Megamodul kompilieren

Wie wird die Software fürs Megamodul erzeugt?

Die gesamte Modul-Software stammt von Alexander Schön. Ich habe etwas aufgeräumt und den Make-Prozess zusammengestellt. Damit kann man die Modulsoftware innerhalb kürzester Zeit komplett neu erstellt werden. Das umfasst das Packen der Daten mit bitbuster, das Kompilieren verschiedenster Programme einschließlich der Modulsoftware und das eigentliche Erstellen der ROM-Dateien. Das ganze funktioniert unter Windows XP in einem CMD-Fenster. Einzige zusätzliche Vorraussetzung zum Selbstkompilieren ist ein installiertes Perl.

Der allgemeine Ablauf

Das Quellcode-Archiv ist zu entpacken. Im Pfad zum Verzeichnis darf es keine Leerzeichen geben, da sonst der Make-Prozess nicht ordnungsgemäß funktioniert.

Beispiel: c:\hobby\z9001\megarom\....

Die ROM-Datei zum Testen in KCEMU von Torsten Paul wird erzeugt durch

```
make depend
make
make
```

Der zweite Durchlauf ist nötig, wenn sich die ROM-Zusammenstellung in packedroms.asm geändert hat.

Es wird eine Datei „super_new.bin“ erzeugt, die in die Ordnerstruktur des Emulators kopiert wird (nach KCemu\roms\z9001\mega_rom.851). Dann kann der neu erstellte ROM im Emulator getestet werden.

Fürs Brennen einzelner Chips für das Modul in Hardware muss die Datei „super_new.bin“ einfach nur der Länge nach in 3 Teile gesplittet werden. 1. und 2. Teil = 1MB, 3. Teil = 0,5 MB.

Aufräumen kann man mit

```
make clean
```

oder ganz doll mit

```
make clobber
```

Hinzufügen von weiteren Programmen

Weitere Programme können relativ leicht ergänzt werden, vorausgesetzt, der Platz im Modul reicht

noch.

Maschinencodeprogramme mit einem Speicherplatz unterhalb 0C000h können als Assemblerquelltext *.ASM, als Tape-Archiv *.TAP oder als reine Binärdatei (Speicherabzug) *.BIN vorliegen. Der Makeprozess macht aus all diesen Formaten eine bitbuster-gepackte Datei *.PCK.

Diese Datei ist zum einen physisch ins Mega-Modul einzubinden, zum anderen in die Menü-Struktur. Erstes erfolgt durch Eintrag in die Datei packedroms.asm. Vor der Marke „hier“ ist einfach eine neue Zeile analog der vorhandenen zu ergänzen. Die 3 Parameter des Makros sind der Dateiname der gepackten Datei, die Anfangsadresse des Programms im Speicher und eine ID-Nummer. Diese wird einfach hochgezählt.

```
addfile "vp/graf.bin.pck" ,08D60h, 316
```

Der Eintrag ins Menü erfolgt durch Hinzufügen eines Eintrags in der passenden Menü-Datei im Verzeichnis bank0. Auch hier wird nur ein Makroeintrag an der passenden (alphabetisch sortierten) Stelle erwartet:

```
addProc "GRAF ", mc_prog, b_316, 0ffffh, p_316
```

Das Beispiel steht in der Datei bank0\treiber.asm. Die Makro-Parameter sind der OS-Programmname (max. 8 Zeichen), der Dateityp (hier ist die Konstante mc_prog zu nutzen), die Banknummer (verbirgt sich hinter b_+vergebene ID-Nummer), die Startadresse des Programms (hier 0FFFFh, da der Treiber nicht gestartet wird), und die Programmnummer (p_ + vergebene ID-Nummer).

Jetzt muss der Make-Prozess gestartet werden und ein weiteres Programm ist ins Megamodul aufgenommen worden!

Basic-Programme sind etwas komplizierter einzurichten. Dafür wird man von einem automatisch startendem BASIC-Programm verwöhnt! Das BASIC-Programm muss in den Emulator geladen werden. Ein echter KC mit ROM-BASIC tut's natürlich auch! Dann ist der Speicherbereich von 300h bis DEEK(983)-1 als Binärdatei zu speichern. 983=3D7h ist SVARPT, der Pointer auf das Ende des BASIC-Programms. Somit wird das BASIC-Programm zusammen mit dem initialisierten Speicherzellen von BASIC gespeichert. Diese Binärdatei (oder *.TAP geht auch wieder) wird wie bei Maschinencodeprogrammen ins Modul eingebunden: Erstens in packedroms.asm. Hier kann das Makro addfile genommen werden. Alternativ gibt es addbasicfile, dass die Anfangsadresse von 300h automatisch einträgt:

```
addbasicfile "basicfiles/sumeria.bin.pck", 35
```

Als zweites ist das Programm in der gewünschten Menüdatei einzutragen. Hier ist das bank0\spiele4.asm:

```
addProc "SUMMERIA", BASIC_prog, b_35, 002fbh, p_35
```

Der Aufbau ist analog zu Maschinencodeprogrammen. Als Dateityp ist BASIC_prog einzutragen. Dadurch schaltet das Megamodul nach dem Laden des BASIC-Programms die Bank1 mit dem BASIC zu und startet ein Miniprogramm auf Adresse 2fbh, was das BASIC startet (WBASIC) und noch das Drücken der RUN-Taste simuliert, so dass die Basic-Programme gleich starten. Deshalb muss als

Startadresse bei addProc auch immer 02FBh eingetragen werden.

ROM-Programme mit einem Speicherplatz über 0C000h werden nicht komprimiert. Sie bekommen einfach eine eigene Bank. In der Bank sollte ein kleiner Code untergebracht werden, der beim Beenden des Programms die Bank 0 durch einen OUT-Befehl wieder einschaltet, möglichst sogar per OS-Erweiterung '#'. Das schaut man sich beispielsweise in bank1.bin oder bank5.asm an. Eine neue Bank wird einfach als 10K große Binärdatei in packedroms.asm per BINCLUDE hinzugefügt (ziemlich weit oben, beim Label startbank). Die Programme der neuen Bank sollten einzeln mit addProc in den Menüdateien ergänzt werden.

Wie arbeitet das Mega-Modul?

Das Megamodul arbeitet mit einem normalen Z9001-OS zusammen. Es muss nichts am System verändert werden. Nach RESET wird die Bank 0 zugeschaltet. Ab Adresse 0C000h steht jetzt ein riesiger Kommandonamensbereich mit allen über 300 startbaren Programmen des Moduls. Das Z9001-OS durchsucht mit der Funktion CPROM beim Aufruf eines Kommandos den Speicher nach dem Kommandonamen und startet das Kommando. Anstelle des direkten Startens wird im Megamodul eine Routine angesprungen, die den bitbuster-Entpacker in den Speicher nach Adresse 0082h lädt, die Bank zuschaltet, in der das Kommando steht und den Entpacker startet. Dieser entpackt das Programm direkt aus dem Mega-ROM an die Zieladresse im Hauptspeicher. Anschließend startet er das Kommando auf der bei addproc angegebenen Startadresse.

Im Megamodul gibt es auch das Systemerweiterungskommando '# '. Wenn der Rechner neu gestartet wurde, erfolgt das Anzeigen des Modul-Bootscreens. Danach wird einfach ins Z9001-OS gesprungen (GOCPM).

Die grafische Menü-Software im Modul dient „lediglich“ zum komfortablem Auswählen des Kommandos.

Damit ist das Modul unabhängig vom konkreten Ausbauzustand des Z9001, seien es RAM-Module, Zusatzmodule oder sogar individuelle OS-Versionen.

From:
https://hc-ddr.hucki.net/wiki/ - **Homecomputer DDR**

Permanent link:
https://hc-ddr.hucki.net/wiki/doku.php/z9001/module_sonstige/megamodul/megamodul2

Last update: **2013/06/12 08:31**