1.0 Abstract

PWMs, or Pulse Width Modulators, are useful in DC motor control, actuator control, synthesized analog output, piezo transducers, etc. PWMs produce a signal of (typically) fixed frequency and vary the width of the pulse to control a peripheral. The following article describes how to use the M16C/62 A Timers as Pulse Width Modulators, referred to as Pulse Width Modulation Mode.

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 5 timer A’s, all of which can operate as PWMs. Timer A has the following additional modes of operation:

- Event Counter Mode
- Timer Mode
- One-Shot Mode

Figure 1 illustrates the operation of timer A. The remainder of this document will focus on setting up timers A0 and A1 in PWM Mode.

---

**Figure 1 Block Diagram of Timer A**

---

**Figure 1 Block Diagram of Timer A**
3.0 PWM Mode Description

PWM Mode has two "sub" modes: 16 bit and 8 bit. In 16-bit mode, the value of the 16-bit Ai Timer register determines the pulse width and the frequency is fixed to \( \frac{f_{in}}{65535} \). Therefore the maximum frequency of the 16-bit PWM (assuming 16 MHz at Xin) is approximately 244 Hz. In 8-bit mode, the high order 8 bits are used to determine the pulse width and the lower 8 bits the frequency, where the frequency is \( \frac{f_{in}}{255} \) or a maximum frequency of approximately 62,745 Hz. Note that the PWM output is free running and interrupts need not be enabled or serviced. Also the user has the option of triggering the start of the PWM output via the timer’s TaiIN pin.

The pulse width (high level) of the 16-bit PWM is:
\[
pulse width (high) = \frac{n}{f_{in}}, \text{ or } \% \text{ duty} = \frac{n}{65535} \times 100,
\]
where \( n \) is the value loaded into the Ai counter register.

The pulse width (high level) of the 8-bit PWM is:
\[
pulse width (high) = \frac{n \times (m+1)}{f_{in}}, \text{ or } \frac{n}{(255 \times f_{PWM})}, \% \text{ duty} = \frac{n}{255} \times 100,
\]
where \( n \) is the value loaded into the Ai counter register’s high order address and \( m \) is loaded into the Ai counter register’s low order address.

The pulse width can be changed at any time by writing to the Ai counter register, but during counting, the write affects only the reload register, and the counter register is updated on the next cycle.

Figure 2 and Figure 3 illustrate the timing for the PWMs.

**Figure 2 Example of How a 16-bit Pulse Width Modulator Operates**
Using the M16C/62 Timer in PWM Mode

Count source (Note1)

TAiIN pin input signal

Underflow signal of
8-bit prescaler (Note2)

PWM pulse output
from TAiOUT pin

Timer Ai interrupt request bit

1 / fi X (m + 1) X (2^n - 1)

Condition: Reload register high-order 8 bits = 0216
Reload register low-order 8 bits = 0216
External trigger (falling edge of TAiIN pin input signal) is selected

Note 1: The 8-bit prescaler counts the count source.
Note 2: The 8-bit pulse width modulator counts the 8-bit prescaler's underflow signal.
Note 3: m = 0016 to FF16; n = 0016 to FE16.

Figure 3 Example of How an 8-bit Pulse Width Modulator Operates

4.0 Configuring PWM Mode

To configure a timer for PWM mode:

1. Load the Timer Ai register with the pulse width value (16-bit) or frequency and pulse width value (8-bit).

Note: For 16-bit 65535 is not valid. For 8-bit 255 is not valid.

2. Load the Timer Ai Mode register, TaiMR:
   - Select PWM mode: bits TMOD0 and TMOD1 = 1.
   - Set the MR0 bit = 1 for PWM Mode.
   - Clear the MR1 bit for a falling edge external trigger, or set it for rising edge.
   - Clear the MR2 bit to use the ‘count start flag’ as a trigger, or set it for external trigger.
   - Clear the MR3 bit for 16-bit PWM, and set it for 8-bit PWM.
   - Select the clock source (f1, f/8, f/32, or fc/32): bits TCK0, TCK1 register.

3. Load the Timer Interrupt Control register (TAlIC) with an interrupt priority level, (ILVL) (load zero if interrupts are not required).

4. Enable interrupts if required (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.

### Figure 4 Timer Ai Mode Register in Pulse Width Modulation Mode

<table>
<thead>
<tr>
<th>Bit symbol</th>
<th>Bit name</th>
<th>Function</th>
<th>R</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAiMR(i=0 to 4)</td>
<td>039616 to 039A16</td>
<td>0016</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TMOD0</td>
<td>Operation mode select bit</td>
<td>1 : PWM mode</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>TMOD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MR0</td>
<td>1 (Must always be “1” in PWM mode)</td>
<td></td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>MR1</td>
<td>External trigger select bit (Note 1)</td>
<td>0 : Falling edge of TAi IN pin’s input signal (Note 2)</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 : Rising edge of TAi IN pin’s input signal (Note 2)</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>MR2</td>
<td>Trigger select bit</td>
<td>0 : Count start flag is valid</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 : Selected by event/trigger select register</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>MR3</td>
<td>16/8-bit PWM mode select bit</td>
<td>0 : Functions as a 16-bit pulse width modulator</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 : Functions as an 8-bit pulse width modulator</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>TCK0</td>
<td>Count source select bit</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCK1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note 1:** Valid only when the TAi IN pin is selected by the event/trigger select bit (addresses 038216 and 038316). If timer overflow is selected, this bit can be “1” or “0”.

**Note 2:** Set the corresponding port direction register to “0”.

### Figure 5

<table>
<thead>
<tr>
<th>Timer Ai register (Note)</th>
<th>Symbol</th>
<th>Address</th>
<th>When reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>TA0</td>
<td>038716, 038616</td>
<td>Indeterminate</td>
<td></td>
</tr>
<tr>
<td>TA1</td>
<td>038B16, 038A16</td>
<td>Indeterminate</td>
<td></td>
</tr>
<tr>
<td>TA2</td>
<td>038D16, 038C16</td>
<td>Indeterminate</td>
<td></td>
</tr>
<tr>
<td>TA3</td>
<td>038F16, 038E16</td>
<td>Indeterminate</td>
<td></td>
</tr>
</tbody>
</table>

### Function Values that can be set | R | W |
---|---|---|
Timer mode | 00016 to FFFF16 | O | O |
Counts an internal count source | |
Event counter mode | 00016 to FFFF16 | O | O |
Counts pulses from an external source or timer overflow | |
One-shot timer mode | 00016 to FFFF16 | X | O |
Counts a one-shot width | |
Pulse width modulation mode (16-bit PWM) | 00016 to FFFF16 | X | O |
Functions as a 16-bit pulse width modulator | |
Pulse width modulation mode (8-bit PWM) | 0016 to FE16 (Both high-order and low-order addresses) | X | O |
timer low-order address functions as an 8-bit prescaler and high-order address functions as an 8-bit pulse width modulator | |
Pulse width modulation mode (8-bit PWM) | |

**Note:** Read and write data in 16-bit units
Using the M16C/62 Timer in PWM Mode

**Figure 6**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Address</th>
<th>When reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>TA0S</td>
<td>038016</td>
<td>0016</td>
</tr>
<tr>
<td>TA1S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA2S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA3S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA4S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TB0S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TB1S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TB2S</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Count start flag

<table>
<thead>
<tr>
<th>Bit</th>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>b7</td>
<td>TABSR</td>
<td></td>
</tr>
<tr>
<td>b6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 0: Counting stops
- 1: Counting starts

**Figure 7**

**Interrupt control register (Note 2)**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Address</th>
<th>When reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBI0C</td>
<td>005516</td>
<td>XXXXXXX0002</td>
</tr>
<tr>
<td>TBI0C</td>
<td>005616</td>
<td>XXXXXXX0000</td>
</tr>
<tr>
<td>TBI0C</td>
<td>005716</td>
<td>XXXXXXX0000</td>
</tr>
<tr>
<td>TBI0C</td>
<td>005816</td>
<td>XXXXXXX0000</td>
</tr>
<tr>
<td>TBI0C</td>
<td>005916</td>
<td>XXXXXXX0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>b7</td>
<td>ILVL0</td>
<td></td>
</tr>
<tr>
<td>b6</td>
<td>ILVL1</td>
<td></td>
</tr>
<tr>
<td>b5</td>
<td>ILVL2</td>
<td></td>
</tr>
<tr>
<td>b4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 0000 : Level 0 (interrupt disabled)
- 0001 : Level 1
- 0010 : Level 2
- 0011 : Level 3
- 0100 : Level 4
- 0101 : Level 5
- 0110 : Level 6
- 0111 : Level 7

- 0: Interrupt not requested
- 1: Interrupt requested

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.
5.0 Reference

Renesas Technology Corporation Semiconductor Home Page
http://www.renesas.com

E-mail Support
support_apl@renesas.com

Data Sheet
• M16C/62 datasheets, 62aeds.pdf

User's Manual
• NC30 Ver. 4.0 User’s Manual, NC30UE.PDF
• M16C/60 and M16C/20 C Language Programming Manual, 6020EC.pdf

6.0 Software Code
Following is a simple program written for Renesas' NC30 compiler to illustrate how to set up a 16-bit PWM Mode on timer A0, and an 8-bit PWM on timer A1. This program runs on the MSV1632/62 Starter Kit Board.

To become familiar with the PWM, try changing the clock source or switch to a different timer (e.g., TA2, TA3 etc.).

/**********************************************************
 *   File Name: pwm.c
 *   
 *   Content: Example program using Timer A in "PWM mode" .This program
 *   is written for the 'Timer A PWM Mode' application note. Timer
 *   A0 is set up as an 8bit PWM (output on TA0out or P7.0),
 *   timer A1 set up as a 16bit PWM (output on TA1out or P7.2), and
 *   the output varied. The frequency of the 16bit PWM is set to 244Hz,
 *   and the 8bit PWM is set to 7843Hz. The outputs can be viewed
 *   on a scope. This program works with the MSV1632/62 starter kit board,
 *   but should work with any M16C/62 system with P7.0 and P7.2
 *   available.
 *
 *   Compiled with NC30 ver. 3.20.00.
 *   
 *   All timing based on 16 Mhz Xtal
 *   
 *   Copyright, 2003 Renesas Technology Corporation, Inc.
 ***********************************************************/

.textField
#include "sfr62.h"

#define PWM8_CONFIG 0x67 /* 01100111 value to load into timer A0 mode register

|||TMOD0,TMOD1: PWM MODE SELECTED
||MR0: = 1 FOR PWM MODE
||MR1,MR2: EXT TRIGGER NOT SELECTED
||MR3: SET TO 1 FOR 8BIT PWM
||TCK0,TCK1: F DIVIDED BY 8 SELECTED */

#define PWM16_CONFIG 0x07 /* 00000111 value to load into timer A1 mode register

|||TMOD0,TMOD1: PWM MODE SELECTED
||MR0: = 1 FOR PWM MODE
||MR1,MR2: EXT TRIGGER NOT SELECTED
||MR3: SET TO 0 FOR 16BIT PWM
||TCK0,TCK1: F DIVIDED BY 1 SELECTED */

#define CNTR_IPL 0x00 // TA0 AND TA1 interrupt priority level

int time_cnt; // loop counter

//prototypes
void init(void);

*****************************************************************************
Name: main()
Parameters: none
Returns: nothing
Description: initializes variables, then goes into an infinite loop. A
simple delay loop is used to "slowly" increase the pulse widths
*****************************************************************************/

void main (void)
{ int pwm16;
  char pwm8;

  pwm16 = 100; // 16bit PWM changes slowly, so give it a reasonable width to
                // start
  pwm8 = 0;
  time_cnt = 0;
  init();
  while (1)
  {
    while(time_cnt <10000)
      time_cnt++; // delay loop
Using the M16C/62 Timer in PWM Mode

```c
void init()
{
    // initial TA0
    _asm ("fclr i"); // turn off interrupts before modifying IPL
ta0ic |= CNTR_IPL; // use read-modify-write instruction to write IPL
ta0mr = PWM8_CONFIG;
    _asm ("fset i");
    ta0s = 1;  //start PWM

    // initialize TA1
    _asm ("fclr i"); // turn off interrupts before modifying IPL
talic |= CNTR_IPL; // use read-modify-write instruction to write IPL
ta1mr = PWM16_CONFIG;
    _asm ("fset i");
    ta1s = 1;  //start PWM
}
```
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.