Praktikum Mikrorechner 5 (Bitadressen, Sprünge und Schleifen)

Prof. Kemnitz

Institut für Informatik, Technische Universität Clausthal

5. November 2014

Bitadressen und Bitbefehle

- Bitadressierbarer Speicher: DS(20h) bis DS(2fh);
  - Bitadresse:

    ```
    badr = (dadr AND 0fh)*8 + BitNr
    ```
- Bitadressierbare SFR: 80h, 88h, 90h, ... f8h;
  - Bitadressen:

    ```
    badr = dadr + BitNr
    ```
- Bitbefehle:

```
mov c, <badr>   ; kopieren
mov badr, c
anl c, <badr>   ; UND
anl c, <badr>
orl c, <badr>   ; ODER
orl c, <badr>
```

```
clr c          ; Löschen
clr <badr>
setb c         ; Setzen
setb <badr>
```

Symbolische Namen für Bitadressen

```
;prozessorspezifische Namen
ov; Überlauf

;Vereinbarung eigener Namen
Flag  equ 20h
Flag0 bit Flag.0
Flag1 bit Flag.1
led0  bit P1.0
```

Beispielprogramm

```
;led0 = btn0 UND  btn1
;-------------------------------
bt0  bit P3.2  ; Taster 0
bt1  bit P3.3  ; Taster 1
LED0 bit P1.0  ; Leuchtdiode 0
;-------------------------------
 setb bt0      ; Vorbereitung für Eingabe
 setb bt1      ;
Schleife:
 mov c, bt0    ; Taster 0 einlesen
 anl c, bt1    ; UND Taster 1
 mov led0, c   ; Ergebnisausgabe auf LED0
ljmp Schleife
```

Sprünge und Schleifen

- Selbst einfache Aufgaben verlangen lange Befehlssequenzen
- Mehrfachverwendung von Befehlssequenzen
- Hochsprachen:
  - for-, while, repeat-until-Schleife
  - if-then-else, case
  - Unterprogramme
- Prozessorbefehle
  - bedingte und unbedingte Sprünge
  - Unterprogrammaufruf und Rücksprung

Sprungbefehle

Der Programmfortsetzung mit einer Befehlsadresse (BA) ungleich der Folgeadresse:

- Absolute Spünge (BA = Konstante)

```
ljmp <adr16> ; Adressbereich 0 bis 0ffffh
ajmp <adr11> ; Adressbereich 0 bis 07ffh
```

- Relative Sprünge (BA = BA+rel)

```
sjmp <rel>    ;  Sprungentfernung -80h≤rel≤7fh
```

- Berechneter Sprung (BA = acc+dptr)

```
jmp @a+dptr;
```

- Marke als Konstante für Sprungziel oder -entfernung:

```
Marke:
 ...
 ljmp Marke
```

Bedingte Sprünge

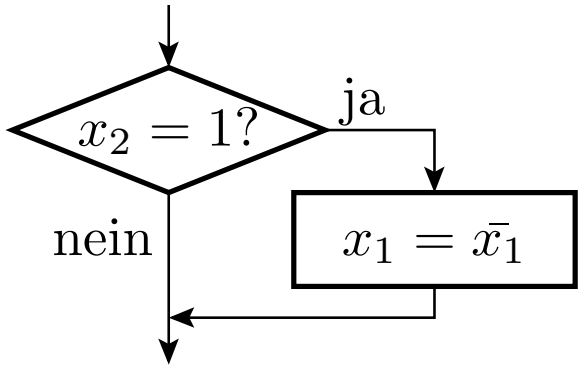
Relative Sprünge; Sprungentfernung -80h≤rel≤7fh

- Springe, wenn ein bestimmtes Bit 0 oder 1 ist

```
jc  <rel>           ; wenn cy=1
jnc <rel>           ; wenn cy=0
jb  <badr>, <rel>   ; wenn adressiertes Bit = 1
jnb <badr>, <rel>   ; wenn adressiertes Bit = 0
```

Macro für bitweises EXOR

```
xorb_mac MACRO x1, x2
 ; x1, x2: Bitadressen
LOCAL  Marke
    jnb x2, Marke
    cpl x1
Marke:
ENDM
```

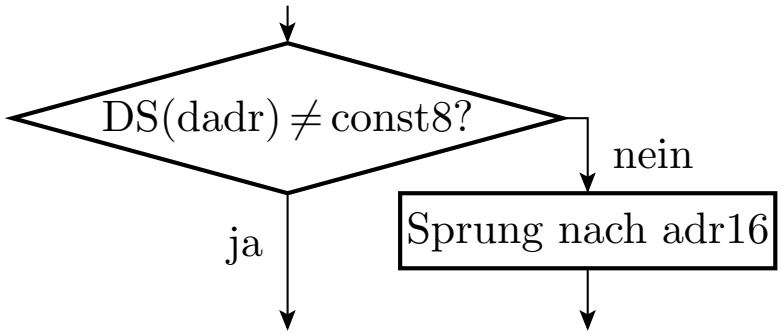


- Springe, wenn ein bestimmter Wert nicht vorliegt

```
jz  <rel>                          ; wenn Acc = 0
jnz <rel>                          ; wenn Acc ≠ 0
cjne a, <dadr>, <rel>              ; wenn Acc ≠ DS(dadr)
cjne a,    #<const8>,<rel>         ; wenn Acc ≠ const8
cjne <rr>, #<const8>,<rel>         ; wenn rr  ≠ const8
cjne @<ri>,#<const8>,<rel>         ; wenn @ri ≠ const8
```

Makro für einen Sprung, wenn DS(dadr)=const8

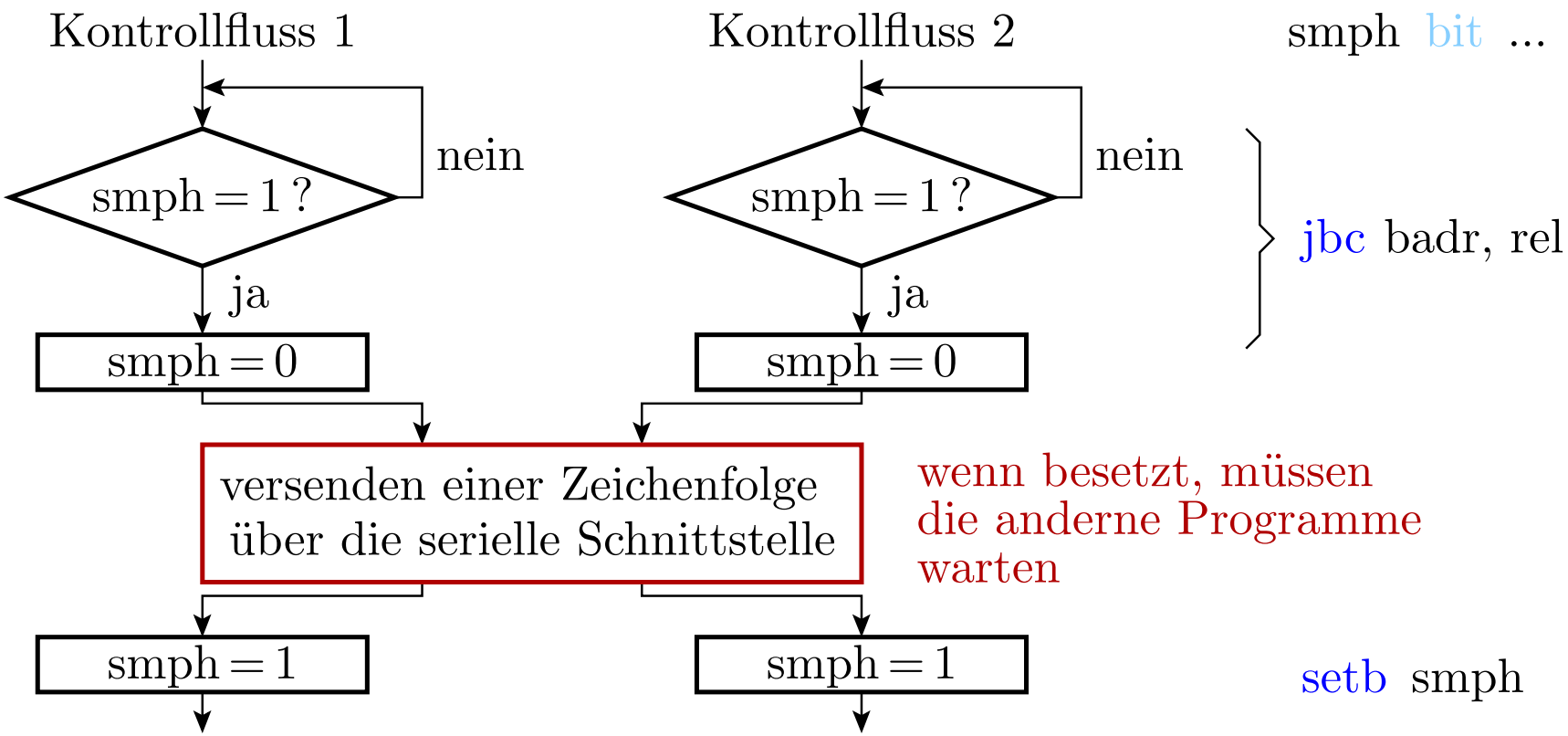
```
cje_mac MACRO dadr, const8, adr16
; V: acc
LOCAL Marke
    mov a, dadr
    cjne a, #const8, Marke
        ljmp adr16
Marke:
ENDM
```



Spezialsprung für die Ressourcenverwaltung

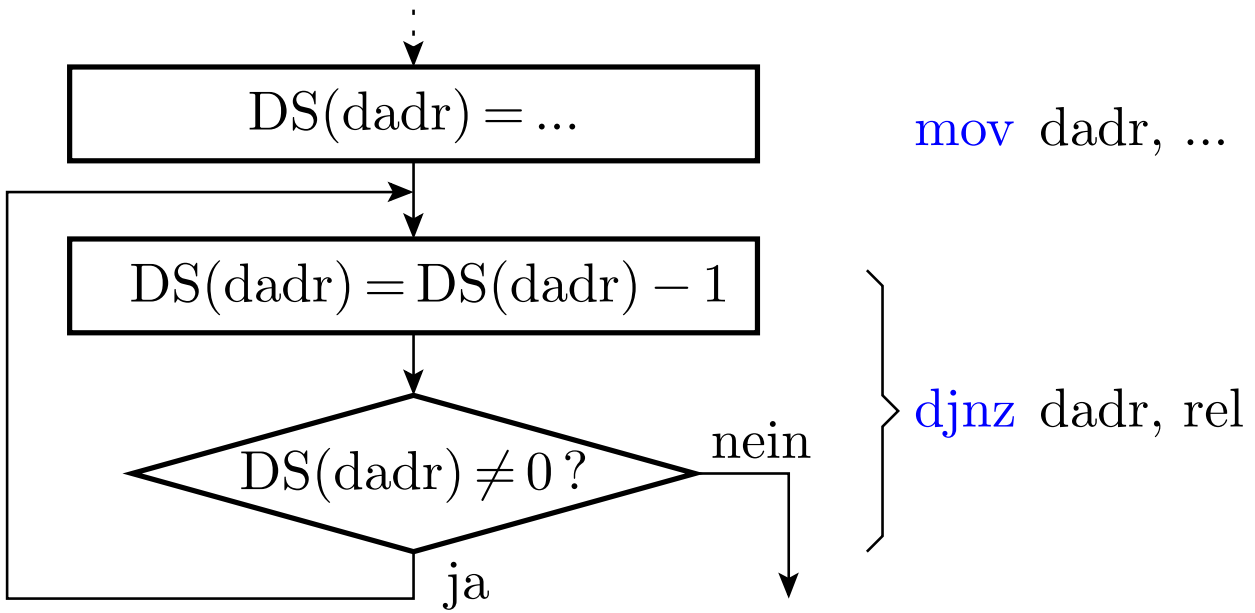
- Semaphor: Bitvariable zum Besetzen von Systemresourcen

  `jbc <badr>, <rel>`

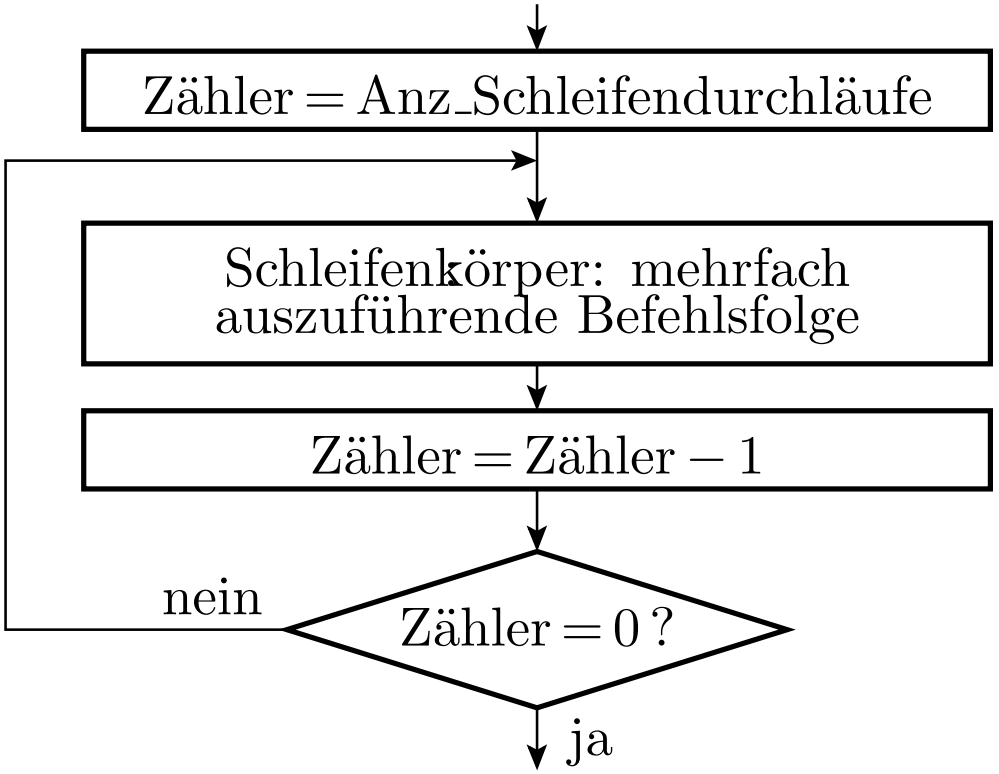


Spezialsprung für Wiederholschleifen

```
djnz <rr>, <rel>   ; rr=rr-1 und Sprung, wenn rr≠0
djnz <dadr>, <rel>; rr=rr-1 und Sprung, wenn Variable
                   ;ungleich 0
```



Wiederholschleife

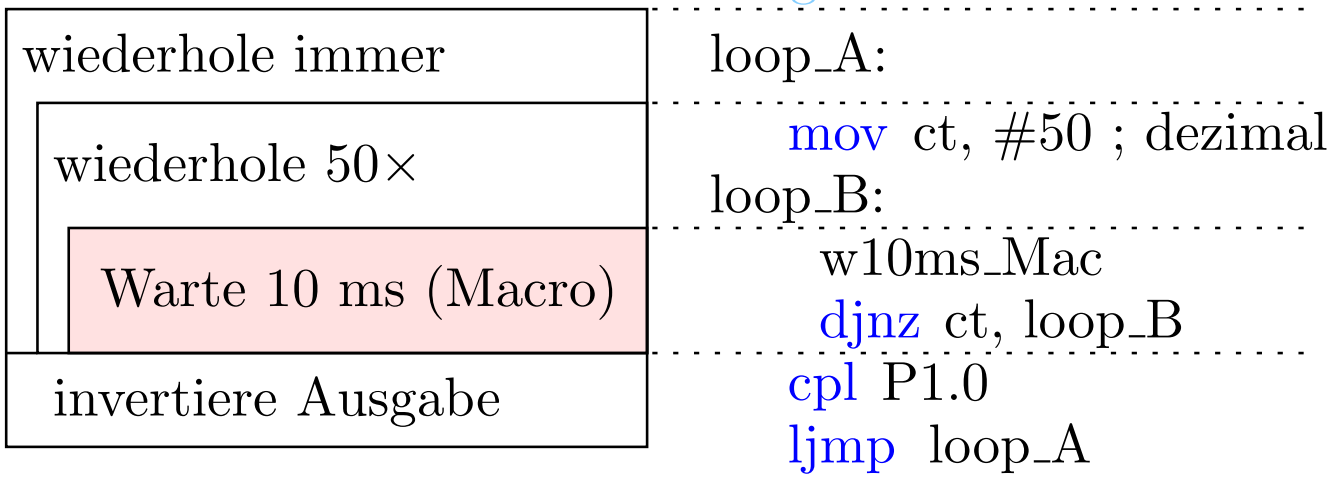



```
ct data 80h
mov ct, #21
Marke:

<Schleifenkörper>

djnz ct, Marke
```

Warteschleife

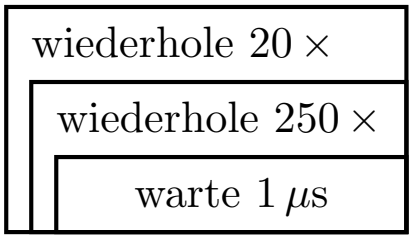
Die Leuchtdiode LED1 an Port P1.0 soll mit einer Frequenz von ungefähr 1 Hz blinken. Periodische Programmfolge:

- tue 500 ms lang nichts
- Ausgabewert invertieren.

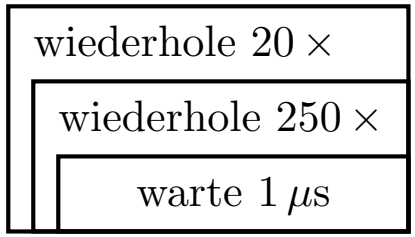


Macro für 10 ms Wartezeit

- Prozessortakt 12 MHz
- 1 bis 4 Zyklen je Befehl (siehe Befehlsliste)
- 6 Takte je Zyklus
- 10 ms Wartezeit $\Rightarrow$ 10.000 Befehle $\Rightarrow$ 20×250 »djnz-Befehle«

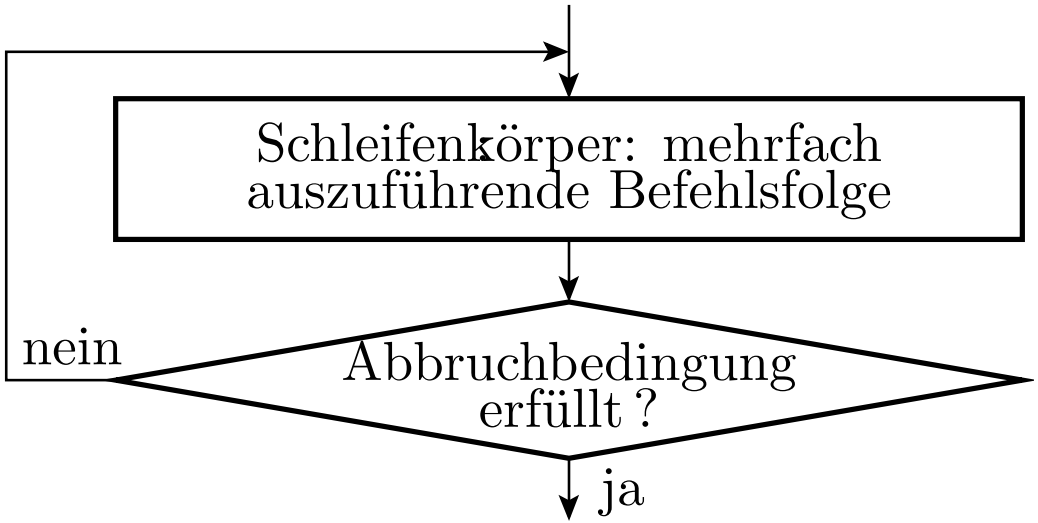


```
Ct0_w10ms_mac data 60h; globale Variablen
Ct1_w10ms_mac data 61h;

w10msMac MACRO
    LOCAL M1, M2
        mov Ct1_w10ms_mac, #20
    M1:
          mov Ct0_w10ms_mac, #248; (*)
          nop                    ; (*)
    M2:
            djnz Ct0_w10ms_mac, M2
          djnz Ct1_w10ms_mac, M1
    ENDM
```



(*) damit es ganz genau stimmt

Abbruchschleife (while)



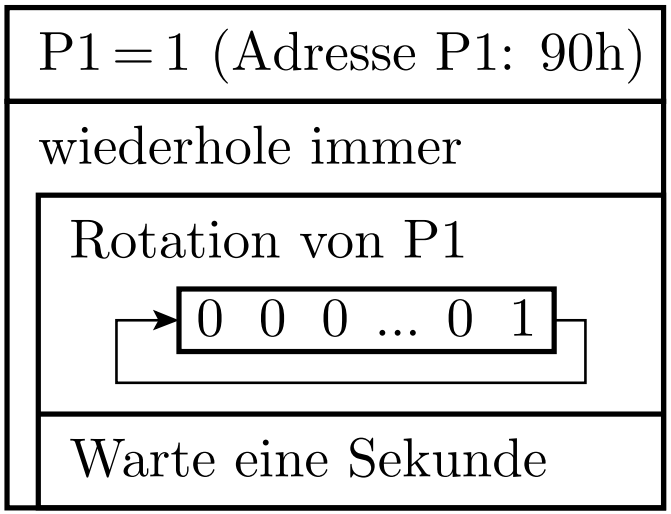
Warte auf Tastendruck

```
Taster bit P3.2
...
setb Taster; Eingabevorbereitung
jb Taster, $
```

($ – Adresse des aktuellen Befehls)

Aufgabe 5.1: Lauflicht

Schreiben Sie ein Progamm, das eine wandernde Eins auf den Leuchtdioden ausgibt.



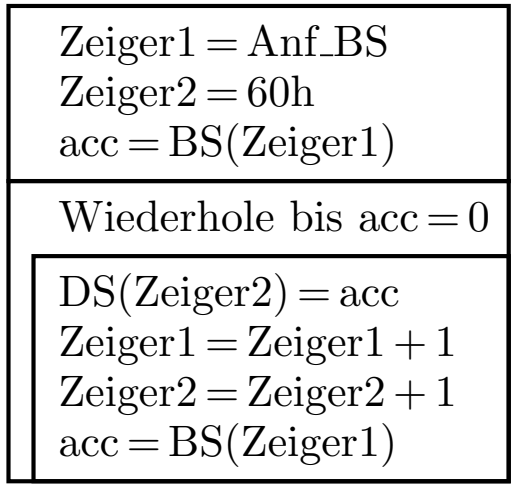
Aufgabe 5.2: Kopierschleife

Legen Sie im Befehlsspeicher eine Folge von Byte-Konstanten an, die mit Null endet:

```
org 200h
    Anf_BS:
    db 35h, 45h, ... , 0h
```

Schreiben Sie ein Programm, dass diese Bytefolge in den Datenspeicher ab Adresse 60h kopiert.

Welche Speicherplätze sind für die Zeiger geeignet?



Aufgabe 5.3: elektronischer Würfel

Schreiben Sie eine Programm, das auf Tastendruck eine Zufallszahl auf den Leuchtdioden ausgibt.