

MP35-3

80535 Controller Modul mit Flash-Memory

1. FUNKTION	2
1.1. DATENBLATT	2
1.1.1. Anwendung	2
1.1.2. Daten	2
1.1.3. Besonderheiten	2
1.1.4. Aufbau	2
1.1.5. Stromversorgung	2
1.2. BLOCKDIAGRAMM	3
1.3. BESCHREIBUNG	3
1.4. ANSCHLUßSIGNALE	4
2. BETRIEB	6
2.1. KONFIGURIERUNG	6
2.1.1. Jumper (fett=vorverdrahtet):	6
2.2. SPEICHERBELEGUNG	6
2.2.1. Setup 0 (S0)	6
2.2.2. Setup 1 (S1)	6
2.2.3. Setup 2 (S2)	7
2.2.4. Setup 3 (S3)	7
2.3. BEDIENUNG	8
2.3.1. Reset	8
2.3.2. Monitor MON51	8
2.3.3. Autostart	10
2.4. PROGRAMMIERUNG	10
2.4.1. Extern	10
2.4.2. Download	10
2.4.3. Temporär im RAM	10
3. FERTIGUNG	11
3.1. MECHANIK	11
3.1.1. Stecker (S1)	11
3.2. ELEKTRONIK	12
3.2.1. Schaltbild	12
3.2.2. Bestückungsplan	13
3.2.3. Stücklisten	13
3.2.4. Kondensatoren für Powerup-Reset	13
3.3. PLDS	13
3.3.1. PAL-Listing MP35_2	13
4. MODIFIKATIONEN	15
4.1. VERSIONEN	15
4.1.1. Vers. 0:	15
4.1.2. Vers. 1:	15
4.1.3. Vers. 2:	15

5. ANHANG.....	16
5.1. BAUSTEINUNTERLAGEN.....	16
5.1.1. <i>Schaltbild</i>	16
5.1.2. <i>Stückliste</i>	16
5.1.3. <i>Bestückungsplan</i>	16
5.1.4. <i>Controller 80C535</i>	16
5.1.5. <i>Memory 62256</i>	16
5.1.6. <i>Flash EPROM Am29F010</i>	16

1. FUNKTION

1.1. Datenblatt

1.1.1. Anwendung

Universelles Controller-Modul mit integriertem RS232 Interface, 5 x 8 Bit Ports, 3 Kanal Timer und 8 x 10 Bit ADC.

Verwendung für eigenständige Messdatenerfassung und Steuerung.

1.1.2. Daten

Parameter	Wert	Bemerkung
Controller	80C535	
RAM	32KB	..64KB erw.bar
Flash-EPROM	128KB	
Systemclock	12MHz	(bzw. 11,0592MHz)
Befehlszeiten	$\geq 1\mu\text{s}$	meiste Befehle

1.1.3. Besonderheiten

- Vordekodierung von 8 Select Leitungen.
- PAL zur flexiblen Einstellung von Steuerleitungen
- Status LED
- 4 Setups (Pages) programmierbar

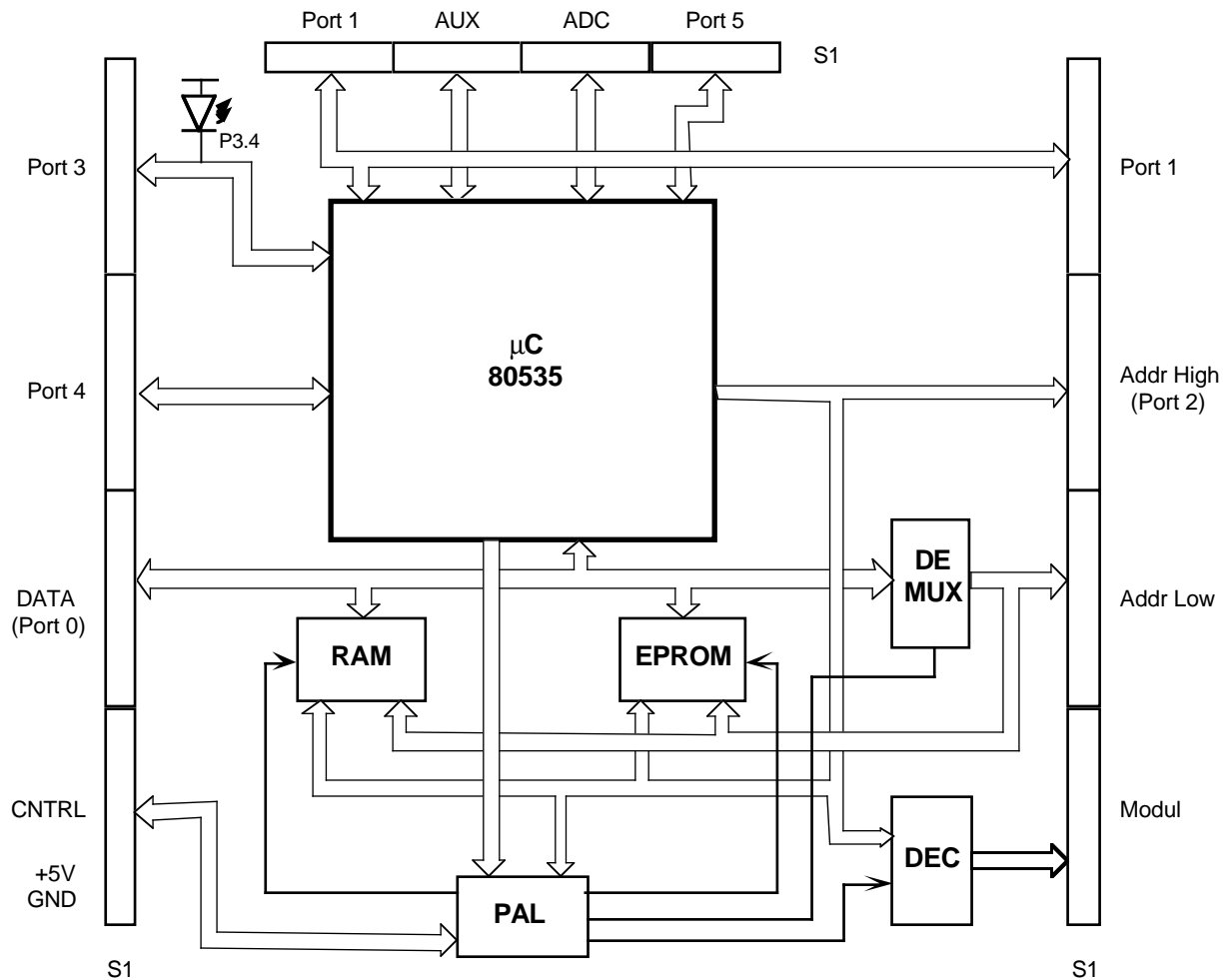
1.1.4. Aufbau

Scheckkartengröße 53 * 83 mm mit seitlichen Busstiften. Frontstecker für zusätzliche Signale.

1.1.5. Stromversorgung

Spannung	Strom	Leistung
+5V	ca. 120mA	600mW
Gesamt		600mW

1.2. Blockdiagramm



1.3. Beschreibung

Ein **Single-Chip-Controller** (80535) aus der 8051 Familie stellt alle grundsätzlichen I/O Funktionen zur Verfügung. Ein statisches **RAM** realisiert einen 32KByte großen Arbeitsspeicher. Die Programme sowie weitere Daten (nichtflüchtig) können in einem **Flash-EPROM** untergebracht werden.

Alle Ports (8 Bits) des Controllers sind an externen Pins zur allgemeinen Verwendung herausgeführt:

Port 0: (bidirektional 8 TTLS). **Datenbus** (D0..7) und **LowAdressbus** (A0..7) gemultiplext. Auf dem Board befindet sich ein Demultiplexer (U3) um für normale Speicher auch diese Adressen separat zur Verfügung zu haben.

Port 1: (bidirektional 4 TTLS). Frei verwendbar oder alternativ Spezialfunktionen für Timer und Interrupt.

Port 2: (bidirektional 4 TTLS). **HighAdressbus** (A8..15).

Port 3: (bidirektional 4 TTLS).

Steuersignale RD (P3.7), WR (P3.6).

Serielle Schnittstelle TxD (P3.1), RxD (P3.0).

Setup (P3.2), (P3.4). Bei Beschränkung auf 1 Setup (anderes PAL) frei verwendbar.

Timer (P3.5) und **Interrupt** (P3.3). Frei verwendbar.

Port 4: (bidirektional 4 TTLS). Frei verwendbar.

Port 5: (bidirektional 4 TTLS). Frei verwendbar.

ADC: 8 Kanal ADC mit 8 Bit (durch Steuerung der Referenz bis 10Bit) Auflösung .

AUX: weitere Steuersignale des Controllerbausteins.

MODUL: Durch einen **Decoder** werden aus dem 32KB Addressbereich 8*4KByte große Fenster vordekodiert und als Select-Leitungen (Low active) zur Verfügung gestellt.

CNTRL: Hier liegen die Pins zur Stromversorgung (GND, +5V) sowie Signale zur Systemsteuerung (RES,INT). 4 weitere Signale (C0..C2, FA16) können über ein **PAL** programmiert und zur komplexeren Ansteuerung weiterer Komponenten verwendet werden.

Eine frei programmierbare **Status-LED** (Pin 3.3) auf dem Board kann z.B. zur Fehleranzeige verwendet werden.

1.4. Anschlußsignale

PIN	Symbol	Eing./Ausg	Funktion
1-3 5-9	P4.0-4.2 P4.3-4.7	E/A	Port 4 ist bitadressierbar, 8 Bit und bidirektional. Er kann 4 TTLS treiben.
4	VPD		Ruhestromversorgung für 40-Byte-RAM (58H bis 7FH). Ist VCC > VPD, wird dieser Bereich von VCC versorgt.
10	_RESET	E	Liegt dieser Pin während 2 Maschinentakten auf LOW, wird der 80515 zurückgesetzt.
11	VAREF		5V Referenzspannung des A/D-Wandlers
12	VAGND		0V Referenzspannung des A/D-Wandlers
13-20	AN7-AN0	E	Gemultiplexte Analog-Eingänge
21-28	P3.0-3.7	_E/_A	Port 3 ist bitadressierbar, 8 Bit und bidirektional. Er kann 4 TTLS treiben. Jedem Pin ist eine alternative Funktion zugeordnet. Wird diese benötigt, ist vorher das Ausgangs-Latch auf HIGH zu programmieren. Die Alternativfunktionen sind:
21	P3.0	E/A	RXD: serieller Port-Daten-Eingang (async), Ein-/Ausgang (sync).
22	P3.1	E/A	TXD: serieller Port-Daten-Ausgang (async), Takt Ausgang (sync).
23	P3.2	E/A	_INT0: Interrupt 0 Eingang, Timer 0 Gate-Eingang.
24	P3.3	E/A	_INT1: Interrupt 1 Eingang, Timer 1 Gate-Eingang.
25	P3.4	E/A	T0: Zähler 0 Eingang
26	P3.5	E/A	T1: Zähler 1 Eingang
27	P3.6	E/A	_WR: Steuersignal für externe Datenspeicher, es übergibt das Datenbyte an Port 0 zum externen Speicher.
28	P3.7	E/A	_RD: Steuersignal für externe Speicher. Es übernimmt das Datenbyte in Port 0.
29-36	P1.7-1.0	E/A	Port 1 ist bitadressierbar, bidirektional und 8 Bit breit. Er kann 4 TTLS treiben. Jedem Pin ist eine alternative Funktion zugeordnet. Wird diese benötigt, ist vorher das Ausgangs-Latch auf HIGH zu programmieren. Die Alternativfunktionen sind :
29	P1.7	E/A	T2: Timer 2 Eingang.
30	P1.6	E/A	CLKOUT: Systemtakt Ausgang.
31	P1.5	E/A	T2EX: externer Reload-Eingang.
32	P1.4	E/A	INT2: externer Interrupt 2.
33	P1.3	E/A	INT6/CC3: Interrupt 6- Capture 3 Eingang, Compare 3
34	P1.2	E/A	INT5/CC2: Ausgang.
35	P1.1	E/A	INT4/CC1: Interrupt 5- Capture 2 Eingang, Compare 2
36	P1.0	E/A	_INT3/CC0: Ausgang. Interrupt 4- Capture 1 Eingang, Compare 1

			Ausgang. Interrupt 3- Capture 0 Eingang, Compare 0 Ausgang.
37	VBB		Substrat Pin. Dieser Pin muß über eine Kapazität (10 bis 100 nF) an GND liegen.
39	XTAL2	E/A	Ausgang des Oszillatorverstärkers. Es ist ein Quarz, Keramikschwinger oder eine externe Taktquelle verwendbar.
40	XTAL1	E	Eingang des Oszillatorverstärkers. Der Pin muß auf Masse liegen, wenn an XTAL2 ein externes Taktsignal eingespeist wird.
41-48	P2.0-2.7	E/A	Port 2 ist bitadressierbar, 8 Bit breit und bidirektional. Er kann bis 4 TTLS treiben. Beim Anschluß externer Speicher gibt P2 das höherwertige Adreßbyte aus.
49	_PSEN	A	Ext. Programmspeicher-Enable Ausgang.
50	ALE	A	Address-Latch-Enable. Mit diesem Signal wird das niederwertige Adreßbyte extern zwischengespeichert.
51	EA	E	Bei HIGH führt die CPU Befehle vom internen Befehlsspeicher aus, wenn der Inhalt des PC kleiner als 8191 (1FFFH) ist. Bei LOW werden Befehle nur vom externen Befehlsspeicher ausgeführt. EA muß beim 80535 auf LOW gelegt werden.
52-59	P0.0-0.7	E/A	Port 0 ist bitadressierbar, 8 Bit breit und bidirektional. Er kann bis 8 TTLS treiben. Beim Anschluß externer Speicher gibt P0 zuerst das niederwertige Adreßbyte aus, dann folgt das Einlesen des Befehlsbyte bzw. das Ein- oder Ausgeben des Datenbyte.
60-67	P5.7-5.0	E/A	Port 5 ist bitadressierbar, bidirektional und 8 Bit breit. Er treibt 4 TTLS.
68	VCC		Spannungsversorgung (+5V)
38	VSS		Masse (0V)

2. BETRIEB

2.1. Konfigurierung

2.1.1. Jumper (fett=vorverdrahtet):

Betriebsart	Jumper	Bemerkung
freie Verwendung	J1: 1-2	Eingang an PAL I6=ALE
freie Verwendung	J1: 2-3	Eingang an PAL I6=P3.5
Externes ROM benutzen	J2: 1-2	_EA low für 80535 (normal!)
Internes ROM benutzen	J2: 2-3	EA für 80515 (ROM-Version)
Ruhestromversorgung f. 40B RAM	J3: 1-2	VPD an Vcc (Versorgung d. Vcc)
Ruhestromversorgung f. 40B RAM	J3: 2-3	VPD an GND (Versorgung d. Vcc)
freie Verwendung	J4: 1-2	Eingang an PAL I1=A14
freie Verwendung	J4: 2-3	Eingang an PAL I1=P3.3

2.2. Speicherbelegung

Das Flash-EPROM ist in 8 Segmente à 16KB (**F0..F7**) aufgeteilt.

Das RAM ist in 2 Segmente à 16KB (**R0..R1**) aufgeteilt.

Je nach Setup (**S0..S3**) sind die einzelnen Segmente im Memory unterschiedlich lokalisiert. Die Setups werden durch die PortBits [P3.4,P3.2] eingestellt.

2.2.1. Setup 0 (S0)

Start	CODE (PSEN)	DATA (RD WR)
	nur lesen	lesen&schreiben
\$C000	F3	R1
\$8000	F2	R0
\$7000	F1	M7
\$6000	F1	M6
\$5000	F1	M5
\$4000	F1	M4
\$3000	F0	M3
\$2000	F0	M2
\$1000	F0	M1
\$0000	F0	M0

Flash F0...F3 enthält das Programm und/oder fixe Daten.

RAM R0...R1 enthält Daten.

Module M0...M7 können angesprochen werden.

2.2.2. Setup 1 (S1)

Start	CODE (PSEN)	DATA (RD/WR)
	nur lesen	lesen&schreiben
\$C000	R1	R1
\$8000	F2	R0
\$4000	F1	F5
\$0000	F0	F4

Flash F0...F2 für Programme und/oder fixe Daten.

Flash F4...F5 für Programmierung (Beschreiben) und Daten.

RAM R0 für Daten.

RAM R1 für Daten und Programme (vonNeumann).

2.2.3. Setup 2 (S2)

Start	CODE (PSEN)	DATA (RD/WR)
	nur lesen	lesen&schreiben
\$C000	R1	R1
\$8000	F2	R0
\$4000	F1	F7
\$0000	F0	F6

Flash F0...F2 für Programme und/oder fixe Daten.

Flash F6...F7 für Programmierung (Beschreiben) und Daten.

RAM R0 für Daten.

RAM R1 für Daten und Programme (vonNeumann).

2.2.4. Setup 3 (S3)

Start	CODE (PSEN)	DATA (RD/WR)
	nur lesen	lesen&schreiben
\$C000	R1	R1
\$8000	R0	R0
\$4000	F7	F1
\$0000	F6	F0

Flash F6...F7 für Programme und/oder fixe Daten (typisch MONITOR).

Flash F0...F1 für Programmierung (Beschreiben) und Daten (Applikation).
RAM R0...R1 für Daten und Programme (vonNeumann).

2.3. Bedienung

2.3.1. Reset

Beim **Reset** ist immer der **Setup 3** eingeschaltet und ein Programm wird an der Code-Speicherstelle **\$0000** (F6) gestartet.

2.3.2. Monitor MON51

Zur Entwicklung von Programmen und zum weiteren Laden von Applikationen ist ein Monitor (**MON51**) im F6,F7-Segment bereits vorhanden, der bei Reset somit gestartet wird.

Als erstes überprüft MON51 ob die Leitung **INT** im LOW-Zustand ist. In diesem Fall wird immer in MON51 fortgesetzt.

Andernfalls überprüft MON51 dann die Adresse **\$0000** in **F0**. Falls die Daten dort ungleich **\$FF** sind, wird dort der Start eines lauffähigen Programms angenommen, welches nach Umschalten in den **Setup 2** gestartet wird. Das dort befindliche Programm übernimmt dann die Kontrolle (Autostart).

ACHTUNG: Da der Monitor im Augenblick der Setupumschaltung nicht mehr existent wäre, wird ein kleines (temporäres) Hilfsprogramm in das Segment R1 geladen, welches zum Umschalten und Starten des Hauptprogrammes verwendet wird.

Das Hauptprogramm muß selber zur vollständigen Kontrolle z.B. in den Setup 0 schalten!

Andernfalls wird in MON51 fortgesetzt.

Zum Start auf dem Terminal dann mehrmals die **SPACE**-Taste drücken, bis der Controller z.B. mit der Systemmeldung:

```
--- MON51 4.0 (9600 Baud @ 11,0592MHz) ---
```

antwortet. Damit erfolgt eine automatische Adaptierung an die verwendete Baudrate. Unterstützt wird bei Systemclock

12MHz: 1200, 2400, 4800 Baud

11,0592 MHz: 1200, 2400, 4800, 9600, 19200 Baud

Alle Kommandos sind in ASCII und werden bereits durch das Senden eines entsprechenden Zeichens (Buchstaben) ohne <CR> ausgelöst.

Parameter (Zahlen) werden vorangestellt und sind durch das Kommando abgeschlossen! Spezielle Kommandos (, ;) erlauben das Setzen bestimmter Parameter (Addr, Data, Size). Ansonsten ergibt sich die Bedeutung des Parameters aus dem Kommando. Die Eingabe kann auch in Hex mit vorgestelltem \$ erfolgen (z.B: \$A1 =161). Gesendete Parameter werden immer mit einem <CR> beendet.

Alle eingegebenen Zeichen werden grundsätzlich zurückgesendet (Echo).

Der Monitor beansprucht keinerlei RAM. Lediglich im Falle der Programmierung des Flash-Speichers wird ein kleiner 256 Byte großer Bereich (Scratch: normalerweise \$FF00...\$FFFF, einstellbar mit &!) beansprucht.

MON51 akzeptiert folgende Kommandos:

- ? Ausgabe aller Befehle in Kurzform
- a& Setzt Speicherbereich (ca. 100 Bytes) im RAM ab a für Flash-Hilfsroutinen.
- & Ausgabe Startadresse
- a. Setzt interne Adressvariable ADDR auf a (z.B.: \$1000.)
- . Ausgabe ADDR
- d, Setzt interne Datenvariable DATA auf d (z.B.: \$FF,)
- , Ausgabe DATA
- s; Setzt interne Grössenvariable SIZE auf s (z.B.: 20;)
- ; Ausgabe SIZE
- a+/- Addiert/Subtrahiert a zu ADDR (z.B.: 1+)
- : Lädt HEX listing und setzt ADDR , DATA, SIZE automatisch.
Daten, die auf \$0000..\$7FFF geladen werden sollen, werden zunächst nach \$8000 geladen und müssen dann mit dem Befehl M nach \$0000 (in F0 und F1) umgespeichert werden.
- aG Startet Programm an ADDR
- ag: Liest Daten aus dem CODE Bereich (PSEN) von Addr a (0..\$FFFF)
- dX Schreibt DATA d an ADDR (0..\$FFFF) in den externen RAM Bereich (WR)
- ax Liest Daten von ADDR a (0..\$FFFF) aus dem externen RAM Bereich (RD)
- dI Schreibt DATA d an ADDR (0..\$FF) in den internen RAM Bereich (Reg.Ind.Addressing)
- ai Liest Daten von ADDR a (0..\$FF) aus dem internen RAM Bereich (Reg.Ind.Addressing)
- dJ Schreibt DATA d an ADDR (0..\$FF) in den internen SFR Bereich (Dir. Byte Addressing)
- aj Liest Daten von ADDR a (0..\$FF) aus dem internen SFR Bereich (Dir. Byte Addressing)
- dP Setzt Port ADDR (0..5) auf DATA d (z.B.: 2.\$FFP)
- ap Liest Daten von Port ADDR a (0..5) (z.B.: 2p)
- dO Verknüpft DATA d mit Portzustand (OR) und setzt Port ADDR (0..5)
- do Verknüpft DATA mit Portzustand (AND) und setzt Port ADDR (0..5)
- aZ Löscht Segment (= \$FF) an ADDR a (0..\$FFFF) im Datensgment.
Um ein entsprechendes Flash-Segment zu löschen, muss dieses durch einen geeigneten Setup in den Databereich gebracht werden!
Im Monitor können unmittelbar nur die Segmente F0 und F1 gelöscht werden
- dM Kopiert Daten mit der Länge SIZE von ADDR nach DATA d. Wird zur automatischen Programmierung der Flash-Segmente F2...F3 (Setup 0) verwendet.
- am gib Daten mit SIZE ab ADDR a auf die serielle Schnittstelle.
- dT d<512: Schreibt Lo(DATA d) nach RAM:ADDR..ADDR+SIZE-1.
d>=512: Schreibt Hi(ADDR)+Lo(ADDR) nach RAM:ADDR..ADDR+SIZE-1.

dt d<256: Vergleicht RAM:ADDR..ADDR+SIZE-1 mit DATA d.
 d<512: Vergleicht CODE:ADDR..ADDR+SIZE-1 mit Lo(DATA d).
 d=512: Vergleicht RAM:ADDR..ADDR+SIZE-1 mit Hi(ADDR)+Lo(ADDR).
 d>512: Vergleicht CODE:ADDR..ADDR+SIZE-1 mit Hi(ADDR)+Lo(ADDR).

2.3.3. Autostart

Programme, die in F0.. geladen sind, werden bei Vorhandensein von **MON51** automatisch bei Reset gestartet.

ACHTUNG: Falls **MON51** nicht existiert, können auch Programme bei F0.. nicht gestartet werden, da der Controller immer im Setup 3 startet und somit ein Programm bei F6, F7 erwartet wird. In diesem Fall muß das Hauptprogramm in F6, F7 gespeichert sein!

2.4. Programmierung

2.4.1. Extern

Entsprechend Setup 3 muß immer ein Programm zunächst im Flash-Segment F6, F7 (Speicheradresse \$18000..) mit einer maximalen Größe von 32KB liegen. Das Flash-EPROM wird hier auf einem Programmiergerät programmiert. Typischerweise ist dies der Monitor **MON51**. Durch Segmentumschaltung kann dann ein weiteres Programm mit einer maximalen Größe von 64KB (F0..F3) gestartet werden.

2.4.2. Download

Programme, die nach F0, F1 geladen werden sollen, werden zunächst (automatisch) in den RAM-Bereich R0, R1 geladen und müssen anschließend mit dem Befehl **M** in die Flash-Segmente F2...F3 kopiert werden. Anschließend werden sie bei Reset durch den Monitor automatisch nach Umschaltung in **Setup 2** gestartet.

ACHTUNG: Es ist hier zu beachten, daß in diesem Fall ein kleiner RAM-Bereich (Scratch: normalerweise \$FF00...\$FFFF einstellbar mit &!) für Hilfsprogramme zur Programmierung des Flash-Speichers verwendet wird.

2.4.3. Temporär im RAM

Im Setup 3 ist der ganze Bereich von \$8000...\$FFFF (In Setup 1...2 nur der Bereich \$C000...\$FFFF !) als kombinierter DATEN&CODE Bereich (vonNeumann) verfügbar. Es können dort in **MON51** Programme geladen und ausgeführt (Befehl **G**) werden.

3. FERTIGUNG

3.1. Mechanik

3.1.1. Stecker (S1)

Vierkant-Stifte: Zu empfehlen bei Einbau in Geräten ohne Sandwich-Aufbau.

Wire-Wrap Buchsen/Stifte (Fischer: BL9): Sandwich-Aufbau

3.2. Elektronik

3.2.1. Schaltbild

(Anhang)

3.2.2. Bestückungsplan

(Anhang)

3.2.3. Stücklisten

(Anhang)

3.2.4. Kondensatoren für Powerup-Reset

10uF: Anstiegszeit 15ms; Abfallzeit 300ms

100uF: Anstiegszeit 20ms; Abfallzeit 300ms

1000uF: Anstiegszeit 150ms; Abfallzeit 300ms

3.3. PLDs

3.3.1. PAL-Listing MP35_2

```
module MP35_02
title 'vwa230798';
```

```
"inputs
```

```
MS,A14,P34,A15,PSEN,P32,I6,RES,RD,WR    PIN 1,2,3,4,5,6,7,8,9,11;
```

```
"outputs
```

```
FA15,OE          PIN 12,19;
```

```
C2,C1,C0        PIN 13,14,15;
```

```
CSB,FA16,CSA    PIN 16,18,17;
```

```
x = .X.;
```

```
Setup  = [P34,P32];
```

```
Setup0 = (Setup==0);
```

```
Setup1 = (Setup==1);
```

```
Setup2 = (Setup==2);
```

```
Setup3 = (Setup==3);
```

```
FSel   = [FA16,FA15];
```

```
F01    = [0,0];
```

```
F23    = [0,1];
```

```
F45    = [1,0];
```

```
F67    = [1,1];
```

```
CODE   = !PSEN;
```

```
DATA   = !RD # !WR;
```

```

CODESeg = !A15 & !Setup3;
PrgmSeg = A15 & Setup0 # A15 & !A14 & (Setup1 # Setup2);
MonSeg = !A15 & Setup3;
NeumSeg = A15 & A14 & !Setup0 # A15 & Setup3;
RAMSeg = A15;
FSeg = !A15 & !Setup0;
IOSeg = !A15 & Setup0;
LCDSeg = IOSeg & A14;
CSRam = CSA;
CSRom = CSB;

```

```

CODESel = CODESeg & CODE;
PrgmSel = PrgmSeg & CODE;
MonSel = MonSeg & CODE;
NeumSel = NeumSeg & (DATA # CODE);
RAMSel = RAMSeg & DATA;
FSel = FSeg & DATA;
LCDSel = LCDSeg & DATA;

```

Equations

```

!C0 = !PSEN # !RD;
!C1 = !WR;
C2 = LCDSel; "for LCD Display!
!CSRam = RAMSel # NeumSel;
!CSRom = CODESel # PrgmSel # FSel # MonSel;
!OE = IOSeg;
FSel = F01 & CODESel
      # F01 & FSeg & Setup3 & !CODE
      # F23 & PrgmSeg & !CODE
      # F45 & FSeg & Setup1 & !CODE
      # F67 & MonSel
      # F67 & FSeg & Setup2 & !CODE;

```

end

4. MODIFIKATIONEN

4.1. Versionen

4.1.1. Vers. 0:

1. Versuchsmuster
verschiedene Leitungen von Hand gepatched!

4.1.2. Vers. 1:

1. Layout entsprechend Vers. 0 geändert.

4.1.3. Vers. 2:

2. Setup (in PAL) so geändert, daß Programme bei \$0000 starten!

5. ANHANG

5.1. Bausteinunterlagen

5.1.1. Schaltbild

5.1.2. Stückliste

5.1.3. Bestückungsplan

5.1.4. Controller 80C535

5.1.5. Memory 62256

5.1.6. Flash EPROM Am29F010