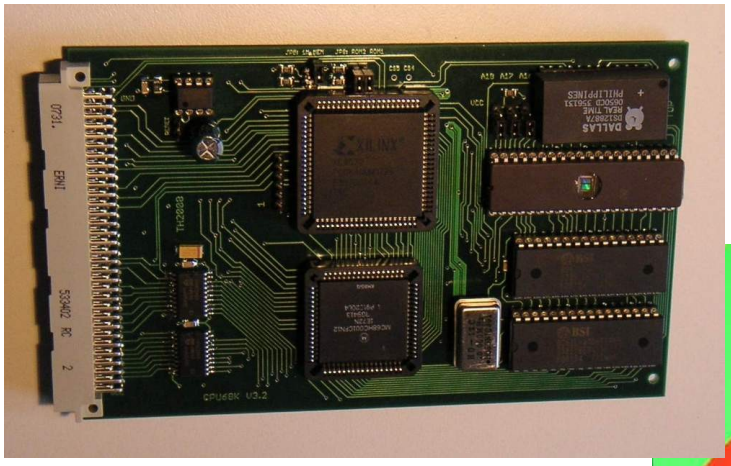
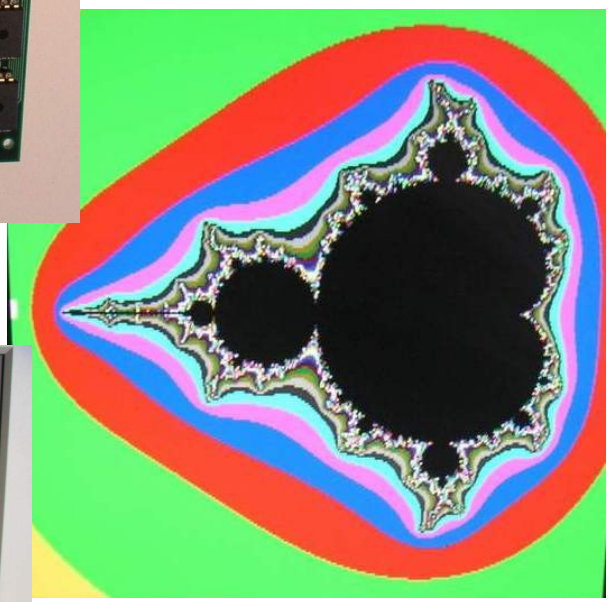
CPU68K

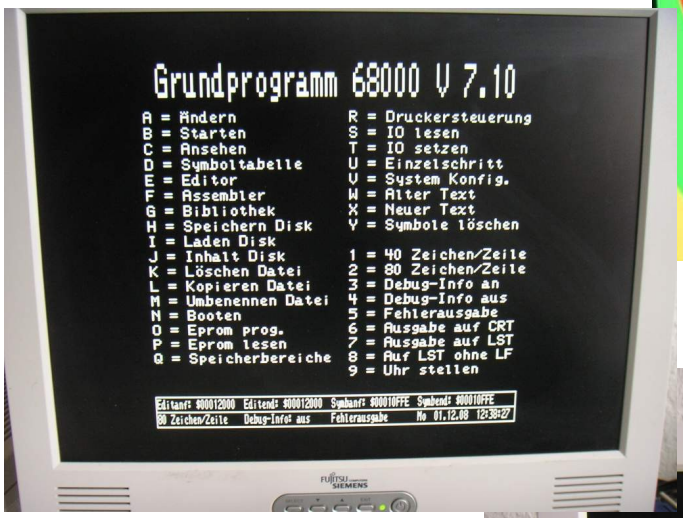
68HC00x Baugruppe u.a. für den NDR-Klein-Computer

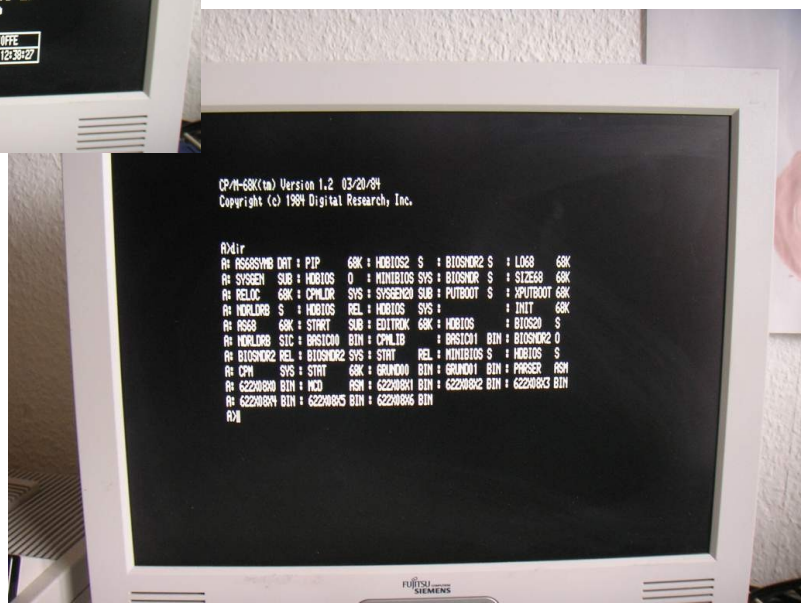
Stand: März 2010

Autor: Torsten Hemmecke









Wichtiger Hinweis:

Die in dieser Anleitung wiedergegebenen Schaltungen und Verfahren werden ohne Rücksicht auf die Patentlage oder Lizenzrechte Dritter mitgeteilt. Sie sind ausschließlich für private Zwecke und Lehrzwecke bestimmt und dürfen nicht gewerblich genutzt werden. *)
Alle Schaltungen und technische Angaben in dieser Anleitung wurden vom Autor sorgfältig erarbeitet bzw. zusammengestellt. Trotzdem sind Fehler nicht aus zuschließen. Daher kann der Autor weder eine Garantie noch die juristische Verantwortung oder irgend eine Haftung für Folgen, die auf fehlerhafte Angaben zurückgehen, übernehmen. Für die Mitteilung eventueller Fehler ist der Autor jederzeit dankbar.
Die Rechte an Firmennamen, Logos und Warenzeichen, die in dieser Anleitung genannt werden, liegen bei den jeweiligen Inhabern.

*) Bei gewerblicher Nutzung ist vorher die Genehmigung des möglichen Lizenz- oder Rechteinhabers einzuholen.

Inhalt

1 Vorwort

Die Baugruppe CPU68k-XC16BIT entstand im Rahmen des Projektes „NKC Redesign“ das zum Ziel hat, den legendären NDR-Klein-Computer mit heute (Stand 2007) erhältlichen Bauelementen neu aufzubauen und bis hin zum Einsatz mit einem kleinen Linux-Kernel. Das ganze System verwendet einen ECB Bus, wobei die Belegung des Busses zum MC-Computer kompatibel ist und darüber hinaus 16bit Erweiterungen vorsieht.
Die CPU68k soll volle Software-Kompatibilität zum original NKC bieten.
Auf der Karte sind bis zu 2MB Speicher, ein konfigurierbares 16bit-ROM, eine Echtzeit-UHR (kompatibel zur Uhren-Baugruppe UHR3 von Jens Mewes) sowie die komplette Banken-Logik integriert. Es stehen wahlweise ein 8/16-bit ECB-Bus-Interface oder ein 8-bit NKC-BUS Interface zur Verfügung.

Zusammen mit der GDP-FPGA von Andreas Voggeneder steht damit ein vollständiges NKC System zur Verfügung, das bereits JADOS über eine SD-CARD bootet.

2 Änderungshistorie

3 Kurzbeschreibung der Funktion

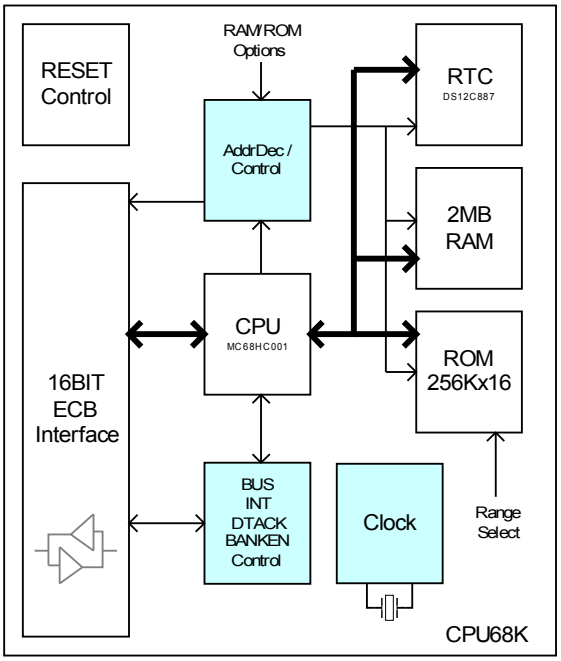
Die Baugruppe CPU68k stellt eine vollständige 16Bit CPU für den NKC dar. Sie enthält zusätzlich zur originalen CPU68000 bereits bis zu 2MB RAM und mehrere durch Jumper wählbare ROM-Bereiche, die komplette Banken-Logik sowie die UHR3 Baugruppe und ist voll softwarekompatibel zu diesen Baugruppen. Somit kann sämtliche Software des NKC darauf betrieben werden.
Die Hardware ist auf einer Europakarte aufgebaut und kann wahlweise mit einen 8/16-Bit ECB-Bus-Anschluß oder am 8Bit NKC-Bus betrieben werden.
Beim Betrieb am ECB Bus besteht die Möglichkeit den Speicherbereich durch Einsatz einer RAM-Karte [] zu erweitern, was beim Experimentieren mit uCLinux unbedingt zu empfehlen ist. Speicher oberhalb 2MB wird ausserdem prinzipiell von CPM68k unterstützt.

4 Technische Daten

Europakarte 160 x 100 mm 4 Lagen
CPU 68HC000
RTC 12C887
ROM/FLASH bis 512K
Bis zu 2MB RAM on Board
ECB-Bus (8/16-bit ab v3.1) optional NKC-Bus (8-bit ab v3.2)
Stromaufnahme ca. … mA
Gerätetyp; Betriebsspannung 5V
Verschiedene Jumper bieten weitere Anpassungsmöglichkeiten. Siehe Abschnitt „Beschreibung der Jumper“.

5 Prinzipbeschreibung

5.1 Blockschaltbild



5.2 Beschreibung des Blockschaltbildes

Die Baugruppe stellt einen kompletten Single Board Computer (SBC) mit 16Bit MC68000 CPU dar. Zusammen mit der GDP-FPGA entsteht so ein vollständiger NKC mit 68000 CPU.

Es ist bereits eine RTC (Real Time Clock) mit einem DS12(C)887 von Dallas integriert, der im CPLD als UHR3 Baugruppe eingeblendet wird. Eine Funktionsbeschreibung findet sich in der Dokumentation von UHR3 [].

Es können bis zu 2MB RAM (NKC Vollausbau) und 256K ROM eingesetzt werden. Das Speicher-Layout kann über Jumper an verschiedenen Ausbaustufen bzw. Anwendungsfälle angepasst werden: So ist es möglich 100% kompatibel zum NKC zu sein (2MBRAM/64KROM) oder bis zu 256K ROM als Monitor zu verwenden, z.B. als Image Speicher für ein komplettes Betriebssystem. Auch können verschiedene Bereiche aus dem ROM in den 64K GP Bereich eingeblendet werden. Man kann damit verschiedenen GP Versionen gleichzeitig im EPROM betreiben, ohne das EPROM wechseln zu müssen.

Die farblich hervorgehobenen Blöcke sind in einem CPLD realisiert.

Die Bus Steuerung sowie DTACK/WAIT Generierung wird im CPLD erledigt. Es können die IRQx, NMI und INT Leitungen des ECB Busses im CPLD beliebig auf die IPL Leitungen der CPU geschaltet werden.

Darüber hinaus kümmert sich das CPLD um die notwendigen Adress-Dekodierungen von IO und MEM Zugriffen (lokal/nicht lokal etc.) und realisiert das BANKEN Register zum Ausblenden des ROM aus dem Adressraum des 68000.

Als Reset Baustein wird ein TLC7725 eingesetzt. Der Baustein sorgt bei Einschalten für einen genügend langen RESET Puls, so dass sich die Versorgungsspannung und die Signale stabilisieren können. Außerdem wird die Versorgungsspannung überwacht und bei Unterschreitung ein Reset ausgelöst um einen undefinierten Zustand des Systems zu vermeiden. Außerdem kann ein externer Reset-Taster angeschlossen werden.

Das 16Bit ECB Bus Interface stellt die Verbindung zu weiteren Karten über den ECB Bus her, z.B. zur GDP-FPGA oder SRAM_24MB.

5.3 Erweiterung des ECB-BUS auf 16/32 Bit

Damit der 68000 und auch der 68020 mit voller Bus-Breite auf Peripherie zugreifen können, wurde der Bus auf 16- bzw. 32-Bit erweitert. Er bleibt dabei Kompatibel zum Z80-Bus (siehe Baugruppe SBC4).

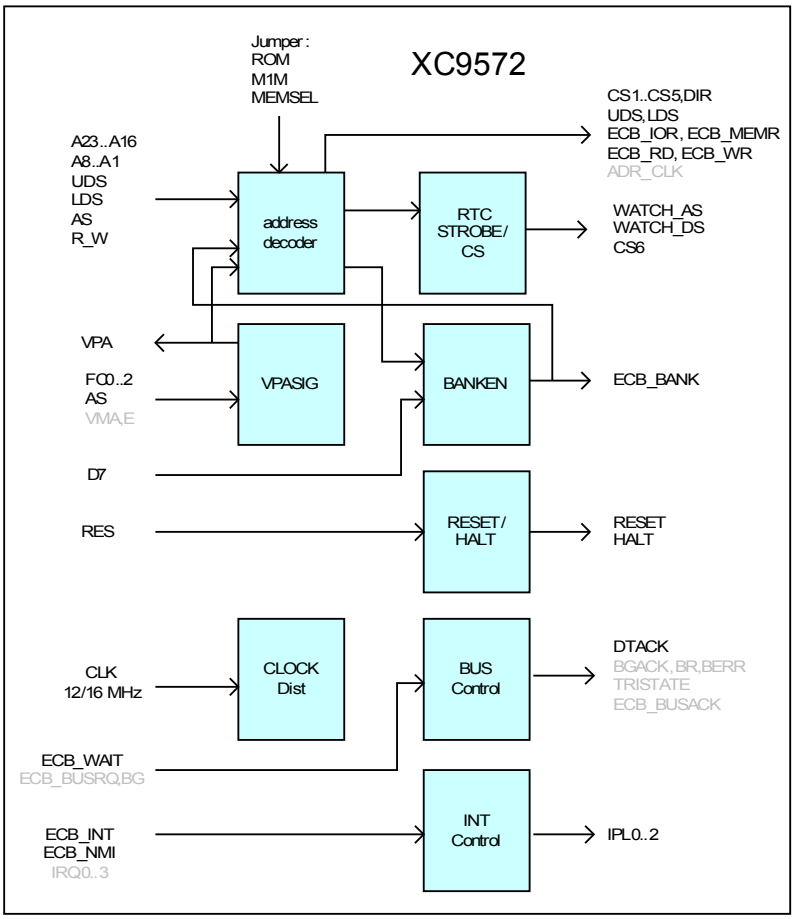
Reihe	A	B	C
1	5V	5V	5V
2	D5	A20	D0
3	D6	A21	D7
4	D3	A22	D2
5	D4	A23	A0
6	A2	D8	A3
7	A4	D9	A1
8	A5	D10	A8
9	A6	D11	A7
10	WAIT\	D12	A24
11	BUSRQ\	D13	A25
12	A18	D14	A19
13		D15	BANKEN
14	A26	D24	D1
15	(020-A0)	D25	A27
16	(020-A1)	D26	A28
17	A17	IRQ4	A11
18	A14	IRQ3	A10
19	A29	IRQ2	A16
20	M1\	IRQ1	NMI\
21	INT\	IRQ0	
22	D27	D16	WR\
23	D28	D17	
24	D29	D18	RD\
25	D30	D19	HALT/
26	D31	D20	
27	IORQ\	D21	A12
28		DS0/UDS/SIZ0	A15
29	A13	DS1/LDS/SIZ1	CLK
30	A9	D22	MREQ\
31	BUSAK\	D23	RESET\
32	GND	GND	GND

GRÜN = NKC-ECB-BUS Belegung
ORANGE = 16Bit Erweiterung (68000)
VIOLETT = 32Bit Erweiterung (68020 – asynchrones Bus Design)
HBLAU = evtl. für synchrones Bus-Design (ab 68040) vorgesehen

6 Das CPLD

Im folgenden wird im Wesentlichen das CPLD in der NKC Version beschrieben. Prinzipiell kann das Verhalten der CPU-Baugruppe auch an andere Anwendungsfälle angepasst werden, z.B. kann das Interrupthandling verfeinert und die BUS Arbitration für die Verwendung mehrerer CPUs angepasst werden. Auch die dekodierten Adressbereiche sind hier rekonfigurierbar.

6.1 Blockschaltbild des CPLD



Signale, die im Standard NKC Design nicht verwendet werden sind ausgegraut.

6.2 Funktionsbeschreibung der CPLDs

Das VHDL File definiert zunächst die Signale, die der Baustein nach Außen hin zur Verfügung stellt. Danach werden einige Konstanten definiert, die eine Anpassung/Änderung von Speicherbereichen und Register-Adressen erleichtern und das ganze Listing lesbarer machen.

Danach folgen einige Prozesse und ein wenig kombinatorische Logik:

Process STROBE:
Dient dem Erzeugen der für den Uhrenbaustein notwendigen Strobe Signale für Read und Write Zugriffe.

Process RESETPROC:
Dieser Prozess realisiert ein Tristate Gate, das beim Anliegen eines Reset Signals die Signale HALT und RESET des Prozessors auf NULL zieht. Die Verwendung als Tristate ist notwendig, da der Prozessor selbst diese Signale über Kommandos setzen kann.

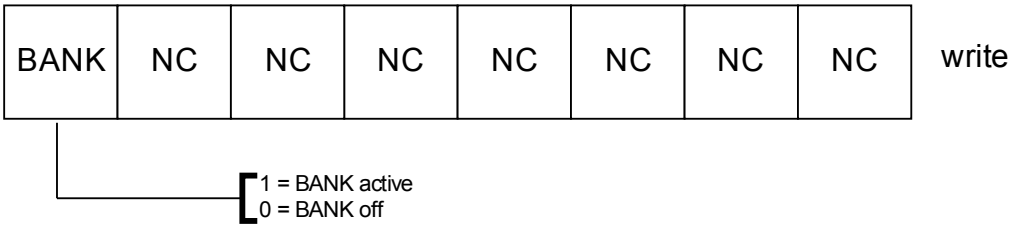
Process BANKPROC
Realisiert das BANKEN Register.

Process VPASIG:
Realisiert die Interrupt-Auto-Vector Steuerung, wie im NKC realisiert. Siehe dazu auch die Dokumentation im MC68000 User Guide [] und hier [].

Die Adress-Dekodierung wird über mehrere interne Signale generiert. Diese Art der Dekodierung gegenüber einer Switch-Anweisung oder If-then-else Verzweigung ist weniger fehleranfällig und besser lesbar.
Zunächst wird festgestellt, ob ein gültiger Speicherzugriff der CPU vorliegt und das Signal *is_68k_mem_cycle* gesetzt. Die Signale *is_io_cs* und *is_mem_cs* unterscheiden weiter zwischen IO und Speicher-Zugriff (dieses Konzept ist aus historischen Gründen für den NKC so realisiert, obgleich der 68000 keinen Unterschied zwischen IO und MEM Zugriff kennt).
Falls ein IO Zugriff vorliegt wird überprüft, ob es ein lokaler Zugriff auf das BANKEN-Register oder die RTC ist (*rtc_data_cs, rtc_addr_cs, banken_cs*) oder ob es sich um einen Zugriff auf eine externe IO-Karte handelt (*ext_io_cs*).
Bei MEM Zugriffen wird je nach den Jumper-Einstellungen (siehe) und dem Zustand des BANKEN Registers das lokale RAM, ROM oder externer Speicher selektiert (*ram_cs, rom_cs, ext_mem_cs*). Bei lokalen Zugriffen werden die Chip-Select Signale entsprechend erzeugt, bei externen Zugriffen werden die Signale entsprechend auf den ECB BUS geschaltet.

Das BANKEN Register ist so realisiert wie im NKC. Es dient hier, anders als der Name vermuten lässt, nicht dem Banking sondern dem Ein- und Ausblenden des ROM Bausteins aus dem Speicherbereich des 68000. Der Namen BANKEN Register stammt aus der Verwendung als BANK-Umschalteregister im Z80 System.

0xC8 – BANK register



6.3 Programmierung der CPLDs

Zum Programmieren des CPLDs benötigt man das ISE-Webpack von Xilinx, dieses kann man (nach kostenloser Registrierung) von der Webseite www.xilinx.com herunterladen. Weiterhin benötigt man einen Programmieradapter, hierfür kann z.B. die Schaltung von www.holger-klabunde.de verwendet. Der Programmierstecker auf der GIDE Platine hat die Belegung dieses Programmers. Die CPLD-Daten sind in der Datei Xilinx.zip, diese entpackt man am besten nach c:\xilinx und nicht nach Eigene Dateien, da versch. Versionen des ISE-Webpacks nicht mit Leerzeichen in Verzeichnisnamen klar kommen. Zum Programmieren startet man den (Xilinx) Projekt Navigator und öffnet die Datei CPU68K.ise, dort startet man unter Processes Implement Design -> Gererate Programming File – Configure Device (iMPACT), iMPACT das eigentliche Programmier Tool.

Bis zur Version 9.1.x ist ispIMPACT als Stand-Alone-Version vorhanden, das sich auch ohne ISE installieren lässt. Falls man also nur die fertigen JEDEC Files in's CPLD brutzeln möchte, eine echte Alternative.

7 Beschreibung der Jumper

Im folgenden werden die Einstellungsmöglichkeiten der Jumper beschrieben. Grundlage dabei ist das Default NKC-CPLD Design wie in [] beschrieben.

x bedeutet Jumper gesetzt
- bedeutet Jumper offen

J5-1 1MB (x) / 2MB (-) RAM OnBoard
J5-2 Use OnBoard Memory (x)

Wenn J5-2 nicht gesetzt ist, müssen ROM/RAM auf einer externen Speicherkarte z.V. gestellt werden.

J8-1	J8-2	ROM-Grösse
x	x	64k (*)
x	-	128k
-	x	256k
-	-	512k

J8 definiert, wie gross der vom EPROM eingeblendete ROM Bereich ist. Sind also beide Jumper gesetzt, sind die ersten 64K des Speichers ROM (solange Bit 7 des BANKEN Registers 0 ist – ansonsten ist das ROM ausgeblendet und statt desses wird RAM eingeblendet)

(*) bezeichnet die Standard Einstellung für NKC Betrieb

J7	J6	J4	EPROM-Offset	Maximale ROM Grösse
A18	A17	A16	$0.0000	512K (*)
A18	A17	1	$1.0000	64K
A18	1	A16	$2.0000	128K
A18	1	1	$3.0000	64k
1	A17	A16	$4.0000	256k
1	A17	1	$5.0000	64K
1	1	A16	$6.0000	128K
1	1	1	$7.0000	64K

Über die Jumnper J4,6,7 können die Adressleitungen A15..17 des EPROMs entweder mit den Adressleitungen A16…19 der CPU verbunden oder fest auf 1 gelegt werden. Damit können verschiedene Offsets innerhalb des EPROMs in den Speicher eingeblendet werden.

Beispiel:
Es sollen 2 GP Versionen über Jumper ausgewählt werden können. Dazu wird GP1 ab $00000 im EPROM abgelegt und GP2 ab $10000 und die Einstellung auf J8 wird auf 64K festgelegt. Wenn alle Adresssignale an das EPROM geschaltet werden, bootet GP1. Wird J4 auf ‚1' ge-jumpert, bootet dagegen GP2.

(*) bezeichnet die Standard Einstellung für NKC Betrieb

Die Stifte CS4 und CS5 dienen als Chip-Select Signale für das zweite MB Speicher, die „huckepack" aufgelötet werden (siehe Aufbauanleitung).

Ab Hardware-Version 3.2.1 gibt es noch die Stifte CS7 und CS8 mit denen ein weiteres MB angesprochen werden könnte.

8 Aufbauanleitung

8.1 Umgang mit IC´s

CMOS-Bausteine sind hochempfindlich gegen elektrostatische Aufladung! Bewahren oder Transportieren Sie CMOS-Bausteine nur auf leitenden Schaumstoff! Alle Pins müssen kurzgeschlossen sein.
Achten Sie darauf, daß Sie Verbindung mit einer Erdungsmöglichkeit haben, bevor Sie mit diesen Bausteinen arbeiten. Geeignete ESD-Artikel gibt es im Fachhandel.

8.2 Aufbau Schritt für Schritt

Beim Layout der Platine wurde Wert auf eine möglichst hohe Nachbausicherheit gelegt. Dennoch war es notwendig einige SMD Bauteile zu verwenden die allerdings aufgrund ihrer Grösse und des Pinabstandes bei den ICs noch weitgehend unkritisch zu verarbeiten sind. Aufgrund der deutlich höheren Störsicherheit und der Anzahl integrierter Funktionen (UHR, Speicher etc.) ist ein 4-Lagen Design notwendig geworden. Daher kann die Platine i.d.R. nicht vom Hobbyisten gefertigt werden. Aufgrund der verwendeten Bauteile kann dieses Projekt nicht mehr als Anfängerprojekt gewertet werden, ein wenig Löterfahrung sollte vorhanden sein. Der Aufbau ist aber weit weniger kompliziert wie z.B. beim Aufbau der GDP-FPGA (FinePitch).

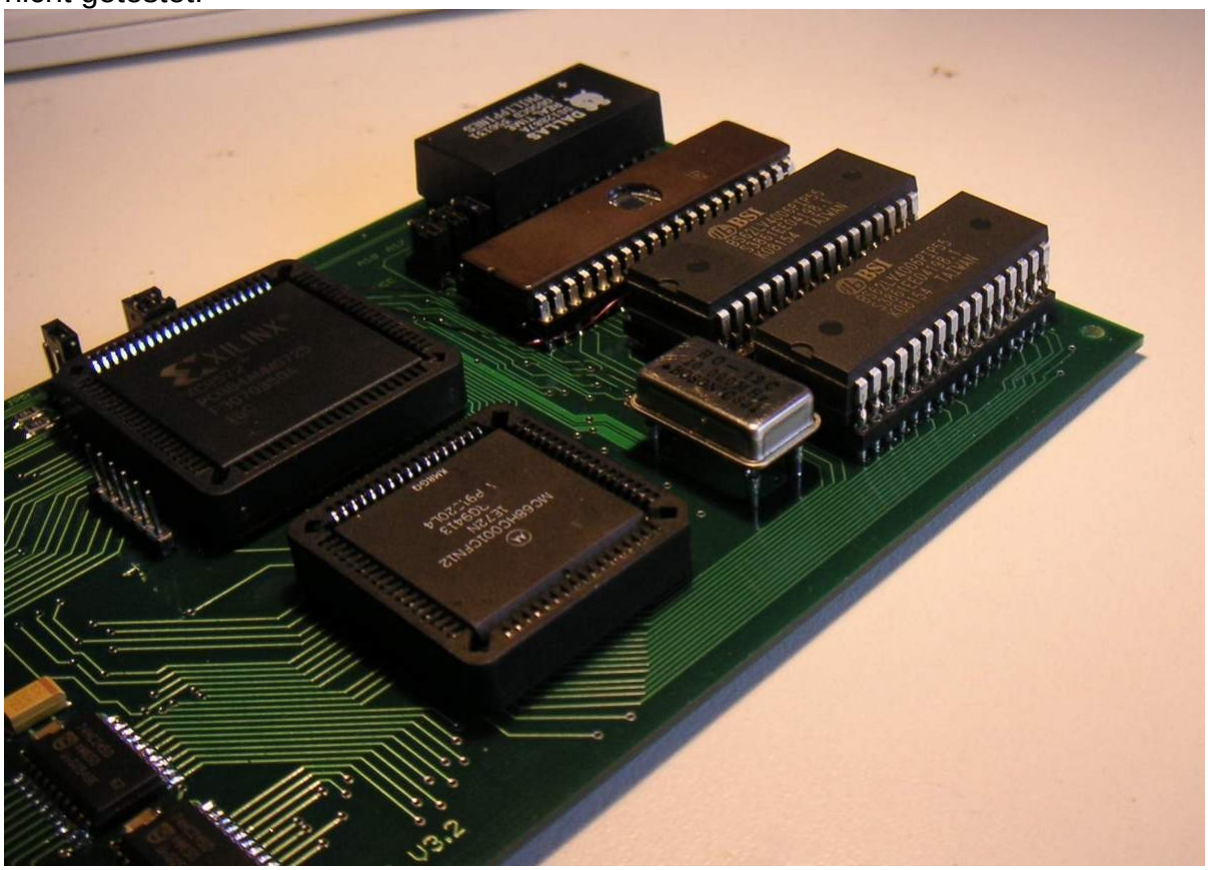
Zunächst sollte angefagen mit den Treiber ICs () alle SMD Bauteile eingelötet werden.
Danach werden alle IC Sockel und der VG Stecker eingelötet.

Sind alle Sockel und fest einzulötenden Bauteile eingebaut, kann ein erster Test der Baugruppe erfolgen. Dazu 5V an die entsprechenden Klemmen des VG Steckers anlegen und überprüfen, ob irgendwelche Bauteile zu warm werden. Sollte dies der Fall sein, müssen die eingelöteten Bauteile überprüft werden (richt herum eingelötet, richtige Werte etc.). Die Stromaufnahme der Baugruppe ohne gesteckte Bauelemente sollte mA nicht überschreiten,

Danach können alle gesockelten Bauteile aufgesteckt un der Test wiederholt werden. Dabei kann die Stromaufnahme mA erreichen.

Um 2MB onBoard zu realisieren, müssen auf die vorhandenen Speicherbausteine zwei weitere „huckepack“ aufgelötet werden. Dabei muss der CS Pin weggebogen und mit den Stiften CS4 und CS5 auf der Platine verbunden werden.

In gleicher Weise könnten jeweils noch 1 Speicherbaustein „gestackt“ und mit den Stiften für CS7 und CS8 angesteuert werden. Allerdings ist das z.Z. (Nov2010) noch nicht getestet.



Zuletzt muss noch das CPLD mit dem JEDEC File programmiert werden. Das kann entweder im System oder wieder durch Anschluss an eine 5V Stromversorgung geschehen.

9 Inbetriebnahme

9.1 Grundprogramme

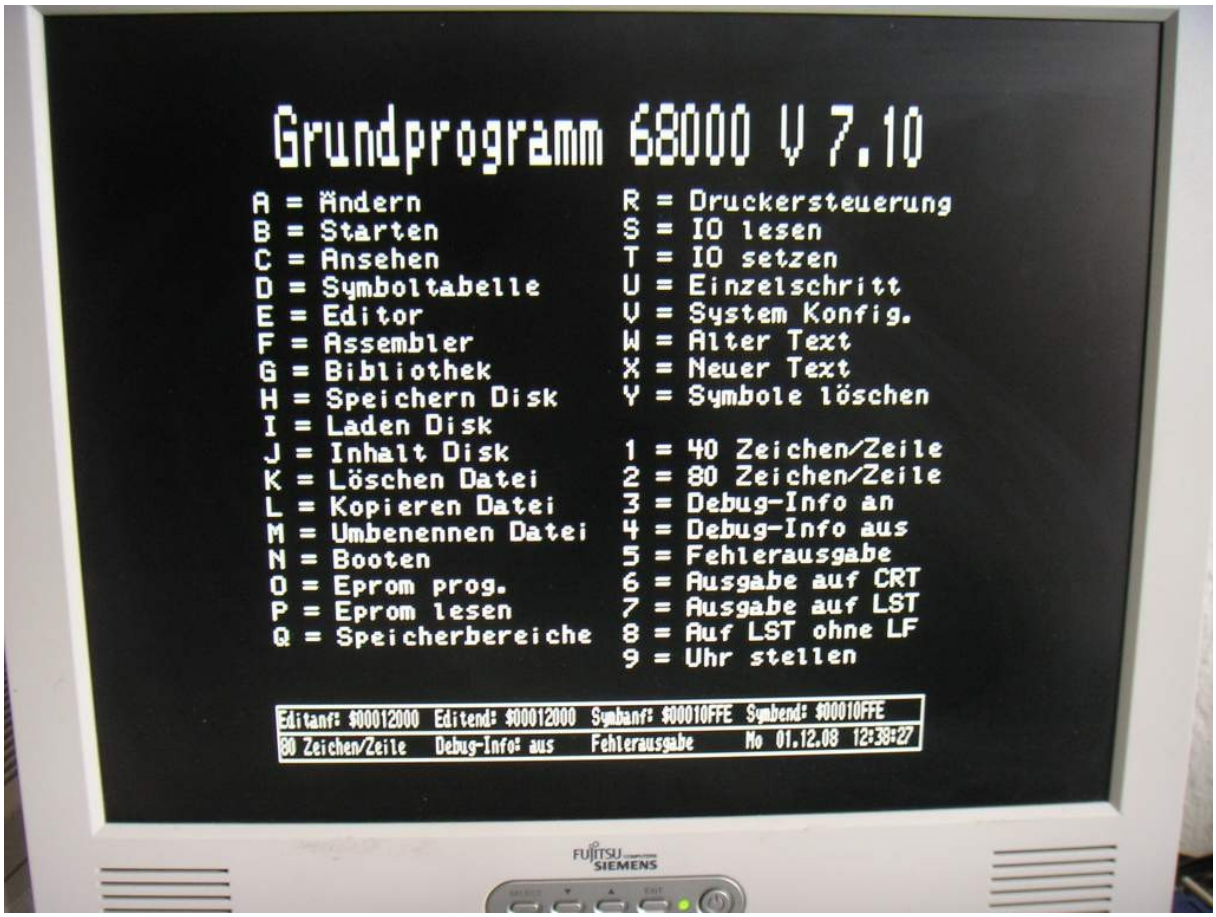
9.1.1 68000 (16-Bit Grundprogramm)

Wenn das GP als bin File vorliegt kann es in das EPROM gebrannt werden. Dazu muss alerdings noch ein Byte-Swapping durchgeführt werden (Litte-/Big-Endian Problem – Im Layout wurden die Bytes nicht getauscht).
Das geht mit dem Programm SPLITTER.EXE:

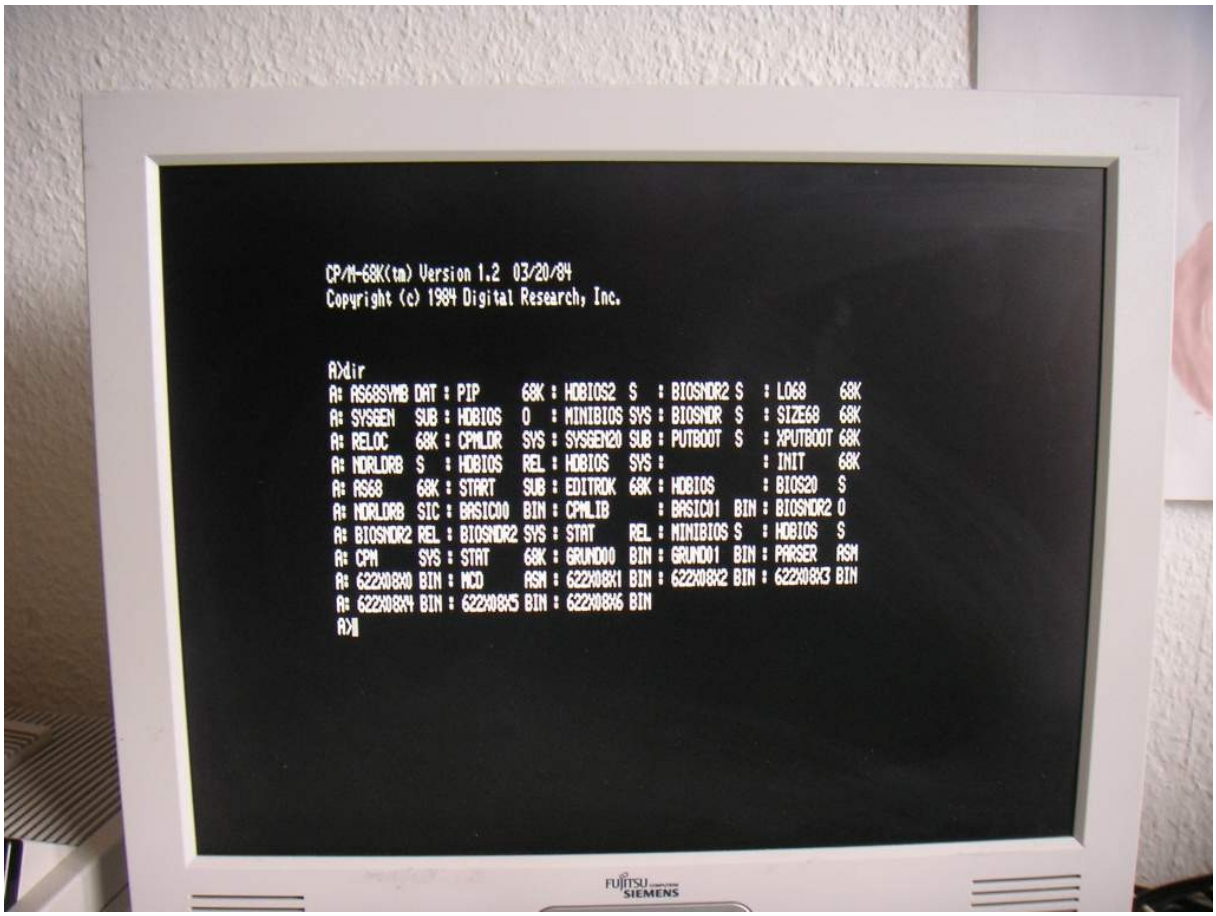
splitter –swap GP.bin GPswp.bin

GPswp.bin kann jetzt in's EPROM gebrannt werden.

Mit eingesetztem EPROM und Standard Einstellungen der Jumper sollte jetzt das GP Booten und das Menü des GP auf dem Bildschirm erscheinen.

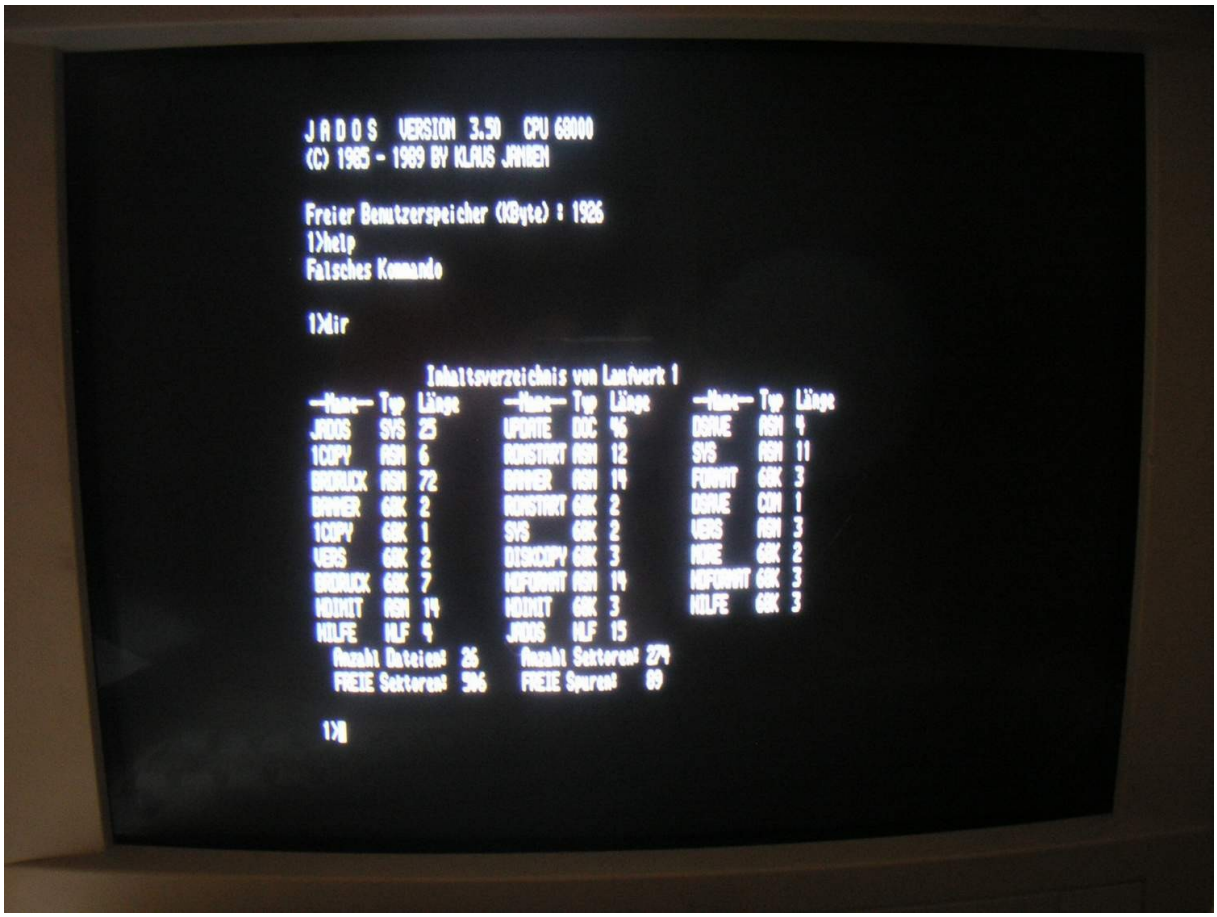


9.2 Betrieb mit CP/M-68K



9.3 Betrieb mit Jados

Jados kann entweder von der SDCARD (GDP-FPGA), GP ab Version 7.x) oder von Diskette gebootet werden.



9.4 Betrieb mit uCLinux

Um Linux zu booten sind folgende Harwarevorraussetzungen notwendig:
CPU68K im Vollausbau und GP Version 7.10
GDP-FPGA mit ScanCode-Mode Support
FLOHD

Das LINUX ROM Image (NKC68K.ROM) wird aus dem Grundprogramm von der GIDE gebootet. Für Entwicklungszwecke ist es vorteilhaft, eine SD-Card mit CD-DIE Interface zu verwenden, dadurch kann das unter Linux erstellte ROM Image schneller vom Entwicklungssystem auf den NKC gebracht werden.

Die SD-CARD ist folgendermaßen vorzubereiten:

1)
LDR_PRT1.BIN in den MBR (Sector 0)
Dadurch wird eine FAT16 Partiotionstabelle angelegt.
Die erste Partition beginnt ab Sector 66 und ist die einzige Partition auf der Card.

LDR_PRT2.BIN in den Sector 1 kopiert werden. Es gibt 2 Varianten des Teil 2 Bootloaders: einen für ein RAM Image und ein solches für ein ROM Image.

2)
SD-Card wird jetzt nicht mehr von Win** erkannt. Einfach formatieren (FAT). Win** akzeptiert die angelegte Partitionstabelle und erzeugt eine MSDOS-FAT16 Partition ab Sector 66.

3)
NKC68K.rom kann jetzt "normal" in Win** auf die Karte kopiert werden.

4)
Beim Booten (N-6) auf dem NKC passiert jetzt folgendes:
- GP liest Sector 0 und führt BootLoader Teil 1 (LDR_PRT1) aus
- BootLoader (PRT1) läd PRT2 aus Sector1 in den Arbeitsspeicher nach und springt dorthin
- PRT2 läd die MTOOLS, die wiederum die Datei NKC68K.rom nach 0x000400 und springt dort hin (Label start: in crt0.S).

Jetzt übernimmt der Bootloader von uCLinux (crt0.asm) das Kommando:
Zunächst wird das ROM-Image in den Zielbereich kopiert und das BSS Segment initialisiert. Danach wird nach kernel_start() gesprungen und das System initialisiert.

Nähere Einzelheiten dazu bitte aus der Dokumentation des uCLinux Patches für den NKC2007 entnehmen.

10 Fehlerkorrekturen

Bisher (Stand Nov. 2010) sind keine Fehler bekannt.
Beim Aufbau sollte die Adresse des Banken-Registers im eingesetzten CPLD und der Software überprüft werden, da während der ersten Test-Phasen diese Adresse auf $C9 gesetzt wurde.

11 Quellenhinweise

CP/M-68K für den c't68000 c't 9/1985
c't 68000 Das Quadro Board c't 10/1984
c't68000 Teil 1 c't 11/1984
c't68000 Teil 2 c't 12/1984
c't68000 Teil 3 c't 1/1985
c't68000 Software c't 2/1985
Herzverpflanzung c't 11/1985
M68000 User's Manual MOTOROLA 1993
M68000 Addendum FREESCALE 2004
CPU68000 / Graf Elektronik Stand Okt 1985
UHR3 Jens Mewes 2007
DS12C887 RTC Datasheet DALLAS 020900
Die Bank Boot Baugruppe / GES
BOOTRAM68k Gerald Ebert 2007
Handbuch Grundprogramm V 7.0 Jens Mewes 2007
Handbuch Jados 3.50 Klaus Janßen 1985-1990

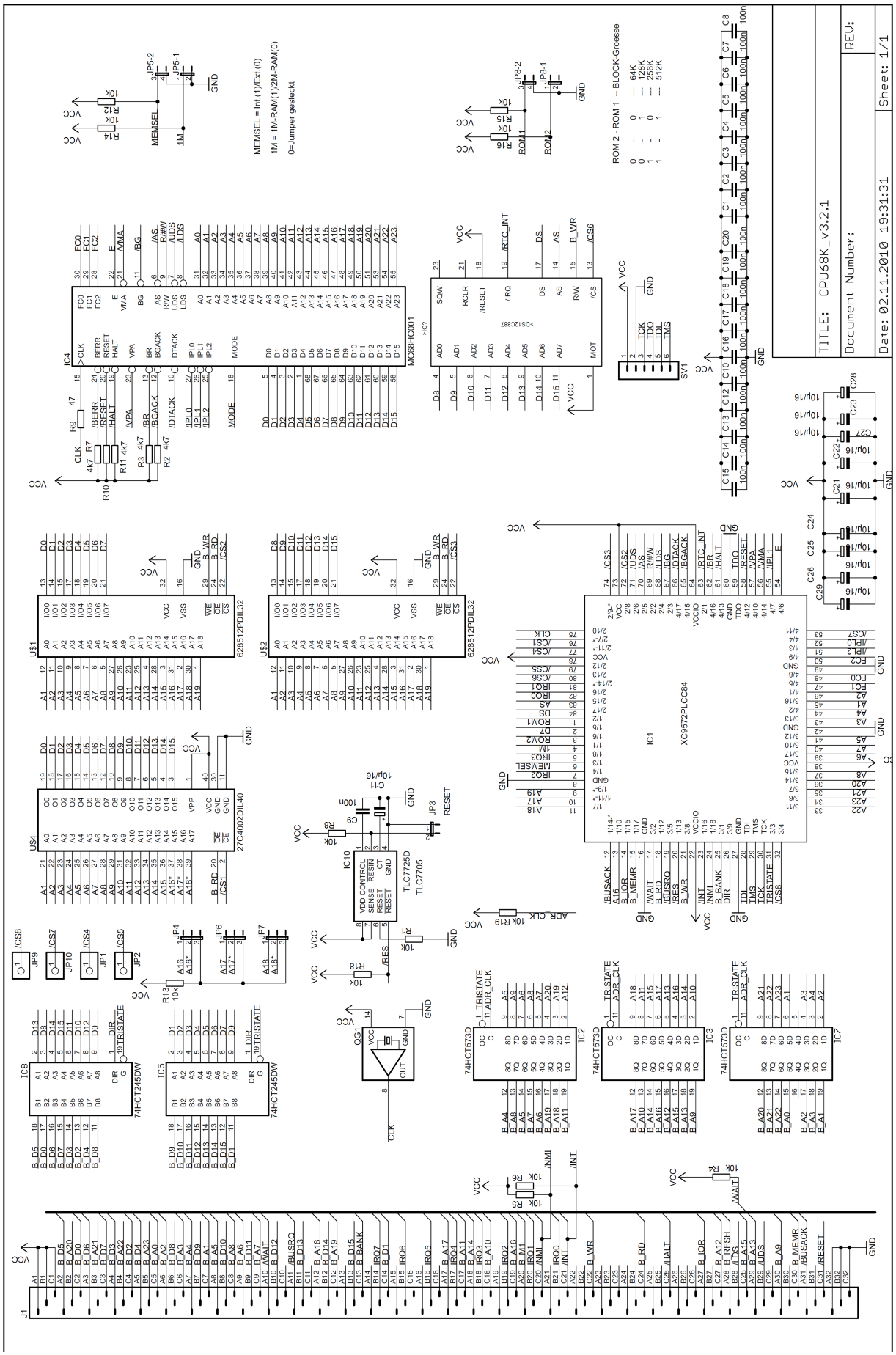
http://www.schuetz.thtec.org/index.html
http://www.drcrazy.de/nkc

12 Anhänge

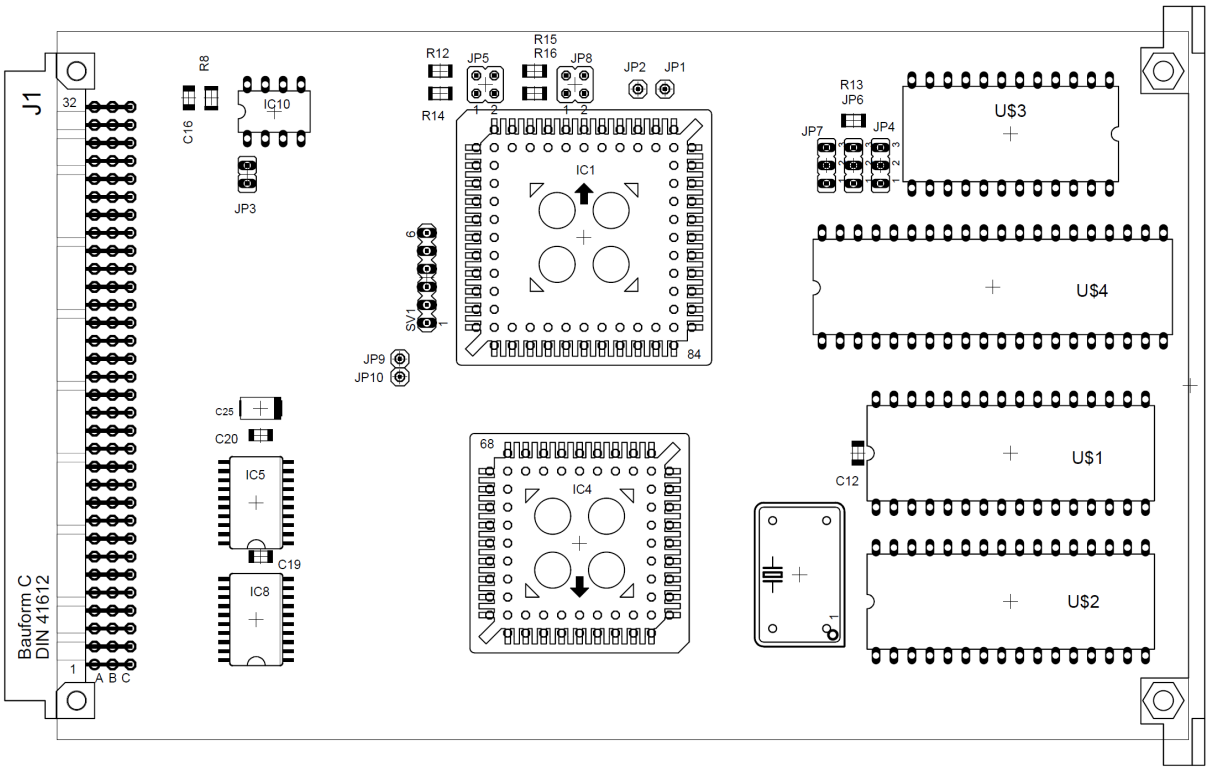
12.1 Stückliste

Menge	Wert	Device	Bauteile
3		JP2E	JP4, JP6, JP7
2		JP2QE	JP5, JP8
1		MA06-1	SV1
4		PINHD-1X1	JP1, JP2, JP9, JP10
1		QG5860	QG1
1		VG96P	J1
1	3K3	R-EU_M1206	R11
4	4k7	R-EU_M1206	R2, R3, R7, R10
10	10µ/16	CPOL-EUSMCC	C11, C21, C22, C23, C24, C25, C26, C27, C28, C29
12	10k	R-EU_M1206	R1, R4, R5, R6, R8, R12, R13, R14, R15, R16, R18, R19
1	27C4002DIL40	27C4002DIL40	U$4
1	47	R-EU_M1206	R9
2	74HCT245DW	74HCT245DW	IC5, IC8
3	74HCT573D	74HCT573D	IC2, IC3, IC7
19	100n	C-EUC1206	C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C12, C13, C14, C15, C16, C17, C18, C19, C20
2	628512PDIL32	628512PDIL32	U$1, U$2
1	DS12C887DIL24	DS12C887DIL24	U$3
1	MC68HC001	MC68HC001	IC4
1	RESET	JP1E	JP3
1	TLC7725D	TLC7725D	IC10
1	XC9572PLCC84	XC9572PLCC84	IC1

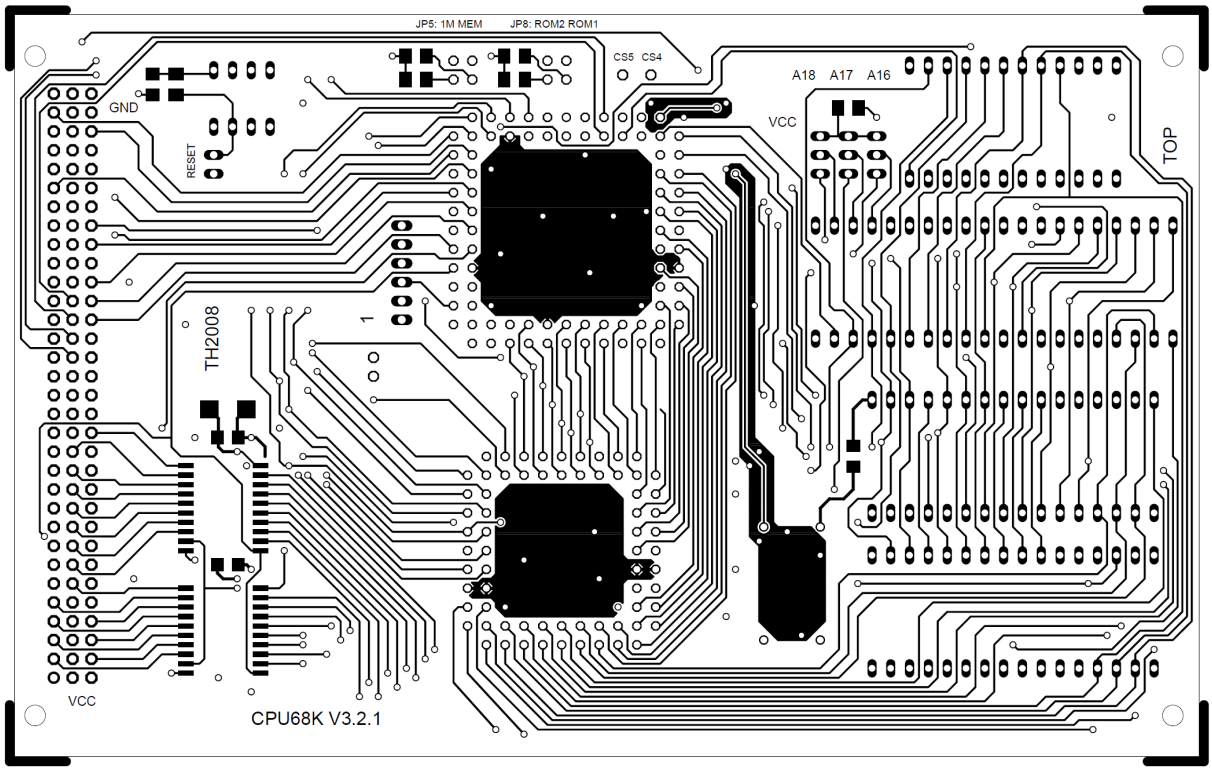
12.2 Schaltplan



12.3 Bestückungsplan



12.4 Layout Top



12.5 Layout Bottom