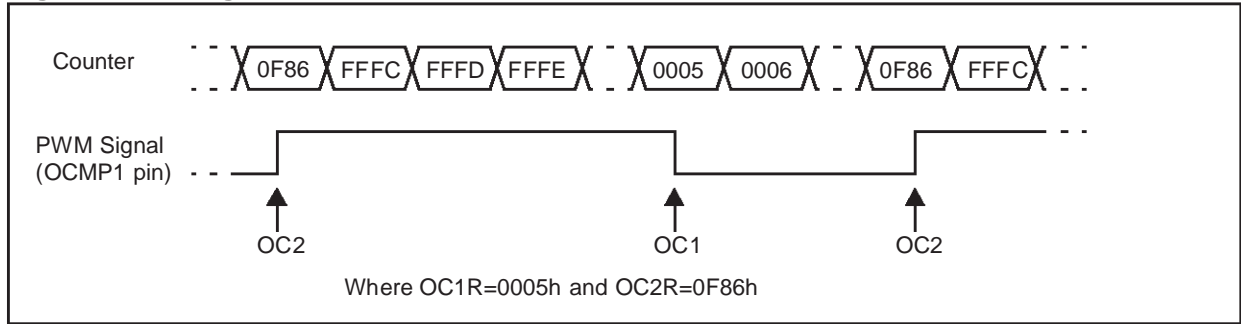


ONE TONE GENERATION (FREQUENCY)

The value written in the OC1R register corresponds to the elapsed time when the OCMP1 pin is maintained to V_{ss} in a period time given by OC2R register (see Figure 2.). To insure these transitions the OLVL1 bit of the Control Register 1 (CR1) is cleared and the OLVL2 bit of the same register is set.

Figure 2. Timing of the ST7 Timer PWM Mode



2 TONE NOTE GENERATION (FREQUENCY)

In a basic music synthesizer application the tone note is given by the output frequency of the microcontroller input of the buzzer (see Table 1). In the ST7 Timer interface this frequency is given by the value written in the OC2R register.

To ensure the best accuracy, the fastest timer free running counter speed has to be chosen. Then, in front of the required frequencies and a f_{CPU} at 4-MHz the best timer clock prescaler is $f_{CPU}/2$ (CC1,CC0="01" in the CR2 register).

As the free running counter of the ST7 Timer (in PWM mode) is reloaded with FFFCh when it matches the Output Compare 2, the corresponding time base for each note is given by the following equation in Table 1.

$$OC2R = \frac{f_{CPU}/2}{f_{TONE}}$$

Table 1. Rough Frequencies of the Basic Music Tones

Music Tone Notes	SI ₀	DO	RE	MI	FA	SOL	LA	SI	DO ₂
Tone Frequency [Hz]	503	524	587	662	701	787	878	1004	1048
OC2R [hex]	0F86	0EE4	0D4A	0BC8	0B20	09E8	08E1	07C3	0774

The "mute" tone can be obtained through a constant level on the pin. This level is obtained when $OC2R < OC1R$.

3 MUSIC VOLUME SETTING (DUTY CYCLE)

The volume control of the basic music synthesizer is done by the duty cycle control of the generated PWM signal. In timer PWM mode, this control is done through the OC1R register for each music tone notes (see Table 2).

The volume control scale depends on the chosen buzzer. The scale chosen in this application has 255 different steps. It is based on an around 0.25% high level duty cycle which allows a range of possible high duty cycle between ~0.25% and ~64%.

Table 2. Rough High Duty Cycle Bases of the Basic Music Tones

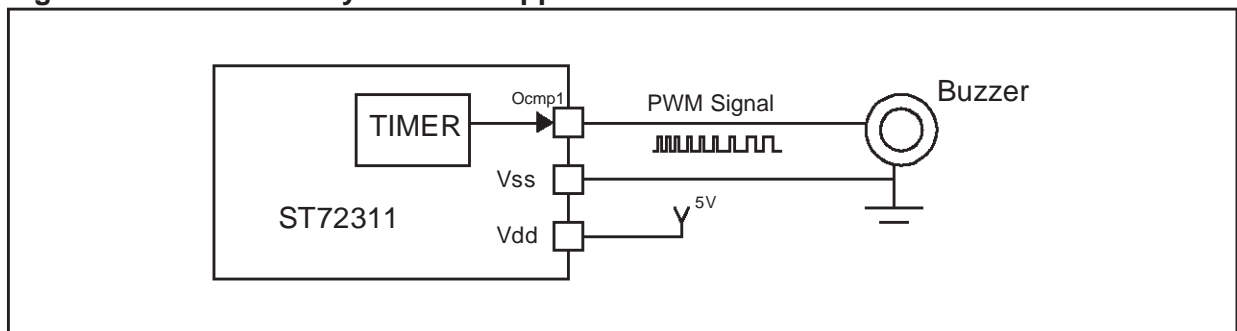
Music Tone Notes	SI ₀	DO	RE	MI	FA	SOL	LA	SI	DO ₂
OC2R [hex]	0F86	0EE4	0D4A	0BC8	0B20	09E8	08E1	07C3	0774
OC1R [hex]	000A	0009	0008	0007	0007	0006	0005	0005	0004
Duty cycle base [%]	0.25	0.24	0.24	0.23	0.25	0.24	0.22	0.25	0.21

4 BASIC MUSIC SYNTHESIZER APPLICATION

4.1 HARDWARE CONFIGURATION

The basic music synthesizer application hardware is made of a ST72311 microcontroller timer which generates a PWM signal to a buzzer VSB35EW-0701B (MURATA) as described in Figure 3.

Figure 3. Basic Music Synthesizer Application



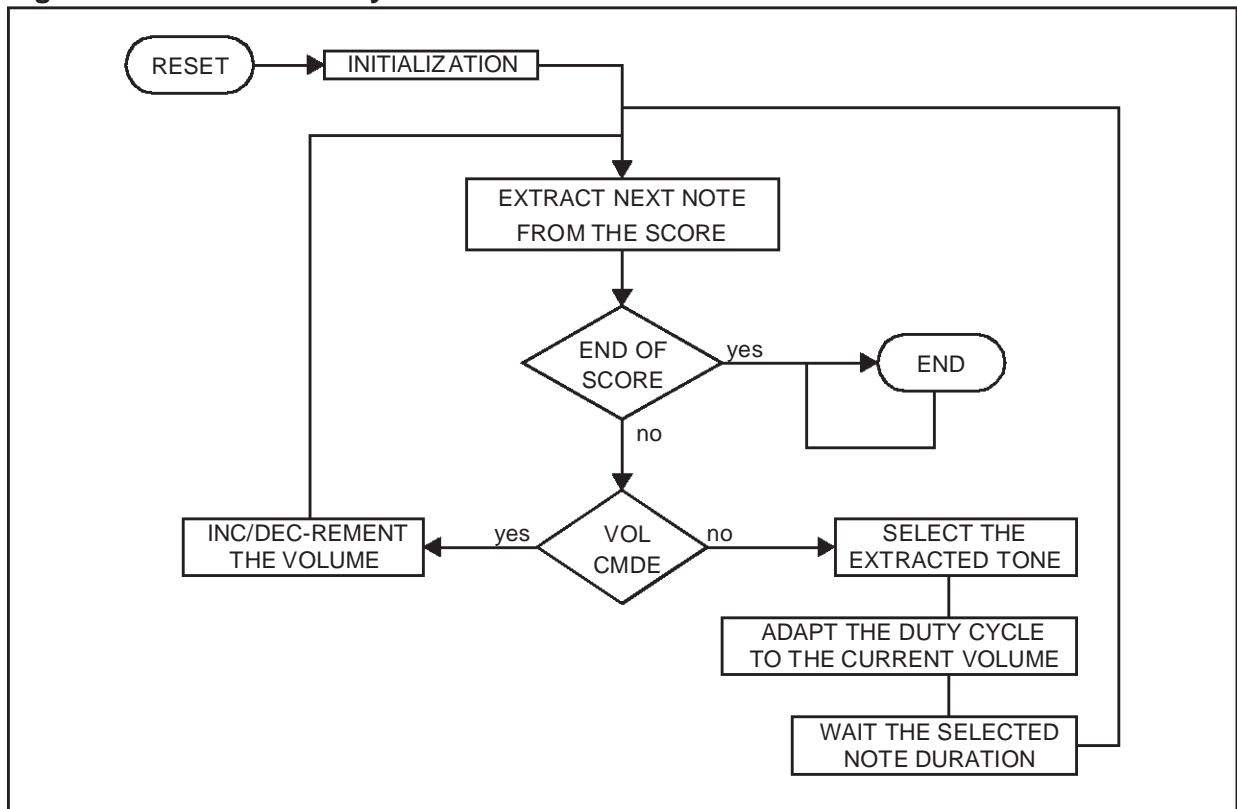
4.2 MUSIC SCORE

A music score is made of a sequence of 8-bit information data which can be:

- a coded music note
- a mute command
- a volume control command
- a score "end" flag.

The flowchart given in Figure 4. shows how a music score is played in this synthesizer application.

Figure 4. Music Score Play Flowchart

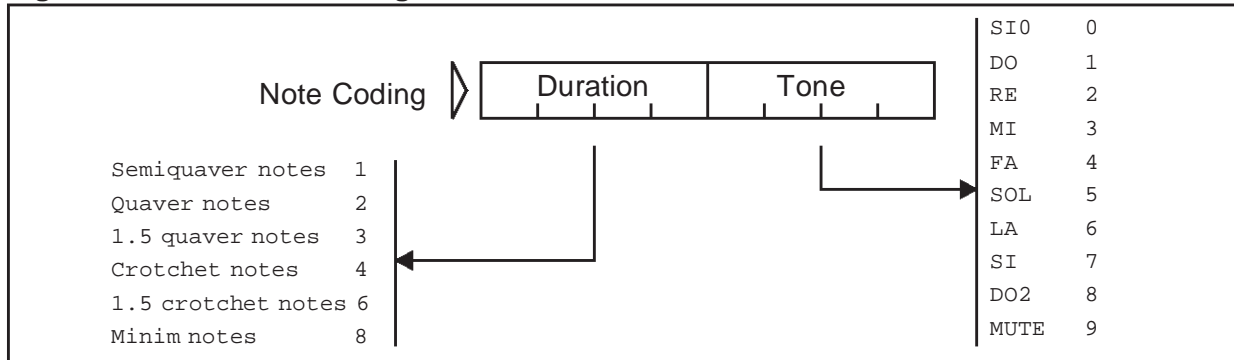


4.3 MUSIC NOTE CODING

To insure an easy music score management, a tricky music note coding is implemented. Each note is composed of two different fields: its tone (frequency) and its duration. These two fields are coded in one data byte as it is shown in Figure 5.

In this application, the music note range is based on an enhanced octave and a mute command.

Figure 5. Music Note Coding



The note duration field is based on a polling wait loop with a same tone. The coded duration [1,2,3,4,6,8] corresponds to the time of basic loop to be computed.

The tone information gives the index in the Output Compare registers table where the music note frequency and their duty cycle are coded (see Table 2). The mute index corresponds to a non audible frequency not shown in the table.

The initialization corresponds to a 100% duty cycle producing a mute tone.

4.4 VOLUME CONTROL

The volume is controlled through the duty cycle of the PWM signal.

A basic duty cycle is associated to each note as it is mentioned in Table 2. From this basic duty cycle, the volume is determined thanks to an 8-bit variable which allows 255 steps.

The volume change during a score is done through two different 8-bit commands which allow a decrement (F0h) or an increment (F1h) of the volume.

The initial volume level is given by the minimum high duty cycle which means around 0.25%.


```

#define q      +$20}                ; Quaver notes.
#define qp     +$30}                ; 1.5 quaver notes.
#define c      +$40}                ; Crotchet notes.
#define cp     +$60}                ; 1.5 crotchet notes.
#define m      +$80}                ; Minim notes.

;*****
;  RAM SEGMENT
;*****

        BYTES                        ; following addresses are 8 bit length
        segment byte at 80-FF 'ram0'

.DutyCycle DS.B 1                    ; DutyCycle percentage in [0..99].

;*****
;  ROM SEGMENT
;*****

        WORDS
        segment 'rom'

; Music note coding ~~~~~~
;          SI0 DO RE MI FA SOL LA SI DO2 MUTE
.Note_h   DC.B $0F,$0E,$0D,$0B,$0A,$09,$08,$07,$07,$00 ; OCR1 high byte
.Note_l   DC.B $86,$E4,$4A,$C8,$20,$E8,$E1,$C3,$74,$44 ; OCR1 low byte
.DutyTime DC.B $0A,$09,$08,$07,$07,$06,$05,$05,$04,$88 ; ~0.25% at Vdd

.Score ; Music score ~~~~~~

; "Au clair de la Lune..."
DC.B _M m,_DO c,_M sq,_DO c,_M sq,_DO c,_RE c,_MI m,_RE m,_M sq
DC.B _DO c,_MI c,_RE c,_M sq,_RE c,_DO m,_VOLi,_VOLi,_VOLi

; "J'ai du bon tabac..."
DC.B _M m,_DO c,_RE c,_MI c,_DO c,_RE m,_M sq,_RE c,_MI c,_FA c,_M sq
DC.B _FA cp,_MI c,_M sq,_MI cp,_M c,_DO c,_RE c,_MI c,_DO c,_RE m
DC.B _M sq,_RE c,_MI c,_FA m,_SOL m,_DO m,_DO m,_VOLi,_VOLi,_VOLi

; "Hymne a la joie"
DC.B _M m,_MI c,_M sq,_MI c,_FA c,_SOL c,_M sq,_SOL c,_FA c,_MI c
DC.B _RE c,_DO c,_M sq,_DO c,_RE c,_MI c,_M sq,_MI m,_RE m,_MI c
DC.B _M sq,_MI c,_FA c,_SOL c,_M sq,_SOL c,_FA c,_MI c,_RE c,_DO c
DC.B _M sq,_DO c,_RE c,_MI c,_RE m,_DO m,_VOLi,_VOLi,_VOLi

; "Star War"
DC.B _M m,_DO m,_SOL m,_FA q,_MI q,_RE q,_DO2 m,_SOL m
DC.B _FA q,_MI q,_RE q,_DO2 m,_SOL m,_FA q,_MI q,_FA q,_RE m,_RE m
DC.B _M c,_DO m,_SOL m,_FA q,_MI q,_RE q,_DO2 m,_SOL m

```

SOFTWARE

```
DC.B _FA q, _MI q, _RE q, _DO2 m, _SOL m, _FA q, _MI q, _FA q, _RE m, _RE m
DC.B _M c, _DO m, _FA c, _MI c, _RE c, _DO c, _M sq
DC.B _DO qp, _RE q, _DO q, _SI0 c, _DO c, _RE c, _M q,
; "End"
DC.B _M m, _END_

; Program code *****

; *****
; * SUB-ROUTINES LIBRARY SECTION *
; *****

; -----
; ROUTINE NAME : TBuzz_Init
; INPUT/OUTPUT : None.
; DESCRIPTION : TIMER initialization for PWM mode.
; COMMENTS :
; -----
.TBbuzz_Init ; IN: None / OUT: None.
    BRES PFOR, #4
    BSET PFDDR, #4 ; PF4 (OCMP1 of timer A used for the buzzer) is defined as an output.
    BRES TACR1, #0 ; OLVL1=0: Vss applied on OCMP1 pin when OC1 occurs.
    BSET TACR1, #2 ; OLVL2=1: Vdd applied on OCMP1 pin when OC2 occurs.
    LD A, #10010100 ; OC1E=1 OCMP1 pin is activated.
    LD TACR2, A ; PWM=1 Active the PWM mode.
    ; CC10=01 Timer clock=Fcpu/2.

    CLR A ; Nothing on the output OC1R=0 and OC2R=0.
    LD TAOC1HR, A
    LD TAOC2HR, A
    LD TAOC2LR, A
    INC A ; 100% Duty cycle at initialization.
    LD TAOC1LR, A
    LD A, #1 ; 0.25% DutyCycle for Music score.
    LD DutyCycle, A
    RET

; -----
; ROUTINE NAME : TBuzz_Wait
; INPUT/OUTPUT : Note duration (4bit MSB information) / None.
; DESCRIPTION : Polling wait routine for note duration (based on 250ms delay).
; COMMENTS : Quaver (Y=2), crotchet (Y=4) or minim (Y=8) duration selection
; -----
.TBbuzz_Wait ; IN: Y=duration / OUT: None.
    LD A, Y
    SWAP A
    AND A, #0F ; Use only the rythme information.
    LD Y, A
.Tbwt1 LD X, #13
.Tbwt2 LD A, #255
```



```

.Tbwt3 DEC A
      JRNE Tbwt3
      DEC X
      JRNE Tbwt2
      DEC Y
      JRNE Tbwt1
      RET

; -----
; ROUTINE NAME : TBuzz_ChgDuty
; INPUT/OUTPUT : Note Index / None.
; DESCRIPTION  : Change the PWM duty cycle in the TIMER for the volume.
; COMMENTS    : Select the music volume for the current note.
; -----
.TBbuzz_ChgDuty ; IN: X=Note Index / OUT: None.
      LD  Y,DutyCycle           ; The duty cycle is given by the ratio
      LD  A,(DutyTime,X)       ; between the period length which gives the
      MUL Y,A                   ; the frequency (OC2 registers) and the OC1
      LD  TAOC1HR,Y            ; registers.
      LD  TAOC1LR,A
      RET

; -----
; ROUTINE NAME : TBuzz_ChgFreq
; INPUT/OUTPUT : New note (4bit LSB information) / None.
; DESCRIPTION  : Change the PWM frequency in the TIMER for the music tone.
; COMMENTS    : Select the music volume for the current note.
; -----
.TBbuzz_ChgFreq ; IN: A=New note / OUT: None.
      AND A,#$0F               ; Use only the note information.
      LD  X,A
      LD  A,(Note_h,X)         ; The frequency is given by the PWM period
      LD  TAOC2HR,A           ; set through the OC2 registers.
      LD  A,(Note_l,X)
      LD  TAOC2LR,A
      RET

; -----
; ROUTINE NAME : TBuzz_PlyScor
; INPUT/OUTPUT : None.
; DESCRIPTION  : Play a music score.
; COMMENTS    :
; -----
.TBbuzz_PlyScor ; IN: None / OUT: None.
      CLR X
.PlyscrLD A,(Score,X)         ; Extract the note from the score.

```

SOFTWARE

```

        CP   A,#_END_                ; Check if the score is finished.
        JREQ Plyend
        CP   A,#_VOLd                ; Check if it is a volume instruction to be decode.
        JRULTPlynnot
        JREQ Plydec    ; Decoding of a volume control instruction in the score.
        INC  DutyCycle                ; Volume increment decoded.
        JRT   Plynxt
.PlydecDEC DutyCycle                ; Volume decrement decoded.
        JRT   Plynxt                ; Next note or instruction in score.
.PlynnotPUSH X                      ; Push X to be used in TBuzz_ChgFreq.
        PUSH A                ; A used in TBuzz_ChgFreq: Push A for the rythme.
        CALL TBuzz_ChgFreq    ; Change the music tone according to the score.
        CALL TBuzz_ChgDuty    ; Change or update the volume level.
        POP  Y                ; Load Y for the rythme.
        CALL TBuzz_Wait       ; Polling wait routine for note duration.
        POP  X                ; Restore the score counter in X from stack.
.PlynxtINC X; Next note.
        JRA  Plyscr
.PlyendRET

; *****
; *      MAIN-ROUTINES SECTION *
; *****

.main  CALL TBuzz_Init                ; Init TIMER peripheral.
.end   CALL TBuzz_PlyScor              ; Play score.
        JRA  end                      ; Infinity main loop.

; *****
; *  INTERRUPT SUB-ROUTINES LIBRARY SECTION *
; *****

.dummy_rt  iret                ; Empty subroutine. Go back to main (iret instruction).

        segment 'vectit'
                DC.W  dummy_rt        ; FFF0-FFF1h location
                DC.W  dummy_rt        ; FFF2-FFF3h location
                DC.W  dummy_rt        ; FFF4-FFF5h location
sci_it:    DC.W  dummy_rt        ; FFE6-FFE7h location
timb_it:   DC.W  dummy_rt        ; FFE8-FFE9h location
tima_it:   DC.W  dummy_rt        ; FFEA-FFEBh location
spi_it:    DC.W  dummy_rt        ; FFEC-FFEDh location
                DC.W  dummy_rt        ; FFEF-FFEFh location
ext3_it:   DC.W  dummy_rt        ; FFF0-FFF1h location
ext2_it:   DC.W  dummy_rt        ; FFF2-FFF3h location
ext1_it:   DC.W  dummy_rt        ; FFF4-FFF5h location
ext0_it:   DC.W  dummy_rt        ; FFF6-FFF7h location

```

```
                DC.W  dummy_rt      ; FFF8-FFF9h location
                DC.W  dummy_rt      ; FFFA-FFFBh location
softtit:        DC.W  dummy_rt      ; FFFC-FFFDh location
reset:          DC.W  main          ; FFFE-FFFEh location
```

END

*** (c) 1998 STMicroelectronics ***** END OF FILE *****

SOFTWARE

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.

Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without the express written approval of STMicroelectronics.

The ST logo is a registered trademark of STMicroelectronics

©1998 STMicroelectronics - All Rights Reserved.

Purchase of I²C Components by STMicroelectronics conveys a license under the Philips I²C Patent. Rights to use these components in an I²C system is granted provided that the system conforms to the I²C Standard Specification as defined by Philips.

STMicroelectronics Group of Companies

Australia - Brazil - Canada - China - France - Germany - Italy - Japan - Korea - Malaysia - Malta - Mexico - Morocco - The Netherlands - Singapore - Spain - Sweden - Switzerland - Taiwan - Thailand - United Kingdom - U.S.A.

<http://www.st.com>