Vademekum

für den

MCS51 – Anwender

Hardware

CPU

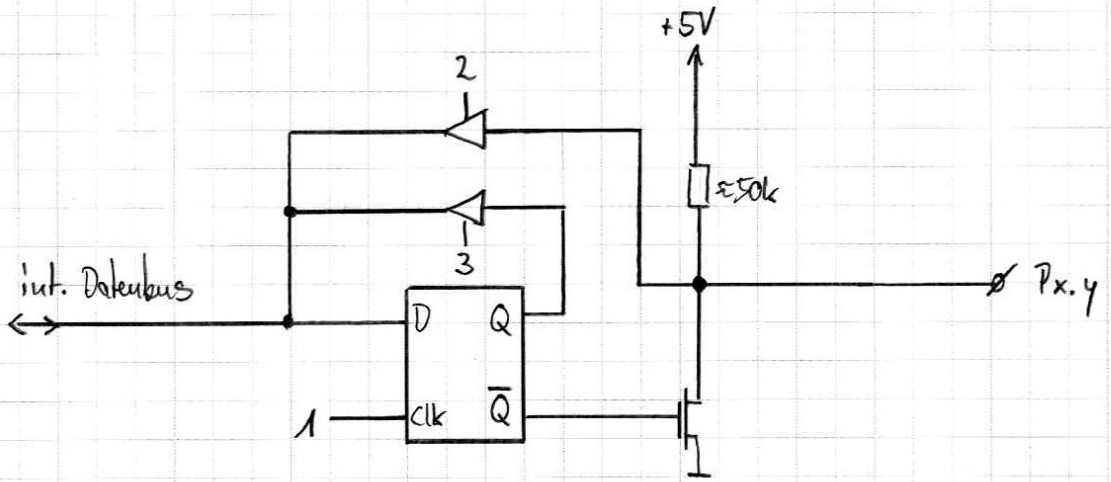
On-Chip-Peripherie

http://VolkerMüller.Info

Pin-Belegung

- Port 0 — 8 I/O — Adressen Low-Byte Daten
- Port 1 — 8 I/O
- Port 2 — 8 I/O — Adressen High-Byte
- Port 3 — 8 I/O — Sonderfunktionen
  - .0 RXD, .1 TXD } ser. Schnittstelle
  - .2 $\overline{INT0}$, .3 $\overline{INT1}$ } ext. Interruptquellen
  - .4 T0, .5 T1 } ext. Taktquellen für Zähler / Timer-Gates
  - .6 $\overline{WR}$, .7 $\overline{RD}$ } Read-/Write-Data-Strobe f. ext. Raum
- +5V, GND } Betriebsspannung
- XTAL1, XTAL2 } Taktquelle, z.B. Quarz
- RESTART
- ALE — Adress-Latch Enable
- $\overline{PSEN}$ — Program-Store Enable
- $\overline{EA}$ — externer Programmspeicher

I/O - Schaltung

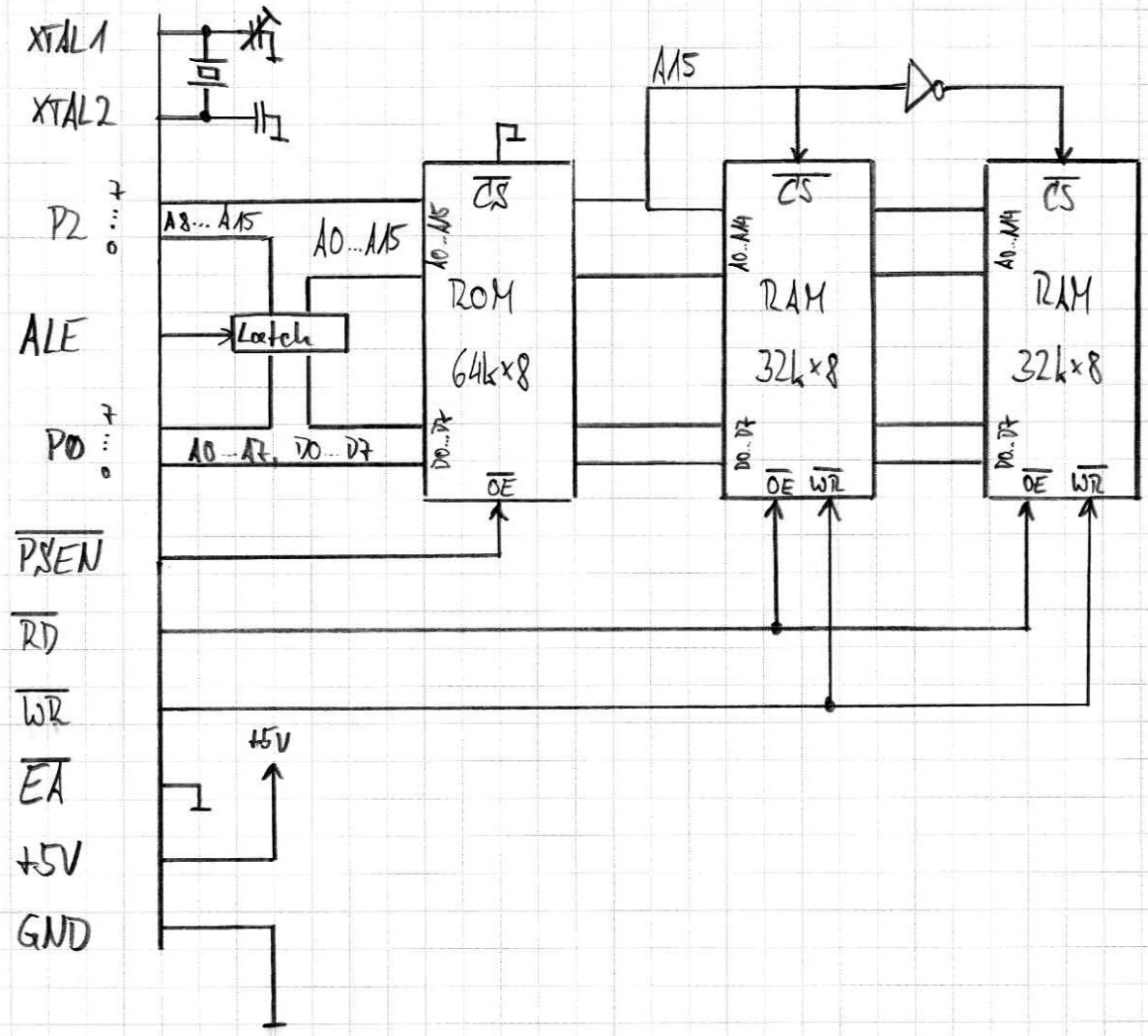


- **Datenpfade**

1	Schreiben	MOV	P3,	#0AAh	
2	Lesen Pin	MOV	A,	P3	
3	Lesen FF	XRL	P3,	#0FFh	(Read-Modify-Write)

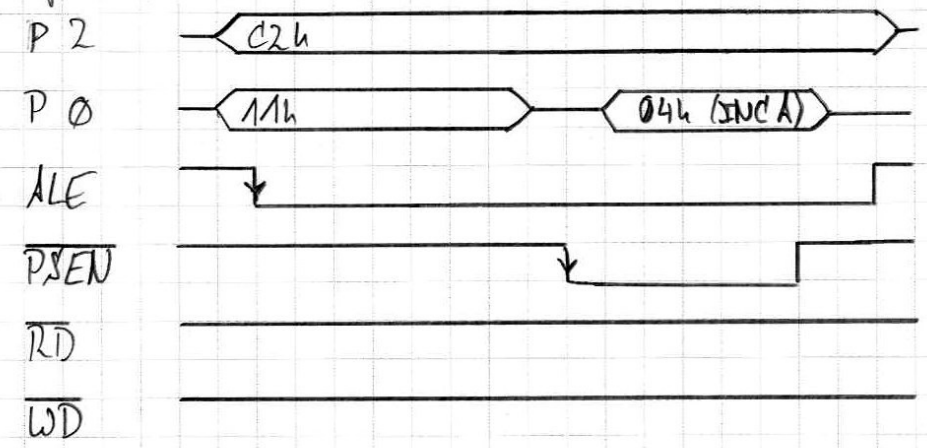
- **Tastenanschluss**
  - → gegen Masse wegen int. Pull-Up (ausser P0)
  - → am Pin '1' ausgeben, damit MOSFET hochohmig

Anschluss ext. Speicher



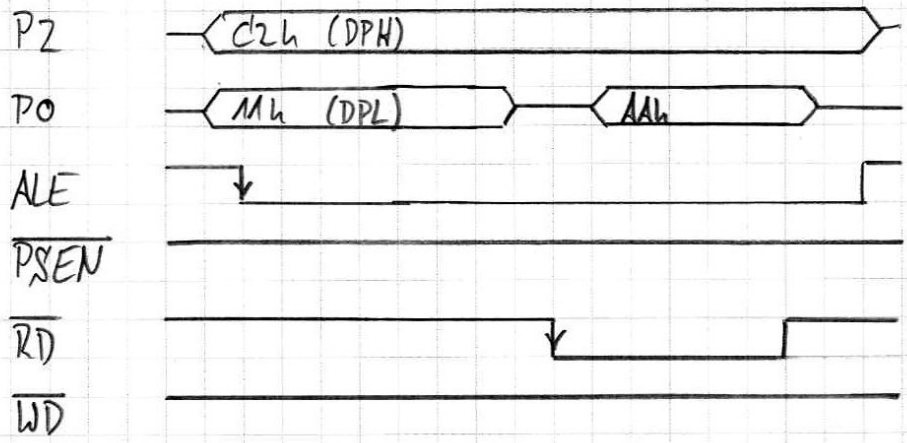
Zugriff ext. Speicher

- Programmspeicher lesen → INC A auf C211h



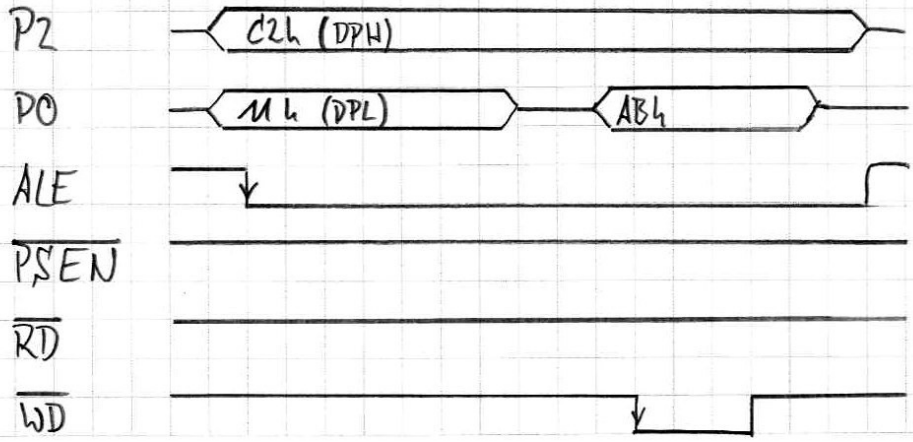
Dauer: 1/2 Zyklus
Befehl der 2. Hälfte wird ignoriert.

- Datenspeicher lesen → AAh von C211h



Dauer: 1 Zyklus

- Datenspeicher schreiben → ABh nach C211h



Dauer: 1 Zyklus

Adressierungsarten

- **Register-A.**
  R0 ... R7 in durch RS0, RS1 (im PSW) gewählter Registerbank
  L> Rn
- **Direkte A.**
  Interne Ram-Adresse wird übergeben
  SFR nur direkt adressierbar (bzw. inhärent)
  L> direct
- **Indirekte A.**
  Inhalt von R0 bzw. R1 dient als Adresse, SFR nicht adressierbar
  L> @Ri
  auch indirekt: Zugriff auf ext. Ram per DPTR (oder Ri)
  (Stack-Adressierung via SP)
- **Unmittelbare A.**
  Übergabe einer Konstante, normalerweise 8 bit, beim DPTR 16 bit
  L> #data
- **Indirekt indizierte A. / Index-A.**
  nur bei MOVC A, @A+DPTR (bzw +PC)
  Zweck: Lesen von Werktabellen im Programm-Rom
- **Inhärente Adressierung**
  Aktu A (und Reg. B) in vielen Befehlen, z. B. INC A

Programmstatuswort

PSW

C	AC	F0	RS1	RS0	OV	/	P

C	Carry	Übertrag von Bit 7
AC	Auxillary Carry	Übertrag von Bit 3 nach Bit 4
F0	User Flag	
RS0, RS1	Registerbank-Selektion	
OV	Overflow	(Übertrag von Bit 6 nach Bit 7) xor C
P	Parity	Anzahl der 1en im Akku ungerade Anzahl der 1en im Akku mit P gerade

Das Carry-Flag dient bei der Subtraktion als Borrow, muss also vor der ersten SUBB-Operation i. A. gelöscht werden. INC und DEC beeinflussen, mit Ausnahme des Parity-Flags, das PSW nicht.
Die BCD-Dezimalkorrektur mit DA A setzt das Carry-Flag bei Zahlen $\geq 100_{BCD}$, löscht es aber im umgekehrten Fall nicht. AC und OV bleiben in jedem Falle unverändert.
Das Parity-Flag kann durch das Programm nicht direkt geändert werden, es ist stets vom Inhalt des Akkus abhängig.
Eine Division durch Null setzt das OV-Flag, ansonsten werden bei DIV A, B Overflow und Carry gelöscht.

Interrupts

- **Quellen**

  ext. $\overline{INT0}$, $\overline{INT1}$; Timer 0, Timer 1 ; seriell gesendet/empfangen

  ↳ TCON

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

  ↳ SCON

...	TI	RI

  ↳ zusammen 1 Int.

  TFx Timer Int. Flag
  TRx Timer Run / $\overline{Stop}$
  IEx Ext. Int Flag
  ITx 1 / 0

  [Flags werden beim Sprung zur ISR gelöscht, außer TI/RI]

- **Enable**

  ↳ IE

EA	/	/	ES	ET1	EX1	ET0	EX0

  0 gesperrt
  1 freigegeben

  EA generelle Freigabe
  ES seriell
  ETx Timer
  EXx extern

- **Priorität**

  ↳ IP

  [wenn gleich:
  Ext0 → T0 →
  Ext1 → T1 → SI]

/	/	/	PS	PT1	PX1	PT0	PX0

  PS seriell
  PTx Timer
  PXx extern

  IPL - Int. Priority Level
  0 niedrig
  1 hoch

Timer / Zähler

- **Steuerung**

  L> TMOD

Gate	C/$\overline{T}$	M1	M0	Gate	C/$\overline{T}$	M1	M0
Timer 1				Timer 0			

Int.-Flags und Run/$\overline{Stop}$ in TCON

Gate — Torsteuerung mit $\overline{Int1}$, $\overline{Int0}$

C/$\overline{T}$ — Zähler / $\overline{Timer}$

M1, M0 } Mode-Steuerung

- **Modes**

0	13 bit	aus Kompatibilitätsgründen, THx.0...7 + TLx.0...4
1	16 bit	
2	8 bit	mit Auto-Reload, bei Überlauf TLx ← THx
3	2x8 bit	nur Timer 0, Timer 1: Stop

TL0 gesteuert von Gate 0, C/$\overline{T}$0, TR0 -> Flag TF0

TH0 gesteuert von TR1, -> Flag TF1 [nur f_{osc}/12]

Timer 1 dann gesteuert von M01, M11, Gate1, C/$\overline{T}$1

[L> kein Int. möglich, aber Baudrate-Generation

läuft ständig, Stop nur durch M01 = M11 = 1]

Serielle Schnittstelle

- **Steuerung**

  ↳ SCON

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

TI, RI	Interrupt-Flags Senden, Empf.
TB8, RB8	9. Bit Senden, Empf.
REN	Empfang zugelassen / gesperrt
SM2	Multiprozessor- / Normal-Mode
SM0, SM1	Mode-Steuerung

PCON

SMOD	...

↳ siehe Mode 1

SBUF

↳ lesen : Empfangs-

schreiben: Sende-

Puffer

- **Modes**

0 Schieberegister (8 bit)

Senden: schreiben in SBUF

Daten an RXD, Takt an TXD, dann TI=1

Empfangen: RI=0 bei REN=1

Daten an RXD, Takt an TXD, dann RI=0, Daten in SBUF

1 UART 8bit $t_{Bit} = 32 \cdot t_{T1ü} / 2^{SMOD}$, RB8 = Stopbit

2 UART 9bit $t_{Bit} = 64 \cdot t_{XTAL} / 2^{SMOD}$

3 UART 9bit $t_{Bit} = 32 \cdot t_{T1ü} / 2^{SMOD}$

- **Baudraten**

für Mode 1 mit T1 im Autoreload $256 - (2^{SMOD} \cdot f_{Osz} / Baudrate / 384)$

für Mode 2: $1/32 \cdot f_{Osz}$ für SMOD; $1/64$ für $\overline{SMOD}$

für Mode 3: wie Mode 1