Christiani Mikroprozessor Labor with Z80-EMUF

Usage Notes

Monitor

Operating

Key	Function	
PC,P1,P2, P3,AC,EX	**Display the contents of registers PC(PC), BC(P1), DE(P2), HL(P3), AF(AC), SP(EX) with decipoints enabled to mark register display. The register display mode is leaved after pressing the same register key twice, or on any other non-register key. After a reset all values are initialized to zero except PC which gets 8100, the user RAM start address. The register store is updated after a breakpoint was hit. If a four-digit value was entered before the key, this value is written to the register buffer. On the next GO, the registers are loaded from the buffers. The index registers IX, IY and the alternate registers can be examined or set with the buffer addresses outlined in the memory map. The register update function for P1, P2 and P3 is also used to enter parameters for the PT functions (see below).**	
AD	**Decrement displayed address and show new address and data.**	
ME	**Increment displayed address and show new address and data.**	
IN	**Store entered data at displayed address and increment address if a two-digit value was entered before, set address if a four-digit value was entered. If no data was entered, it behaves like the ME key.**	
GO	**Start program from the displayed address in realtime until a breakpoint is reached or the program terminates of its own using a RET instruction, which returns to the monitor.**	
BP	**Insert a breakpoint (RST2, D7H) instruction at the displayed address. When the breakpoint is hit, it is replaced with the original instruction, all registers are saved in the buffers and the breakpoint address is displayed. If a breakpoint was not hit and RESET was pressed, the D7H is still in memory and must be replaced with the original instruction manually.**	
SI	**Print an empty line on the printer.**	
PT	**This is a multi functional key. The function executed depends on the 1-key value entered before PT.**	
	1-PT	**Print register dump. Only meaningful after a breakpoint was hit. In this case, a register dump can be printed with a singe PT keystroke without the leading 1**
	2-PT	**Print hexdump from (P1) to (P2, inclusive). If (P1) is zero, 1800 (RAM start) is used. If (P2) is zero or smaller than (P1), a single line (8 bytes) is printed.**
	3-PT	**Print disassembly listing from (P1) to (P2, inclusive). If (P1) is zero, 1800 (RAM start) is used. If (P2) is zero or smaller than (P1), a single instruction is printed.**
		The printing functions have a repeat mode. If no other PT function was executed since the last call, a simple PT keypress continues in hexdump and disassembler modes from the last printed address, one line (8 bytes) in hexdump, one instruction in disassembler. If the printer if not connected or offline, the monitor is reentered after half a second. (P1) and (P2) values remain unchanged. A running printjob can be cancelled using SENSEB.
	4-PT	**Writes memory from (P1) to (P2, inclusive) to tape. A five-second sync tone is output before the data, which must also be recorded, otherwise load will not start. At the end, a one second sync and three seconds of silence are output to allow the TPSS function of the recorder to work. The DB2 output can be used to start/stop the recorder. Strap the LED/Relay board so the DB2 controls one of the relays.**
	5-PT	**Load data from tape to memory, starting at the address stored on tape. The CY/L is lit until the tape input SENSEB goes high and stays high for at least one second, the sync signal must be received for this to happen. This supresses noise which might arise when starting the recorder. After successful load, the end address is displayed. (PC) holds the start address, can be used to run the program (PC - GO). For load errors see the error handler section.**
	6-PT	**Load data from tape to memory, starting at the address stored in (P1). See also 5-PT. If end of RAM is reached, loading stops with error-blink to prevent overwriting system RAM, because 8000 to 87FF is mirrored in 8800 to 8FFF.**

	8-PT	**Load binary data from external RS-232 Device and store at 1800, uses DB7 as data input. Parameters are 1200,N,8,1. F1 LED is lit to show 'waiting for 1st startbit'. While receiving data, F1 LED changes state every four bytes to indicate progress. Because the RAM is mirrored from 8800 to 8FFF a check is done if 8800 is reached, would overwrite system RAM at 8000. If reached, blinkcode 7 is output, must be confirmed with SENSEB. A verify of written data is also done, on error blinkcode 6 is output, also to be confirmed with SENSEB. Receiving stops after about 500mS of silence or on error. If loading ends on error, F1 is lit until the host stops sending. Then the last written address is displayed. To run the downloaded program, PC brings back the start address. A special cable is required which converts and inverts RS232 signal to TTL. DB7 is nomally high. Note: Do not send data while the monitor is in input mode, keypresses will be triggered because DB7 gets the input data. Leads to irregular behaviour or crashes.**
	9-PT	**Same as '8' expept for the received data is stored at (P1), which is checked for inside user RAM. If outside, range error blinkcode is output.**
	A-PT	**Fills memory from (P1) to (P2) with constant in (P3 hibyte). Pay attention not to overwrite monitor- or user stack (8000 to 80FF). P1 is shown when done.**
	B-PT	**Moves memory from (P1) to (P2) to (P3). Handy to debug ROM functions by copying them to RAM. P3 is shown when done.**
0802	**Toggles DB2 output, which is used for tape motor control. Required to be able to use Play to position the tape.**	
0803	**Toggles DB4 output.**	
0..F	**Used to enter addresses and data or the function for PT. If more than four digits are entered in sequence, invalid key blink is output and input must be restarted.**	

Error Handler

The CY/L LED is used to signal errors with blink codes.

Blinks	Meaning
1	**A invalid number of digit keys were entered before the IN key was pressed (four for address, two for data) or an invalid command key was pressed prior to the PT key.This code is also used if more than four digit keys were entered in sequence.**
2	**The range check failed on commands which require a from (P1) and to (P2) address. The to-address smaller than the from-address.**
3	**A unknown function number was entered prior to the PT key.**
4	**Loading from tape caused a checksum error. After pressing the SENSEB key, the load address is displayed.**
5	**Verify error while loading from tape. Most likely there is ROM or no RAM at the load address. The error address is displayed after pressing SENSEB.**
6	**Serial Receive verify error. Probably no RAM or ROM at the address. After pressing the SENSEB key, the error address is displayed.**
7	**Tape Read or Serial Receive reached RAM end. Protects system RAM from being overwritten. After pressing the SENSEB key, 8800 (RAM end +1) is displayed.**

Memory and I/O Map

I/O Ports

From	To	Function
00H		IC3, PIO 0 Port A Data, used by monitor
02H		IC3, PIO 0 Port A Control, used by monitor
01H		IC3, PIO 0 Port B Data, used by monitor
03H		IC3, PIO 0 Port B Control, used by monitor
10H		IC4, PIO 1 Port A Data
12H		IC4, PIO 1 Port A Control
11H		IC4, PIO 1 Port B Data
13H		IC4, PIO 1 Port B Control

Stack

From	To	Function
8000	8040	User Stack
8040	8080	Monitor Stack

RST Vectors

From	To	Monitor jumps here on a RSTn instruction
		All vectors are initialized to JP 0000 on reset. RST1 and RST2 are used by the monitor.
8080	8082	RST1 vector is used by monitor for GO function
		RST2 vector is used by monitor for breakpoint
8083	8085	RST3 vector
8086	8088	RST4 vector
8089	808B	RST5 vector
808C	808E	RST6 vector
808F	8091	RST7 vector
8092	8094	NMI vector

Breakpoint Register Save

From	To	Register
8095	8096	AF
8097	8098	BC
8099	809A	DE
809B	809C	HL
809D	809E	PC
809F	80A0	IX
80A1	80A2	IY
80A3	80A4	SP
80A5		I
80A6		IFF
80A7	80A8	AF'
80A9	80AA	BC'
80AB	80AC	DE'
80AD	80AE	HL'

PIO 1 Interrupt

From	To	Function
80AF	80B0	CH-A Application interrupt handler address
80B1	80B2	CH-B
80B3		CH-A Interrupts counter
80B4		CH-B

Monitor Scratchpad

From	To	Function
80B5		Last entered key
80B6		Digits input counter
80B8	80BC	Keyboard buffer
80BD	80BE	Input buffer backup
80BF	80C4	Display buffer
80C5	80E4	Printer line buffer
80E5		Breakpoint opcode save
80E6	80E7	Breakpoint address
80E8	80E9	Status of AUX outputs
80EA		Wait-for-key-release flag
80EB		Tape save checksum
80EC	80ED	Monitor stackpointer save
80EE		Tape motor manual on/off flag
80EF		Printer last-run flags
80F0		Status of DB2 and DB4 toggles
80F1	80F7	Display pattern buffer for text (shared with disassembler)
80F1	80F2	Disassembler current address
80F3	80F4	Disassembler current position
80F5	80F6	Disassembler current print buffer position
80F7	80F9	Disassembler scratch

Monitor Routines

Display / Keyboard

03CA	dfKYB	Purpose.....: Scan and read keyboard Output......: A: Code of pressed key (if CY=1) CY: 0= no key pressed, 1= a key was pressed Registers...: AF destroyed
043F	dfDSPN	Purpose.....: Update display with numeric values Input.......: HL: Contents of address digits A : Contents of data digits Registers...: All saved
0451	dfDSPADR	Purpose.....: Display data in HL on display in address segments Input.......: HL: address to display (vINLST): If <>0, displays decipoints. This is true only if register contents are currently displayed. Registers...: AF,BC destroyed
046E	dfDSPDAT	Purpose.....: Display data in data digits on display Input.......: A: Data to display Registers...: AF,BC,D destroyed
047D	dfDSPBYT	Purpose.....: Display data in two digits on display Input.......: A: Data to display B: Digit to start with, shifted 2x to left on exit Registers...: AF,BC,D destroyed
04AE	dfDSPA	Purpose.....: Display data from pattern buffer, call in sequence Input.......: DE: Address of pattern buffer, must be 6 chars A: Brightness adjust, smaller=brighter. If called without delay, a value of 30d is sufficient. Registers...: AF destroyed
04D7	dfDSPSEL	Purpose.....: Clear all segments, select digit to display next data Input.......: A: digit to select, 04= data1, 08= data2 10= adr1, 20= adr2, 40= adr3, 80=adr4 Registers...: AF destroyed
04EA	dfDSPCVD	Purpose.....: Convert digit to display pattern Input.......: A: digit to convert Output......: A: display pattern Registers...: AF,C destroyed
050C	dfDSPCVB	Purpose.....: Convert ASCII buffer to display pattern Input.......: HL: Address of ASCII buffer DE: Address of pattern buffer Output......: (DE)..(DE+5): display pattern Registers...: AF destroyed
051E	dfDSPCVA	Purpose.....: Convert ASCII character to display pattern Input.......: A: character to convert Output......: A: display pattern, FF (=space) if not displayable Registers...: AF destroyed

Printing

1103	dfPTXT	Purpose.....: Print ASCII text, uppercase only Input.......: HL: from address DE: end address (including) Output......: CY: 1= normally terminated 0= address range error Registers...: AF destroyed
1193	dfPBCLR	Purpose.....: Clear print buffer Registers...: All saved
11A2	dfPREADY	Purpose.....: Wait until printer is ready or timeout occurs Output......: Z: 1= Timeout occurred 0= Printer is ready C: Bit7 set if SENSEB was pressed, NOT reset otherwise Registers...: AF destroyed

11B9	dfPLINE	Purpose.....: Print one text line Output......: CY: 1= SENSEB was pressed while printing or printer timeout Registers...: AF destroyed
11F6	dfPMOFF	Purpose.....: Turn printer motor off Registers...: AF destroyed

Tape I/O

1200	dfTR	Purpose.....: Read memory block from tape Input.......: HL: From address CY: 0= use load address stored on tape (HL ignored) 1= use load address in HL Registers...: None saved
12E0	dfTW	Purpose.....: Write memory area to tape Input.......: HL: From address DE: To address Output......: CY: 1= normally terminated 0= range error Registers...: None saved
1382	dfTMOT	Purpose.....: Turn tape recorder motor on or off The signal is output on DB2, the LED/Relay board must be connected and jumpered so that DB2 is connected to a relay. When turning on, there will be a 500mS delay to give the motor time to fully spin up. Input.......: CY: 1= turn it on 0= turn it off Registers...: All saved

Utilities

13A1	dfSR	Purpose.....: Receive serial data on DB7 input using bitbanging: - Wait for startbit (DB7 goes low), set FLAG1 LED on. - Start timeout counter, once a startbit is received, after half a second of silence the program exits and displays the last received address. - Receive a byte, store and verify, on error exit with CY/L blinks and wait until SENSEB is pressed, then display error address. Also check for end of user RAM reached, RAM is mirrored from 8800 to 8FFF. - Toggle FLAG1 LED every four bytes to show progress. - Repeat until timeout or error Input.......: HL: Store address Registers...: None saved
144A	dfMFILL	Purpose.....: Fill memory area with constant Input.......: HL: From address DE: To address A: Constant to write Registers...: All saved
146C	dfMMOVE	Purpose.....: Move memory area Input.......: HL: From address DE: To address BC: Destination address Registers...: All saved

I/O

1492	dfLEDWRN	Purpose.....: Pulse CY/L LED n times for 150mS to signal an error Input.......: A: Number of pulses per sequence HL: Pause between sequences, only if CY=1 CY: 1= Repeat sequences 0= only one Registers...: All saved

14C8	dfLEDCY	Purpose.....: Turn CY/L LED on or off Input.......: CY: 1= turn it on 0= turn it off Registers...: All saved
14DC	dfOUTAUX	Purpose.....: Set/clear auxilary outputs Input.......: Bit combination of dbLCYL, dbLOV, dbLIE, dbDB2, dbDB4 H: bits to set L: bits to clear Registers...: All saved
14FD	dfSENSEB	Purpose.....: Read status of SENSEB key Output......: Z: 0= SENSEB is active 1= is is not Registers...: AF destroyed
1502	dfOUTCLK	Purpose.....: Clock the LS174 and LS175 output flipflops Input.......: A=1C: display digits (IC6) A=1D: display segments (IC7 & 2) A=1E: printer (IC5 & 1) A=1F: LEDs & DB2/4 (IC3) Registers...: All saved
150D	dfPIOINP	Purpose.....: Set PIO port A as input, enable 8834 bus drive Registers...: All saved
151C	dfPIOOUT	Purpose.....: Disable 8834 bus drive, set PIO port as output Registers...: All saved
152A	dfCKRAM	Purpose.....: Check if address is in user RAM Input.......: HL: Address to check Output......: CY: 1= HL is in user RAM 0= it is somewhere else Registers...: All saved
1540	dfCKRNG	Purpose.....: Check if address range is valid (end >= start) Input.......: HL: start address DE: end address Output......: CY: 1= range valid 0= range invalid (end <= start) Registers...: AF destroyed

Converting

155D	dfCVB4HS	Purpose.....: Convert 16 bit value to HEX-ASCII, append 2 spaces Input.......: HL: Binary value DE: Where to store result Output......: DE: DE+6 Registers...: AF,DE destroyed Labels......: C)onV)ert B)inary to 4) H)ex digits append two S)paces
1567	dfCVB4H	Purpose.....: Convert 16 bit binary value to HEX-ASCII Input.......: HL: Binary value DE: Where to store HEX digits Output......: DE: DE+4 Registers...: AF,DE destroyed Labels......: C)onV)ert B)inary to 4) H)ex digits
1570	dfCVB2HS	Purpose.....: Convert 8 bit value to HEX-ASCII, append 2 spaces Input.......: A: Binary value DE: Where to store result Output......: DE: DE+4 Registers...: AF,DE destroyed Labels......: C)onV)ert B)inary to 2) H)ex digits append two S)paces
157A	dfCVB2H	Purpose.....: Convert 8 bit value to two HEX digits Input.......: A: Binary value DE: Where to store HEX digits Output......: DE: DE+2 Registers...: AF,DE destroyed Labels......: C)onV)ert B)inary to 2) H)ex digits

1591	dfCVB1H	Purpose.....: Convert 4 bit value to HEX ASCII Input.....: A: value to convert (low nibble) Output.....: A: ASCII equivalent ('0'..'F') Registers...: AF destroyed Labels.....: C)onV)ert B)inary to 1) H)ex digit
159E	dfDLYMS	Purpose.....: Delay in mS. Time constants are evaluated using an osci, with CPU clock of 2MHz Input.....: HL: Number of milliseconds to delay CY: 0= Ignore SENSEB 1= Abort if active Output.....: CY: 1= SENSEB was pressed, delay interrupted 0= no SENSEA Registers...: F destroyed

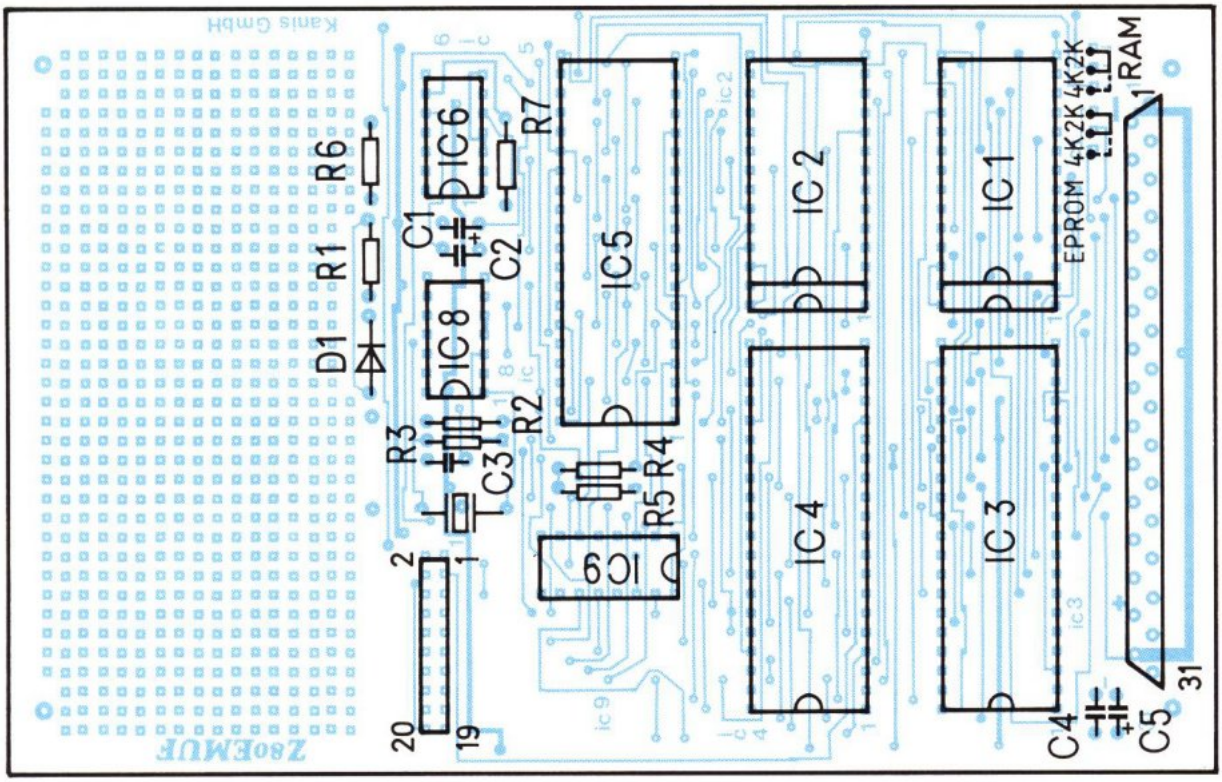
PIO 1 Support

15C6	dfP1S	Purpose.....: Initialize one PIO1 channel, call with interrupts disabled Input.....: D: channel, 0=A, 1=B, or one of dbPx E: operation mode, one of dbPMx B: interrupt control, one of dbPIxx A: input/output mask, 1=input, 0=output (mode 3 only) C: interrupt mask (mode 3 only) HL: address of interrupt handler CY: 1= setup PIO 0= disable channel interrupts, set mode 1 (all inputs) Output.....: Interrupts are disabled Registers...: AF destroyed Labels.....: P1Sxxxx: P)IO 1) S)etup
1625	dfP1AR	Purpose.....: Read PIO1 port A Input.....: A: Bitmask to compare Output.....: A: Port status, not masked with input mask Z: 1= Input status matches input bitmask 0= It does not Registers...: AF destroyed Labels.....: P)IO 1) port A) R)ead
162C	dfP1AW	Purpose.....: Write PIO1 port A Input.....: A: Bits to set B: Bits to clear Output.....: A: Port status Registers...: AF destroyed Labels.....: P)IO 1) port A) W)rite
1639	dfP1BR	Purpose.....: Read PIO1 port B Input.....: A: Bitmask to compare Output.....: A: Port status, not masked with input mask Z: 1= Input status matches input bitmask 0= It does not Registers...: AF destroyed Labels.....: P)IO 1) port B) R)ead
1640	dfP1BW	Purpose.....: Write PIO1 port B Input.....: A: Bits to set B: Bits to clear Output.....: A: Port status Registers...: AF destroyed Labels.....: P)IO 1) port B) W)rite
164D	dfP1IS	Purpose.....: Set CPU interrupt mode, does not enable interrupts Input.....: A: interrupt mode to set (0,1 or 2) Registers...: AF, I destroyed Labels.....: P)IO 1) I)nterrupt S)etup
1664	dfP1RETI	Purpose.....: The PIO can be reset only by power-on. If it was stopped by breakpoint while in interrupt and RESET was hit, the PIO does not process any further interrupts until it encounters a RETI instruction. So we do a RETI here to avoid this problem. Registers...: All saved Labels.....: P)IO 1) R)E)T)I)

Hardware

EMUF

Layout



Schematic

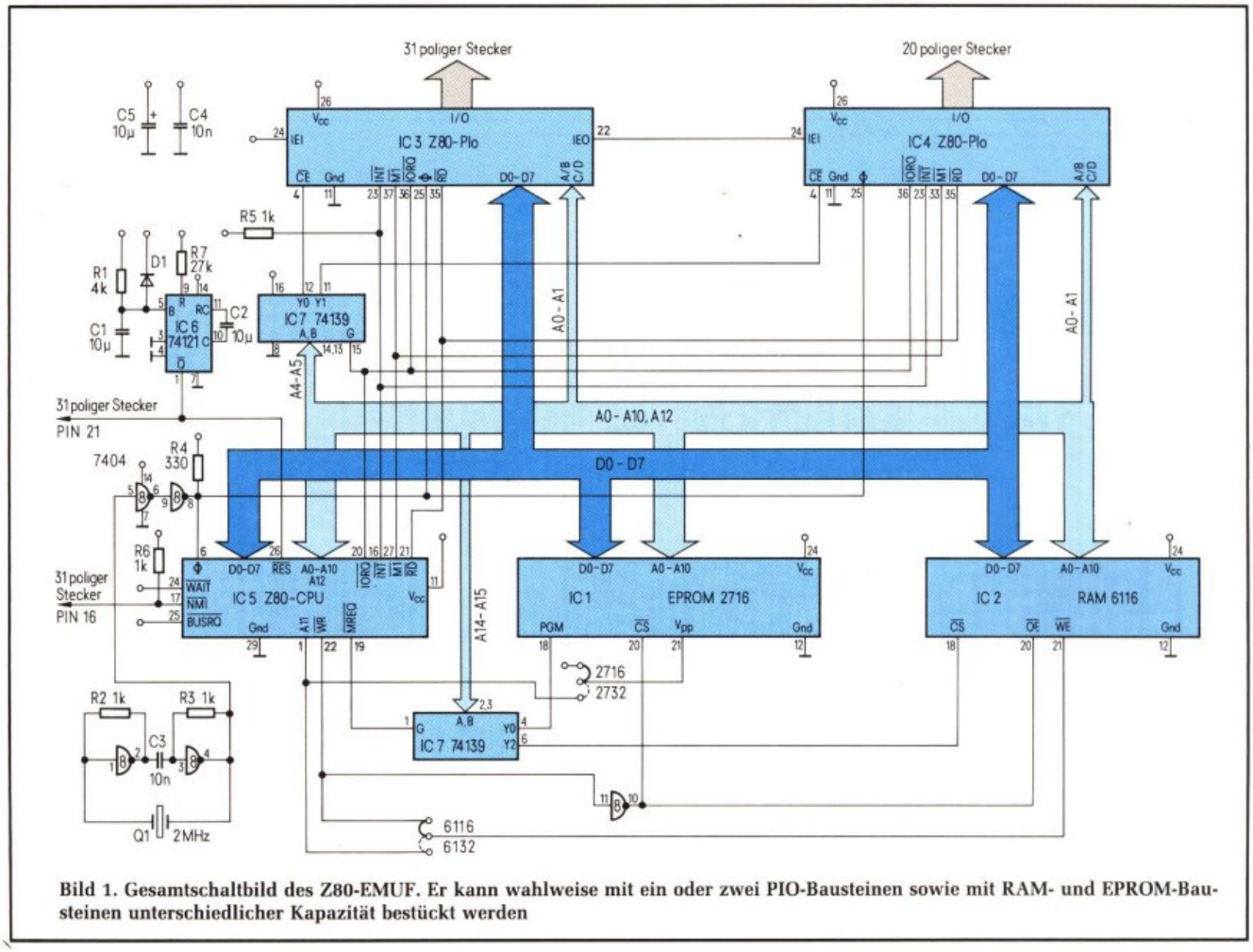


Bild 1. Gesamtschaltbild des Z80-EMUF. Er kann wahlweise mit ein oder zwei PIO-Bausteinen sowie mit RAM- und EPROM-Bausteinen unterschiedlicher Kapazität bestückt werden

Connectors

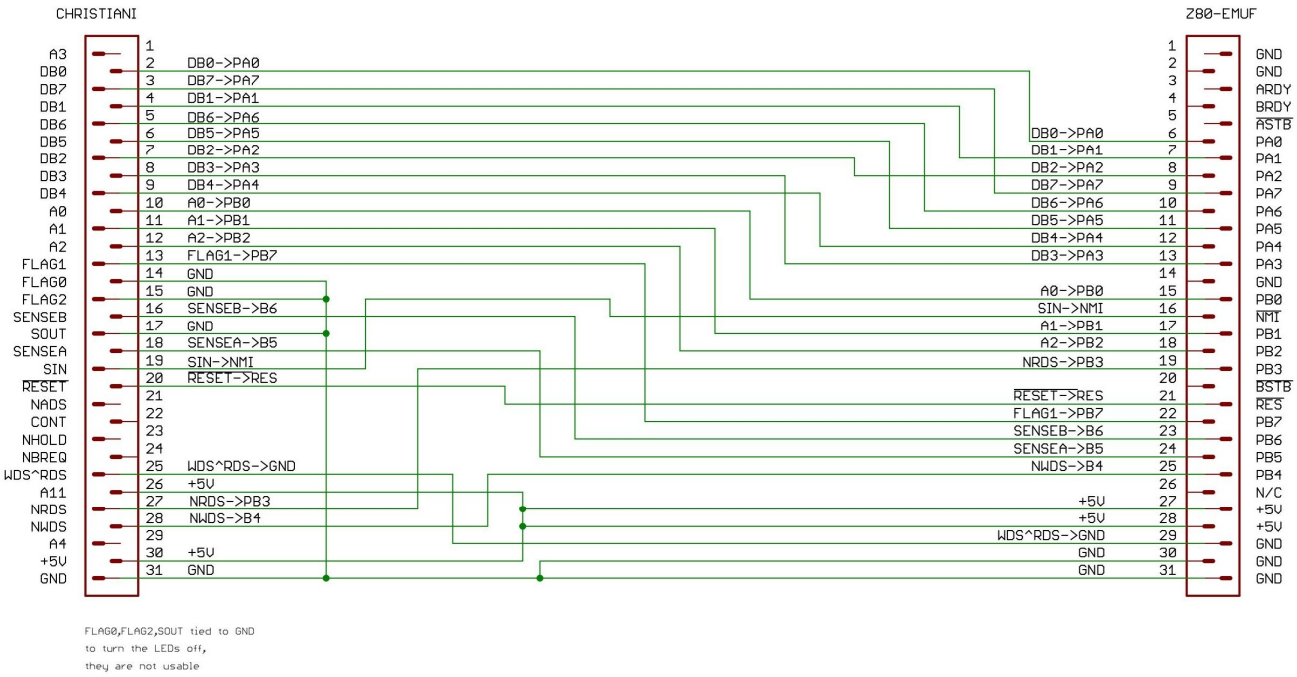
Tabelle 2: Belegung des 31poligen Steckers (PIO – IC3)

1 – Masse	16 – $\overline{\text{NMI}}$
2 – Masse	17 – B1
3 – ARDY	18 – B2
4 – BRDY	19 – B3
5 – $\overline{\text{ASTB}}$	20 – $\overline{\text{BSTB}}$
6 – A0	21 – $\overline{\text{RES}}$
7 – A1	22 – B7
8 – A2	23 – B6
9 – A7	24 – B5
10 – A6	25 – B4
11 – A5	26 – NC
12 – A4	27 – +5 V
13 – A3	28 – +5 V
14 – Masse	29 – Masse
15 – B0	30 – Masse
	31 – Masse

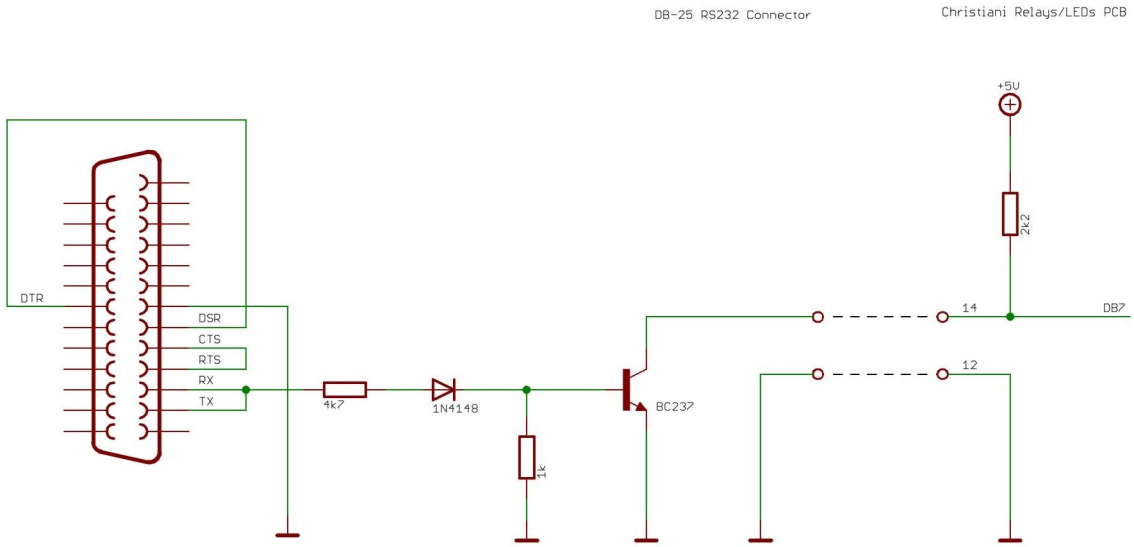
Tabelle 3: Belegung des 20poligen Steckers (PIO – IC4)

1 – ARDY	11 – A3
2 – BRDY	12 – B3
3 – $\overline{\text{ASTB}}$	13 – A4
4 – $\overline{\text{BSTB}}$	14 – B4
5 – A0	15 – A5
6 – B0	16 – B5
7 – A1	17 – A6
8 – B1	18 – B6
9 – A2	19 – A7
10 – B2	20 – B7

Christiani to EMUF Adapter

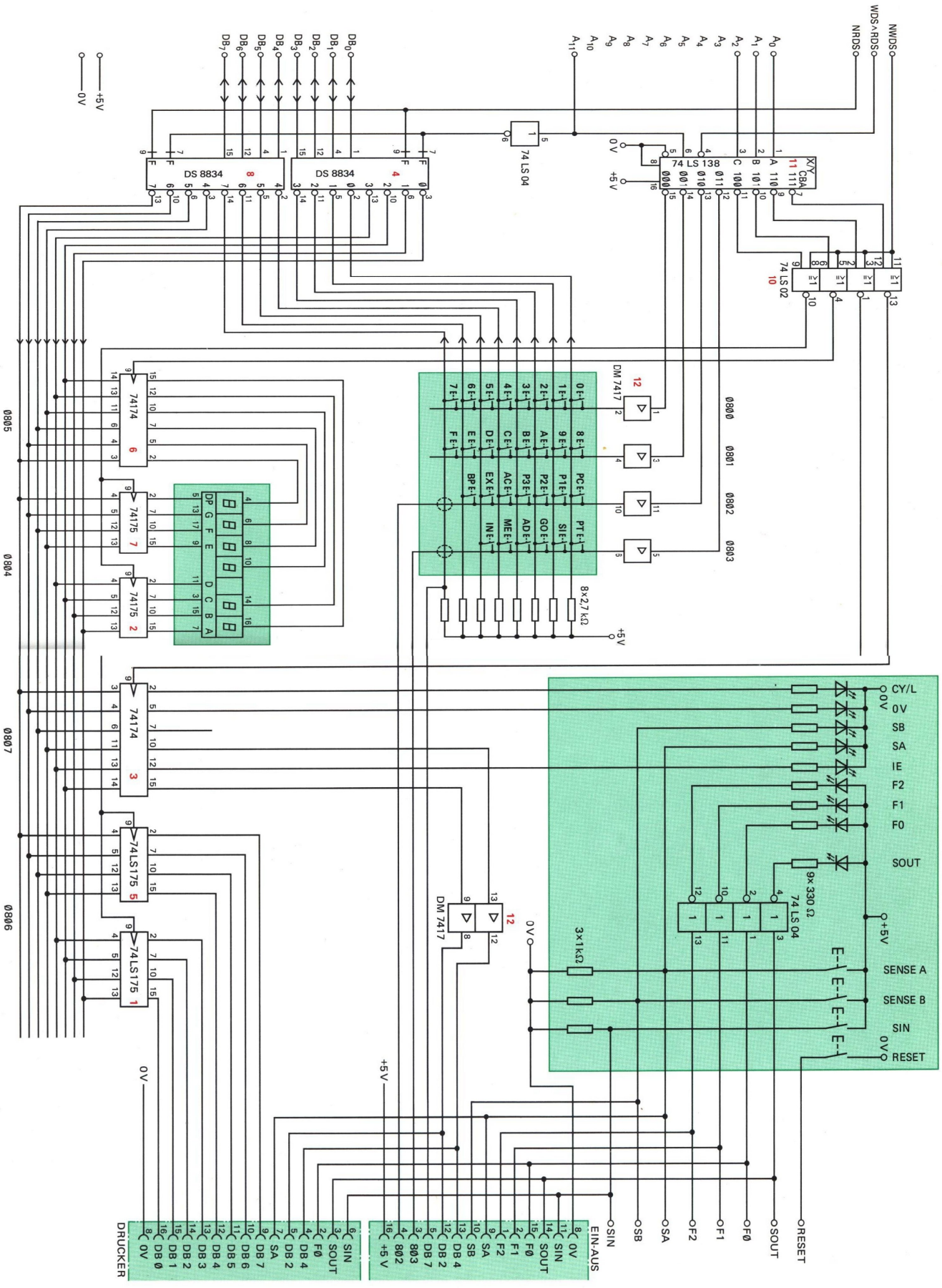


RS-232 to DB7 Cable

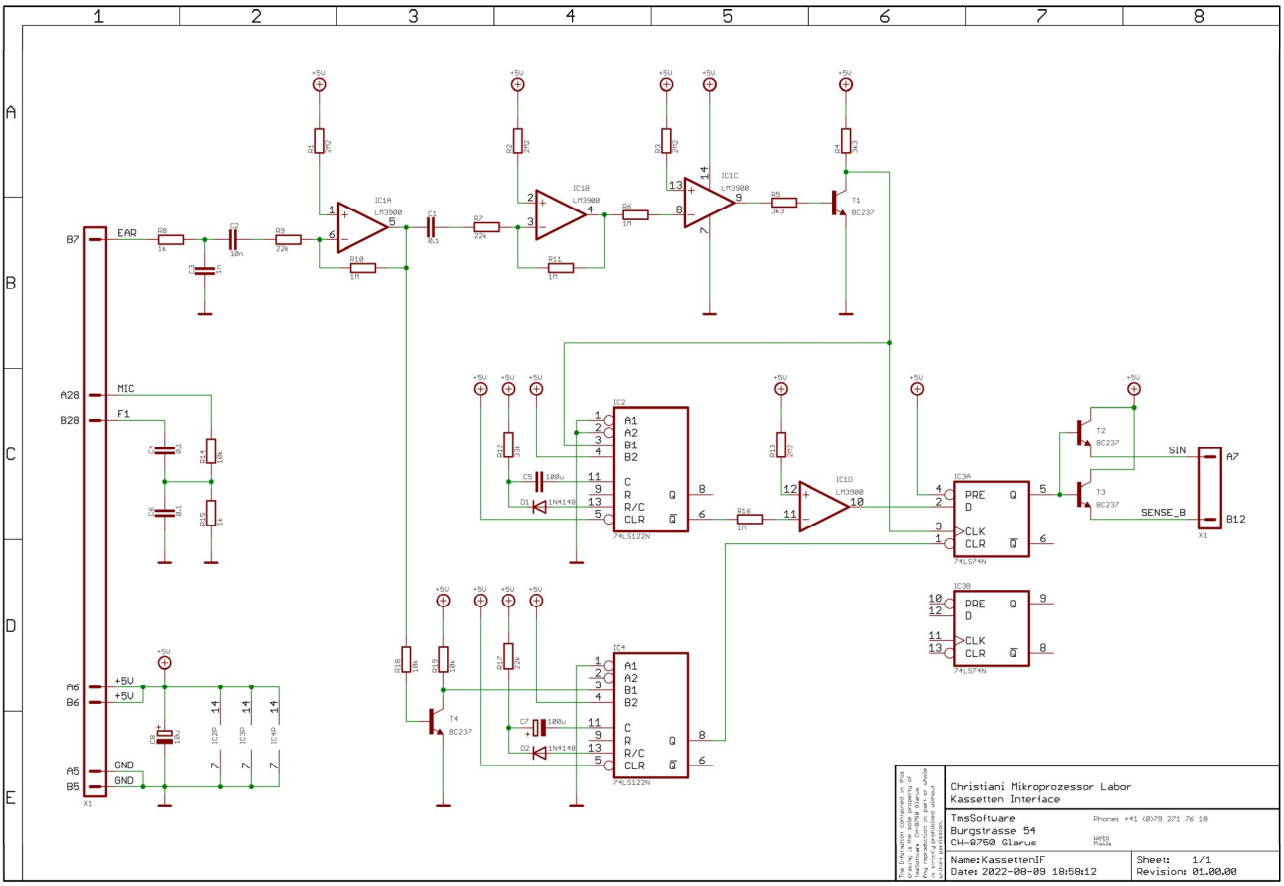


MP-Labor

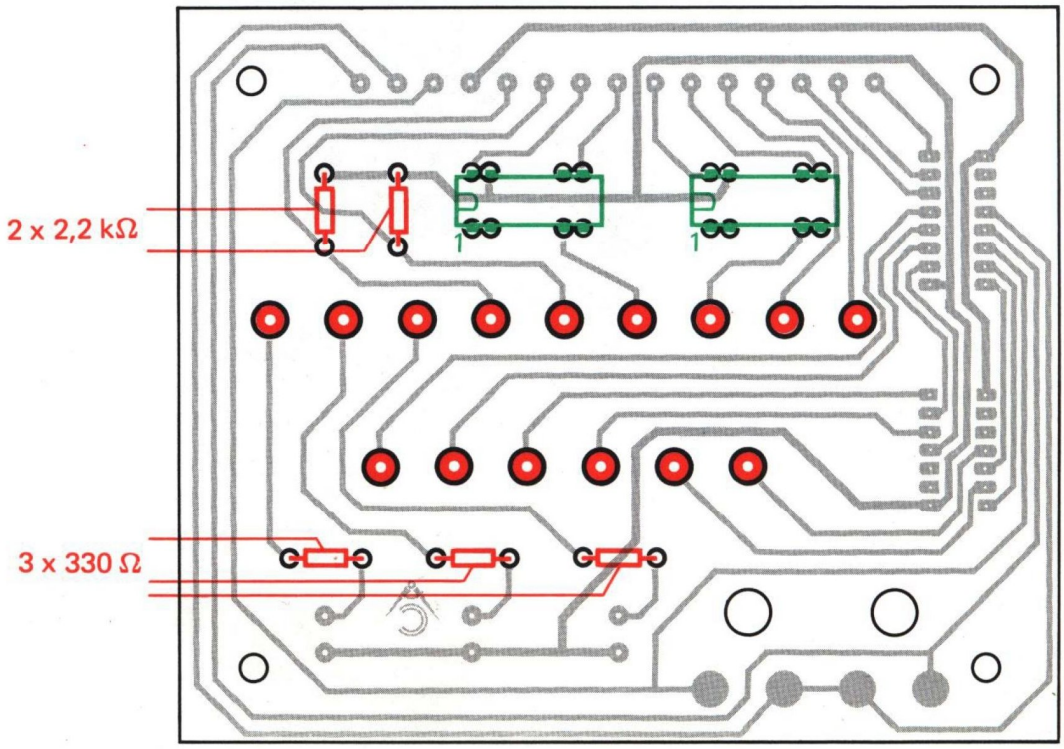
Base Unit Schematic



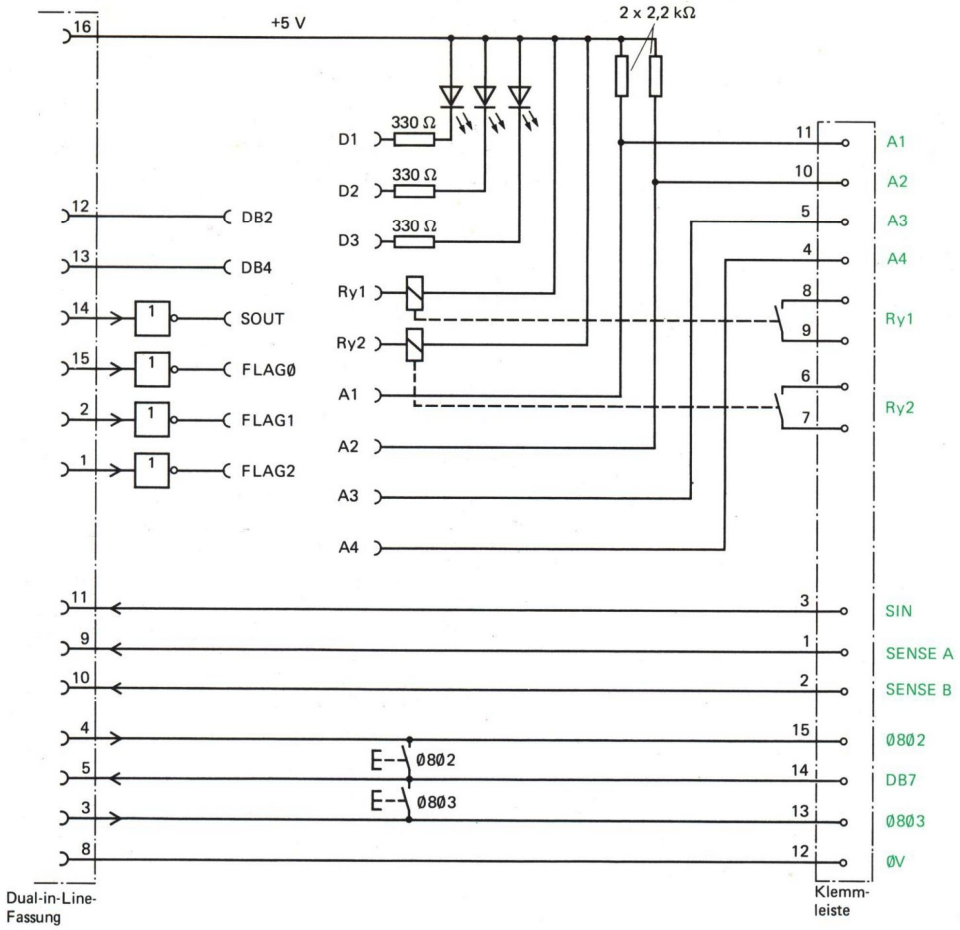
Cassette Interface Schematic



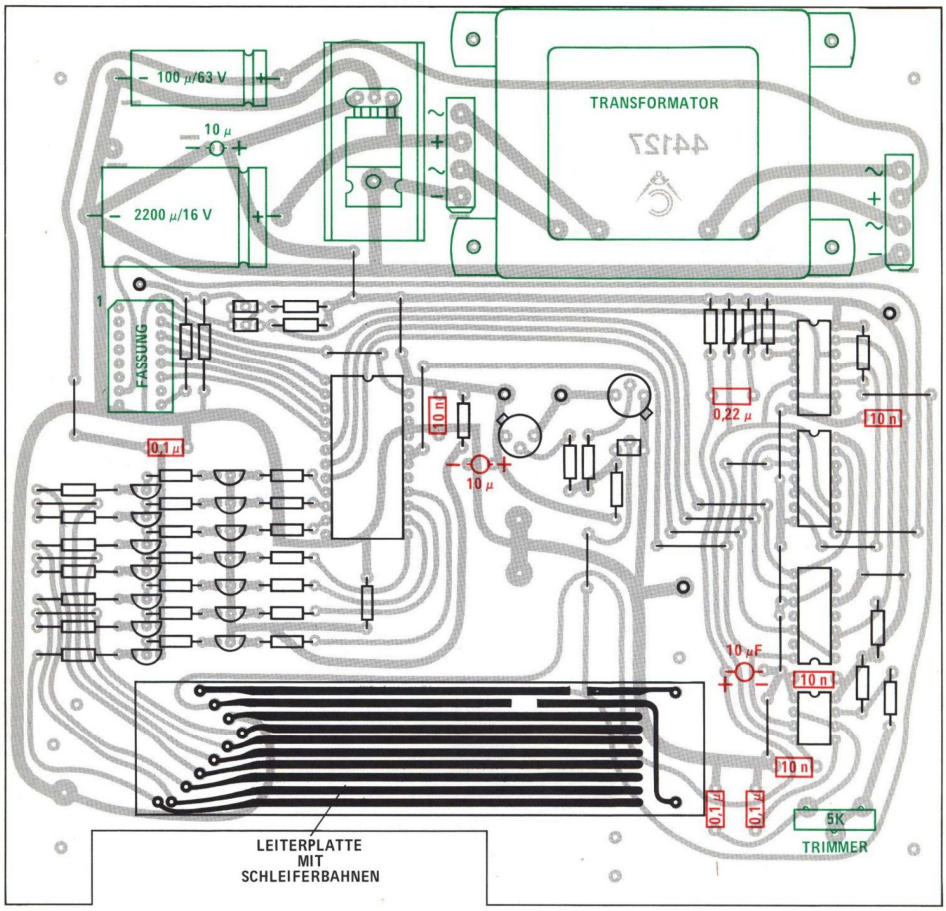
Relays/Leds Layout



Relays/LEDs Schematic



Printer Layout



Printer Schematic