Application Note AN059

Flash-Eprom auf MODULAR-4/486 und Multi-COM

Autor: HB/JS Datei: AN059.DOC (8 Seiten)

Historie dieses Dokumentes

Datum	Auflage	Grund der Änderung
20.11.1998	4	Ergänzungen
28.08.1998	3	Umbau auch für Multi-COM, Rev. D und "kleine" MODULAR-4/486 erforderlich
17.08.1998	2	Korrektur
28.07.1998	1	Neu

Flash-Eprom

Auf den SORCUS-Karten "kleine"-MODULAR-4/486 (ML7), "große" MODULAR-4/486 (ML8) und Multi-COM (ML6) kann statt des standardmäßig ausgelieferten Eproms auch ein Flash-Eprom eingesetzt werden. Zur Zeit wird nur der Typ Am29F040 bzw. die neuere Version Am29F040B, beide mit 512 kByte, unterstützt.

SORCUS-Karte	Kurz bez.	Rev.	Umbau erford.	Flash-Eprom Typ	Gehäuse
Multi-COM	ML6	C	ja	Am29F040 *	DIL-32
		D	ja	Am29F040 *	DIL-32
"kleine" MODULAR-4/486	ML7	B	ja	Am29F040 *	PLCC-32
"große" MODULAR-4/486	ML8	C	ja	Am29F040 *	DIL-32

* oder Am29F040B

Umbau bei Multi-COM (ML6), Rev. C und D

Für ein Flash-Eprom Am29F040 oder Am29F040B müssen die Widerstände R25, R28 und R29 bestückt sein. R26, R27, R30 und R31 dürfen nicht bestückt sein. Das Eprom muß gegen ein Flash-Eprom vom Typ Am29F040 oder Am29F040B (DIL-32) ausgetauscht werden.
Alle bisher ausgelieferten Karten von Rev. C und Rev. D sind auch für den Betrieb mit Flash-Eprom geeignet.
Die Änderungen der Bestückung der Widerstände und des Eproms bzw. Flash-Eproms sind in das EEPROM der Basiskarte einzutragen.

Umbau bei "kleiner" MODULAR-4/486 (ML7), Rev. B

Für ein Flash-Eprom Am29F040 oder Am29F040B müssen die Widerstände R34 und R36 bestückt sein. R35, R37, und R38 dürfen nicht bestückt sein. Das Eprom muß gegen ein Flash-Eprom vom Typ Am29F040 oder Am29F040B (PLCC-32) ausgetauscht werden.
Die Änderungen der Bestückung der Widerstände und des Eproms bzw. Flash-Eproms sind in das EEPROM der Basiskarte einzutragen.

Umbau bei "großer" MODULAR-4/486 (ML8), Rev. C

Rev. B der Karte läßt sich nicht für das Flash-Eprom umbauen!

Für ein Flash-Eprom Am29F040 oder Am29F040B müssen die Jumper J7 folgendermaßen gesteckt sein: 2-4, 5-7, 8-10, 11-12. Das Eprom muß gegen ein Flash-Eprom vom Typ Am29F040 oder Am29F040B (DIL-32) ausgetauscht werden. Zusätzlich müssen die beiden GALs IC14 und IC15 gegen solche mit dem Design "ML8C-14B" bzw. "ML8C-15B" ausgetauscht werden.

Die Änderungen der Jumper, des Eproms bzw. Flash-Eproms und der GALs sind in das EEPROM der Basiskarte einzutragen.

Flash-Eprom Mechanismus

Flash-Eproms sind im System mehrfach programmierbare Speicher-Chips, die ihren Inhalt auch bei Abschalten der Stromversorgung nicht verlieren. Flash-Eproms tragen eine Kennung (Hersteller-ID und Device-ID), die, ebenso wie die Organisation des Flash-Eproms, per System-Call oder Makro-Befehl abgefragt werden kann. Die auf den SORCUS-Karten einsetzbaren Flash-Eproms sind Sektor-orientiert, d.h. die kleinste löschbare Einheit ist ein Sektor. Das Am29F040 und Am29F040B besitzt 8 Sektoren, die jeweils 64 kByte groß sind.
Sektoren können werksseitig auch gegen Löschen geschützt werden, z.B. der Bereich, in dem das Betriebssystem steht. Vor dem Beschreiben (=Programmieren) eines Flash-Eproms müssen alle Sektoren, in denen die neu zu beschreibende Teile liegen, gelöscht werden (auch wenn nur ein Byte neu programmiert werden soll, muß der gesamte Sektor gelöscht werden).
Wenn mehrere Flash-Eproms im System sind, die gelöscht und programmiert werden sollen, so kann das zeitlich überlappend geschehen. Alle Befehle bzgl. Flash beziehen sich immer nur auf den jeweils angegeben Chip (Steckplatz, IC-Nr.).

Stand-Alone Betrieb (Auto-Boot)

Programme, die nach einem Power-On oder nach einem Reset der Karte installiert werden sollen, müssen zusammen mit einer modifizierten .INS Datei und einem Loader Programm in das Flash der Karte programmiert werden. Hierzu steht ein sogenannter Flash-Assistent in SNW32 zur Verfügung. Dieser modifiziert eine "name.INS" Datei und schreibt diese und alle notwendigen Dateien in das Flash. Die Originaldatei "name.INS" bleibt dabei unverändert erhalten. Die modifizierte Datei steht als "name.INF" Datei zur Verfügung. Die "name.INS" Datei muß gemäß Anhang M der Handbücher erstellt sein (z.B. keine überflüssigen Leerzeichen)!

Voraussetzung für das Auto-Boot: Wenn der in nachfolgender Tabelle aufgeführte Jumper aufgesteckt ist, wird immer das Eprom-OsX geladen und der Auto-Boot-Vorgang gestartet.

SORCUS-Karte	Jumper für Auto-Boot
"große" MODULAR-4/486	St1: 7-8
"kleine" MODULAR-4/486	St1: 14-16
Multi-COM	St3: 14-16

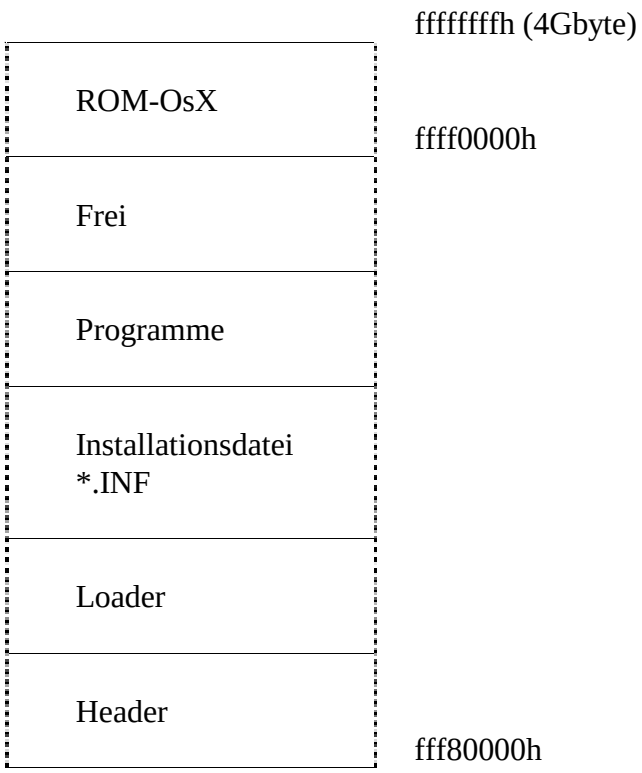
In diesem Fall kann kein RAM-OsX geladen werden. Hierzu muß der Jumper wieder entfernt werden.
Sollte eine fehlerhafte Auto-Boot Liste oder ein fehlerhaftes Programm programmiert worden sein und die Karte zum Absturz bringen, kann durch Entfernen des Jumpers wieder das MINI-OS aktiviert werden. Nun kann mit dem Assistenten eine neue Auto-Boot Liste in das Flash geladen werden.

Auto-Boot Mechanismus

Das ROM-OsX überprüft bei jedem Neustart, ob der Header des Flash gültig ist (das RAM-OsX ignoriert den Header). Der Flash-Header befindet sich am Anfang des Flash-Eproms.
Ist die Kennung im Header gültig und das Headerbyte 4 = ffh, dann wird ein installierbares Echtzeit-Program als Boot-Loader ab der im Header angegebenen Adresse erwartet. Das Programm wird in das RAM kopiert und als NI-Task installiert und gestartet. Dieses Loader-Programm arbeitet dann die modifizierte .INS Datei im Flash ab.
Ist das Headerbyte ungültig oder Headerbyte 4 ungleich ffh, wird normal im Betriebssystem weitergearbeitet.

Flash Speicheraufteilung

Die physikal. Länge (Anzahl Byte) des Flash-Eprom ergibt sich aus seinem Typ. Sie kann z.B. mit "FLASH_STATUS" gelesen werden. Bei ML6, ML7 und ML8 ist die physikal. Endadresse immer das Ende des 4GB Bereichs der Karte. Am oberen Ende des Flash-Eprom steht immer das Betriebssystem OsX. Der Beginn des Betriebssystems im Flash-Eprom liegt immer in den obersten 64k des Adreßbereichs und ist werkseitig schreibgeschützt, so daß dieser nicht gelöscht werden kann.
Nach der Programmierung des Flash mit Hilfe des Assistenten hat dieses folgenden Speicheraufbau:



Aufbau des Flash-Headers nach der Programmierung

Flash-Adresse	Inhalt	Bedeutung
00000h	55h aah aah 55h	gültige Kennung (4 Byte)
00004h	ffh	Valid-Byte: ffh = Flash ist gültig <> ffh = Flash ist ungültig
00005h	02h	Version dieser Flash-Definition
00006h	oL oH	Offset zum Installierungs-Record des Boot-Loaders (= 0030h, wenn x = 0)
00008h	nL nH	Länge (Anzahl Byte) des Boot-Loaders
0000ah	aE aF aG aH	Auto-Installierung: Anfangsadresse der Liste (ähnlich .INS-Datei)
0000eh	eE eF eG eH	Endadresse der Liste
00012h	uE uF uG uH	Freier User-Bereich: Anfangsadresse
00016h	vE vF vG vH	Endadresse
0001ah	bE bF bG bH	Betriebssystem: Anfangsadresse
0001eh	10 Byte	Datum (ASCII) tt.mm.jjjj der Programmierung dieses Flash (nicht User-Bereich)
00028h	8 Byte	Uhrzeit (ASCII) hh.mm.ss der Programmierung dieses Flash (nicht User-Bereich)
00030h	x Byte	ASCII-Text (Kommentar des Flash-Programmierers
00030h+x	22 Byte	Installierung-Record des Boot-Loaders (Tiny-Programm, Flag Bit 8,7,6 = 0) Die Adresse a in diesem Record ist die Anfangsadresse der PDT im Flash, bezogen auf den Anfang des Flash-Eproms. a steht bei Record-Adr. + 18. Die Länge des Records steht im 1. Byte des Records (z.Zt. = 22).
a		PDT des Boot-Loaders (Echtzeit-Programm)

Anm.: alle Adressen rel. zum Anfang des Flash-Eproms

Makro-Befehle für das Flash-Eprom

Code/Data	Funktion
2ah 44h	**FLASH_STATUS setzen bzw. lesen**
SP IC	SP=Modul-Steckplatz, IC=IC-Nr.
d p	d=Dienst (s.u.), p=Parameter

d	p	Funktion
0	0	**Hersteller-ID** des Flash-Eproms ermitteln
1	0	**Device-ID** des Flash (Device-ID) ermitteln
2	0	**Länge** Flash (L) und Sector (H) ermitteln: Länge (in Anzahl Byte) = 2**L bzw. 2**H
3	0	**Organisation** des Flash lesen (L=Org., H=0) L (untere 4 Bit) =0: Bit, =1: Byte, =2: Word L (obere 4 Bit) =0: kein Sector geschützt, =1: Bottom, =2: Top, H=0
4	s	**Check, ob Sektor s Write-Protected** L=0: Write-Protected, L=1: Not-Protected, H=0
5	s	**Status nach Erase von Sector/en s** L=0, H=0: Löschen fertig L=1, H=0: Löschen noch nicht fertig L=2, H=0: Fehler bei Löschen aufgetreten
10	0	**Lies Soft-State** (L = Soft-State, H=0)
11	0	**Setze "Erase enable"** (Soft-State 0 => 1)
-	-	Makro-Befehl: Erase Flash (Soft-State 1 => 2)
12	0	**Setze "Erase disable"** (Soft-State 0, 1 oder 2 => 0)
13	0	**Setze "Program enable"** (Soft-State 0 => 3)
-	-	Makro-Befehl "Programmiere Flash" (Soft-State bleibt => 3)
14	0	**Setze "Program disable"** (Soft-State 0 oder 3 => 0)

2ah e L H	**Antwort:** e=Error; L, H: Ergebnis je nach Dienst
2bh 4ah	**FLASH_ERASE: Erase Flash-EPROM direkt**
SP IC	SP=Modul-Steckplatz, IC=IC-Nr.
aE aF aG aH	Erste zu löschende Adresse (bzg. auf Anfang Flash)
eE eF eG eH	Letzte zu löschende Adresse (bzg. auf Anfang Flash)
2bh e x 0	**Anwort:** x = Nr. eines gelöschten Sectors
2ch 4dh	**FLASH_PROGRAM: Flash direkt programmieren**
nL nH	Anzahl Byte + 6 (n max. = 262)
SP IC	SP=Modul-Steckplatz, IC=IC-Nr.
rE rF rG rH	Relative Adresse des ersten zu progr. Bytes
d0...dn-6	
2ch e wL wH	**Antwort:** e = error, w = Anzahl verworfener Datenbyte

2dh d6h	**FLASH_READ: Flash direkt lesen**
mL mH	Anzahl zu lesender Byte (m max. = 256)
SP IC	SP=Modul-Steckplatz, IC=IC-Nr.
rE rF rG rH	Rel. Adresse des ersten zu lesenden Bytes
2dh e	**Antwort:** e = error
iL iH	i = Anzahl tatsächlich gelesener Bytes
d1...di	Data 1...i

Neue System-Routinen für Flash-Eprom

FLASH_STATUS (Nr. 104): Setze bzw. lies Status von Flash

Entry	bl	Dienst (Erkl. siehe Makro-Befehl FLASH_STATUS)
	bh	Parameter
	cl	Steckplatz (0 = Basiskarte) des Flash
	ch	IC-Nr. (ML6 und ML7 = IC6, ML8 = IC17)
Changed	f,eax,bx,cx,dx	
Exit (CY=0)	ax	kein Fehler, Ergebnis
Exit (CY=1)	ax	Fehlercode (s.u.)

FLASH_ERASE (Nr. 105): Lösche 1 oder mehrere Sektoren im Flash

Entry	eax	Erste Adresse (auf Flash-Anfang bezogen und auf Sektoranfang aligned)
	ebx	Letzte Adresse (auf Flash-Anfang bezogen und auf Sektorende aligned)
	cl	Modul-Steckplatz (0=Basiskarte) des angespr. Flash
	ch	IC-Nr. (ML6 und ML7 = IC6, ML8 = IC17)
Changed	f,eax,ebx	
Exit (CY=0)	al	Einer der Sektoren, der gelöscht wurde
	ah	0
Exit (CY=1)	ax	Fehlercode

FLASH_PROGRAM (Nr. 106): Programmiere Block im Flash

Entry	eax	Pointer auf Datenblock
	ebx	Erste Adresse im Flash (rel. Adresse)
	cx	Anzahl Byte zu programmieren (max. 65535)
	dl	Modul-Steckplatz (0=Basiskarte) des angespr. Flash
	dh	IC-Nr. (ML6 und ML7 = IC6, ML8 = IC17)
Changed	f,eax,ebx,cx,dx	
Exit (CY=0)	-	
Exit (CY=1)	ax	Fehlercode (s.u.)

FLASH_READ (Nr. 107): Lies Block aus Flash

Entry	eax	Pointer, wo die Daten hin sollen
	ebx	Erste Adresse im Flash (rel. Adresse)
	cx	Anzahl Byte zu lesen (max. 65535)
	dl	Modul-Steckplatz (0=Basiskarte) des angespr. Flash
	dh	IC-Nr. (ML6 und ML7 = IC6, ML8 = IC17)
Changed	f,eax,ebx,cx,dx	
Exit (CY=0)	-	
Exit (CY=1)	ax	Fehlercode (s.u.)

Neue Fehlercodes

84h Falscher Soft-State (Flash-Eprom) oder ein anderer Flash-Eprom Befehl läuft noch
85h Eingesetztes Flash-Eprom wird nicht unterstützt
86h Falscher Dienst (Flash-Eprom)
87h Kein Flash-Eprom lt. EEPROM
88h SP (Steckplatz) oder IC-Nr. falsch
89h: Parameter nicht aligned, z.B. n ungerade bei Wort, auch bei Adresse a oder e nicht auf Sectorgrenze eines Flash-Eprom
8ah Adresse falsch
8bh Flash bzw. Sector nicht gelöscht
8ch Flash nicht programmierbar/löschbar (Time-Out bzw. defekt)