```
Variable V$TimeSlotReady
Variable V$MstByte

\ TIMER2A ------------------------------------------------------------------

Variable V$TIMER2A> \ Action of Timer2A

LABEL <TIMER2A>
   \ Save registers
   R16 PUSH
   SREG R16 IN
   R16 PUSH R17 PUSH
   ZL PUSH ZH PUSH

   V$TIMER2A>    ZL LDS
   V$TIMER2A> 1+ ZH LDS
   R31 LSR
   R30 ROR
   ICALL     \ call word in V$TIMER2A>

   \ Restore registers
   ZH POP ZL POP
   R17 POP R16 POP
   R16 SREG OUT
   R16 POP
   RETI   END-CODE

<TIMER2A> OC2Aaddr INTERRUPT

\ TIMER2B ------------------------------------------------------------------

Variable V$TIMER2B> \ Action of Timer2B

LABEL <TIMER2B>
   \ Save registers
   R16 PUSH
   SREG R16 IN
   R16 PUSH R17 PUSH
   ZL PUSH ZH PUSH

   V$TIMER2B>    ZL LDS
   V$TIMER2B> 1+ ZH LDS
   R31 LSR
   R30 ROR
   ICALL     \ call word in V$TIMER2B>

   \ Restore registers
   ZH POP ZL POP
   R17 POP R16 POP
   R16 SREG OUT
   R16 POP
   RETI   END-CODE

<TIMER2B> OC2Baddr INTERRUPT

\ Hardware -----------------------------------------------------------------
Code  VDD 4 PORTD SBI RET End-Code
Code /VDD 4 PORTD CBI RET End-Code

Code /DQ 3 DDRD SBI RET End-Code \ output
Code  DQ 3 DDRD CBI RET End-Code \ input

: DQ? ( -- f )
   PIND C@ $08 And
   ;

Code StartTimer2
```

```
   R16 CLR   R16 TCNT2 OUT \ Clr Timer
   %00000010 R16 LDI   R16 TCCR2B OUT \ Start timer  Prescaler / 8
   RET End-Code

Code StopTimer2
   %00000000 R16 LDI   R16 TCCR2B OUT \ Stopt timer
   RET End-Code

\ Timeconstants -----------------------------------------------------------------
100 CPUCLOCK 8000000 */ EQU 100uS
  3 CPUCLOCK 8000000 */ EQU   6uS \ time corrected
 12 CPUCLOCK 8000000 */ EQU  15uS \ time corrected
 60 CPUCLOCK 8000000 */ EQU  60uS
 70 CPUCLOCK 8000000 */ EQU  70uS

\ Resetpulse --------------------------------------------------------------------
Variable V$100uS
Variable V$Presence

Label MstResetA \ 100uS Counter
   R16 CLR  R16 TCNT2 OUT  \ Clr Timer Counter Register
   V$100uS R16 LDS   R16 INC  R16 V$100uS STS \ cout up 100us
    5 R16 CPI   0= If
      3 DDRD CBI  \ DQ -> high
      70uS R17 LDI   R17 OCR2B OUT \ init timerB
    Then
   10 R16 CPI   0= If
      $FF R17 LDI   R17 V$TimeSlotReady STS \ ready flag setzen
      R17 CLR   R17 TCCR2B OUT \ Stopt timerThen
   Then
   RET End-Code

Label MstResetB \ fetch presence
   PIND R16 IN   Bit3 R16 ANDI  R16 V$Presence STS \ DQ?
   $FF R17 LDI   R17 OCR2B OUT \ init timerB
   RET End-Code

: MstReset ( -- )
   0 V$100uS ! 0 V$TimeSlotReady !  0 V$Presence !
   MstResetA V$TIMER2A> !  100uS OCR2A C!
   MstResetB V$TIMER2B> !  $FF   OCR2B C!
   /DQ StartTimer2
   Begin V$TimeSlotReady @ Until
   ;

: Presence? ( -- f )
   V$Presence @ 0= If True Else False Then
   ;

\ Bit Transfer ------------------------------------------------------------------

Label setDQhigh
    3 DDRD CBI
    RET End-Code

Label TimeSlotReady
    %00000000 R16 LDI   R16 TCCR2B OUT \ Stopt timer
    $FF R16 LDI   R16 V$TimeSlotReady STS  \ ready flag
    RET End-Code
TimeSlotReady $FF And EQU TimeSlotReady_L
TimeSlotReady $100 /  EQU TimeSlotReady_H

Label MstSample
   PIND R16 IN   Bit3 R16 ANDI   0= If R16 CLR Else $80 R16 LDI Then
   V$MstByte R17 LDS   R17 LSR   R17 R16 OR   R16 V$MstByte STS
   TimeSlotReady_L R16 LDI R16 V$Timer2B>    STS \ next timerB action
   TimeSlotReady_H R16 LDI R16 V$Timer2B> 1+ STS \ next timerB action
   70uS R16 LDI   R16 OCR2B OUT
   RET End-Code
```

```
: MstWrBit1 ( -- )
   0 V$TimeSlotReady !
   setDQhigh     V$TIMER2A> !   6uS OCR2A C!
   TimeSlotReady V$TIMER2B> !  70uS OCR2B C!
   /DQ  StartTimer2
   Begin V$TimeSlotReady @ Until
   ;

: MstWrBit0 ( -- )
   0 V$TimeSlotReady !
   setDQhigh     V$TIMER2A> !  60uS OCR2A C!
   TimeSlotReady V$TIMER2B> !  70uS OCR2B C!
   /DQ  StartTimer2
   Begin V$TimeSlotReady @ Until
   ;
: MstWrBit ( -- )
   V$MstByte C@ 1 And If MstWrBit1 Else MstWrBit0 Then
   V$MstByte C@ 2/ V$MstByte C!
   ;

: MstRdBit
   0 V$TimeSlotReady !
   setDQhigh V$TIMER2A> !   6uS OCR2A C!
   MstSample V$TIMER2B> !  15uS OCR2B C!
   /DQ  StartTimer2
   Begin V$TimeSlotReady @ Until
   ;


\ Byte transfer ------------------------------------------------------------

: CWrite ( C -- )
   V$MstByte C! 8 0 Do MstWrBit Loop
   ;
: CRead ( -- C )
   8 0 Do MstRdBit Loop V$MstByte C@
   ;

\ initialisation -----------------------------------------------------------

: MstInit
   <NOP> V$TIMER2A> !
   <NOP> V$TIMER2B> !
   StopTimer2
   %00000110 TIMSK2 C! \ Interrupt mask A&B
   ;
```