ECE 3120: Computer Systems
Chapter 8: ECE-3120-A Musical

Manjeera Jeedigunta
http://blogs.cae.tntech.edu/msjeedigun21
Email: msjeedigun21@tntech.edu
Tel: 931-372-6181, Prescott Hall 120
Output Compare Function

- The HCS12 has eight output compare functions.
- Each output compare channel consists of
  - A 16-bit comparator
  - A 16-bit compare register TCx (also used as inout capture register)
  - An output action pin (PTx, can be pulled high, pulled low, or toggled)
  - An interrupt request circuit
  - A forced-compare function (CFOCx)
  - Control logic
Operation of the Output-Compare Function

- One of the applications of the output-compare function is to trigger an action at a specific time in the future.

- To use an output-compare function, the user
  - Makes a copy of the current contents of the TCNT register
  - Adds to this copy a value equal to the desired delay
  - Stores the sum into an output-compare register (TCx, x = 0..7)

- The actions that can be activated on an output compare pin include:
  - Pull up to high
  - Pull down to low
  - Toggle
Operation of the Output-Compare Function

- The action is determined by the Timer Control Register 1 & 2 (TCTL1 & TCTL2):
- A successful compare will set the corresponding flag bit in the TFLG1 register.
- An interrupt may be optionally requested if the associated interrupt enable bit in the TIE register is set.

<table>
<thead>
<tr>
<th>Value after reset</th>
<th>OM7</th>
<th>OL7</th>
<th>OM6</th>
<th>OL6</th>
<th>OM5</th>
<th>OL5</th>
<th>OM4</th>
<th>OL4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

(a) TCTL1 register

<table>
<thead>
<tr>
<th>Value after reset</th>
<th>OM3</th>
<th>OL3</th>
<th>OM2</th>
<th>OL2</th>
<th>OM1</th>
<th>OL1</th>
<th>OM0</th>
<th>OL0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

(b) TCTL2 register

- read: anytime
- write: anytime

OMn  OLn : output level
- 0  0  no action (timer disconnected from output pin)
- 0  1  toggle OCn pin
- 1  0  clear OCn pin to 0
- 1  1  set OCn pin to high

Figure 8.18 Timer control register 1 and 2 (TCTL1 & TCTL2)
Making Sound Using the Output-Compare Function

- A sound can be generated by creating a digital waveform with appropriate frequency and using it to drive a speaker or a buzzer.
- The circuit connection for a buzzer is shown in Figure 8.21.
- The simplest song is a two-tone siren.

![Circuit connection for a buzzer](image-url)
Algorithm for Generating a Siren

- **Step 1**
  - Enable an output compare channel to drive the buzzer (or speaker).

- **Step 2**
  - Start an output compare operation with a delay count equal to half the period of the siren and enable the OC interrupt.

- **Step 3**
  - Wait for the duration of the siren tone (say half a second). During the waiting period, interrupts will be requested many times by the output compare function. The interrupt service routine simply restart the output compare operation.

- **Step 4**
  - At the end of the siren tone duration, choose a different delay count for the output compare operation so that the siren sound may have a different frequency.

- **Step 5**
  - Wait for the same duration as in Step 3. During this period, many interrupts will be requested by the output compare operation.

- **Step 6**
  - Go to Step 2.
Example 8.7 Write a program to generate a two-tone siren that oscillates between 300 Hz and 1200 Hz.

Solution:

- Set the prescaler to TCNT to 1:8.
- The delay count for the low frequency tone is \( \frac{24000000}{8} \div 300 \div 2 = 5000 \).
- The delay count for the high frequency tone is \( \frac{24000000}{8} \div 1200 \div 2 = 1250 \).

```assembly
#include "c:\miniide\hcs12.inc"
hi_freq equ 1250 ; delay count for 1200 Hz (with 1:8 prescaler)
lo_freq equ 5000 ; delay count for 300 Hz (with 1:8 prescaler)
toggle equ $04 ; value to toggle the TC5 pin
org $1000

delay ds.w 1 ; store the delay for output-compare operation
org $1500
lds #1500
movw #oc5_isr,UserTimerCh5 ; initialize the interrupt vector entry
movb #$90,TSCR1 ; enable TCNT, fast timer flag clear
movb #$03,TSCR2 ; set main timer prescaler to 8
```
bset   TIOS,OC5       ; enable OC5
movb   #toggle,TCTL1   ; select toggle for OC5 pin action
movw   #hi_freq,delay  ; use high frequency delay count first
ldd    TCNT           ; start the low frequency sound
addd   delay          ; "
std    TC5            ; "
bset   TIE,OC5        ; enable OC5 interrupt
cli               ; "
forever ldy   #5            ; wait for half a second
    jsr   delayby100ms     ; "
movw   #lo_freq,delay ; switch to low frequency delay count
    ldy   #5
    jsr   delayby100ms
    movw   #hi_freq,delay ; switch to high frequency delay count
bra    forever
oc5_isr ldd   TC5       ; enable OC5 interrupt
    addd   delay
    std    TC5
    rti
#include c:\miniide\delay.asm”
end
Playing Songs Using the OC Function

- Place the frequencies and durations of all notes in a table.
- For every note, use the output-compare function to generate the digital waveform with the specified frequency and duration.
- The next example plays the US national anthem.
The Star-Spangled Banner

#include "c:\miniide\hcs12.inc"
G3 equ 7653 ; delay count to generate G3 note (with 1:8 prescaler)
B3 equ 6074 ; delay count to generate B3 note (with 1:8 prescaler)
C4 equ 5733 ; delay count to generate C4 note (with 1:8 prescaler)
C4S equ 5412 ; delay count to generate C4S (sharp) note
D4 equ 5108 ; delay count to generate D4 note (with 1:8 prescaler)
E4 equ 4551 ; delay count to generate E4 note (with 1:8 prescaler)
F4 equ 4295 ; delay count to generate F4 note (with 1:8 prescaler)
F4S equ 4054 ; delay count to generate F4S note (with 1:8 prescaler)
G4 equ 3827 ; delay count to generate G4 note (with 1:8 prescaler)
A4 equ 3409 ; delay count to generate A4 note (with 1:8 prescaler)
B4F equ 3218 ; delay count to generate B4F note (with 1:8 prescaler)
B4 equ 3037 ; delay count to generate B4 note (with 1:8 prescaler)
C5 equ 2867 ; delay count to generate C5 note (with 1:8 prescaler)
D5 equ 2554 ; delay count to generate D5 note (with 1:8 prescaler)
E5 equ 2275 ; delay count to generate E5 note (with 1:8 prescaler)
F5 equ 2148 ; delay count to generate F5 note (with 1:8 prescaler)
notes equ 101
toggle equ $04 ; value to toggle the TC5 pin
org $1000

delay ds.w 1 ; store the delay for output-compare operation
rep_cnt ds.b 1 ; repeat the song this many times
ip ds.b 1 ; remaining notes to be played

org $1500
lds #$1500

; establish the SRAM vector address for OC5
movw #oc5_isr,UserTimerCh5
movb #$90,TSCR1 ; enable TCNT, fast timer flag clear
movb #$03,TSCR2 ; set main timer prescaler to 8
bset TIOS,OC5 ; enable OC5
movb #toggle,tctl1 ; select toggle for OC5 pin action
ldx #score ; use as a pointer to score table
ldy #duration ; points to duration table
movb #1,rep_cnt ; play the song once
movb #notes,ip
movw 2,x+,delay ; start with zeroth note
ldd TCNT ; play the first note
addd delay ; "
std TC5 ; "
bset TIE,C5I ; enable OC5 interrupt
cli ; "
forever  pshy ; save duration table pointer in stack
ldy  0,y ; get the duration of the current note
jsr  delayby10ms ; "
puly ; get the duration pointer from stack
iny ; move the duration pointer
iny ; "
ldd 2,x+ ; get the next note, move pointer
std  delay ; "
dec  ip
bne forever
dec  rep_cnt
beq  done ; if not finish playing, re-establish
ldx  #score ; pointers and loop count
ldy  #duration ; "
movb  #notes,ip ; "
movw  0,x,delay ; get the first note delay count
ldd  TCNT ; play the first note
addd  #delay ; "
std  TC5
bra  forever
done  swi
; The following subroutine creates a time delay which is equal to [Y] times 10 ms. The timer prescaler is 1:8.

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

; delayby10ms

again1

wait_lp1

oc5_isr

l

add

std

rti

bset TIOS,OC0 ; enable OC0

ldd TCNT

addd #30000 ; start an output-compare operation

std TC0 ; with 10 ms time delay

brclr TFLG1,C0F,wait_lp1

ldd TC0

dbne y,again1

bclr TIOS,OC0 ; disable OC0

rts
score dw D4,B3,G3,B3,D4,G4,B4,A4,G4,B3,C4S
   dw D4,D4,D4,B4,A4,G4,F4S,E4,F4S,G4,G4,D4,B3,G3
   dw D4,B3,G3,B3,D4,G4,B4,A4,G4,B3,C4S,D4,D4,D4
   dw B4,A4,G4,F4S,E4,F4S,G4,G4,D4,B3,G3,B4,B4
   dw B4,C5,D5,D5,C5,B4,A4,B4,C5,C5,C5,B4,A4,G4
   dw F4S,E4,F4S,G4,B3,C4S,D4,D4,G4,G4,G4,F4S
   dw E4,E4,E4,A4,C5,B4,A4,G4,G4,F4S,D4,D4
   dw G4,A4,B4,C5,D5,G4,A4,B4,C5,A4,G4
.

; Each of the following entries multiplied by 10 ms gives the duration of a note.
.
duration dw 30,10,40,40,40,80,30,10,40,40,40
   dw 80,20,20,60,20,40,80,20,20,40,40,40,40
   dw 30,10,40,40,40,80,30,10,40,40,40,80,20,20
   dw 60,20,40,80,20,20,40,40,40,40,20,20
   dw 40,40,40,80,20,20,40,40,40,80,40,60,20,40
   dw 80,20,20,40,40,80,40,40,40,20,20
   dw 40,40,40,40,20,20,20,20,40,40,40,20,20
   dw 60,20,20,20,80,20,20,60,20,40,80
end