Recap

Pointers
Indexed mode
Arrays and strings

Overview

Timer
Fixed-time delay
Debugging
Intrusiveness
Monitors and dumps

4.5. 16-bit timer

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
$0044																	TCNT

Addr	Bit 7	6	5	4	3	2	1	Bit 0	Name
$0044	**Bit 15**	**14**	**13**	**12**	**11**	**10**	**9**	**Bit 8**	**TCNT**
$0045	**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**Bit 0**	**TCNT**
$0046	**TEN**	TSWAI	TSFRZ	TFFCA	0	0	0	0	TSCR1
$004D	TOI	0	0	0	TCRE	PR2	PR1	PR0	TSCR2
$004F	TOF	0	0	0	0	0	0	0	TFLG2

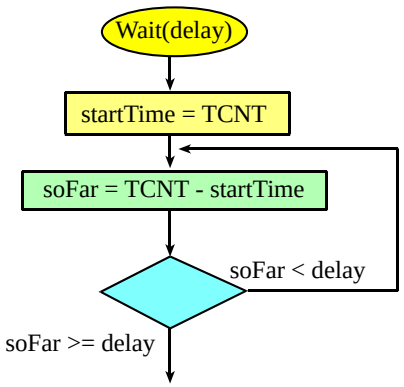
Table 4.11. 9S12 timer ports.

				E = 8 MHz		E = 24 MHz	
PR2	PR1	PR0	Divide by	TCNT period	TCNT frequency	TCNT period	TCNT frequency
0	0	0	1	125 ns	8 MHz	41.7 ns	24 MHz
0	0	1	2	250 ns	4 MHz	83.3 ns	12 MHz
0	1	0	4	500 ns	2 MHz	167 ns	6 MHz
0	1	1	8	1 µs	1 MHz	333 ns	3 MHz
1	0	0	16	2 µs	500 kHz	667 ns	1.5 MHz
1	0	1	32	4 µs	250 kHz	1.33 µs	667 kHz
1	1	0	64	8 µs	125 kHz	2.67 µs	333 kHz
1	1	1	128	16 µs	62.5 kHz	5.33 µs	167 kHz

Table 4.12. Given an E clock frequency, the PR2 PR1 and PR0 bits define the TCNT rate.

Fixed time delay software using the built-in timer

```
void Timer_Init(void){
  TSCR1 = 0x80; // enable TCNT
  TSCR2 = 0x00; // 125ns
}
void Timer_Wait(unsigned short cycles){
unsigned short startTime = TCNT;
  while((TCNT-startTime) <= cycles){}
}
****Timer_Init**********
* Initialize Timer
* Input: none
* Outputs: none
* error: none
Timer_Init
      movb #$80,TSCR1   enable TCNT
      movb #$00,TSCR2  ;125ns in RUN mode
      rts
```



```
;***Timer_Wait**********
; Time delay function
; Input: RegD time to wait (125ns)
; Outputs: none
; error: input must be less than 60000
Timer_Wait
      std  delay      ;time to wait
      movw TCNT,start ;TCNT at start
Wloop ldd  TCNT       ;now
      subd start      ;soFar=TCNT-Start
      cpd  delay
Wchk  blo  Wloop      ;loop if soFar<delay
      rts
```

Show Timer_Wait starter file

Run until `start` is 50000 `RegD` is about 14000

Put a scan point at `Wchk`

Watch `TCNT` roll over

It works because all data are 16-bit unsigned

Real time systems

Bounded latency

Input interface:

input interface latency RDRF -> Read SCIDRL

Output interface:

output interface latency TDRE -> Write SCIDRL

Periodic process (square wave, Labs 7, 8 and 9):

software activity occurs at a periodic rate, Δt

let t_n be the nth time the process executes

goal to make $t_n - t_{n-1} = \Delta t$

δt=jitter $\Delta t-\delta t < t_i - t_{i-1} < \Delta t+\delta t$ for all i

******************TimerWait*************************

Does the LED flash at exactly 5 Hz?

Put a ScanPoint at PTP toggle, measure jitter

show the sliding time

measure the jitter

$100ms-\delta t < t_n - t_{n-1} < 100ms+\delta t$

fix to have almost perfect timing

measure the jitter again

Intrusiveness

degree of perturbation caused by the debugging itself

how much the debugging slows down execution

Nonintrusive

characteristic or quality of a debugger

allows system to operate as if debugger did not exist

e.g., logic analyzer, ICE, BDM

Minimally intrusive

negligible effect on the system being debugged

e.g., dumps (ScanPoint) and monitors

Highly instrusive

e.g., print statements, breakpoints and single-stepping

Dump or ScanPoint

Question: Is my program outputting to the motor?

Method: Add instruments, assemble, download, run

Initialization

```
dbuf rmb 100 first 100 outputs to stepper
dpt  rmb 2   pointer to next place
      ldx  #dbuf
      stx  dpt
      ldaa #100
dloop clr  1,x+
      dbne A,dloop
```

or

```
      ldy  #dbuf
      sty  dpt
dloop clr  1,y+
      cpy  #dbuf+100
      bne  dloop
```

Debugging Instrument

```
      org  *
dscan pshx
      ldx  dpt
      cpx  #dbuf+100
      bhs  ddone
      movb PTT,1,x+
      stx  dpt
      pulx
ddone rts
      nop
dsize equ *-dscan
```

Is it Intrusive??

1) count cycles

```
$F05E                          org  *
$F05E 34              [ 2](  0)dscan pshx
$F05F FE0800          [ 3](  2)      ldx  dpt
$F062 8E0866          [ 2](  5)      cpx  #dbuf+100
$F065 2409            [ 3](  7)      bhs  ddone
$F067 1809300024      [ 5]( 10)      movb PTT,1,x+
$F06C 7E0800          [ 3]( 15)      stx  dpt
$F06F 30              [ 3]( 18)      pulx
$F070 3D              [ 5]( 21)ddone rts
$F071 A7              [ 1]( 26)      nop
$0014                          dsize equ *-dscan
```

2) TCNT measurement

```
test1  movb #$80,TSCR1
       movw TCNT,first
       jsr  dscan
       ldd  TCNT
       subd first
       std  delay
```

3) Attach scope to unused output, PT7

```
test2  bset DDRT,#$80
tloop  ldx  #dbuf
       stx  dpt
       bset PTT,#$80
       jsr  dscan
       bclr PTT,#$80
       bra  tloop
```

LED monitor (PP7)
Question: Is my program running?
Method: Add instruments, assemble, download, run
Initialization

```
DDRP |= 0x80;
bset DDRP,#$80
```

Debugging Instrument (toggle PP7)

```
PTP ^= 0x80;
ldaa PTP     ;[3] cycles
eora #$80    ;[1] cycle
staa PTP     ;[3] cycles
```

How intrusive is the heartbeat?

The bottom line

Timer is an accurate wait to create delays
Monitor is a real-time visualization
Dump records data for later analysis
Simulation (testing), prototype (test), final system (test)