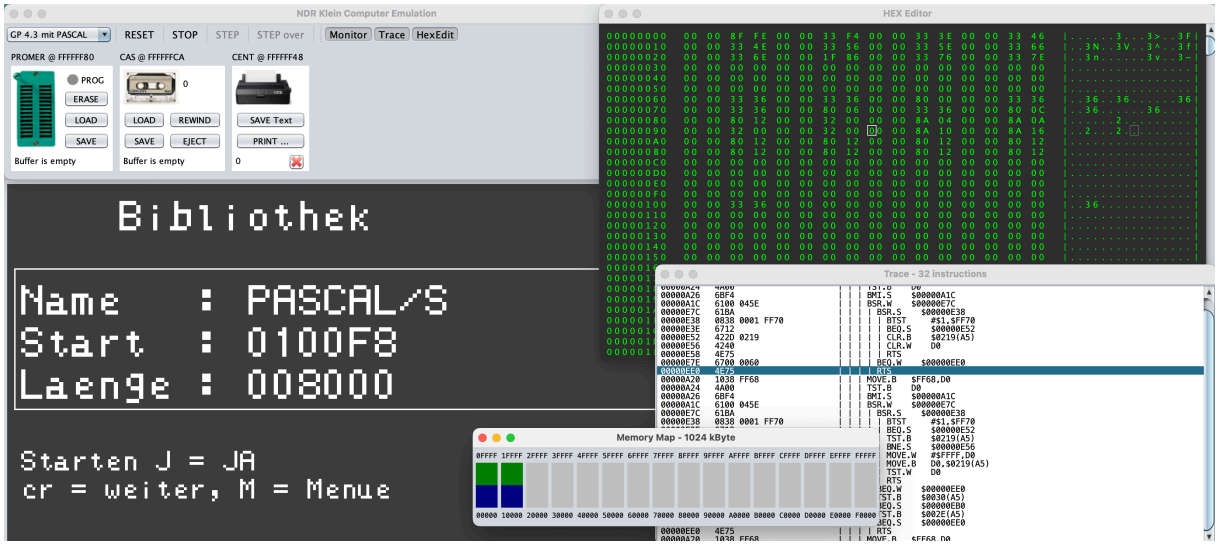
NDR Klein Computer Emulation

40 Jahre NDR-Klein-Computer



Vorwort

Endlich ist es so weit, zum 40-jährigen Geburtstag des NDR-Klein-Computers gibt es eine erste Version einer Emulation des sehr gut konzipierten und flexiblen Lernsystems.

Seit meiner Jugendzeit hat mich der von Rolf-Dieter Klein konzipierte und im deutschen Fernsehen vorgestellte Computer fasziniert. All mein Taschengeld ist in den Erwerb der Baugruppen geflossen. Der Computer hat mich viele Jahre lang begleitet und mein weiteres Berufsleben geprägt

Später habe ich beschlossen, mein angehäuftes Wissen anderen Menschen weiterzugeben und habe eine Homepage kreiert, die sich bis heute ausschließlich dem NKC widmet. Leider hat mich das Berufsleben sehr häufig davon abgehalten, die umfangreiche Seite angemessen zu pflegen.

Schon vor knapp sechs Jahren hatte ich angefangen, eine Software-Emulation für den NKC umzusetzen, das Projekt ist dann leider liegengeblieben und vergessen worden. Im Sommer des Jahres 2023 hat mich Martin Merck kontaktiert, der einen eigenen Emulator entwickelt. Davon inspiriert, habe ich mein Projekt wieder aufgenommen und versucht so schnell wie möglich zumindest in einen Zustand zu bringen, den man als erste Version veröffentlichen kann. Ich danke Martin auch für die Bereitstellung seiner Quellen sowie Tipps bei der Lösung von Problemen.

Für die Umsetzung des Emulators waren die Vorarbeiten von Tony Headford im Rahmen seines Projektes „Java Amiga MachineCore“ essenziell. Die Emulation des Mikroprozessors MC 68000 ist gut aber war noch nicht perfekt. Das Projekt „Hexadecimal File Editor“ von Keith Fenske ist ebenfalls in den Emulator eingeflossen, um einen direkten Einblick in den Hauptspeicher der Emulation zu gewähren. Beide Projekte mussten in den Emulator kopiert werden, da etliche Erweiterungen zur Integration in den Emulator notwendig waren. Quellen und Links finden sich im Anhang dieses Dokuments.

Ein abschließender Dank geht an meine Frau, die es erträgt, wenn ich stundenlang ohne für sie erkennbares Ziel vor meinem MacBook sitze.

Inhaltsverzeichnis

Die Emulation

Der zentrale Grund, warum Java als Grundlage für diese Software gewählt wurde, liegt in seiner Plattformunabhängigkeit. Die Fähigkeit von Java, auf verschiedenen Systemen ausgeführt zu werden, ermöglicht es den Benutzern, die Software sowohl unter OS X als auch Windows und Linux zu benutzen.

Darüber hinaus bietet Java eine stabile und sichere Umgebung, was zu einer zuverlässigen Leistung und einem erhöhten Schutz vor potenziellen Sicherheitsrisiken führt.

Zur Emulation der Hardware-Baugruppen des NKC wurde ein Konzept entwickelt, mit dem sich verschiedene Treiber aktivieren lassen, die jeweils genau eine Baugruppe behandeln. Die Treiber sind fest in den Emulator integriert und lassen sich bei zukünftigen Revisionen erweitern.

Die Basisadressen der Hardware-Baugruppen können frei festgelegt werden, allerdings ist die allgemein verfügbare Software des NKC darauf nicht ausgelegt. Daher sollten die Adressen in der Konfigurationsdatei gemäß dem NKC-Standard vergeben werden.

Hardware-Treiber bestehen mindestens aus dem Treiber selbst und einem Memory-Handler. Zusätzlich kann ein Darstellungsbereich (Frame) und ein parallel zum Emulator ablaufendes Programm (Thread) implementiert werden. Die Frames dienen zur Darstellung im Haupt-Fenster des Emulators, die Threads zur Aktualisierung des Inhaltes dieser Frames.

Die Entwicklung ist lange noch nicht abgeschlossen. Hier eine kleine Liste der Sachen, die ich noch auf dem Plan habe:

- Umsetzung weiterer Baugruppen wie zum Beispiel IOE, SOUND, SER …
- Umsetzung des Mikroprozessors Z80
- Integration eines Disassemblers
- Unterstützung anderer Diskettenformate
- Fehlerbehebungen
- Verarbeiten von Rückmeldungen der Benutzer

Systemvoraussetzungen

Um den Emulator ausführen zu können ist eine aktuelle Java-Laufzeitumgebung notwendig. Bei der Entwicklung wurde das Java Development Kit JDK17 verwendet.

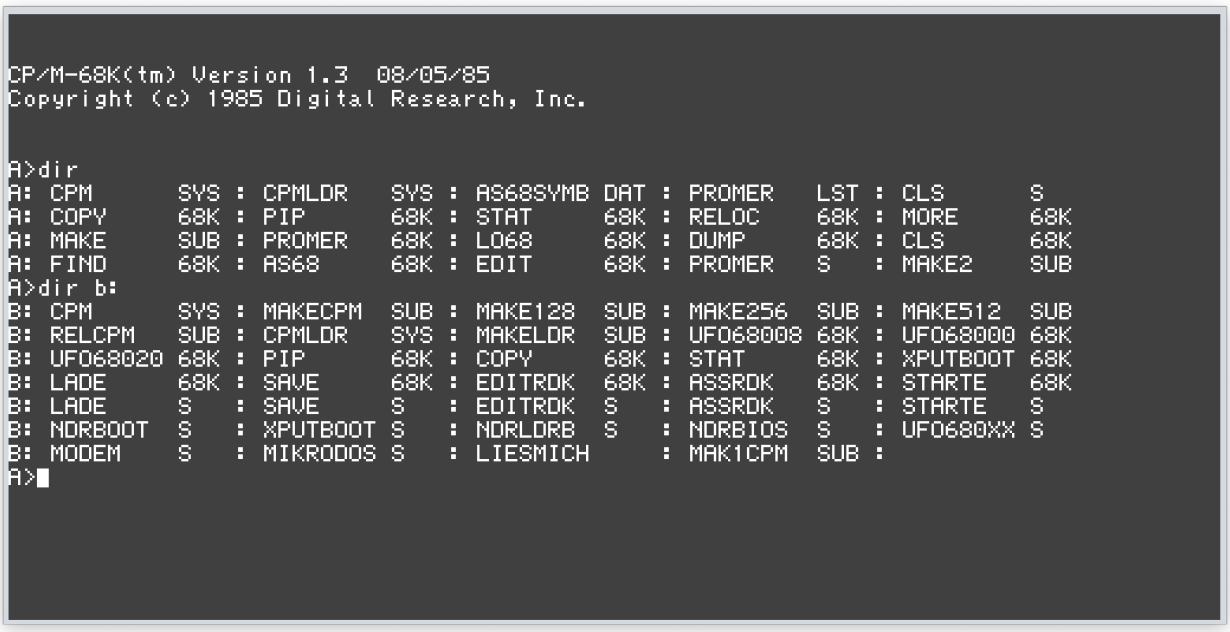
Starten der Emulation

Ein beherzter Doppelklick auf die Datei „NKCEmu.jar“ startet die Emulation. Beim Erststart wird eine Standardkonfiguration aktiviert, die jederzeit durch eine Auswahl im Dropdown-Menü des Hauptfensters geändert werden kann. Die zuletzt gewählte Konfigurationsdatei wird bei einem Neustart wiederhergestellt.

Das Hauptfenster

Das Hauptfenster dient zur Steuerung der Emulation. Unter der Symbolleiste werden die Baugruppen angezeigt, welche durch die Konfigurationsdateien ausgewählt wurden. Im Bild sind die Frames der Baugruppen FLO2, PROMER, CAS und CENT sichtbar.

Direkt unter dem Hauptfenster wird ein Fenster für den Bildschirm des NKC angezeigt. Dieses Fenster bleibt beim Verschieben des Hauptfensters immer unter dem Hauptfenster kleben.



Symbol-Leiste

Über die Symbolleiste lassen sich die wichtigsten Funktionen des Emulators steuern. Das DropDown-Menü zeigt alle Konfigurationsdateien aus dem Unterordner CONFIG an und gestattet das schnelle umkonfigurieren der Hard- und Softwarezusammenstellung, die bei der Emulation genutzt werden.

RESET

Setzt das emulierte System zurück. Geladene Dateien und Pufferinhalte bleiben erhalten.

STOP

Hält die Emulation an und aktiviert den Einzelschrittmodus.

RUN

Beendet den Einzelschrittmodus und startet den automatischen Ablauf der CPU.

STEP

Führt einen einzelnen Befehl der CPU aus. Wenn das Trace-Fenster aktiv ist, wird dort die disassemblierte Instruktion ausgegeben.

STEP over

Führt einen Befehl der CPU aus. Falls in ein Unterprogramm gesprungen wird, wird dieses bis zum Ende des Unterprogrammes ausgeführt.

Monitor

Zeigt oder verbirgt den Bildschirm, der durch die Baugruppe GDP64K angesteuert wird.

Trace

Zeigt oder verbirgt das Trace-Fenster zur Anzeige der Instruktionen im Einzelschrittmodus.

HexEdit

Zeigt oder verbirgt das Fenster zur Anzeige und Manipulation des Speicherinhaltes.

Menü

Die Funktionen des Emulators können auch über das Menü gesteuert werden, hier sind auch Tastaturkürzel verfügbar.

Edit

- Paste to NKC – Inhalt der Zwischenablage wird an die Baugruppe KEY gesendet.
- Take Screenshot – Der Inhalt des Monitors wird in die Zwischenablage kopiert.
- Select all – Der gesamte Inhalt des Trace-Fensters wird markiert.
- Copy selection – Kopiert den markierten Trace-Inhalt in die Zwischenablage

Run

- Start Emulation – Startet die Emulation und beendet den Einzelschrittmodus.
- Stop Emulation – Hält die Emulation an und aktiviert den Einzelschrittmodus.
- Single Step – Führt eine einzelne Instruktion der CPU aus.
- Step Over – Führt eine einzelne Instruktion oder ein ganzes Unterprogramm aus.
- Reset – Setzt die CPU und die Hardware zurück.

View

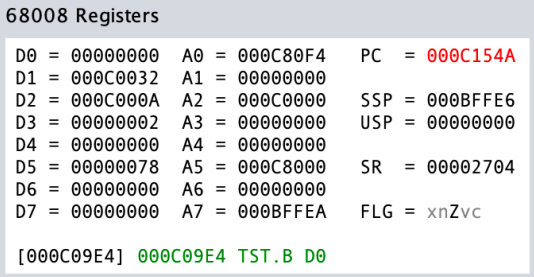
- Monitor Window – Zeigt den Monitor an
- Trace Window – Zeigt das Trace-Fenster an
- HexEdit Window – Zeigt das das Hex Editor Fenster an
- Memory Map Window – Zeigt die Speicherbelegung an

Help

- About – Zeigt Informationen zum Emulator an

Emulierte Hardware

CPU – Central Processing Unit



Die Emulation basiert auf der Implementierung des Motorola MC68000 Prozessors von Tony Headford mit ein paar spezifischen Erweiterungen, welche die Nutzung im NKC-Emulator erleichtern und verbessern. Änderungen beziehen sich im Wesentlichen auf das Berechnen des Timings der verschiedenen Prozessor-Varianten.

Über die Konfigurationsdateien können verschiedene Prozessoren ausgewählt werden. Dies bestimmt sowohl den maximal nutzbaren Speicherbereich als auch die Ablaufgeschwindigkeit des Emulators. Die Geschwindigkeit der Emulation entspricht etwa der realen Geschwindigkeit im NKC.

Da sich die Emulationsgeschwindigkeit auf die Systemzeit des emulierenden Computers bezieht, kann die Geschwindigkeit der Emulation je nach verwendetem Computer von der vorgegebenen Taktfrequenz abweichen. In der Regel beträgt die Differenz jedoch unter 2%.

In der Konfigurationsdatei kann optional ein Frame aktiviert werden, der die Inhalte aller Register der CPU in Echtzeit anzeigt. Dabei werden die aktuell geänderten Registerinhalte hervorgehoben dargestellt. Der gerade ausgeführte Befehl wird in disassemblierter Form angezeigt. Im Zusammenhang mit dem Trace-Fenster und den Breakpoints ist das sehr hilfreich bei der Entwicklung eigener Programme.

CPU 68000

- Maximaler Speicherbereich 2 MByte 0x00000000 – 0x001FFFFF
- Nominelle Taktfrequenz 16 MHz

CPU 68008

- Maximaler Speicherbereich 1 MByte 0x00000000 – 0x000FFFFF
- Nominelle Taktfrequenz 8 MHz

Optional kann die emulierte Taktfrequenz der CPU abweichend von der Vorgabe eingestellt werden. Dazu wird das Schlüsselwort CLOCK in den Konfigurationsdateien verwendet.

MEMORY – Hauptspeicher

Bei der Emulation des Hauptspeichers wird nicht zwischen den verschiedenen Baugruppen wie ROA64, RAM64/256 usw. unterschieden. Stattdessen wird der gesamte Speicherbereich wie ein zusammenhängender Bereich mit unterschiedlichen Eigenschaften betrachtet.

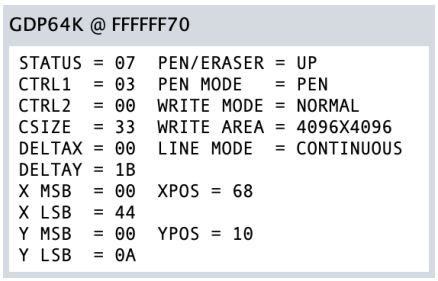
Die Größe des verwalteten Hauptspeichers richtet sich nach der CPU und umfasst den gesamten möglichen Adressbereich des gewählten Mikroprozessors. Der Speicher kann durch die Konfigurationsdatei frei definiert werden. Dadurch ist es möglich, die Bestückung aller Speicherbaugruppen festzulegen.

Über die Konfigurationsdateien ist es möglich, die Größe des maximal adressierbaren Speichers für die Emulation anzupassen. Falls diese Möglichkeit nicht genutzt wird, werden die Vorgaben des gewählten Mikroprozessors genutzt.

BANKBOOT – Speicher ab Adresse 0

Die Emulation verwaltet neben dem Hauptspeicher 32 kByte Speicher der BANKBOOT-Karte, die nach dem Start des Systems dazu dient, RAM-Speicher ab Adresse 0 einzubinden. Auch hier kann die Bestückung mit ROM oder RAM über die Konfigurationsdateien frei festgelegt werden. Die Baugruppe BANKBOOT hat keinen eigenen Anzeigebereich im Hauptfenster.

GDP64K – Grafik-Interface und Monitor

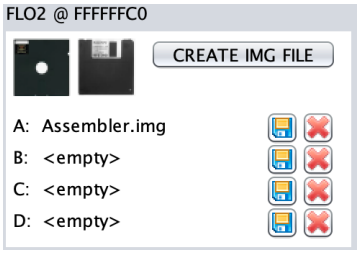


Der Treiber emuliert das Verhalten des Grafik-Prozessors EF9366 auf der Baugruppe GDP64K. Ein 64 kByte umfassender Bildschirmspeicher wird mit entsprechenden Befehlen an den Treiber manipuliert.

Ein eigenes Fenster stellt den Inhalt dieses Bildschirm-Speichers dar. Der Inhalt des Fensters wird durch einen getrennten Thread 50-mal pro Sekunde aktualisiert und läuft unabhängig von der Emulation des restlichen Systems. Um die Proportionen der Ausgabe einigermaßen korrekt darzustellen, wird jedes Pixel als 3x2 Block dargestellt.

In der Konfigurationsdatei können optional andere Skalierungen angegeben werden, die zu einer mehr oder weniger verzerrten Ausgabe führen. Ebenfalls optional kann ein Frame aktiviert werden, der in Echtzeit die Inhalte der Register des Grafikprozessors im Hauptfenster darstellt.

FLO2 – Floppy-Disk-Interface



Mit dem Treiber FLO2 können insgesamt vier Diskettenlaufwerke emuliert werden. Welche Formate erkannt werden hängt dabei natürlich von der Implementierung innerhalb der betriebenen Software ab. Um das Arbeiten mit dem Emulator einfacher zu gestalten, können leere Diskettenimages im Standard-Format des NKC direkt über die Oberfläche erstellt werden.

Das Standard-Format des NKC bietet eine Bruttokapazität von 800 kByte entsprechend der Speicherkapazität einer 5,25 Zoll Diskette im Double Density Format. Eine Formatierung wie bei physischen Disketten ist in der Emulation nicht notwendig. Der Treiber arbeitet unmittelbar auf den Dateien des emulierenden Computers, nach der Arbeit muss also nicht manuell gespeichert werden.

In der beim NKC vorgesehenen Konfiguration sind die Laufwerke in CP/M wie folgt belegt

- A: 5,25“ Standard NKC Format beidseitig
- B: 5,25“ Standard NKC Format beidseitig
- C: 8“ Vorderseite der Diskette (derzeit nicht unterstützt)
- D: 8“ Rückseite der Diskette (derzeit nicht unterstützt)

Die mitgelieferten Disketten-Images liegen ausschließlich im Standard-Format vor. Im Anhang findet sich eine Liste der Inhalte dieser Images.

Create Image File

Erzeugt ein neues Image einer leeren Diskette, welches danach sofort in eines der Laufwerke geladen und benutzt werden kann. Eine Formatierung ist nicht notwendig.

Disketten Image laden

Mit den Diskettensymbol-Buttons neben den Laufwerksbuchstaben kann ein Image in eines der Laufwerke geladen werden. Nach der Auswahl wird der Dateiname der Image-Datei neben dem Laufwerksbuchstaben angezeigt. Eine Neuauswahl ist jederzeit auch während des Betriebs möglich, ohne dass ein Datenverlust auftritt.

Disketten Image auswerfen

Mit dem zweiten Button neben dem Dateinamen der Imagedatei kann dieses Diskettenimage ausgeworfen werden. Auch hier sind keine Datenverluste zu erwarten, da der Treiber direkt auf den Image-Dateien arbeitet.

PROMER – EPROM-Programmier-Interface

Der Treiber für die Baugruppe PROMER ist auf die übliche Konfiguration für 8 kByte EPROMS des Typs 2764 eingestellt. EPROMS können gelesen, programmiert und gelöscht werden. Die Bedienung der Baugruppe erfolgt über den Frame im Hauptfenster.

Die Programmierung erfolgt entweder über das Grundprogramm oder über das Programm PROMER.68K direkt aus CP/M. Dieses Programm befindet inklusive Quellcode auf der NKC-Zusatzdiskette. Wie im Original leuchtet während des Programmierens eine LED auf.

LOAD – Einsetzen bestehender EPROMs

Lädt den Inhalt des Puffers mit einer ROM-Datei aus dem Unterverzeichnis PROMER. Nach dem Laden wird der Dateiname der geladenen Datei unten im Frame angezeigt. Der Inhalt des Puffers kann jetzt mit dem Grundprogramms in den Hauptspeicher übertragen werden.

SAVE – Speichern programmierter EPROMs

Speichert den Inhalt des Puffers in das Unterverzeichnis PROMER. Nach dem Speichern wird der Dateiname der gesicherten Datei unten im Frame angezeigt. Dateien, die hiermit erzeugt wurden, können in die Unterverzeichnisse ROMS und BOOT kopiert werden, um diese bei der Konfiguration des Hauptspeichers bzw. des Speichers auf der Baugruppe BANKBOOT zu verwenden.

ERASE – Löschen von EPROMs

Löscht den 8 kByte umfassenden Puffer, alle Bytes enthalten den Wert 0xFF. Anstelle des Dateinamens wird der Text „EPROM is empty“ angezeigt. Im Gegensatz zu einem realen EPROM lassen sich auch gesetzte Bits in den Puffer schreiben, man muss also nicht den Puffer löschen, um erneut programmieren zu können.

CAS – Kassetten-Interface

Der Treiber für die Baugruppe CAS emuliert das Kassetteninterface des NKC und verwaltet einen Daten-Puffer unbestimmter Größe. In diesem Puffer werden alle an den Baustein 6850 der Baugruppe CAS übertragenen Bytes gesichert beziehungsweise von dort geladen.

Der Frame zeigt neben den vier Buttons zur Bedienung ein Zählwerk (Byte-Zähler) und den Dateinamen der aktuell geladenen oder gespeicherten CAS-Datei aus dem Unterverzeichnis CAS des Emulators an.

Die Bedienung der Baugruppe erfolgt über die Menüpunkte **Sichern CAS**, **Laden CAS** und **Pruefen CAS** des Grundprogramms. Beim Speichern wird das Timing des Grundprogramms nachgebildet, das Laden und Prüfen erfolgt hingegen sehr schnell, auf die Umsetzung einer Verzögerung wurde absichtlich verzichtet.

LOAD – Einlegen von Kassetten

Lädt eine zuvor mit CAS gesicherte Datei in den Puffer. Das Zählwerk wird automatisch auf 0 zurückgesetzt. Das Grundprogramm wartet nach Aufruf des Menüpunkts „Laden CAS“ so lange bis eine CAS-Datei geladen wurde. Da das Grundprogramm beim Laden keinen Dateinamen abfragt, kann man effektiv nur das erste gesicherte Programm einer CAS-Datei verwenden.

SAVE – Sichern des Pufferinhaltes

Sichert den Pufferinhalt in eine CAS-Datei im Unterverzeichnis CAS des Emulators zur späteren Verwendung. Nach dem Speichern zeigt das Zählwerk die Anzahl der in den Puffer geschriebenen Bytes an.

REWIND – Zurückspulen von Kassetten

Das Zählwerk wird auf 0 zurückgesetzt, die geladenen oder gespeicherten Daten im Puffer bleiben erhalten.

EJECT – Auswerfen von Kassetten

Löscht den Puffer. Es ist zu beachten, dass möglicherweise geschriebene Daten zuvor gesichert werden müssen.

CENT – Centronics-Drucker-Interface

Der Treiber für die Baugruppe CENT emuliert das Druckerinterface des NKC und erlaubt das Ausdrucken sowie das Speichern aller an die Baugruppe gesendeten Daten. Der Treiber enthält einen Puffer, in dem alle gesendeten Daten gesammelt werden. In der unteren linken Ecke wird der Füllstand des Puffers angezeigt.

Der Pufferinhalt kann wahlweise als Textdatei gespeichert oder direkt zu einem physikalisch vorhandenen Drucker gesendet werden. Nach dem Speichern oder Ausdrucken kann der Puffer manuell gelöscht werden. Als Textdatei gesicherte Druckerausgaben werden im Unterverzeichnis CENT des Emulators abgelegt. Diese Funktion ist hilfreich bei der Erstellung von Dokumentationen eigener Programme.

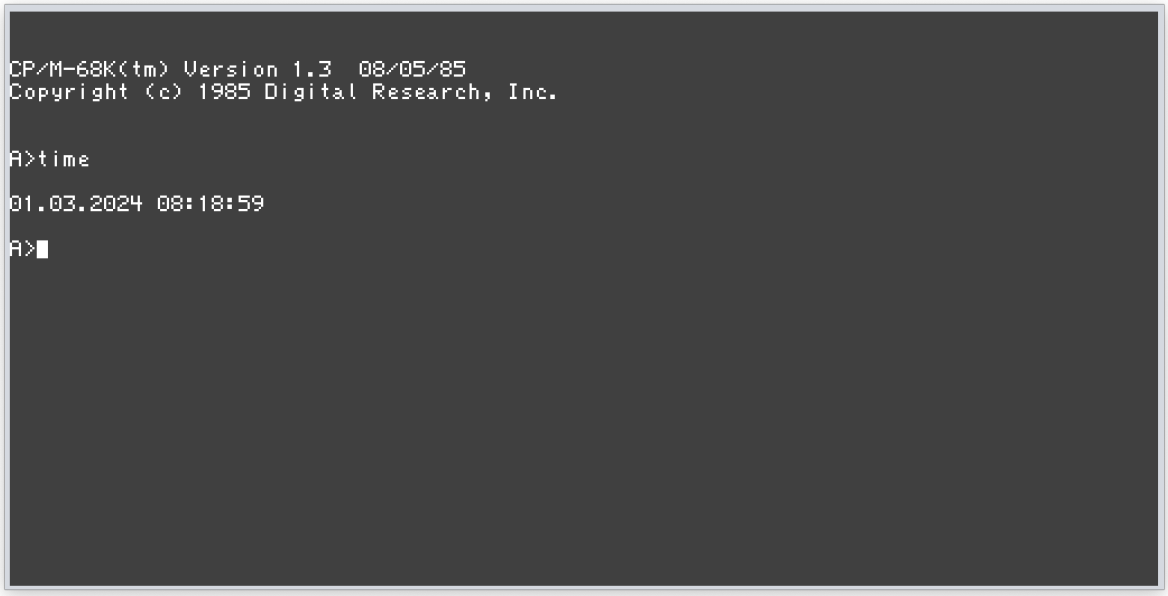
Unter CP/M kann mit der Tastenkombination Ctrl+P erreicht werden, dass alle Ausgaben auf dem Bildschirm gleichzeitig zum Drucker gesendet werden. Die Tastenkombination wechselt ein Flag im Treiber und zeigt abhängig von Status einen roten Punkt neben dem Drucker-Symbol an.

Da der Emulator keine Möglichkeit hat, festzustellen wann CP/M gestartet wird oder ob das Betriebssystem aktuell ausgeführt wird, sollte man darauf achten, dass das Flag nicht aktiv ist, bevor man CP/M startet.

UHR – Real Time Clock

Die Baugruppe UHR stellt eine batteriegestützte Echtzeituhr im NKC zur Verfügung.

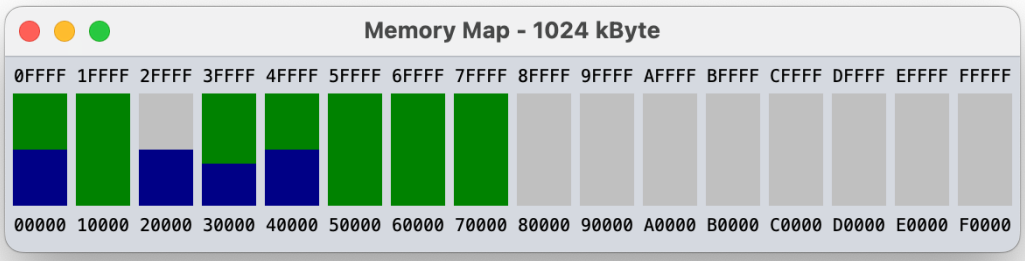
In der Emulation ist nur der lesende Zugriff auf die Uhrzeit umgesetzt, es wird immer die Zeit des Hostcomputers zurückgegeben. Das Stellen der Uhrzeit durch den NKC hat keinen Effekt, führt aber auch nicht zu einem Fehler.



Weitere Funktionen

Memory-Map Fenster

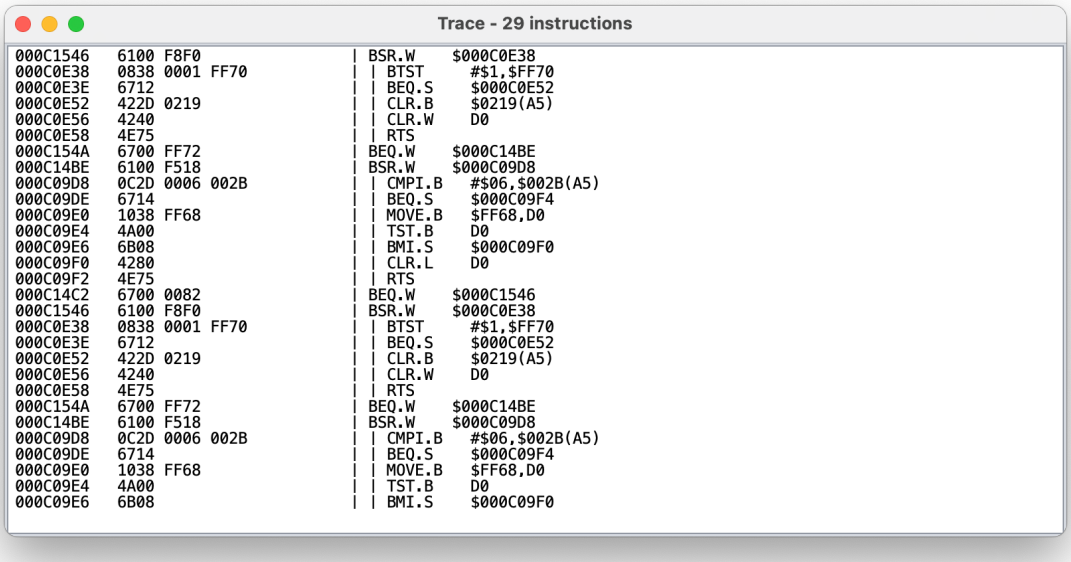
Im diesem Fenster wird die Belegung des Hauptspeichers dargestellt. So kann man jederzeit schnell überprüfen, an welchen Adressen ROMs geladen und an welchen Adressen RAM definiert wurde. Grundsätzlich wird nur das erste Megabyte des Speichers angezeigt. Bereiche mit RAM werden grün, ROM-Bereiche werden blau dargestellt. Unbelegte Bereiche erscheinen in grauer Farbe.



Das Fenster kann über das Menü View des Hauptfensters oder mit der Tastenkombination ALT+4 beziehungsweise Command+4 unter OSX aufgerufen werden.

Trace Fenster

Das Trace-Fenster dient zur Anzeige der ausgeführten Befehle des Mikroprozessors. Sinnvoll ist diese Anzeige jeder einzelnen Instruktion nur im Einzelschrittmodus, damit man den Ablauf eines Programmes exakt nachvollziehen kann. Im normalen Betrieb werden die Instruktionen auch bei geöffnetem Trace Fenster nicht wiedergegeben. Die Ausgabe der disassemblierten Befehle verbraucht sehr viel Zeit, so dass die Geschwindigkeit der Emulation deutlich geringer wäre.



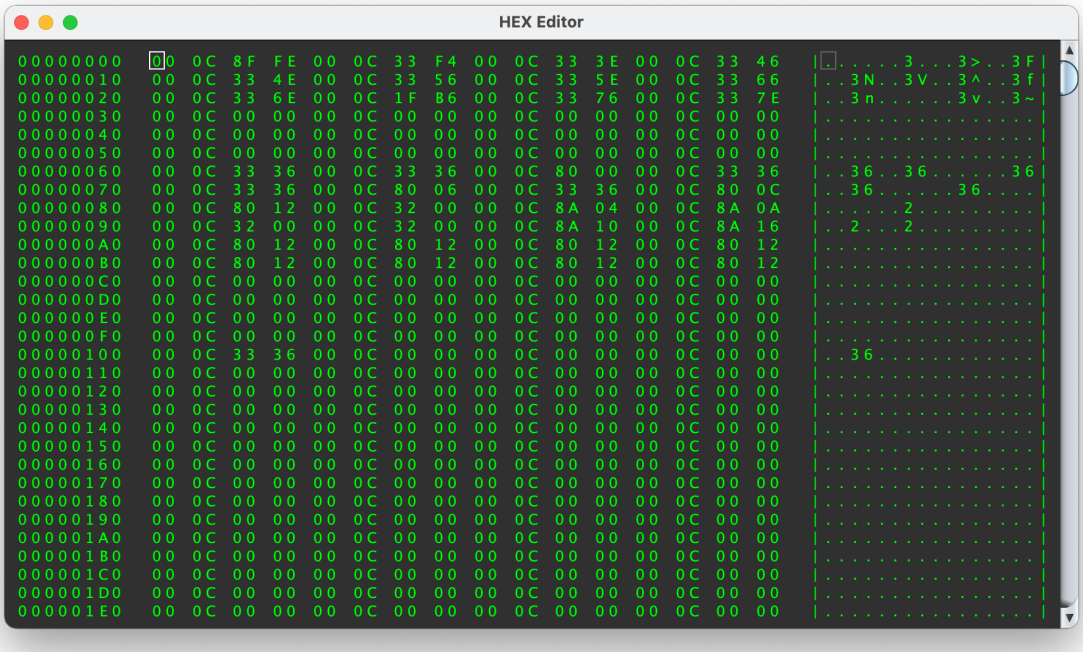
Jeder von der CPU ausgeführte Befehl wird in einer Zeile dargestellt. Die Zeilen bestehen aus

- Aktueller Programmzählerstand
- Bytes des ausgeführten Befehls
- Disassemblierte Instruktion

Die Instruktionen sind entsprechend der Verschachtelung der Unterprogramme eingerückt. Über das Menü kann der gesamte Text ausgewählt und in die Zwischenablage kopiert werden. Im Fenster ist zudem die Auswahl von Teilbereichen des Listings möglich.

Hex-Editor Fenster

Der Hex-Editor bietet einen direkten Zugriff auf den gesamten Hauptspeicher der Emulation. Die Anzeige wird bei sichtbarem Fenster immer aktuell gehalten und bietet so wertvolle Informationen beim Debuggen von Programmen.



Zusätzlich kann der gesamte Speicher unmittelbar überschrieben werden. Dabei sollte man Vorsicht walten lassen, da Änderungen im aktuell ausgeführten Programmbereich fast immer zu ungültigen Befehlssequenzen führen.

Der rechteckige Cursor kann frei im HEX-Bereich oder im ASCII-Bereich positioniert werden. Eingaben werden unmittelbar in den Hauptspeicher der Emulation geschrieben.

Konfigurationsdateien

Die Konfigurationsdateien sind zeilenorientierte Textdatei, die zur Konfiguration der Hard- und Software für den Emulator dienen. Beim Start des Emulators wird die zuletzt genutzte Konfigurationsdatei zuerst analysiert und danach der Emulator mit den darin enthaltenen Vorgaben gestartet. Es können mehrere Konfigurationsdateien definiert werden, die im Unterverzeichnis CONFIG abgelegt sein müssen.

Während der Laufzeit des Emulators kann man im Hauptfenster über eine Auswahl der gewünschten Konfiguration zwischen den verschiedenen Konfigurationen wechseln, was jeweils einen Neustart der Software auslöst.

Bei Fehlern in einer ausgewählten Konfiguration wird eine Meldung mit den Fehlern angezeigt und die zuvor gewählte Konfiguration bleibt aktiv. Bei der Auswahl einer gültigen Konfiguration wird diese gespeichert. Beim nächsten Start der Emulation wird automatisch die letzte Konfiguration wiederhergestellt.

Konfigurationsdateien sind zeilenorientierte Textdateien mit den nachfolgend beschriebenen Möglichkeiten. Die Dateien dürfen Kommentare enthalten, die mit // eingeleitet werden oder optional mit * am Anfang einer Zeile. Die Reihenfolge der Schlüsselworte ist nicht relevant, die Reihenfolge der Parameter hingegen muss eingehalten werden.

Prozessordefinition

CPU

Dient zur Auswahl zwischen den emulierten Mikroprozessoren. Es ist genau eine CPU-Angabe in einer Konfigurationsdatei zulässig und notwendig.

```
CPU 68008          // CPU Motorola MC 68008 (8 MHz)
CPU 68000          // CPU Motorola MC 68000 (16 MHz)
```

CLOCK

Mit diesem Eintrag kann optional man die von der CPU vorgegebene Taktfrequenz überschrieben werden. Die Angabe erfolgt in Hertz.

```
CLOCK 4000000      // 4 MHz Taktfrequenz der CPU
```

Auf einem MacBook Pro M1 liegt die maximale Emulationsgeschwindigkeit bei ca. 300 MHz. Zu niedrige Werte sollte man nicht angeben, Werte unter 100 kHz sind nicht sinnvoll.

Speicherkonfiguration

MEMORY

Legt abweichend von der durch die CPU vorgegebenen maximalen Speicherbereich eine andere Größe des Hauptspeichers fest. Die Angabe erfolgt in kByte.

```
MEMORY 1024        // 1 MByte adressierbarer Speicherbereich
```

Die Angabe von MEMORY ist nicht notwendig, je nach gewähltem Mikroprozessor erfolgen sinnvolle Vorgaben, die jeweils den maximalen Adressbereich für diese CPU im NKC umfassen.

ROM

Definiert einen Adressbereich als Read Only Memory. Es müssen zwei Parameter für den Adressbereich und ein Parameter für den Namen der Image-Datei mit dem Inhalt des ROM

folgen. Die beiden Adressen können dezimal oder hexadezimal angegeben werden. Die Image-Dateien werden im Unterverzeichnis ROMS des Emulators erwartet.

```
ROM 0x000000 0x007FFF EASS43.ROM      // GP ab Adresse 0x00000
ROM 0x0C0000 0x0C7FFF EASS43.ROM      // GP ab Adresse 0xC0000
```

Es dürfen beliebig viele Zeilen zur Definition der verschiedenen ROM-Blöcke benutzt werden. Werte, die außerhalb des physikalischen Adressraums der CPU liegen, werden ignoriert. Später definierte ROM-Blöcke können vorherige Definitionen überschreiben.

RAM

Definiert einen Adressbereich als Random Access Memory. Es müssen zwei Parameter folgen, welche die Startadresse und Endadresse des Speichers angeben. Die beiden Adressen können dezimal oder hexadezimal angegeben werden.

```
RAM 0x008000 0x009FFF                 // 8 kByte RAM ab Adresse 0x8000
RAM 0x000000 0x0BFFFF                 // 768 kB RAM ab Adresse 0x00000
```

Es dürfen beliebig viele Zeilen zur Definition der verschiedenen RAM-Blöcke benutzt werden. Werte, die außerhalb des physikalischen Adressraums der CPU liegen, werden ignoriert. Später definierte RAM-Blöcke können vorherige Definitionen überschreiben.

BOOTROM

Wie ROM, jedoch für die Baugruppe BANKBOOT. Der maximale Adressbereich liegt zwischen 0x0000 und 0x7FFF, da die Baugruppe BANKBOOT genau 32 kByte Speicherbereich bietet.

```
BOOTROM 0x0000 0x1FFF BOOT.ROM        // Boot ROM ab Adresse 0x0000
```

BOOTRAM

Wie RAM, jedoch für die Baugruppe BANKBOOT. Auch hier liegt der zulässige Adressbereich zwischen 0x0000 und 0x7FFF.

```
BOOTRAM 0x2000 0x7FFF                 // RAM auf BANKBOOT ab Adresse 0x2000
```

Modifizieren geladener Speicherbereiche

PATCHBYTE / PATCHWORD / PATCHLONG

Dient zum Patchen bereits geladener ROM-Bereiche. Es ist zu beachten, dass die Adressen geändert werden müssen, wenn die ROMs in abweichende Speicherbereiche geladen werden.

Beispiele:

```
PATCHWORD 0x003432 0x000A     // schnellere Copyright Meldung
PATCHWORD 0x0030BC 0x0011     // 80 Zeichen pro Zeile im Editor
PATCHLONG 0x000000 0x00F000   // Abweichender initialer Stack-Pointer
```

PATCHNOP

Dient zum Patchen bereits geladener ROM-Bereiche mit No-Operation-Befehlen (NOP). Als zweiter Parameter wird die Anzahl der zu speichernden NOP-Befehle angegeben. Jeder NOP-Befehl belegt 2 Bytes des Speichers.

```
PATCHNOP 0x00342C  5          // GP 4.3 kein Copyright ausgeben
```

PATCHBOOTBYTE / PATCHBOOTWORD / PATCHBOOTLONG

Wie oben, jedoch für die Baugruppe BANKBOOT

PATCHBOOTNOP

Wie oben, jedoch für die Baugruppe BANKBOOT

Konfiguration der Baugruppen

Jede Baugruppe darf aktuell nur einmal definiert werden. Die Angabe der Basisadressen ist immer optional, ohne Angabe wird die Standard-Adresse wie im NKC vorgesehen verwendet. Abweichende Basisadressen der Baugruppen führen in der Regel zu einer nicht korrekt funktionierenden Emulation, da die gesamte Software darauf nicht ausgerichtet ist.

BANKBOOT

Instanziiert den Hardware-Treiber für die Baugruppe BANKBOOT.

```
BANKBOOT 0xFFFFFFC8
```

Ohne Angabe des Treibers kann kein RAM-Speicher ab Adresse 0x0000 definiert werden und der Betrieb von zum Beispiel CP/M 68k ist nicht möglich. Wenn BANKBOOT angegeben wird, muss mindestens auch ein ROM ab Adresse 0x0000 mit einem BANKROM-Eintrag definiert werden.

KEY

Instanziiert den Hardware-Treiber für die Baugruppe KEY. In fast allen Konfigurationen wird die Baugruppe KEY benötigt, um das System überhaupt bedienen zu können.

```
KEY 0xFFFFFF68
```

GDP64K

Instanziiert den Hardware-Treiber für die Baugruppe GDP64K. Es sind 0 bis 3 Parameter gemäß den nachfolgenden Beispielen möglich. Die Standard-Werte für die Vergrößerung sind 2 für horizontal und 2 für vertikal. Daraus ergibt sich eine Größe des Monitor-Fensters von 1024*512 Pixel.

Für den korrekten Betrieb der Emulation mit den Grundprogrammen muss die Baugruppe GDP angemeldet sein. Ohne GDP erscheint nur das Hauptfenster und der Monitor lässt sich nicht aktivieren.

```
GDP64K                  // Standardeinstellung
GDP64K 0xFFFFFF70       // abweichende Basisadresse
GDP64K 2 3              // geänderte Vergrößerung 2x3
GDP64K 2 3 0xFFFFFF70   // abweichende Adresse und Vergrößerung
```

Bei einer Vergrößerung von 2 horizontal und 3 vertikal ergibt sich ein weitestgehend dem NKC entsprechendes Seitenverhältnis. Mir persönlich gefällt die vorgegebene Auflösung besser.

GDPFONT

Ermöglicht das Austauschen des im Grafikprozessor EF9366 integrierten Zeichensatzes. Ohne Angabe wird der originale Zeichensatz verwendet.

```
GDPFONT 0               // Originaler Zeichensatz
GDPFONT 1               // Verbesserter Zeichensatz
```

FLO2

Instanziiert den Treiber für die Baugruppe FLO2. Dieser Treiber stellt vier Laufwerke zur Verfügung, die mit Image-Dateien aus dem Unterverzeichnis DISKS geladen werden können.

Aktuell werden nur die ersten beiden Laufwerke im Mini-Format unterstützt.

```
FLO2
FLO2 0xFFFFFFC0                 // Angabe der Basisadresse
```

CAS

Instanziiert den Treiber für die Baugruppe CAS zum Speichern von Dateien auf einem externen Kassettenrekorder.

```
CAS
CAS 0xFFFFFFCA                  // Angabe der Basisadresse
```

PROMER

Instanziiert den Treiber für die Baugruppe PROMER zum Programmieren von EPROMS.

```
PROMER
PROMER 0xFFFFFF80               // Angabe der Basisadresse
```

CENT

Instanziiert den Treiber für die Baugruppe CENT zur Ansteuerung von Druckern.

```
CENT
CENT 0xFFFFFF48                 // Angabe der Basisadresse
```

UHR

Instanziiert den Treiber für die Baugruppe UHR zum Ermitteln von Datum und Uhrzeit.

```
UHR
UHR 0xFFFFFFFE                  // Angabe der Basisadresse
```

Bedienelemente

Einige der Baugruppen stellen Bedienelemente zur Verfügung, die im Hauptfenster der Emulation dargestellt werden können. Die Reihenfolge der Bedienelemente kann unabhängig von den Treibern eingestellt werden.

```
FRAMES FLO2 CAS PROMER CENT     // Reihenfolge der Bedienelemente
FRAMES CPU PROMER CAS
FRAMES FLO2 CENT
```

Die Angabe von FRAMES ist essenziell für die Bedienung des Emulators. Ohne Bedienelemente lassen sich die meisten Funktionen des Emulators und der geladenen Baugruppen nicht korrekt einsetzen. Schreibfehler oder nichtexistierende Bedienelemente werden vom Emulator ignoriert.

Die folgenden Bedienelemente sind aktuell verfügbar

```
FLO2        - Laden und auswerfen von Disk-Images
CAS         - Laden und Speichern von Kassetten-Dateien
```

PROMER - Programmieren von EPROMS

CENT - Druckerausgabe

CPU - Anzeige der Register der CPU

GDP - Anzeige der Register von GDP64K

Weitere Einstellungen

DISK

Dient zum automatischen Einlegen von Disketten-Images für die Baugruppe FLO2. Es muss das Laufwerk und die Image-Datei angegeben werden. Der Wertebereich für die Nummer des Laufwerks liegt zwischen 0 und 3, andere Angaben werden ignoriert.

```
DISK 0 Assembler.IMG        // Belegen des ersten Laufwerks
DISK 1 NKC1CPM68K.IMG       // Belegen des zweiten Laufwerks
```

Nicht existente Image-Dateien führen zu einem Fehler während der Emulation, ähnlich als sei keine Diskette in das betreffende Laufwerk eingelegt.

KEYS

Dient zum Hinterlegen einer Tastatursequenz, die sofort nach dem Start der Emulation in den Tastatur-Puffer übertragen wird. Dadurch ist es zum Beispiel möglich, bestimmte Menüpunkte des Grundprogramms automatisch aufzurufen

```
KEYS w cr w cr 4 cr         // Autostart CP/M im Grundprogramm 4.3
KEYS w cr 3 cr              // Anzeige der Bibliotheksprogramme
```

BP

Dient zum Definieren von beliebig vielen Breakpoints. Sobald die Instruktion an einer Adresse in der Liste der Breakpoints ausgeführt wurde, wird die Emulation in den Einzelschrittmodus versetzt und das Trace-Fenster mit der zuletzt ausgeführten Instruktion angezeigt.

Es können mehrere Schlüsselworte BP in einer Konfigurationsdatei verwendet werden.

```
BP 0x0033F4                 // 1 Breakpoint
BP 0x003C40 0x001012        // 2 Breakpoints
```

Mit Hilfe der weiteren Funktionen des Emulators kann der korrekte Ablauf von Programmen leicht kontrolliert werden. Die Ausführung des Programmes kann im Einzelschrittmodus weitergeführt werden, die Fortführung des automatischen Ablaufs ist jederzeit möglich.

NOAUTOSTART

Bestimmt, dass die Emulation nach der Konfiguration nicht automatisch gestartet wird. Dadurch ist es möglich, die Emulation ab dem Start im Einzelschrittmodus zu betreiben.

```
NOAUTOSTART
```

Die CPU muss manuell über den START-Button im Hauptfenster gestartet werden.

Vorgegebene Konfigurationen

Im Lieferumfang befinden sich mehrere vorgegebene Konfigurationsdateien, welche direkt über das Hauptfenster ausgewählt werden können. Die Auswahl einer neuen Konfiguration bewirkt einen sofortigen Neustart der Emulation. Eventuell nicht gespeicherte Dateien gehen verloren.

CP/M 68K 768k RAM

- CPU 68000 bei 40 MHz Taktfrequenz
- 768 kByte RAM ab Adresse 0x00000
- Grundprogramm 4.3 ab Adresse 0xC0000
- Baugruppen KEY, GDP64K, FLO2, PROMER, CAS und CENT
- Keine Einschaltmeldung, Autostart des CP/M

CP/M 68K mit GP 6.22

- Wie oben, jedoch mit Grundprogramm in der Version 6.22

CP/M 68K mit GP 7.10

- Wie oben, jedoch mit Grundprogramm in der Version 7.10

GP 4.3 Minimal

- CPU 68008 bei 8 MHz Taktfrequenz
- Grundprogramm 4.3 ab Adresse 0x00000
- 8 kByte RAM hinter Grundprogramm
- Baugruppen KEY, GDP64K, PROMER und CAS

GP 4.3 mit PASCAL

- CPU 68008 bei 8 MHz Taktfrequenz
- Grundprogramm 4.3 ab Adresse 0x00000
- 32 kByte RAM hinter Grundprogramm
- PASCAL/S ab Adresse 0x10000
- 32 kByte RAM hinter Pascal/S
- Baugruppen KEY, GDP64K, PROMER, CAS und CENT
- Keine Einschaltmeldung
- Automatisch 80 Zeichen pro Zeile

GP 4.3 mit RL-BASIC

- CPU 68008 bei 8 MHz Taktfrequenz
- Grundprogramm 4.3 ab Adresse 0x00000
- 32 kByte RAM hinter Grundprogramm
- RL-BASIC ab Adresse 0x10000
- 168 kByte RAM hinter RL-BASIC
- Baugruppen KEY, GDP64K, PROMER, CAS und CENT
- 32 kByte RAM hinter Pascal/S
- Keine Einschaltmeldung
- Automatisch 80 Zeichen pro Zeile

GP 6.22 mit Tools

- CPU 68008 bei 8 MHz Taktfrequenz
- Grundprogramm 6.22 ab Adresse 0x00000
- 64 kByte RAM ab Adresse 0x10000

- Bibliotheksprogramme DEMO, GOSI, BASIC, PASCAL
- Weitere RAM-Bereiche
- Baugruppen KEY, GDP64K, PROMER, CAS und CENT

GP 7.10 mit Tools

- CPU 68008 bei 8 MHz Taktfrequenz
- Grundprogramm 7.10 ab Adresse 0x00000
- 64 kByte RAM ab Adresse 0x10000
- Bibliotheksprogramme DEMO, GOSI, BASIC, PASCAL
- Weitere RAM-Bereiche
- Baugruppen KEY, GDP64K, PROMER, CAS und CENT

NKC DEMO

- CPU 68000 bei 16 MHz Taktfrequenz
- Grundprogramm 4.3 ab Adresse 0x00000
- 8 kByte RAM hinter Grundprogramm
- Bibliothek DEMO ab Adresse 0x0A000
- Baugruppen KEY, GDP
- Keine Einschaltmeldung
- Autostart der Demo
- Live-Anzeige der CPU-Register

Anhänge

Quellenangaben

Bücher und Hefte

Hauptsächliche Quelle für die Umsetzung der Hardware-Treiber waren die Bücher und Hefte, die von Rolf Dieter Klein über den Francis-Verlag herausgegeben wurden.

- Mikrocomputer selbstgebaut und programmiert ISBN 3-7723-7161-2
- Rechner modular ISBN 3-7723-8721-7
- Die Prozessoren 68000 und 68008 ISBN 3-7723-7651-7
- Anwenderhandbuch CP/M 68K ISBN 3-7723-9751-4

Zusätzlich die Datenblätter der im System verwendeten integrierten Schaltkreise.

- Motorola M68000 Microprocessors User's Manual
- Motorola M68000 Programmer's Manual
- Thomson EF9366 Data Sheet
- 6850 Data Sheet
- Western Digital FD 179X Floppy Disk Controller

68000 Emulation

Grundlage für die Emulation der 32 Bit Motorola Mikroprozessoren 68000 und 68008 war die Vorarbeit von Tony Headford, der eine gut funktionierende Emulation der Mikroprozessoren für Java umgesetzt und veröffentlicht hat.

- Projektseite https://github.com/tonyheadford/m68k

Es waren einige Fehlerbehebungen, Erweiterungen und Anpassungen notwendig, um eine möglichst gute Integration in diese Emulation des NKC zu sichern. Die modifizierten Stellen sind im Quellcode entsprechend markiert.

- Anpassung der Zyklen gemäß Motorola Manual
- Ableitung des 68008 Prozessors
- Zusätzliche Methoden der Cpu
  - getProcessorName()
  - getNominalFrequency()
  - getAddressMask()
- Anpassung der Klasse DisassembledInstruction()
- Verarbeitung der Unterprogramm-Ebenen

Hex Editor

Der integrierte Hex-Editor für den Hauptspeicher basiert auf dem Hexadecimal File Editor von Keith Fenske. Auch hier waren einige Erweiterungen notwendig, um dem Editor Zugriff auf den Hauptspeicher der Emulation zu gewähren.

- Projektseite https://kwfenske.github.io

Die zusätzlichen Funktionen wurden in einer eigenen Klasse umgesetzt, welche die originale Klasse erweitert und als Ersatz für die main() Methode dient.

Dateiformate

Baugruppe FLO2

Die Image-Dateien für die Baugruppe FLO2 sind jeweils in einer durchgehenden Binärdatei im Unterverzeichnis DISKS des Emulators abgelegt. Neben den reinen Daten sind keine weiteren Informationen in der Datei enthalten. Das Format der Image-Datei wird anhand der Dateigröße erkannt. Jede Diskette ist unterteilt in Seiten, Spuren und Sektoren mit einer bestimmten Anzahl an Bytes. Aktuell werden die folgenden Diskettenformate unterstützt:

NKC Mini Format – 5,25 oder 3,5 Zoll – 800 kByte Brutto

Dieses ist das standardmäßig im NKC verwendete Diskettenformat welches unter anderem auch für den Betrieb von CP/M 68K verwendet wird. Dieses Format wird von der Baugruppe FLO2 auf den ersten beiden Laufwerken A: und B: unterstützt.

- Spuren 80 pro Seite
- Seiten 2 - doppelseitig
- Sektoren 5 pro Spur
- Kapazität 1024 Bytes pro Sektor
- Imagedatei 2 x 80 x 5 x 1024 = 819200 Bytes = 800 kByte

Anhand des Beispielcodes kann man den Aufbau der Image-Dateien und die Reihenfolge der Sektoren nachvollziehen.

```
for (int track = 0; track < 80; track ++)
      for (int side = 0; side < 2; side++)
            for (int sector = 0; sector < 5; sector ++)
                  for (int data = 0; data < 1024; data++)
                        // Datenbyte schreiben oder lesen
```

NKC Maxi Format – 8 Zoll – 250 kByte Brutto

Dieses Format wurde auf älteren 8 Zoll Laufwerken verwendet, die nur einen Schreib- und Lesekopf besitzen. Die Disketten konnten beidseitig beschrieben werden, nachdem man sie andersherum eingelegt hatte. Das Format wird von der Baugruppe FLO2 auf den Laufwerken C: und D: unterstützt.

- Spuren 77 pro Seite
- Seiten 1 - einseitig
- Sektoren 26 pro Spur
- Kapazität 126 Bytes pro Sektor
- Imagedatei 1 x 77 x 26 x 128 = 256256 Bytes = 250 kByte

Anhand des Beispielcodes kann man den Aufbau der Image-Dateien und die Reihenfolge der Sektoren nachvollziehen.

```
for (int track = 0; track < 77; track ++)
      for (int sector = 0; sector < 26; sector ++)
            for (int data = 0; data < 128; data++)
                  // Datenbyte schreiben oder lesen
```

Baugruppe CAS

Die Emulation der Baugruppe CAS ermöglicht das Speichern von Daten, die an die Baugruppe gesendet werden und in realer Hardware auf einem Kassettenrekorder gespeichert worden wären. Ein Puffer speichert alle gesendeten Bytes inklusive der Synchronisationssequenzen. Beim Speichern wird der gesamte Puffer als Binärdatei im Unterverzeichnis CAS des Emulators gesichert. Diese Dateien können auch wieder geladen werden. Der Aufbau einer Aufzeichnung wird durch die Implementation in den Grundprogrammen vorgegeben.

Schreiben von Textdateien

• 40 Bytes	0xFF	Synchronisation
• 2 Bytes	0x00 0x2F	
• N Bytes	DATEINAME 0x0D	
• 40 Bytes	0xFF	Synchronisation
• 2 Bytes	0x00 0x3B	
• 4 Bytes	Startadresse	
• 4 Bytes	Endadresse	
• N Bytes	Daten	Nutzdaten
• 4 Bytes	Prüfsumme	
• 20 Bytes	0xFF	Ende

Schreiben von Datendateien

• 40 Bytes	0xFF	Synchronisation
• 2 Bytes	0x00 0x2F	
• N Bytes	DATEINAME 0x0D	
• 32 Bytes	0xFF	Synchronisation
• 1 Bytes	0x00	
• N Bytes	Text	Nutzdaten
• 1 Byte	0x00	Textende
• 2 Bytes	Prüfsumme	
• 20 Bytes	0xFF	Ende

Enthaltene Dateien

Unterverzeichnis ROMS

68008_EASS.BIN

Grundprogramm Version 4.3 vom Entwickler des Systems Rolf Dieter Klein als einzelnes 32 kByte Image mit grundlegenden Funktionen und einem integrierten Editor und Assembler. Das Grundprogramm erwartet mindestens 8 kByte RAM ab der Startadresse + 0x08000

68008_622X08X0.BIN bis 68008_622X08X6.BIN

Grundprogramm Version 6.22 von Ralph Dombrowski. Die Images müssen an fortlaufenden Adressen im Abstand von 8 KByte in den Hauptspeicher geladen werden. Das Grundprogramm erwartet mindestens 8 kByte RAM-Speicher ab der Startadresse + 0x10000.

68008_GRUND710R4#0 bis 68008_GRUND710R4#7

Grundprogramm Version 7.10 von Jens Mewes. Die Images müssen an fortlaufenden Adressen im Abstand von 8 KByte in den Hauptspeicher geladen werden. Das Grundprogramm erwartet mindestens 8 kByte RAM-Speicher ab der Startadresse + 0x10000.

68008_GOSI.BIN

Umsetzung der Grafisch Orientierten Sprache I (GOSI) in Form eines Bibliotheksprogramms. Zum Einsatz wird eines der Grundprogramme benötigt. Das ROM-Image hat einen Umfang von 24 kByte und kann an einer beliebigen freien Stelle des Hauptspeichers geladen werden.

68008_PASCAL.BIN

PASCAL/S ist eine Umsetzung des PASCAL Compilers der ETH Zürich von Rolf Dieter Klein. Das Bibliotheksprogramm besteht aus zwei Teilen, dem Compiler und der Run-Time-Umgebung. Quellcodes werden mit dem Editor des Grundprogramms eingegeben, danach mit PASCAL/S übersetzt und mit PCODE ausgeführt. Die beiden Bibliotheken belegen 32 kByte Speicher.

BASIC_68K.BIN

RL-Basic ist eine Umsetzung der Programmiersprache BASIC für den NKC. Nach dem Start sollte zunächst der Arbeitsbereich gesetzt werden, das das Programm nicht automatisch den größten freien RAM-Bereich erkennt.

DEMO11.BIN

Demonstration der grafischen Möglichkeiten des NKC von Rolf Dieter Klein. Das ROM kann an einer beliebigen freien Stelle des Hauptspeichers geladen werden und über die Bibliotheks-Funktion der Grundprogramme gestartet werden.

Unterverzeichnis BOOT

BOOTORIG.ROM

Dient zum Einsatz auf der Baugruppe BANKBOOT und sucht innerhalb des gesamten Speichers nach der Signatur des Grundprogrammes. Wenn es gefunden wurde wird BANKBOOT deaktiviert und das Grundprogramm gestartet. Zur Nutzung muss in der Konfiguration Hauptspeicher ab Adresse 0x0000 definiert werden. RAM ist auf der BANBOOT Baugruppe nicht notwendig.

Unterverzeichnis DISKS

Alle Diskettenimages können derzeit nur unter CP/M 68K genutzt werden.

NKC1CPM68K.IMG bis NKC8CPM68K.IMG

Diese Images beinhalten das eigentliche CP/M System, wie es im originalen Umfang für den NKC zusammengestellt wurde.

1. Startdiskette mit NKC spezifischen Programmen
2. Assembler und Debugger
3. Linker und Relocator
4. Terminal, Formatier-Programm
5. BIOS, CP/M Relocator
6. Winchester-Treiber
7. C-Compiler und Tools
8. CP/M Z80 Emulation

NKC1CPM68K.IMG

```
B: CPM      SYS : MAKECPM  SUB : MAKE128  SUB : MAKE256  SUB : MAKE512  SUB
B: RELCPM   SUB : CPMLDR   SYS : MAKELDR  SUB : UFO68008 68K : UFO68000 68K
B: UFO68020 68K : PIP      68K : COPY     68K : STAT     68K : XPUTBOOT 68K
B: LADE     68K : SAVE     68K : EDITRDK  68K : ASSRDK   68K : STARTE   68K
B: LADE     S   : SAVE     S   : EDITRDK  S   : ASSRDK   S   : STARTE   S
B: NDRBOOT  S   : XPUTBOOT S   : NDRLDRB  S   : NDRBIOS  S   : UFO680XX S
B: MODEM    S   : MIKRODOS S   : LIESMICH     : MAK1CPM  SUB :
```

NKC2CPM68K.IMG

```
B: AS68     REL : RELOC    REL : PIP      REL : INIT     REL : COPY     REL
B: STAT     REL : RELOC1   SUB : README   TXT : DDT      REL : DDT68000 68K
B: CPM      SYS : AS68INIT     : DUMP     REL : DDT10    REL : DDT68010 68K
B: RELOC2   SUB
```

NKC3CPM68K.IMG

```
B: LO68     REL : S        O   : CLIB         : LIBE     A   : LIBF     A
B: CLINK    SUB : CLINKE   SUB : CLINKF   SUB : RELOC3   SUB : ED       REL
B: CP68     REL : ASSERT   H   : CTYPE    H   : ERRNO    H   : OPTION   H
B: OSIFERR  H   : PORTAB   H   : SETJMP   H   : SIGNAL   H   : STDIO    H
B: AR68     REL : FIND     REL : MORE     C   : MORE     REL : NM68     REL
B: OSATTR   H   : OSIF     H   : RELOC4   SUB : C        SUB : CE       SUB
B: C068     REL : C168     REL : RELOC5   SUB
```

NKC4CPM68K.IMG

```
B: LINK68   REL : LOADR    O   : OVHDLR   O   : SENDC68  REL : SIZE68   REL
B: TERM     C   : TERMA    S   : TERM     REL : XFER86   C   : XFER86   REL
B: FORMAT   S   : FORMAT   REL : INIT     S   : RELOC6   SUB : CONFIG   C
B: CONFIG   REL : BDOS     S   : BIOS     C   : BIOS     O   : BIOSA    S
B: BIOSA    O   : BIOSTYPS H   : CBIOS    S   : ELDBIOS  S   : ERGBIOS  S
B: LDBIOSA  S   : LDBIOSA  O   : LDBIOS   O   : LOADBIOS H   : LOADBIOS SUB
B: NOBIOSHI S   : NOBIOSLO S   : NORMBIOS H   : NORMBIOS SUB : PUTBOOT  S
B: PUTBOOT  REL : VT52     C   : VT52     O   : BOOTER   S   : BOOTER   O
B: RELOC7   SUB
```

NKC5CPM68K.IMG

```
B: CPMLIB       : LDRLIB       : CPM      REL : CPM15000 MAP : CPM15000 SR
B: CPMLDR   SYS : LCPM     SUB : LCPM10   SUB : MAKELDR  SUB : RELCPM   SUB
B: RELOC8   SUB : XPUTBOOT S   : CPM400   SR  : XPUTBOOT REL : XBOOTER  S
B: XBOOTER  O   : XBIOS    C   : XBIOSA   S   : XBIOSA   O   : XLDBIOSA S
B: XLDBIOSA O   : XLOADBIO H   : XNORMBIO H   : XFLDBIOS H   : XFNMBIOS H
B: XNORMBIO SUB : XMAKELDR SUB : XLCPM    SUB : XBIOS    O   : XCPM     SYS
B: XCPMLDR  SYS : RELOC9   SUB : XCPM     REL : CPM400   MAP : XLOADBIO SUB
```

NKC6CPM68K.IMG

```
B: C068     REL : LIBF     A   : LIBE     A   : WIFORM68 A68 : WI68BIOS S
B: WIFORM68 68K : WIFORM68 S   : LIESMICH BAK : LIESMICH TXT : C168     REL
```

NKC7CPM68K.IMG

```
B: PIP      68K : STAT     68K : EDITRDK  68K : SIZE68   68K : RELOC    68K
B: CLINK    SUB : LOADR    O   : CPM      SYS : C068     68K : C168     68K
B: CP68     68K : CLINKF   SUB : C        SUB : CE       SUB : MORE     C
B: ERRNO    H   : LO68     68K : SETJMP   H   : SIGNAL   H   : STDIO    H
B: OSATTR   H   : MORE     O   : ASSERT   H   : XBIOS    O   : OSIF     H
B: S        O   : CTYPE    H   : LIBF     A   : OPTION   H   : OSIFERR  H
B: PORTAB   H   : AS68     68K : AS68SYMB DAT : CLIB         : AS68INIT
B: CLINKE   SUB : LIBE     A
```

NKC8CPM68K.IMG

```
B: READ     ASM : DU       DOC : REZ80    DOC : EMUIO    S   : MOVPAT   ASM
B: XSUB     COM : CRCGEN   COM : DDTZ     COM : DU       COM : @        COM
B: INIDIR   COM : REZ80    COM : GEMU     DOC : EMU      DOC : LOOP     TXT
B: READ     COM : DDTZ     DOC : SCOPY    COM : MOVPAT   BAT : TMT      COM
B: RCV      COM : CPMZ80   68K : EMUIO    DVR : GRUND4,3 68K
```

ASSEMBLER.IMG

Beinhaltet notwendige Anwendungen, um eigene Assembler-Programme zu erzeugen. Von diesem Image kann gebootet werden. Unter Anderem findet dich hier ein 68K Programm, um die Baugruppe PROMER direkt aus CP/M ansteuern zu können.

```
A: CPM      SYS : CPMLDR   SYS : AS68SYMB DAT : PROMER   LST : CLS      S
A: COPY     68K : PIP      68K : STAT     68K : RELOC    68K : MORE     68K
A: MAKE     SUB : PROMER   68K : LO68     68K : DUMP     68K : CLS      68K
A: FIND     68K : AS68     68K : EDIT     68K : PROMER   S   : MAKE2    SUB
```

Source-Code

Der Quellcode des Emulators ist derzeit nicht öffentlich verfügbar, es gibt noch zu viele offene Baustellen. Sobald die Programmierung und die Dokumentation abgeschlossen sind, werde ich den Quellcode auf meiner Homepage zur Verfügung stellen.