APPLICATION NOTE

ST7 SOFTWARE LCD DRIVER

by Microcontroller Division Application Team

1 INTRODUCTION

This note describes a technique for driving a Liquid Crystal Display (LCD) with any standard ST72 Microcontroller (MCU) i.e without any specific on-chip LCD driver hardware. This technique offers a solution for applications which require a small display at low cost together with the versatile capabilities of the standard ST72 MCU.

The Section 2 of this note describes the typical waveforms required to drive an LCD with a multiplexing rate of 1 or 2 (duplex). The Sections 3 and 4 present two solutions based on standard ST72 MCUs directly driving an LCD. Both solutions can be implemented with any ST72xx MCU as they only require the standard I/O ports and one timer (first solution) which are standard features of the ST72 MCUs. The first solution uses one timer, more precisely its output compare feature to generate the LCD timing. The second solution targets low power applications, as the ST7 goes into Halt mode between 2 I/O refreshes. An external RC circuit is used to generate the timing, waking up the ST7 (external I/O interrupt).

In both cases the program size (132 bytes / 212 bytes), the CPU load required for controlling LCD, and the number of external components are minimized. As a result many additional tasks can be added to the application program.

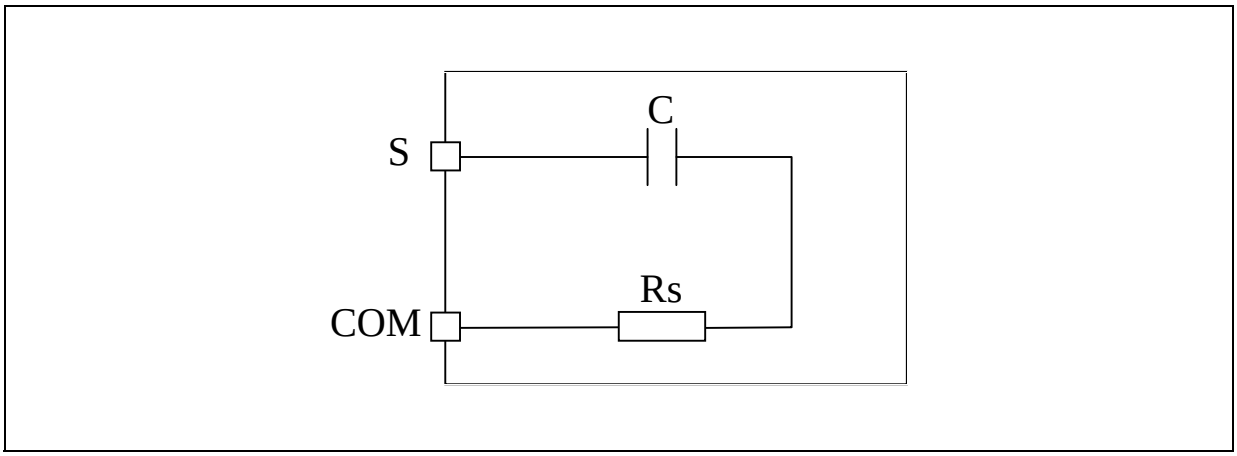
2 LCD REQUIREMENTS

With a zero Root Mean Square (RMS. i.e.: $\sqrt{Mean(Signal^2)}$) voltage applied to it, an LCD is practically transparent. The LCD contrast, which makes the segments turn dark or opaque and thus "ON", is caused by the difference between the applied RMS. LCD voltage applied and the LCD threshold voltage, specific to each LCD type.

The applied LCD voltage must alternate to give a zero DC value in order to ensure a long LCD life time. The higher the multiplexing rates , the lower the contrast. The signal period has also to be short enough to avoid visible flickering on the display.

The LCD voltage for each segment is equal to the difference between the S and COM voltages (see figure 1).

Figure 1. Equivalent Electrical Schematic of an LCD Segment



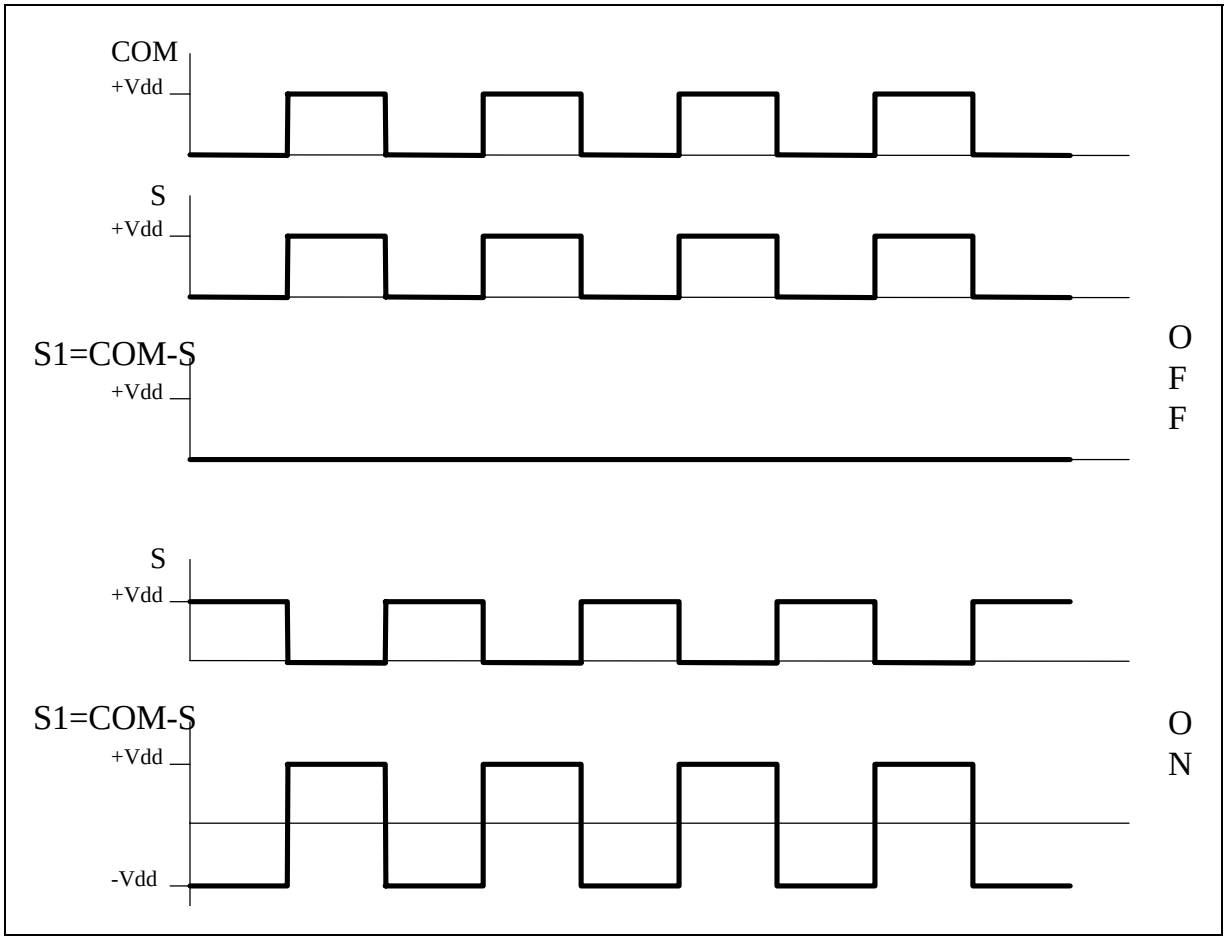
Notes: The DC Value should never be more than 100mV. Otherwise the life time can be shortened.
The frequency range is 30 - 2000Hz typically. If it is less, it flickers; if it is more, the consumption grows.

2.1 ONE BACKPLANE LCD DRIVE

Each LCD segment is connected to an I/O "Segment" and to one backplane common to all the segments. A display using S segments is driven with S+1 MCU output lines. The backplane is driven with a "COM" signal controlled between 0 and Vdd with a duty cycle of 50%.

When switching a segment "ON", a signal with opposite polarity to "COM" is sent to the corresponding "Segment" pin. When the non-inverted signal "COM" is sent to the "Segment" pin, the segment is "OFF". Using an MCU, the I/O operates in output mode either at logic 0 or 1.

Figure 2. LCD Signals for Direct drive



2.2 DUPLEXED LCD DRIVE

in a duplexed drive, two backplanes are used instead of one. Each LCD pin is connected to two LCD segments, each one connected on the other side to one of the two backplanes. Thus, only (S/2)+2 MCU pins are necessary to drive an LCD with S segments.

Three different voltage levels have to be generated on the backplanes : 0, $V_{DD}/2$ and V_{DD}. The "Segment" voltage levels are 0 and Vdd only. The LCD segment is inactive if the RMS. voltage is below the LCD threshold voltage and is active if the LCD RMS. voltage is above the threshold voltage. Figure 4 shows typical Backplane, Segment and LCD waveforms. The intermediate voltage $V_{DD}/2$ is only required for the Backplane voltages. The ST72 I/O pins selected as "Backplanes" are set by software to output mode for 0 or Vdd levels and to high impedance input mode for $V_{DD}/2$. This Vdd/2 voltage is defined by two equal valued resistors externally connected to the I/O pin. By using an MCU with flexible I/O pin configuration such as the ST72251, a duplexed LCD drive can be implemented with only 4 additional resistors.

Figure 3. Basic LCD Segment Connection in duplexed mode

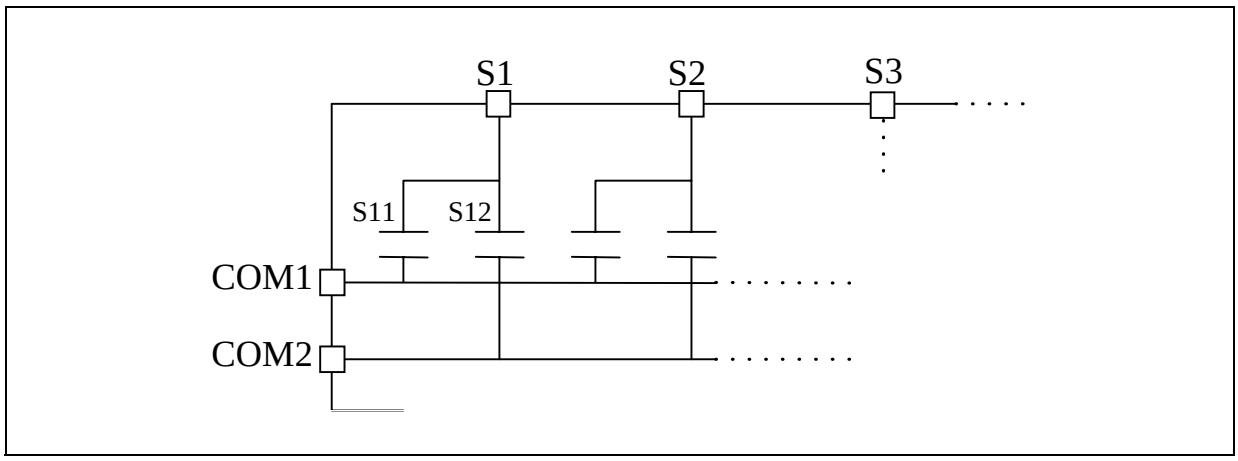
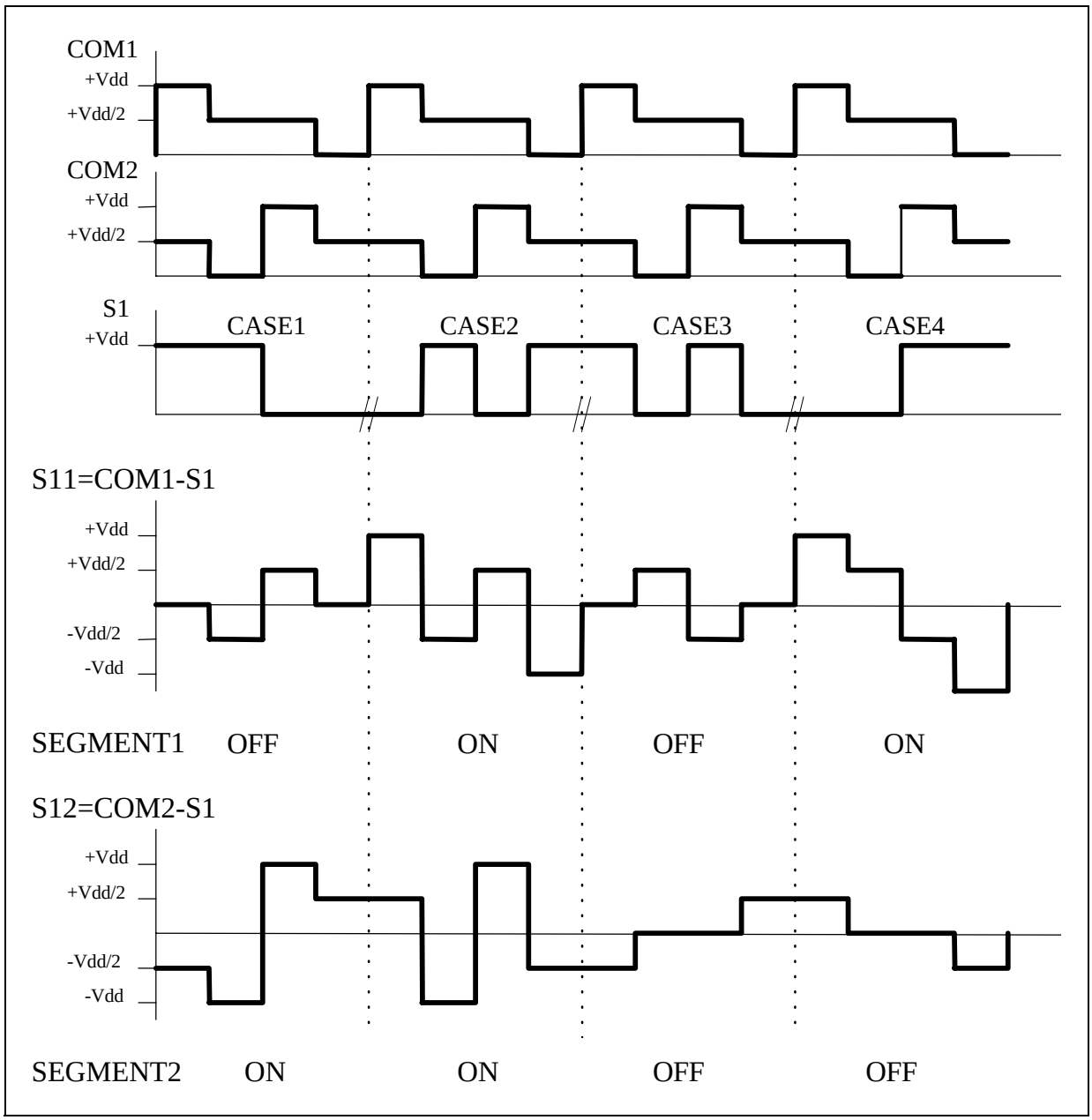


Figure 4. LCD signals for duplexed mode (used in the ST7 example)



The segment RMS. signal is $\frac{Vdd}{\sqrt{8}}$ in cases 1 & 3 for segment 1 (3 & 4 for segment 2) and $\frac{(\sqrt{5})Vdd}{\sqrt{8}}$ otherwise.

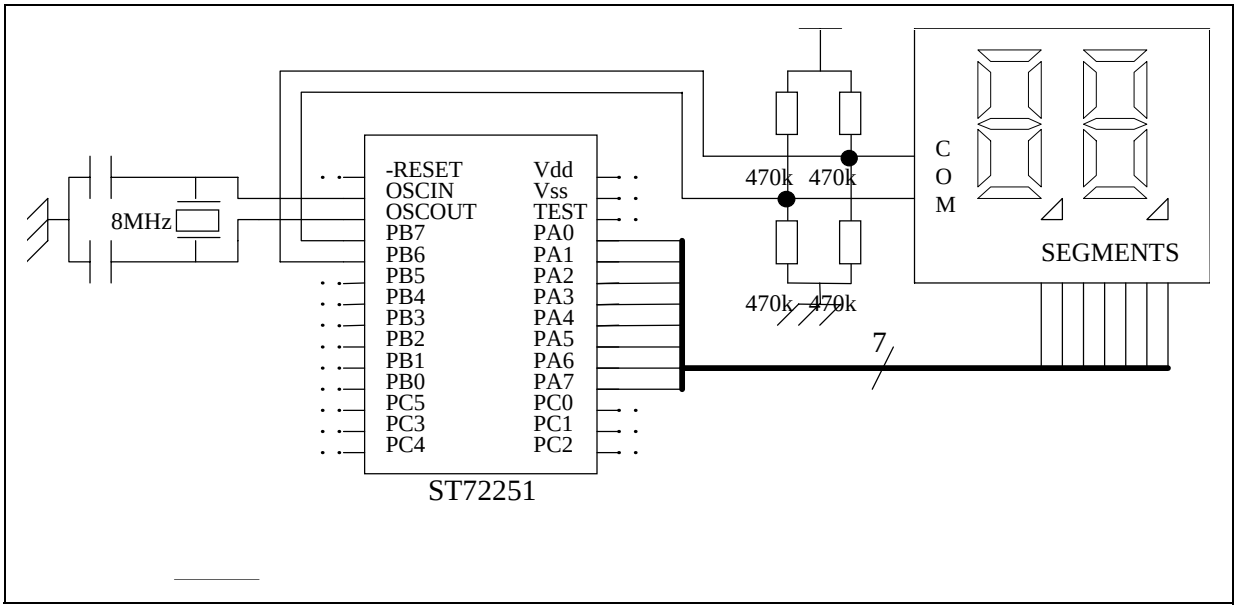
3 EXAMPLE OF A DUPLEXED LCD WITH THE ST72251

The following example describes a drive for duplexed mode (2COM) LCD with the ST72251. The only external components needed for driving the LCD are 4 resistors.

The ST72251 has 22 I/O pins (only 8 are configurable as open drain outputs) and can therefore drive an LCD up to a 32 segments and 2 backplanes. In this example only PA7..PA0 are used for the segment signals. PB7 and PB6 are used to generate both COM1&2 signals, this driver is able to drive a 16 segment LCD. Nevertheless this program can be easily changed to sopport a maximum of 32 segments.

The LCD timing is generated by the TimerA output compare interrupt. The LCD driver consists of 2 initialization routines, the TimerA interrupt routine "tima_rt" (main part of the driver), and 3 variables: "lcdcr", "seg1" and "seg2". To activate the LCD, two initialization routines have to be called: the port initialization routine "port_init" and the TimerA initialization routine "timer_config". Once these routines are called the ST7 will get the Timer output compare 1 & 2 interrupts. According to which output compare and to the LCD control register "lcdcr" the COM1&2 signals will be changed (0, Vdd/2, or Vdd) so that the timing described in Figure 4 will be generated. For the segment outputs the variables "seg1" and "seg2" are also needed. Seg1 and seg2 enable the user program to switch the segment ON or OFF: if the bit N of seg1 is set the driver will turn the Nth segment connected to the COM1 backplane.

Figure 5. ST72251 duplexed LCD example - Hardware



3.1 LISTING 1. ST72251 EXAMPLE - SOFTWARE

The code given below is for guidance only.

st7/

```
;*******************************************************************
; TITLE:        LCD.ASM
; AUTHOR:           Microcontroller Application Team
; DESCRIPTION:     Software driver for a 2 backplane LCD display.
;            Use one timer to generate the appropriate lcd timing
;
;*******************************************************************

    TITLE  "LCD.ASM"
    TITLE  "LCD.ASM"

    MOTOROLA
    .NOLIST
    #INCLUDE "ST72251.inc" ; include st72251 registers and memory mapping file
    #INCLUDE "variable.inc"     ; include general variable file
    .LIST

;*******************************************************************


;*******************************************************************
;    Variables, constants defined and referenced locally
;    You can define your own values for a local reference here
;*******************************************************************
;*******************************************************************
;*******************************************************************
;    Public routines  (defined here)
;*******************************************************************
; routines


; We could have used a dot in front of the label's name to declare the routine
; as visible by others modules in the project.


;*******************************************************************
;    Extern routines  (defined elsewhere)
;
;    The EXTERN directive will be seen by the linker as a call
;    instruction to another routine written in another file
```

```
;************************************************************************
; routines

;*********************************************************
;*********************************************************
;    Program code
;*********************************************************
       WORDS          ; define subsequent addresses as words
                      ; meaning that all instructions are located
                      ; in the address field after 0FFh in the ST72251
                      ; memory mapping

       segment 'rom'
;*************************************************************
;*************************************************************
; TimerA driver - TimerA used for LCD timing generation: generate
;         an output compare interrupt every 2 ms.
;
; DESCRIPTION:
; - PB7 dedicated to COM1, and PB6 to COM2. They provide 3 levels using 2 external
pull up
;  and pull down resistors for each pin, and the open drain output feature: 0, Vdd/
2, and Vdd
; - For segment signals which are normal digital signal (0 or 1) PA[0..7] are used
as ouput
;  push pull
; - The lcd timing is divided in 4 "segments": - oc1_1 & oc1_2: following oc1 IT
;                          - oc2_1 & oc2_2: following oc2 IT
;
;            TIMING (SEGi linked to COMi): cf figure 6
;
;*************************************************************
.tima_rt
        BTJT tasr,#6,oc1
        CLR taclr         ; reset timer counter by any write to the LSB
        LD A,taoc2lr       ; access OCR2 after reading TSR to clear the IT request
        CPL padr          ; NOTE (1): segment signal toggles every oc2 IT
        LD X,lcdcr
        TNZ X           ; test lcdcr(0 or 1) = 1 + lcdcr
        JRNE oc2_2
            ;### actions for oc2_1 segment (1)###
            LD A,pbdr    ; in oc2_1, COM1 = Vdd and COM2 = Vdd/2
            OR A,#$80    ; PB7 is already an output push pull,
            LD pbdr,A    ; PB6 is already an output open drain
                         ; so only need to set bit 7 of pbdr
```

```
            ;### end of actions for oc2_1 segment ###
         JRT out_oc2
            ;### actions for oc2_2 segment (1)###
oc2_2          LD A,pbdr    ; in oc2_1, COM1 = Vdd and COM2 = Vdd/2
            OR A,#$40    ; PB7 is already an output push pull,
            LD pbdr,A    ; PB6 is already an output open drain
            ;### end of actions for oc2_2 segment ###
out_oc2     LD A,lcdcr
         XOR A,#1         ; toggles lcdcr
         LD lcdcr,A
         JRT out
oc1       LD A,taoc1lr   ; access OCR1 after reading TSR to clear the IT request
         LD X,lcdcr
         LD A,(seg1,X)  ; in this segment oc1_i, the segment output value is the
         LD padr,A    ; one of the segment i: 1 if segment is ON. Otherwise 0.
         TNZ X          ; test lcdcr(0 or 1) = 1 + lcdcr
         JRNE oc1_2
            ;### actions for oc1_1 segment ###
            LD A,pbdr
            AND A,#$7F
            LD pbdr,A
            LD A,pbor    ; in oc1_1, COM1 = 0 and COM2 = Vdd/2
            OR A,#$80    ; so PB7 is an output push pull,
            AND A,#$BF    ; and PB6 an output open drain
            LD pbor,A
            ;### end of actions for oc1_1 segment ###
         JRT out
            ;### actions for oc1_2 segment ###
oc1_2          LD A,pbdr
            AND A,#$BF
            LD pbdr,A
            LD A,pbor    ; in oc1_2, COM1 = Vdd/2 and COM2 = 0
            AND A,#$7F    ; so PB7 is an output open drain,
            OR A,#$40    ; and PB6 an output push pull
            LD pbor,A
            ;### end of actions for oc1_2 segment ###
out       IRET
;*************************************************************
; TimerA initialisation
;*************************************************************
.timer_config
         CLR A
         LD lcdcr,A       ; initialize lcd software control register
         LD A,#$03
         LD taoc1hr,A      ; output compare mode chosen.
```

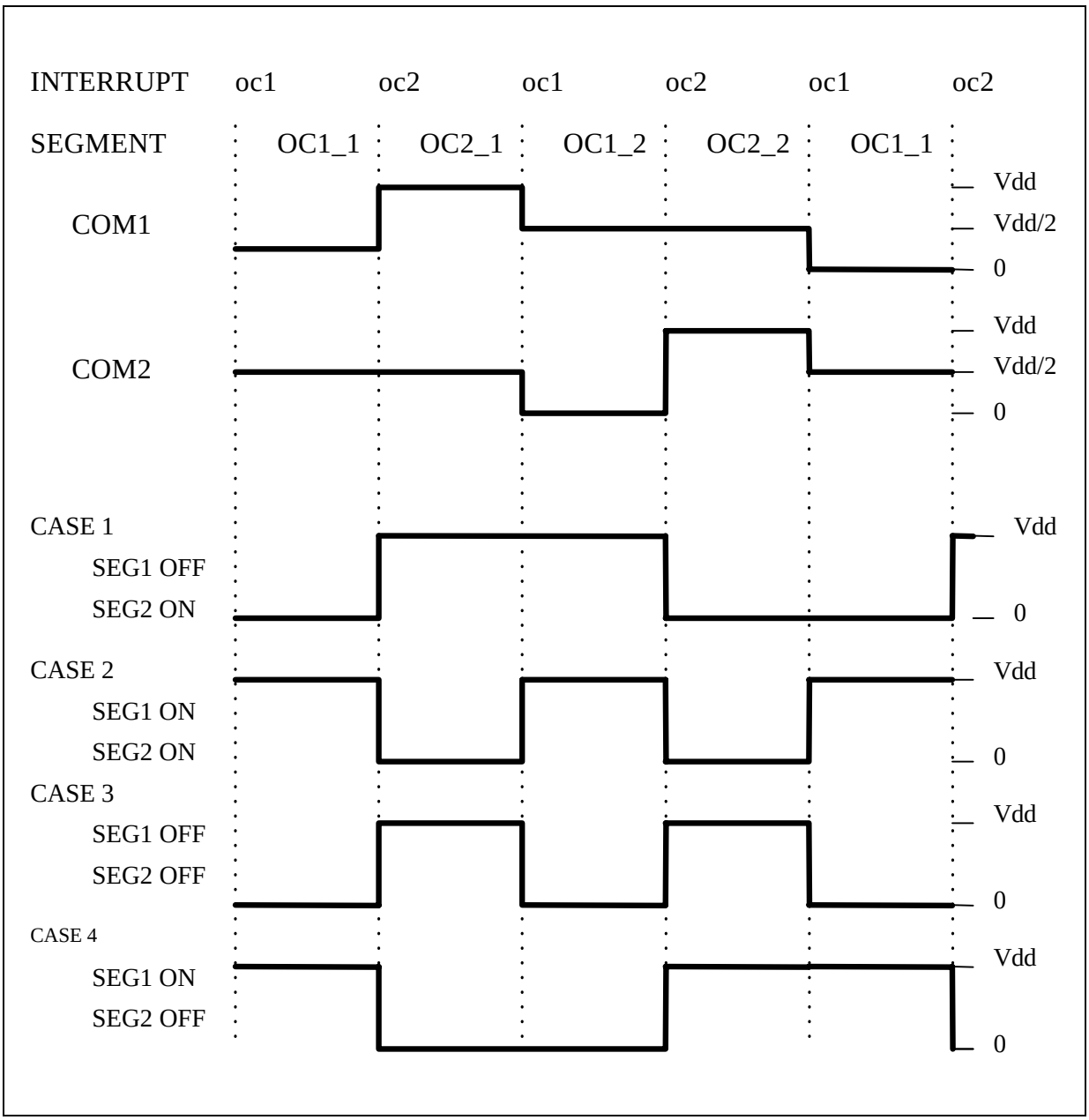
```
        LD A,#$E6
        LD taoc1lr,A       ; ouput compare 1 IT when the counter reaches 270Dh
        LD A,#$07
        LD taoc2hr,A       ; ouput compare 2 IT when the counter reaches 4E1D
        LD A,#$CE
        LD taoc2lr,A
        LD A,#$40
        LD tacr1,A        ; Timer output compare interrupt enabled
        LD A,#$08
        LD tacr2,A        ; Timer counter clock is CPU clk/8
        RET

;*************************************************************
; Port initialisation for LCD pins
;*************************************************************
                    ; CAUTION!! This subroutine has to be runned just before
                    ;the lcd program is run,
                     ; because lcd pins must never be connected
                     ; to steady levels. As long as lcd is not used, leave
                     ; the ST7 pins in reset state (input) so that the lcd
                     ; is not supplied. You can configure them in any other
                     ; state except in ouput push pull!
.port_init
        LD A,#$FF         ; configure the whole portA in output push-pull
        LD paddr,A         ; they will be dedicated to the segment outputs
        LD paor,A

        LD A,#$C0        ; configure PB6 & PB7 as output open drain
        LD pbddr,A
        clr pbor         ; they will be dedicated to the backplane outputs

        ret
      END
```

Figure 6. LCD Timing and Timer interrupts



4 EXAMPLE OF DUPLEXED LOW CONSUMPTION LCD WITH THE ST72251

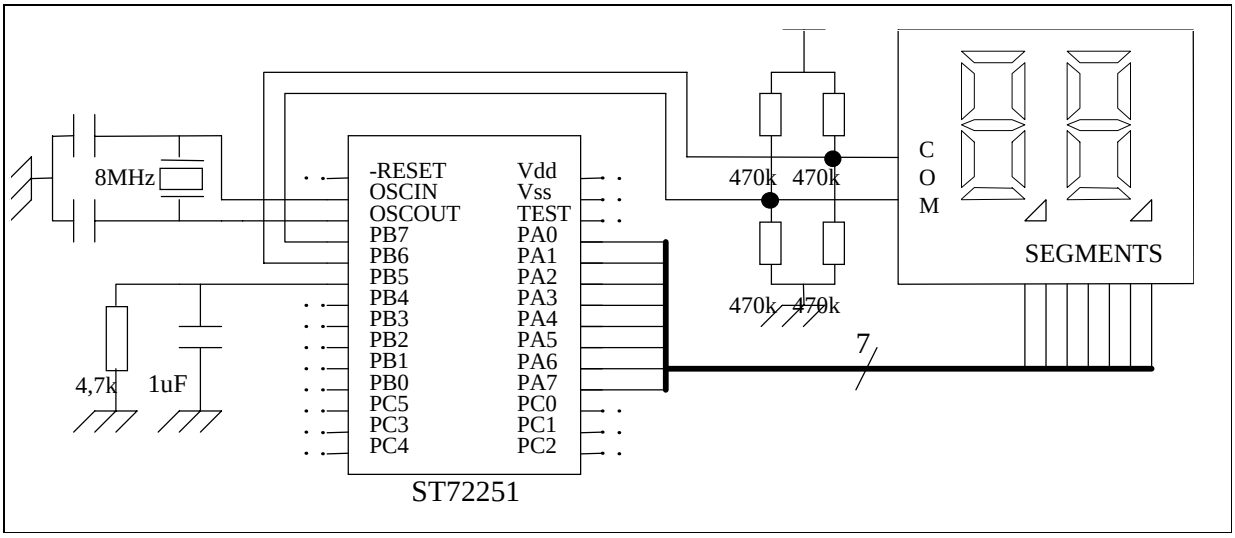
The following example describes the drive for a duplexed mode (2COM) LCD with the ST72251. Moreover this example targets low powered applications, as the ST7 goes in halt mode between two LCD refreshes. The only external components needed for driving the LCD are 4 resistors for the 3 level COM signals and 1 resistor and capacitor for the wake-up mechanism.

The ST72251 has 22 I/O pins (only 8 are configurable as open drain outputs) and can therefore drive an LCD up to a 32 segments and 2 backplanes. In this example only PA7..PA0 are used for the segment signals. PB7 and PB6 are used to generate both COM1&2 signals, this driver is able to drive a 16 segment LCD. Nevertheless this program can be easily changed to support a maximum of 32 segments.

The LCD timing generation is not done using the TimerA because to reduce power consumption to a very low level the ST7 is put in Halt mode between 2 LCD pin refreshes . To wake the ST7 up after it has been put in Halt mode, an external interrupt has to be generated. This is done by an external RC circuitry. While the program refreshes the LCD pins, the capacitor is charged by configuring the PA5 pin as output push pull with a high level. Before going in Halt mode the PA5 pin is configured as input pull up with interrupt. As a result the capacitor discharges until the PA5 pin reaches the low level specification (0.3. Vdd) and the ST7 is woken up. The LCD pins can be refreshed again before the ST7 is put in Halt mode once more....

The LCD driver consists of 2 initialization routines (the port initialization routine "port_init" and the LCD start routine "start_lcd"), the portB&C interrupt service routine (main part of the driver) and 3 variables: "lcdcr", "seg1" and "seg2". To activate the LCD two initialization routines have to be called: the port initialization routine "port_init" and "start_lcd" which will cause the first PA5 external interrupt. Once these routines are called the ST7 will get the PA5 external interrupt, which will generate the correct LCD signals, according to "seg1" and "seg2" indicating which segment has to be turned ON or OFF. After the interrupt it is up to the user program to put the ST7 in Halt mode. You can therefore add your own program to this one.

Figure 7. ST72251 example - Hardware



4.1 LISTING 2. ST72251 EXAMPLE - SOFTWARE

The code given below is for guidance only.

```
st7/

;******************************************************************************
; TITLE:         LCD.ASM
; AUTHOR:        Microcontroller Application Team
; DESCRIPTION:      Software driver for a 2 backplane LCD display
;             This lcd driver version is designed for low power
;             applications: The ST7 can go in halt mode between two
;             lcd refreshes: An external RC circuit generates an I/O
;             interrupt, which wakes up the ST7
;
;******************************************************************************
       TITLE  "LCD.ASM"

    MOTOROLA
    .NOLIST
    #INCLUDE "ST72251.inc" ; include st7225 registers and memory mapping file
    #INCLUDE "variable.inc"     ; include general variable file
    .LIST

;******************************************************************************


;******************************************************************************
;    Variables, constants defined and referenced locally
;    You can define your own values for a local reference here
;******************************************************************************

;******************************************************************************
;    Public routines  (defined here)
;******************************************************************************
; routines

; We could have used a dot in front of the label's name to declare the routine
; as visible by others modules in the project.

;******************************************************************************
;    Extern routines  (defined elsewhere)
;
;    The EXTERN directive will be seen by the linker as a call
;    instruction to another routine written in another file
```

```
;*************************************************************************
; routines

;*********************************************************
;    Program code
;*********************************************************

    WORDS

  segment 'rom'

;*************************************************************
; lcd driver - Several I/O pins are used to emulate segment and common lcd signals.
;Between 2 lcd refreshes (pins level update) the ST7 can go in halt mode in order to
;reduce power consumption to a very low level.
;One ST7 pin is used to generate an external interrupt for every lcd refresh.
;To get this external interrupt an RC circuit is connected to this pin. When the
;the ST7 is in halt mode the pin is configured as an input pull-up with interrupt.
; As a result of the capacitor discharge the pin goes low until it reaches
; the low level specification. An external interrupt is then generated, that wakes
; up the ST7. First the RC circuit is charged again configuring the pin as output
; push push with a high level. Then the lcd segment and common pins are refreshed
; before setting the ST7 in halt mode again. And so on ...
;
;       Calibrating the RC circuit: cf figure 8
;
;        If you want to choose different values, these are the key points:
;             - If you choose a different C value, take care to update the wait
;               loop in the interrupt routine so that the capacitor is fully charged
;               when leaving the interrupt. 100us is needed for these values.
;             - For R, choose a far smaller value than the internal resistor so that
;               the unprecise internal resistor value does not have a big influence
;               in the lcd frequency.
;             - Choose R so that the discharged value R/(R+Rint).Vcc is lower than
;               the interrupt voltage (0.3 Vdd). Then choose C to reach the lcd
;               frequency specification
;
; DESCRIPTION:
; - PB7 is dedicated to COM1, and PB6 to COM2. They provide 3 levels thanks to an
;external pull up
; and pull down resistor for each pin, and the open drain output feature: 0, Vdd/2,
;and Vdd
; - For segment signals which are normal digital signals (0 or 1) PA[0..7] are used
;as output
;  push pull
```

```
; - One lcd period is made of 4 time subdivisions: subdiv1, subdiv2, subdiv3,
;subdiv4. To know
; which subdivision is the next one at the next ST7 I/O interrupt, the 2 lcd
;control register lcdcr
;low bits are used: subdivx = lcdcr[1..0] + 1
;
;            TIMING (SEGi linked to COMi): cf figure 9
;
;*****************************************************************
;***********I/O INTERRUPT ROUTINE*********************************
.ext1_rt
        BTJF pbdr,#5,begin_lcd   ; test if the I/O interrupt source is lcd PB5 pin
        jp no_lcd

begin_lcd    LD A,pbdr         ;|\
        OR A,#$20          ;|
        LD pbdr,A          ;|
        LD A,pbor
        AND A,#$DF
        LD pbor,A          ; configure PB5 as an ouput push pull in high state
        LD A,pbddr
        OR A,#$20
        LD pbddr,A
        LD A,pbor           ;|
        OR A,#$20           ;|
        LD pbor,A           ;|/

    ; ###############User program computed in the wake-up I/O interrupt########

        LD Y,#$3F          ;|\
        LD cpt1,Y          ; wait loop so that the capacitor can be fully
wait1      DEC cpt1          ; charged again
        JRNE wait1         ;|/

        LD A,lcdcr
        AND A,#$01
        TNZ A
        JREQ oc1
        CPL padr          ; NOTE (1): segment signal toggles every oc2 IT
        LD A,lcdcr
        AND A,#$02
        TNZ A
        JRNE oc2_2
            ;### actions for oc2_1 segment (1)###
            LD A,pbdr    ; in oc2_1, COM1 = Vdd and COM2 = Vdd/2
```

```
            OR A,#$80    ; PB7 is already an output push pull,
            LD pbdr,A    ; PB6 is already an output open drain
                    ; so only nead to set bit 7 of pbdr
            ;### end of actions for oc2_1 segment ###
        JRT out
            ;### actions for oc2_2 segment (1)###
oc2_2          LD A,pbdr    ; in oc2_1, COM1 = Vdd and COM2 = Vdd/2
            OR A,#$40    ; PB7 is already an output push pull,
            LD pbdr,A    ; PB6 is already an output open drain
            ;### end of actions for oc2_2 segment ###
        JRT out
oc1       CLR X
        LD A,lcdcr
        AND A,#$02
        TNZ A
        JREQ no_incX
        INC X
no_incX     LD A,(seg1,X)  ; in this subdivision oc1_i
        LD padr,A    ; one of the segment i: 1 if segment is ON. Otherwise 0.
        TNZ X
        JRNE oc1_2
            ;### actions for oc1_1 segment ###
            LD A,pbdr
            AND A,#$7F
            LD pbdr,A
            LD A,pbor    ; in oc1_1, COM1 = 0 and COM2 = Vdd/2
            OR A,#$80    ; so PB7 is an output push pull,
            AND A,#$BF    ; and PB6 an output open drain
            LD pbor,A
            ;### end of actions for oc1_1 segment ###
        JRT out
            ;### actions for oc1_2 segment ###
oc1_2          LD A,pbdr
            AND A,#$BF
            LD pbdr,A
            LD A,pbor    ; in oc1_2, COM1 = Vdd/2 and COM2 = 0
            AND A,#$7F    ; so PB7 is an output open drain,
            OR A,#$40    ; and PB6 an output push pull
            LD pbor,A
            ;### end of actions for oc1_2 segment ###
out
            LD A,lcdcr    ;|\
            INC A      ;|
            CP A,#$04    ; 2bits counter lcdcr[1..0]. Used to know
            JRC end_inc   ; which subdivision is to be applied next
```

```
            CLR A      ;|
end_inc         LD lcdcr,A   ;|/
    ; ######################end of user program################################

        LD A,pbor         ;|\
        AND A,#$DF         ;|
        LD pbor,A
        LD A,pbddr
        AND A,#$DF     ; configure PB5 as a pull-up with interrupt again
        LD pbddr,A
        LD A,pbor         ;|
        OR A,#$20         ;|
        LD pbor,A         ;|/

no_lcd
        IRET

;***********lcd start program*********************************

.start_lcd
        CLR A          ; init lcdcr, lcd control register
        LD lcdcr,A
;    >PORT A initialisation for lcd
        LD A,#$FF        ; configure the whole portA as output push-pull
        LD paddr,A        ; they will be dedicated to the segment outputs
        LD paor,A
;    >PORT B initialisation for lcd
        LD A,pbddr        ;|\
        OR A,#$C0        ;|
        AND A,#$DF        ; configure PB6 & PB7 as output push-pull
        LD pbddr,A        ; they will be dedicated to the backplane outputs
        LD A,pbor        ; configure PB5 as a pull-up with interrupt. As a result
        AND A,#$DF        ; the external capacitance will discharge, leading to
        LD pbor,A        ; the first I/O interrupt for lcd.
        LD A,pbor        ;
        OR A,#$E0        ;>>WARNING<<: This configuration implies that PB[7..5]
        LD pbor,A        ; and PBx were in their reset state (input pull_down no
                    ; interrupt)when calling this routine. If this is not the
                    ; case check carefully if no forbidden I/O pin
                    ; state switch is applied (see the ST72251 databook).
                    ; If necessary, change this configuration.
                   ;|
        RET           ;|/
```

```
;*****************************************************************
; Ports drivers
;*****************************************************************
.port_init

;***********PORT A initialisation ************
;Port A is dedicated to lcd segment signals. Before starting to drive the lcd
;(before calling start_lcd port A pins must remain unsupplied
;(port pins configured as either input or open drain ). As a result the pin reset
;states (inputs)don't need to be changed.

;***********PORT B initialisation ************

        LD A,#$20        ; configure PB5 as output push pull initially
        LD pbddr,A       ; in order to get the capacitor charged when starting
        LD pbdr,A        ; the lcd.
                    ; As long as the lcd is not used, PB7 and PB6, used for
                    ; the lcd backplane signal, can remain in their reset
                    ; state for the same reason as for the port B pins.
                    ; Warning: overwrite the PB4..0 pins configuration.!!!
                    ; You perhaps need to modify this to include the
                    ; configuration of PB4..0. Don't forget to avoid
                    ; forbidden I/O configuration switches (see data sheet)!
        RET
      END
```

Figure 8. Calibrating the RC circuit

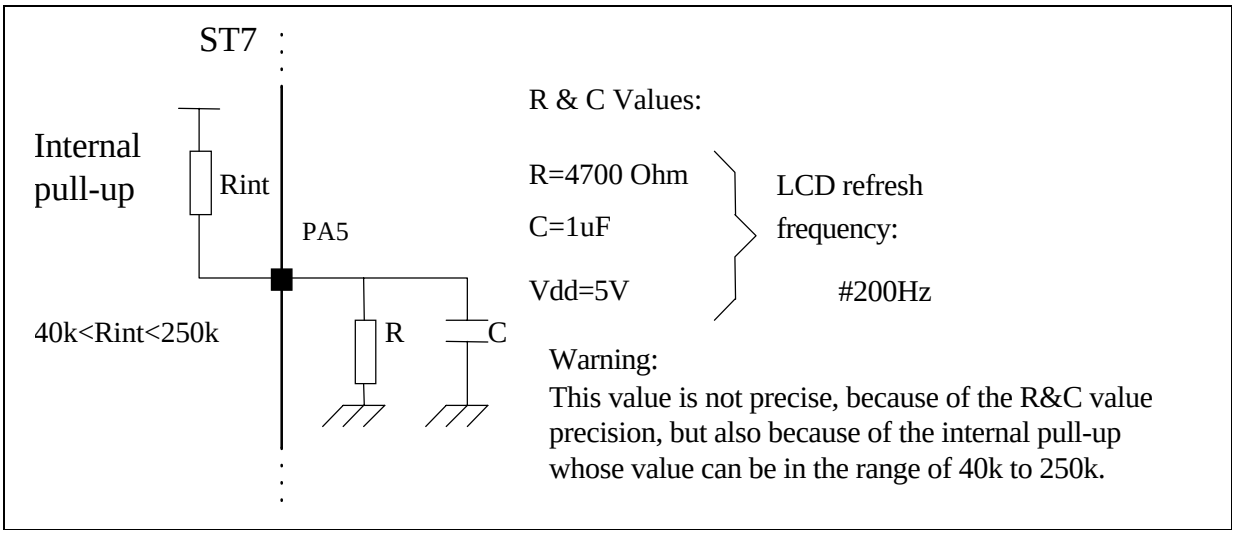
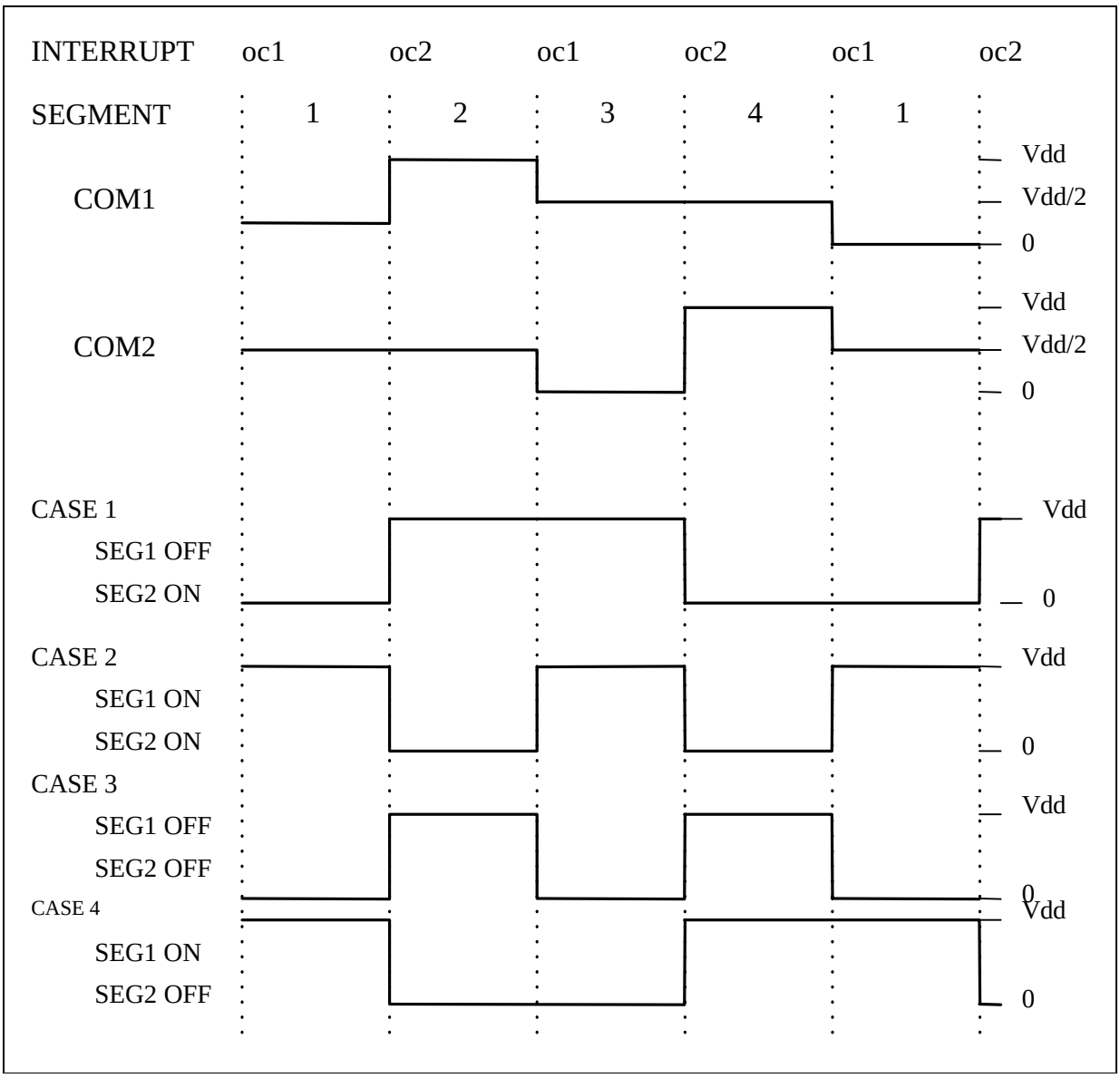


Figure 9. LCD timing



THE PRESENT NOTE WHICH IS FOR GUIDANCE ONLY AIMS AT PROVIDING CUSTOMERS WITH INFORMATION REGARDING THEIR PRODUCTS IN ORDER FOR THEM TO SAVE TIME. AS A RESULT, STMICROELECTRONICS SHALL NOT BE HELD LIABLE FOR ANY DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WITH RESPECT TO ANY CLAIMS ARISING FROM THE CONTENT OF SUCH A NOTE AND/OR THE USE MADE BY CUSTOMERS OF THE INFORMATION CONTAINED HEREIN IN CONNEXION WITH THEIR PRODUCTS.