M16C/62

Using the M16C/62 Timers in Timer Mode

1.0 Abstract

The following article describes how to use timers A and B as basic timers, referred to as Timer Mode. Timers are useful for updating multiplexed display, scanning inputs, real time clocks, hardware watchdogs, etc.

2.0 Introduction

The M16C/62 is a 16-bit MCU, based on the M16C CPU core, with features including 10-bit A/D, D/A, UARTS, timers, DMA, etc., and up to 256KB of user flash. The MCU has 11 timers. The timers are separated into two categories by functionality, 5 Timer A's and 6 Timer B's. All 11 timers can operate in Timer Mode.

Timer A has the following additional modes of operation:

- Event Counter Mode
- PWM Mode
- One-Shot Mode

Timer B has the following additional modes of operation:

- Event Counter Mode
- Pulse Width Measurement Mode

Figure 1 illustrates the operation of timer A and Figure 2 illustrates Timer B. Note that there are some differences between the two timers but both operate similarly in Timer Mode. The remainder of this document will focus on setting up timer A0 in Timer Mode.

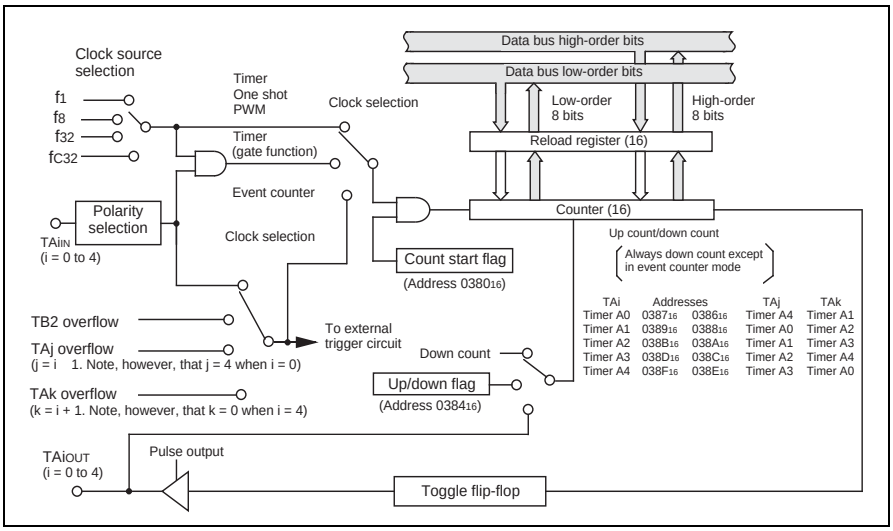


Figure 1 Bock Diagram of Timer A

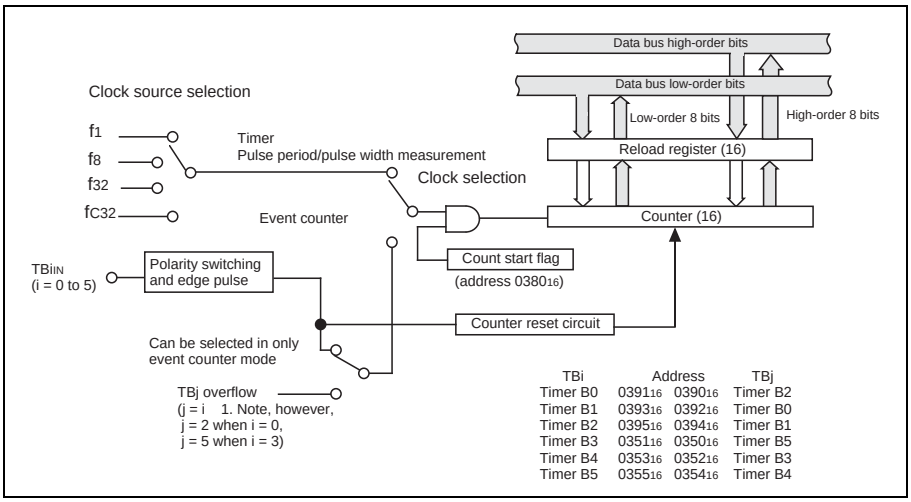


Figure 2 Block Diagram of Timer B

3.0 Timer Mode Description

In Timer Mode, the counter register counts down using the selected clock source until the counter underflows (0000 to FFFFh). At this point, the timer interrupt request bit is set and the contents of the reload register are loaded back into the counter and countdown continues. An interrupt will be accepted when all of the following conditions are met:

- interrupt enable flag (I flag) = "1"
- interrupt request bit = "1"
- interrupt priority level > IPL (Processor Interrupt Priority Level)

If at any time during countdown the count start flag is cleared, counting is suspended until set. Figure 3 illustrates this.

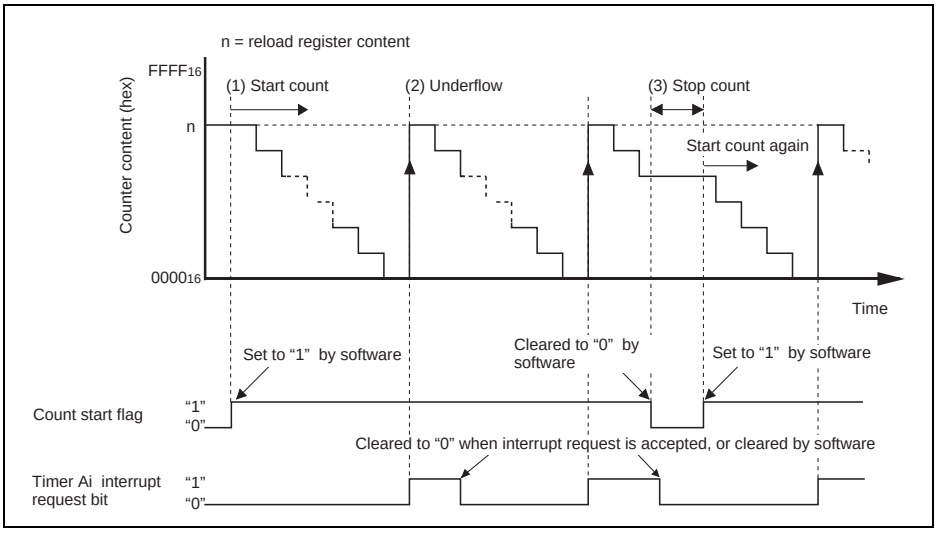


Figure 3 Operation Timing of Timer Mode

4.0 Configuring Timer Mode

To configure a timer for Timer Mode:

1. Load the Timer Ai register, TAi (which also loads the reload register) with the count source.
2. Load the Timer Mode register, TAiMR:
   - Select timer mode: bits TMOD0, TMOD1 = 0.
   - Select the clock source (f1, f/8, f/32, or fc/32): bits TCK0,TCK1.
3. Load the Timer Interrupt Control register (TAiIC) with an interrupt priority level (ILVL) value of at least 1 if interrupts are desired.
4. Ensure interrupts are enabled (I flag set).
5. Set the 'start count' flag bit, TAiS in the Count Start Flag register, TABSR.

It is not necessary to perform these steps in the order listed, but the count register should be loaded before the 'start count' flag is set. Also, the priority level should not be modified when there is a possibility of an interrupt occurring.

The required registers are shown in Figure 4 through Figure 7.

Timer Ai register (Note)

(b15 b7 b8) b0 b7 b0

Symbol	Address	When reset
TA0	0387_{16}, 0386_{16}	Indeterminate
TA1	0389_{16}, 0388_{16}	Indeterminate
TA2	$038B_{16}$, $038A_{16}$	Indeterminate
TA3	$038D_{16}$, $038C_{16}$	Indeterminate
TA4	$038F_{16}$, $038E_{16}$	Indeterminate

Function	Values that can be set	R	W
• Timer mode Counts an internal count source	0000_{16} to $FFFF_{16}$	O	O
• Event counter mode Counts pulses from an external source or timer overflow	0000_{16} to $FFFF_{16}$	O	O
• One-shot timer mode Counts a one-shot width	0000_{16} to $FFFF_{16}$	X	O
• Pulse width modulation mode (16-bit PWM) Functions as a 16-bit pulse width modulator	0000_{16} to $FFFF_{16}$	X	O
• Pulse width modulation mode (8-bit PWM) timer low-order address functions as an 8-bit prescaler and high-order address functions as an 8-bit pulse width modulator	00_{16} to FE_{16} (Both high-order and low-order addresses)	X	O

Note: Read and write data in 16-bit units

Figure 4 Timer Ai Register

Timer Ai mode register

b7	b6	b5	b4	b3	b2	b1	b0
		0				0	0

Symbol: TAiMR(i=0 to 4) — Address: 039616 to 039A16 — When reset: 0016

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 0 0 : Timer mode	○	○
TMOD1			○	○
MR0	Pulse output function select bit	0 : Pulse is not output (TAiOUT pin is a normal port pin) 1 : Pulse is output (Note 1) (TAiOUT pin is a pulse output pin)	○	○
MR1	Gate function select bit	b4 b3 0 X (Note 2): Gate function not available (TAiIN pin is a normal port pin) 1 0 : Timer counts only when TAiIN pin is held “L” (Note 3) 1 1 : Timer counts only when TAiIN pin is held “H” (Note 3)	○	○
MR2			○	○
MR3	0 (Must always be fixed to “0” in timer mode)		○	○
TCK0	Count source select bit	b7 b6 0 0 : f1 0 1 : f8 1 0 : f32 1 1 : fC32	○	○
TCK1			○	○

Note 1: The settings of the corresponding port register and port direction register are invalid.
Note 2: The bit can be "0" or "1".
Note 3: Set the corresponding port direction register to "0".

Figure 5 Timer Ai Mode Register

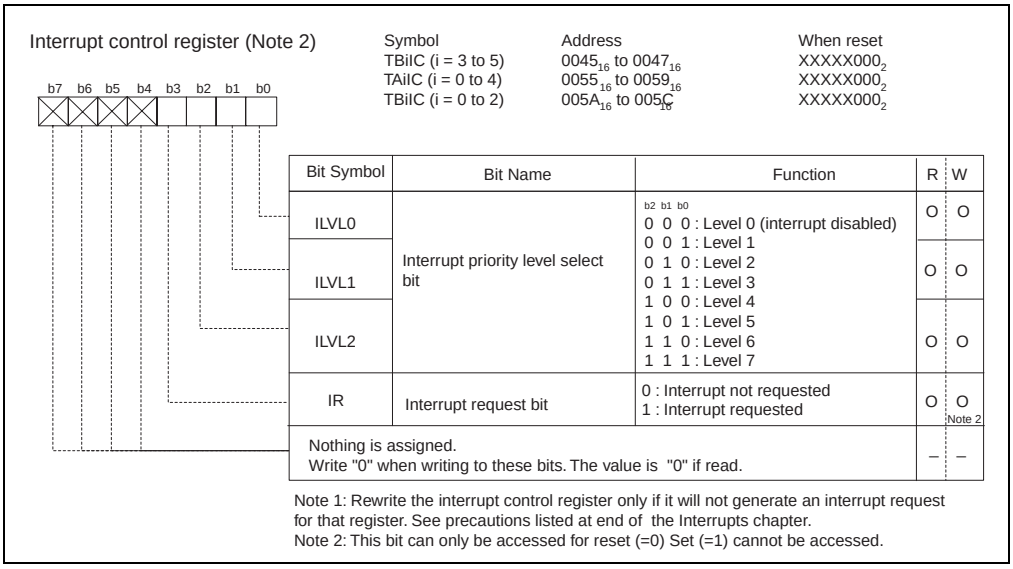
Count start flag

b7	b6	b5	b4	b3	b2	b1	b0

Symbol: TABSR — Address: 0380_{16} — When reset: 00_{16}

Bit Symbol	Bit Name	Function	R	W
TA0S	Timer A0 count start flag	0 : Counting stops 1 : Counting starts	O	O
TA1S	Timer A1 count start flag		O	O
TA2S	Timer A2 count start flag		O	O
TA3S	Timer A3 count start flag		O	O
TA4S	Timer A4 count start flag		O	O
TB0S	Timer B0 count start flag		O	O
TB1S	Timer B1 count start flag		O	O
TB2S	Timer B2 count start flag		O	O

Figure 6 Count Start Flag Register



Interrupt control register (Note 2)

Symbol	Address	When reset
TBiIC (i = 3 to 5)	0045_{16} to 0047_{16}	$XXXXX000_2$
TAiIC (i = 0 to 4)	0055_{16} to 0059_{16}	$XXXXX000_2$
TBiIC (i = 0 to 2)	$005A_{16}$ to $005C_{16}$	$XXXXX000_2$

Bit Symbol	Bit Name	Function	R	W
ILVL0	Interrupt priority level select bit	b2 b1 b0 0 0 0 : Level 0 (interrupt disabled) 0 0 1 : Level 1 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7	O	O
ILVL1			O	O
ILVL2			O	O
IR	Interrupt request bit	0 : Interrupt not requested 1 : Interrupt requested	O	O Note 2
Nothing is assigned. Write "0" when writing to these bits. The value is "0" if read.			–	–

Note 1: Rewrite the interrupt control register only if it will not generate an interrupt request for that register. See precautions listed at end of the Interrupts chapter.
Note 2: This bit can only be accessed for reset (=0) Set (=1) cannot be accessed.

Figure 7 Timer Interrupt Control Register

5.0 Reference

Renesas Technology Corporation Semiconductor Home Page

http://www.renesas.com

E-mail Support

support_apl@renesas.com

Data Sheets

- M16C/62 datasheets, 62aeds.pdf

User's Manual

- M16C/62 User's Manual, 62eum.pdf
- M16C/60 and M16C/20 C Language Programming Manual, 6020EC.pdf
- Application Note: Writing Interrupt Handlers in C for the M16C
- NC30 Ver. 4.0 User's Manual, NC30UE.PDF

6.0 Software Code

Following is a simple program written for Renesas' NC30 compiler to illustrate how to set up a timer mode on timer A0. This program runs on the MSV1632/62 Starter Kit Board and flashes the green LED (LED4) at a 1Hz rate.

To become familiar with the timer, try changing the flash rate, the clock source, or even switch to a different timer (e.g., TA1, TB0, etc.).

```
/***********************************************************************
*
*       File Name: timer_mode.c
*
*       Content: Example program using Timer A0 in "Timer Mode" .This program
*             is written for the Timer Mode application note.  This program
*             works with the MSV1632/62 starter kit board and flashes the
*             red LED (LED2) connected to p7_2.
*
*        Compiled with NC30 ver. 3.20.00.
*
*        All timing based on 16 Mhz Xtal
*
*       Copyright, 2003 Renesas Technology Corporation, Inc.
*=======================================================================
*       $Log:$
*=======================================================================*/
#include "sfr62.h"

#define  TIME_CONFIG 0x40  /*  01000000 value to load into Timer Ai mode register
                               ||||||||_  TMOD0,TMOD1: TIMER MODE SELECTED
                               ||||||_____ MR0:         NO PULSE OUTPUT
                               |||||______ MR1,MR2:     GATE FUNCTION NOT SELECTED
                               |||________ MR3:         SET TO 0 IN TIMER MODE
                               ||_________ TCK0,TCK1:   F DIVIDED BY 8 SELECTED */

#define  CNTR_IPL 0x03   // TA0   interrupt priority level
#define  LED   p7_2

int time_cnt;
int count;   //Global count value, incremented every second

//prototypes
 void init(void);

#pragma INTERRUPT /B TimerA0Int
 void TimerA0Int(void);
```

```
/***********************************************************************
Name:     TimerA0Int()
Parameters: none
Returns:  nothing
Description:Timer A0 Interrupt Service Routine. Interrupts  every 5 msec,
            toggles LED every second, and increments global'count'
*********************************************************************** */

void TimerA0Int(void)
{

  if ((time_cnt += 5) > (1000))     // = 1 second
  { LED ^= 1;                       // toggle LED
    count++;                        // counts the number of seconds since the last
                                    // reset
    time_cnt = 0;
  }
}


/***********************************************************************
Name:    main()
Parameters: none
Returns:  nothing
Description: initializes variables and LED port. Then does nothing but
wait for TA0 interrupts.
*********************************************************************** */

void main (void)
{
  time_cnt = 0;
  count = 0;
  pd7_2 = 1;
  init();
  while (1);      //LED flashing is interrupt driven
}

/***********************************************************************
Name:  init()
Parameters: none
Returns:  nothing
Description: Timer TA0 setup for 5msec interrupts.
*********************************************************************** */
 void init()
  {
   ta0 = 10000; // 16meg xtal, divide by 8, times 10,000 counts-> 5msec interrupts.
```

```
/* the following procedure for writing an Interrupt Priority Level follows that as
described in the M16C data sheets under 'Interrupts' */

   _asm ("   fclr i") ;      // turn off interrupts before modifying IPL
   ta0ic |= CNTR_IPL;        // Interrupt Control register for Timer A0
                             // use read-modify-write instruction to clear IPL
                             // (disable Timer A0 interrupts)
   ta0mr = TIME_CONFIG;      // Timer A0 mode register set to pulse output mode
   _asm ("   fset i");

   ta0s = 1;                 // Count Start Flag register bit0, for Timer A0
                             // start counting!
  }
```

In order for this program to run properly, timer A0's interrupt vector needs to point to the function. The interrupt vector table is near the end of the startup file "sect30.inc". Insert the function label "_TimerA0Int" into the interrupt vector table at vector 21 as shown below.

```
;***********************************************************************
;
;      NC30 C Compiler for M16C/62
;
;   Copyright,2003 Renesas Technology Corporation, Inc.
;       All Rights Reserved.
;
;      Modified  for use on MSV1632 Starter Kit.
;       sect30.inc     : section definition file
;       This program is applicable when using KD30 and the ROM Monitor.
;***********************************************************************
                      :
                      :
                      :

       .lword dummy_int            ; A-D(for user)(vector 14)
       .lword dummy_int            ; uart2 transmit(for user)(vector 15)
       .lword dummy_int            ; uart2 receive(for user)(vector 16)
       .lword dummy_int            ; uart0 transmit(for user)(vector 17)
       .lword dummy_int            ; uart0 receive(for user)(vector 18)
       .lword 0ff900h              ; uart1 transmit(for user)(vector 19)
       .lword 0fff900h             ; uart1 receive(for user)(vector 20)
```

```
.glb       _TimerA0Int
.lword _TimerA0Int          ; timer A0(for user)(vector 21)
.lword dummy_int            ; timer A1(for user)(vector 22)
.lword dummy_int            ; timer A2(for user)(vector 23)
.lword dummy_int            ; timer A3(for user)(vector 24)
.lword dummy_int            ; timer A4(for user)(vector 25)
.lword dummy_int            ; timer B0(for user)(vector 26)
.lword dummy_int            ; timer B1(for user)(vector 27)
.lword dummy_int            ; timer B2(for user)(vector 28)
.lword dummy_int            ; int0 (for user)(vector 29)
.lword dummy_int            ; int1 (for user)(vector 30)
.lword dummy_int            ; int2 (for user)(vector 31)
                :
                :
                :
```

Keep safety first in your circuit designs!

- Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
  Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
- Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
  The information described here may contain technical inaccuracies or typographical errors.
  Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
  Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).
- When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
- Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
  Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.