Dr. Tao Li

EEL 4744C: Microprocessor Applications
Lecture 8

Timer

Reading Assignment

- Software and Hardware Engineering (new version): Chapter 14
- SHE (old version): Chapter 10
- HC12 Data Sheet: Chapters 12, 13, 11, 10

Introduction

- We want separate timing circuitry that runs independent of our main program
  - Our main program can't keep good timing due to unexpected things like IRQ's
- A timer is just a digital counter

Timer Overview

Timer functionality we will studied...
1) Main Free Running Timer (TCNT)
2) Timer Output Compare
3) Timer Input Capture
4) Pulse Accumulator
5) Real Time Interrupt
6) Pulse Width Modulator

Examples of application that use timer functionality
1) Count items on an assembly line
2) Generate a 30/70 duty cycle to control a motor
3) Measure phase of an incoming signal
4) Generate an edge every n seconds
5) Latch data from peripheral when an edge occurs
6) Generate interrupt every 50ms

An Overview of HC12 Timer

- 16-bit free-running counter based on system bus clock (e.g. E CLK)
- 8 timer channels, each configurable as:
  - Output compare: can generate variety of waveforms by comparing counter vs. programmable register
  - Input capture: latch value of counter on selected edge of timer input pins
- 16-bit pulse accumulator to count external events, or act as gated timer of internal pulses
- Programmable, periodic interrupt generator called RTI (real-time interrupt)

Programming HC12 Timer

- Most complex subsystem of the HC12, many control registers and bits
- All timer functions similarly programmed
- All have separate interrupt controls and vectors
  - Interrupts enabled/disabled by bit in control register
- All have flags that get set when some programmable condition is satisfied (reset by program)
- Thus, when operation of one timer is learned, procedures similar for all others
**Basic Timer**

- The heart of the timers is a 16-bit, free running, Main Timer (TCNT)
  - Other Timing functions are based off of this timer
  - Its input clock from system bus clock, but may be prescaled via division by 1, 2, 4, 8, 16, or 32
    - PR2:PR1:PR0 prescale factors in control register

<table>
<thead>
<tr>
<th>PR2 : PR1 : PR0</th>
<th>Bus Ck Dividers</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>1</td>
</tr>
<tr>
<td>001</td>
<td>2</td>
</tr>
<tr>
<td>010</td>
<td>4</td>
</tr>
<tr>
<td>011</td>
<td>8</td>
</tr>
<tr>
<td>100</td>
<td>16</td>
</tr>
<tr>
<td>101</td>
<td>32</td>
</tr>
</tbody>
</table>

**Handling Timer Overflow**

- TOF (bit-7 in TFLG2 register) can be used in two ways: polling or interrupting
  - Polling – program polls value of TOF; after asserted, must reset each time by writing ‘1’ to TOF bit
  - Example: Using TOF to generate delay of ~1s assuming 8MHz bus clock
    - 122 overflows, each of 64K periods of 125ns (8.192ms) ⇒ 122×64K×125ns ≈ 1 second!

**TOF Polling Example**

```
; Clears the TOF flag
ldaa #TOF
staa TFLG2

; Enable the timer
bset TSCR, TEN

; Initialize the counter and wait for NTIMES
ldaa #NTIMES
staa counter

; Spin while TOF isn't set
spin1:
  tst TFLG2
  bpl spin1

; Clear the TOF first
ldaa #TOF
staa TFLG2

; Decrement the counter
dec counter

; If counter != 0 spin
bne spin1
```

**TOF Interrupt**

- TOF can generate interrupt if TOI bit in TMSK2 register enabled. TOF interrupt vector is initialized in vector table, and I-bit unmasked in CCR
  - Example: Use TOF interrupt to generate ~1s delay

```
; Initialization interrupt vector
ORIG TOFVect
DC.W isr
```

**Timer Overflow Hardware**

- TOF can generate interrupt if TOI bit in TMSK2 register enabled, TOF interrupt vector is initialized in vector table, and I-bit unmasked in CCR
  - Example: Use TOF interrupt to generate ~1s delay
TOF Interrupt Example

1. Clear the TOF bit
2. Enable the timer
3. Enable the interrupt system
4. Do forever
   a. Wait for the interrupt
   b. If timer incremented by the ISR reaches a maximum
   c. Do some work and reset the counter

Output Compare Timers

- Each of 8 timer channels can be configured as input capture (from Port T) or output compare (to Port T), or Port T used for GP I/O as before if timers not used; choice via TIOS reg
- Output compare allows more accurate timing delays than the TOF
- Each of the 8 timer channels has a 16-bit timer capture/compare register (TCn: TC7...TC0), may be loaded or stored with 16-bit value

OC Time Delays

- Example: Use output compare to achieve 1ms delay (8000 cycles of 8MHz bus clock)

Output Compare Timers

- TCn register compared with TCNT every clock cycle; when equal then flag for that channel (CnF: C7F...C0F) is set; can poll this flag, or...
  - If interrupt enabled for that channel (CnI: C7I...C0I) and I-bit in CCR unmasked, then OC interrupt occurs

OC Time Delays

- Example: Use output compare to achieve 1ms delay (8000 cycles of 8MHz bus clock)
Dr. Tao Li

**Changing the Timer Prescaler**

- In last example, delay limited to ≤8.192ms when 8MHz bus clock used (125ns × 64K)
- But, can divide TCNT increment of bus clock cycles using PR2:PR1:PR0 prescale bits
- For example: for generation of a 10ms delay
  - Uses PR2:PR1:PR0 = 010 (i.e. ÷4) ⇒ each clock pulse is 500ns
  - ⇒ delay = 500ns/cycle × 20000 cycles = 10ms

Dr. Tao Li

**OC Time Delays**

- Example: generating 1s delay using OC flag to generate interrupt
  - Achieved by waiting 250 complete 4ms delay times (125ns/cycle × 32000 cycles = 4ms) generated by OC
  - ~8ms for setup, interrupt occurs every 4ms, total 250 interrupts

Dr. Tao Li

**OC Interupts**

- Longer delays can also be generated by waiting for more output comparisons to be made
- Example: generating 1s delay using OC flag to generate interrupt
  - Achieved by waiting 250 complete 4ms delay times (125ns/cycle × 32000 cycles = 4ms) generated by OC
  - ~8ms for setup, interrupt occurs every 4ms, total 250 interrupts

Dr. Tao Li

**OC Time Delays**

- Example: generating 1s delay using OC flag to generate interrupt

Dr. Tao Li

**Changing the Timer Prescaler**

- TMSK2: EQU $8D; Timer mask 2
- PR2: EQU %00000010; Prescale bit 2
- PR1: EQU %00000010; Prescale bit 1
- PR0: EQU %00000001; Prescale bit 0
- Get the current prescaler value and save it to be restored later
  - push TMSK2
  - Set the prescaler to divide by 4
    - bclr TMSK2,PR2|PR0
    - bset TMSK2,PR1
  - Now restore the original prescaler values
  - pulb stab TMSK2

Dr. Tao Li

**OC Interrupts**

- Enable the timer system
  - bset TSCR,TEN
- Enable output compare channel 2
  - bset TIOS,IOS2
- Generate a 1 s delay
  - Now have it ms to set up the system
    - Set up interrupts
      - ld #C2F
        - st FA
        - Unmask global interrupts
      - Wait until the counter is 0
        - Wait for interrupt
        - br spin

Dr. Tao Li
OC Bit Operation

- OC flags can auto. set or reset Port T bits when flag is set
- When successful OC occurs, one of fours actions may occur at output pin on Port T: disconnected, toggled, cleared, or set
- Selection made for each Port T output via OMn and OLn control bits in TCTL1 and TCTL2 regs

Example: OM2:OL2 = 01 to auto. toggle Port T-2 when OC occurs periodically
- Program outputs square wave w/ period of 2*OC delay (50% duty cycle)

TCTL2: EQU $89
OM2: EQU %00100000
OL2: EQU %00010000

; Set up Output capture action to toggle the bit-2
bset TCTL2,OM2
bset TCTL2,OL2

Input Capture

- Allows TCNT value latched when program-selected external event occurs via Port T
  - e.g. period of pulse train found by storing TCNT at start of period (i.e. rising or falling edge), then capture count at end of period (next rising or falling edge), and take difference
- Two bits for each IC channel, EDGnB (EDG7B...EDG0B) and EDGnA control when signal on Port T causes capture to occur (i.e. rising, falling or both edges activate)
- IC interrupts operate just like OC interrupts

Measure Waveform Period

(1) bset I/BIO, %10000000
(2) bclr TIOL, %10000000
(3) ld00000010
(4) ldd TI1, %10000000
(5) ldd TI1, %00000000
(6) ldd TI1, %00000000
(7) ldd T1, %00000000
(8) ldd T1, %00000000
(9) ldd T1, %00000000
(10) ldd T1, %00000000

Pulse Accumulator

- Port T, bit7 can be configured as a PA input
  - a system that "counts" events
  - there are two operating modes
1) "Event Counting Mode"
   - each time an edge occur on PT7, a 16-bit counter is incremented
2) "Gated Accumulator Mode"
   - when PT7 is asserted, a 16-bit counter will increment based on the Bus Clock / 64
Pulse Accumulator Operating Modes

- May write to or read from PA at any time, again should do so via 16-bit load/store
- May select the edge (positive or negative) for event counting, or the level (high or low) for gated time accumulation, via several control bits
- Two flags and corresponding interrupts available: (1) PA overflow (PAOVF flag); and (2) selected input edge occurs (PAIF flag)
  - e.g. sensor on conveyor belt counting products as they pass, it wants to take action after 24 counts ⇒ initialize PA counter (PACNT) = -24, set up and use interrupt on PAOVG flag, and in ISR take appropriate action

Pulse Accumulator Setup

<table>
<thead>
<tr>
<th>Register</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>PACNT</td>
<td>16-bit Pulse Accumulator Counter</td>
</tr>
<tr>
<td>PACTL</td>
<td>16-bit Pulse Accumulator Control Register</td>
</tr>
</tbody>
</table>

- PA Flags
  - A flag is set upon PACNT Overflow (PAOVF)
  - A flag is set upon an input edge (PAIF)
- PA Flags & Interrupts
  - IRQs can be generated upon PACNT Overflow (PAOVI)
  - IRQs can be generated upon an input edge (PAI)

Other Options for TCNT Clock Generator

- Two clock-select bits (CLK1:CLK0) in PACTL reg. to select clock source for TCNT reg
- When PA disabled (PAEN=0), bus clock prescaled by PR2:PR1:PR0 bits as before
  - Bus clock = 8MHz ⇒ clock TCNT @8MHz (0:0:0) down to 8MHz/32 = 250kHz (1:0:1)
- When PA enabled (PAEN=1), TCNT source can be derived from either an event signal on PT-7 in EC mode (PAMOD=0) or bus clock further divided (PAMOD=1)
- PAEN:CLK1:CLK0 are used for TCNT clock mux selection
Real-Time Interrupt (RTI)

- RTI operates like TOF interrupt, except the periodic rate of generating interrupts is selectable
- Has its own vector in the vector table
- Enabled by RTIE bit, flag RTIF set at interval specified, and RTIF reset by ‘1’ as before
- RTI rate generated by a 13-bit counter that divides bus clock by \(2^{13} = 8192\) or more via 3 RTI prescalar bits (RTR2:RTR0), where 000 = off, 001 = \(\frac{1}{2}\), 010 = \(\frac{1}{4}\), …, 111 = \(\frac{1}{2^{19}}\)

Bus Clock = 8MHz \(\Rightarrow\) intervals of 125ns \(\times 2^{13} = 1.024\text{ms (001)}\) up to 125ns \(\times 2^{19} = 65.536\text{ms (111)}\)

Bus clock = 4MHz \(\Rightarrow\) intervals of 250ns \(\times 2^{13} = 2.048\text{ms (001)}\) up to 250ns \(\times 2^{19} = 131.072\text{ms (111)}\)

- Real Time Interrupt (RTI) vs. Timer Overflow Interrupt
  - Useful for slower Timer IRQ’s than TOF
  - This can be easier than counting many TOP’s when looking for slower events

External interrupts using Timer Interrupts

- External inputs that generate timer interrupts may be used as GP vectored, external interrupts if pins are not o/w being used for I/O or timer functions
- PT-7 (PA input edge or IC7 interrupt), PT-6 (IC6 interrupt), …, PT-0 (IC0 interrupt), using enable bits (PAI, C7I, …, C0I), flags (PAIF, C7F, …, C0F), and vectors as before
  - See Table 10.6

PWM on B32

- PWM (Pulse Width Modulator) module outputs up to 4 pulse-width modulated waveforms at once

Programming PWM

1) We program period in PWMPER register (s)
   - give in terms of clock cycles
2) We program duty cycle in PWMDTY register (s)
   - can select 25%, 50%, 75%
3) Can select Pulse Alignment
   - can select LEFT or CENTER
4) Can select Clock Input
   - Bus Clock / n
Example of PWM Waveforms

PWM registers and counter may be concatenated in pairs for 16-bit timing resolution:
- PWCT2 and PWCT0 together, PWCT1 and PWCT0 together

- Gives longer period and higher duty-cycle resolution
- May have four 8-bit, two 16-bit, or one 16-bit and two 8-bit PWM registers

PWM Clock Control

- Four clock sources derived from bus clock: Clock A, Clock B, Clock S0, and Clock S1
- Clocks A and B each may be produced as bus clock divided by 1, 2, 4, ..., 64, 128
- Clocks S0 and S1 produced by further dividing A and B, respectively, by 2, 4, 6, 8, ..., 512

PWM Clock Circuit

Example

Question:
What is the longest PWM period available assuming 8MHz bus clock and left-aligned waveform?

Answer:
- Achieved by using concatenated PWPER register (16-bit) and slowest clock available (i.e. S0 or S1).
- Bus clock period × Clock A/B multiple of 128 × Clock S0/S1 multiple of 512 × max. # of counts
  = 125ns × 128 × 512 × 64K = 536.87s ≈ 9 minutes!

Timer Summary

Timers we have studied...
1) Main Free Running Timer (TCNT)
2) Timer Output Compare
3) Timer Input Capture
4) Pulse Accumulator
5) Real Time Interrupt
6) Pulse Width Modulator

What would best fit the following application?
1) Count items on an assembly line
2) Generate a 30/70 duty cycle to control a motor
3) Measure phase of an incoming signal
4) Generate an edge every n seconds
5) Latch data from peripheral when an edge occurs
6) Generate interrupt every 50ms