'      call     rqwait      ; wait for request
00B8' DB FC      in      a,(sasiin)      ; fetch second result
00BA' F1      pop      af
00BB' E6 02      and      2      ; nz = error
00BD' C9      ret

```

```

;*****
;*      internal subroutine hard-disk-error      *
;*****
00BE' CD 008A'  herror: call    select
00C1' 3E 03          ld      a,3          ; request sense status
00C3' CD 00A2'          call    cmdout
00C6' 0E FC          ld      c,sasiin
00C8' 21 00E2'          ld      hl,hdres          ; result area
00CB' CD 0083'  herror: call    rqwait
00CE' CB 5F          bit      cd,a
00D0' 28 04          jr      z,herr9
00D2' ED A2          ini
00D4' 18 F5          jr      herror
00D6' 3A 00E2'  herr9: ld      a,(hdres)          ; result-byte
00D9' E6 7F          and      7fh          ; error?
00DB' C9          ret

;*****
;*      command-table for SCSI-controller      *
;*****
00DC' 00  task: defb      0          ; command
00DD'          drive:
00DD' 00  ha:      defb      0          ; drive and highest address
00DE' 00  ma:      defb      0          ; middle address
00DF' 00  la:      defb      0          ; lowest address
00E0' 01  blkcnt: defb      1          ; block-count
00E1' 80          defb      80h          ; control-byte

;*****
;*      result-table for SCSI-controller      *
;*****
00E2' 0000 0000  hdres: defb      0,0,0,0          ; result area
;*****
;*      unit, track, sector, dma-address      *
;*****
00E6' 00  $drv:  defb      0          ; unit 0
00E7' 0000  $trk:  defw      0          ; track 0
00E9' 00  $sect: defb      0          ; sector 0
00EA' 00EC'  $dma:  defw      buffer          ; dma-buffer
00EC'          buffer: defs      512          ; sector-size
; = 512 byte

end

```

```

                                title    memtime vers. 1.0 (29.08.83)
;*****
;*      real-time clock MEM      on FDC II      *
;*****
                                .z80
00F8      fpioa    equ     0f8h
00FA      fpiob    equ     0fah
0007      csmem    equ     7          ; chipselect (fpioa bit 7)
0006      clmem    equ     6          ; clock      (fpiob bit 6)
0007      damem    equ     7          ; data       (fpiob bit 7)
;*****
;*      initialize FDC-II      *
;*****
0000' 3E CF      init:    ld      a,0cfh
0002' D3 F9              out     (fpioa+1),a
0004' 3E 1F              ld      a,1fh
0006' D3 F9              out     (fpioa+1),a
0008' 3E CF              ld      a,0cfh
000A' D3 FB              out     (fpiob+1),a
000C' 3E 01              ld      a,1
000E' D3 FB              out     (fpiob+1),a
0010' C9              ret
;*****
;*      set the clock          *
;*****
0011' CD 0077'      write:  call   start
;***** continuous write *****
0014' 06 04              ld      b,4          ; 4 bit command
0016' 0E E0              ld      c,11100000b    ; continuous write
0018' CD 009B'          call   outb
;***** send data to mem-clock *****
001B' 3A 00E5'          ld      a,(Shour)
001E' CD 0092'          call   outb8
0021' 3A 00E6'          ld      a,(Smin)
0024' CD 0092'          call   outb8
0027' 21 00E1'          ld      hl,save
002A' 06 04              ld      b,4
002C' 7E              writ1: ld      a,(hl)
002D' CD 0092'          call   outb8
0030' 23              inc      hl
0031' 10 F9              djnz    writ1
0033' 3A 00E7'          ld      a,(Ssec)
0036' CD 0092'          call   outb8
;***** chip-select off *****
0039' DB F8      exit:    in      a,(fpioa)
003B' C8 FF              set     csmem,a
003D' D3 F8              out     (fpioa),a
003F' C9              ret

```

```

;*****
;*      read MEM-clock      *
;*****
0040'      read: ;***** start values *****
0040' CD 0077'      call      start
;***** send read-command *****
0043' 06 04      ld        b,4
0045' 0E F0      ld        c,11110000b
0047' CD 009B'      call      outb
;***** change I/O-mode of fpiob *****
004A' DB FA      in        a,(fpiob)
004C' F5      push      af
004D' 3E CF      ld        a,0cfh          ; bit-mode
004F' D3 FB      out      (fpiob+1),a
0051' 3E 81      ld        a,81h          ; bit 7 to input
0053' D3 FB      out      (fpiob+1),a
0055' F1      pop      af
0056' D3 FA      out      (fpiob),a
;***** fetch 7 bytes from MEM-clock *****
0058' 21 00E5'      ld        hl,$hour
005B' CD 00BC'      call      inb
005E' 21 00E6'      ld        hl,$min
0061' CD 00BC'      call      inb
0064' 21 00E1'      ld        hl,$ave
0067' 06 04      ld        b,4
0069' CD 00BC'      read1: call      inb          ; fetch byte
006C' 23      inc      hl
006D' 10 FA      djnz     read1
006F' 21 00E7'      ld        hl,$sec
0072' CD 00BC'      call      inb
;***** chip-select off *****
0075' 18 C2      jr      exit
;*****
;*      set starting mode and pins for clock      *
;*****
0077' DB FA      start: in        a,(fpiob)
0079' F5      push      af
007A' 3E CF      ld        a,0cfh          ; bit-mode
007C' D3 FB      out      (fpiob+1),a
007E' 3E 01      ld        a,01h          ; bit 7 to output
0080' D3 FB      out      (fpiob+1),a
0082' F1      pop      af
0083' D3 FA      out      (fpiob),a
0085' DB FA      in        a,(fpiob)
0087' CB F7      set      clmem,a          ; set clock to high
0089' D3 FA      out      (fpiob),a
008B' DB F8      in        a,(fpioa)
008D' CB BF      res      csmem,a          ; chipselect to low
008F' D3 F8      out      (fpioa),a
0091' C9      ret

```

```

;*****
;*      send byte to clock      *
;*****
0092' C5      outb8:  push    bc
0093' 4F              ld      c,a
0094' 06 08              ld      b,8
0096' CD 009B'          call    outb
0099' C1              pop     bc
009A' C9              ret

;*****
;*      output <b> bits from <c> to clock      *
;*****
009B' F5      outb:  push    af
009C' C5              push    bc
009D' DB FA      outl:  in      a,(fpiob)
009F' 07              rlca
00A0' CB 01              rlc     c
00A2' 1F              rra
00A3' D3 FA              out     (fpiob),a
00A5' DB FA              in      a,(fpiob)
00A7' CB B7              res     clmem,a
00A9' D3 FA              out     (fpiob),a
00AB' CD 00DA'          call    wait
00AE' DB FA              in      a,(fpiob)
00B0' CB F7              set     clmem,a
00B2' D3 FA              out     (fpiob),a
00B4' CD 00DA'          call    wait
00B7' 10 E4              djnz    outl
00B9' C1              pop     bc
00BA' F1              pop     af
00BB' C9              ret

;*****
;*      fetch byte from MEM-clock      *
;*****
00BC' C5      inb:  push    bc
00BD' 06 08              ld      b,8
00BF' DB FA      inl:  in      a,(fpiob)
00C1' CB B7              res     clmem,a
00C3' D3 FA              out     (fpiob),a
00C5' CD 00DA'          call    wait
00C8' DB FA              in      a,(fpiob)
00CA' 07              rlca
00CB' CB 1E              rr      (hl)
00CD' DB FA              in      a,(fpiob)
00CF' CB F7              set     clmem,a
00D1' D3 FA              out     (fpiob),a
00D3' CD 00DA'          call    wait
00D6' 10 E7              djnz    inl
00D8' C1              pop     bc
00D9' C9              ret
00DA' C5      wait:  push    bc
00DB' 06 09              ld      b,9
00DD' 10 FE      waitl: djnz    waitl
00DF' C1              pop     bc
00E0' C9              ret

```

```
00E1'      save:
00E1' 00    date:  defb  0      ; all in packed BCD
00E2' 00    month: defb  0
00E3' 00    year:  defb  0
00E4' 00    day:   defb  0
00E5' 00    $hour: defb  0
00E6' 00    $min:  defb  0
00E7' 00    $sec:  defb  0
              end
```