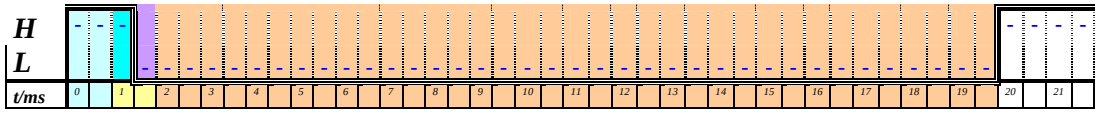
Impulsdiagramm: ***PWM-Periode: High(0...1,5ms), Low(1,5ms..20ms),***
>> Auflösung PWM-Stufe: 10us



-Vorüberlegung: ***Periodenaufbau: High(1ms) + VariablerPuls(1ms) + Low(18ms)***
Variabler Pulsanteil: High-Puls ist von 0..1000us in 100*10us-Steps variabel,
Pulsbreitenzähler Hightime R0: 0..100, Lowtime =100-R0

Struktogramm:

B4.2: PWM-Servotester
PWM-Out: P1.7
Tastgrad-Input: P1.1 / P1.0
00: bleibt, 01: Down ..0, 10: Up ..100, 11:Reset auf Mitte(50)
Pulsbreitenzähler Hightime R0: 0..100, Lowtime = 100-R0

Init P1.0, P1.1 als InputPins
Zähler R0 = 50

P1.0 , P1.1 in A einlesen und ausmaskieren

A: 0	1	2	ansonsten(3)
∅	DOWN	UP	RESET: R0 = 50

P1.7 = 1

WAIT1ms

A = R0

(A == 0)
Y: ∅
N:
- P1.7 = 1
- Wait10us
- A --

solange (A != 0)

A = 100 - R0

(A == 0)
Y: ∅
N:
- P1.7 = 0
- WAIT10us
- A --

solange (A != 0)

P1.7 = 0

WAIT18ms

UP

(R0<100)
Y: R0 ++
N: ∅
RETURN

DOWN

(R0>0)
Y: R0 --
N: ∅
RETURN

AVR-ASM-Programm:

```
; ÜBUNG PWM Servotester   , Z statt R0, A statt ACC
; PWM-Output:             PORTB7
; Tastgrad-Input up/down: PINB1 / PINB0
; 00: bleibt, 01: Down ..0 , 10: Up ..100, 11: Reset auf 0
; PWM-Periode(20ms): FixHigh(1ms) + VariablerPuls(1ms) +  FixLow(18ms)
; Variabler Pulsanteil: High-Puls ist von 0..1000us(1ms) in 100*10us-Steps variabel,
; Pulsbreitenzähler Z (PWM-Zähler, Hightime): 0..100, Lowtime =100-Z
; Register-Definitionen:
.def A = r20  ; vgl. Accu
.def Z = r21  ; vgl. Zähler
;######### Hauptprog. ################################################################
.org 0x0100
INIT: ldi A,0xFF    ; DDRB-Register über Register mit immediate-Wert füllen ...
      out DDRB,A    ; PORTB (bit7..0) zunächst komplett als Output initialisieren
      cbi DDRB,1    ; PINB1 als Input, Taster "UP"
      cbi DDRB,0    ; PINB0 als Input, Taster "DOWN"
      sbi PORTB,1   ; PINB1 mit internem PullUp
      sbi PORTB,0   ; PINB0 mit internem PullUp
      ldi Z,50      ; Tastgrad-Zähler initialisieren Z=50
      cbi PORTB7    ; PWM-Pin PORTB7 = 0
MAIN:
 ;####### Eingabe ###############################################################
      in A,PINB         ; Mode an PINB1/PINB0 einlesen:  PINB >> A
      andi A,0b00000011 ; Bits 7..2 ausblenden, Bit 1,0 bleibt (dez. 0..3)
 ;####### Mode (0-3) -Auswahl ##############################
      cpi A,0           ; A==0 ?
      breq LOS          ; case: Mode0 (00): weiter, Z unverändert
      cpi A,1           ; A==1 ?
      breq DIR_DOWN     ; case: Mode1 (01): DOWN , Z--
      cpi A,2           ; A==2 ?
      breq DIR_UP       ; case: Mode2 (10): UP , Z++
      cpi A,3           ; A==3 ?
      breq RESET        ; case: Mode3 (11): Beide gedrückt:  RESET, Z=0
      rjmp LOS          ;
  DIR_UP:    rcall UP   ; Zählrichtung "UP"
             rjmp  LOS  ; und ausgeben
  DIR_DOWN:  rcall DOWN ; Zählrichtung "DOWN"
             rjmp  LOS  ; und ausgeben
  RESET:     ldi Z,0    ; Z=0
;#### Anteil FixHigh (1ms) ##################################################
 LOS: sbi PORTB,7     ; PWM-Pin PORTB7 = 1 HIGH für 1ms
      rcall WAIT1ms
;#### VARIABLE PWM AUSGABE ###################################################################
  ON:   mov A,Z           ; A=Z , Output HIGH, wait Hightime(Z)
        cpi A,0           ; A==0 ?
        breq ON_END       ; Kein HIGH bei Tastgrad 0
   ONLOOP: sbi PORTB,7     ; HIGH für Z*10us (1..100*10us)
           rcall WAIT10us
           dec A          ; A--
           cpi A,0        ; A==0?
           brne ONLOOP    ; while(A!=0)
   ON_END:
;-------------------------------------------------------------------------------------------
  OFF:    ldi A,100        ; A=100-Z, Output Low, wait Lowtime(100-Z)
          sub A,Z
          cpi A,0          ; A==0 ?
          breq OFF_END     ; Kein LOW bei Tastgrad 100
   OFFLOOP: cbi PORTB,7     ; LOW für (100-Z)*10us (100..1*10us)
            rcall WAIT10us
            dec A          ; A--
            cpi A,0        ; A==0?
            brne OFFLOOP   ; while(A!=0)
   OFF_END:
;#### Anteil FixLow (18ms) ##################################################
         cbi PORTB7    ; PWM-Pin PORTB7 = 0 LOW für 18ms
         rcall WAIT18ms
    rjmp MAIN  ; Endlosschleife
;#######  eigene Unterprogramme ###############################################################
.org 0x0400
UP:         cpi Z,100          ; if (Z<100)
            brlo LESS100       ;     Z++
            ldi Z,100          ; else kein ++, optional Z=100
            ret                ;
  LESS100: inc Z               ;
            ret                ;
;-------------------------------------------------------------------------------------------
DOWN:       cpi Z,0            ; if (Z>0)
            brpl PLUS          ;    Z--
            ret                ; else kein --
    PLUS:   dec Z              ;
            ret                ;
;#######  gegebene Unterprogramme ###############################################################
Wait10us:  ; 10us warten...      ret
Wait1ms:   ; 1ms warten...       ret
Wait1ms:   ; 18ms warten...      ret
Wait100ms: ; 100ms warten...     ret
```

Implementierung des Struktogramms als lauffähiges (ATmega2560-)Assembler-Programm .

Allgemeine Hinweise : *Das Assemblerprogramm soll bei 0x0100 beginnen, die Unterprogramme ab 0x0400*
Die Pins von Port_B sind sinnvoll zu initialisieren, nicht genutzte Pins sollen Ausgänge sein .
(Direction-Register: 0= In, 1= Out; Port-Register: 1 aktiviert internen Pullup)
Der Programmcode ist mit sinnvollen Kommentaren zu dokumentieren.
