Engineering Bulletin

EB394
11/2002

Initializing the MC68HC908AB32 Output Compare Feature

By: Edgar Saenz
Microcontroller Applications Engineering
Austin, Texas

Introduction

To set up an M68HC08 timer module in C language, a few C functions are required to set up prescaler, action, and channel control bits. This engineering bulletin describes how these program functions are called to achieve timer initialization and proper operation. The timer in the MC68HC908AB32 will be used to toggle PTE2's logical state at every 20 counts of the PTD6 external clock stimulus input.

Interrupt Service Routine

In the following interrupt service routine (ISR), `Channel_Var_Temp` is used as temporary variable to load the output compare channel 0 match register which adds an offset of 20 and preloads the next match value to arm the next event. The MCU is running in monitor mode at 4.9 MHz crystal and 1.225 MHz bus speed, and has plenty of time to service the output compare interrupt within 20 external clock pulses provided at 32 kHz. After loading the output compare channel 0 register, the ISR clears the channel 0 flag to prepare for a new match event. The ISR is executed every time the output compare channel 0 match register value matches the timer counter register value. The ISR vector is fetched at every match event, and PTE2 toggles its logical value automatically. See the code below.

```
void   interrupt Timer1_ISR_OC0(void)
{
int unsigned Channel_Var_Temp;
Channel_Var_Temp=T1CH0;
Channel_Var_Temp= Channel_Var_Temp+20; //Add 20 count match event
T1CH0= Channel_Var_Temp;  // Load new value
CH0F_1=0;  // Clear timer flag.
}
```

© Motorola, Inc., 2002

Main Program

At the start of the main program, two functions are called from within `Init_Timer_Port()` to initialize the timer. A *while* statement is used to generate a *run always* loop.

```
void  main(void)
{//Start of main

DDRB=0xff;    // configure PORTB as outputs.
DDRD=0xff;    // configure PORTD as outputs.
Init_Timer_Port();         // Invoke init routine
EnableInterrupts;

   while(1)// run for ever.
   {
   }// end of while
}// end of main
```

`TIMER_Init_ICOC(OC,channel_1,toggle)` configures the timer module to enable the PTE2 as an output compare and sets the pin action to toggle. With `#define`, the strings `channel_1` and `toggle` are defined as 1.

The `TIMER_General_Init(ON,channel_1,external_clock,ON)` routine turns on the timer and enables the timer clock to be used externally through port pin PTD6. It does this by writing the prescaler bits by passing `external_clock` a string defined to a value of 7. It also enables interrupts for timer channel 0 which is controlling PTE2. These steps are sufficient to initialize the timer which will allow the timer to demonstrate an output compare operation.

```
void  init_port(void)
{//Start of init
TIMER_Init_ICOC(OC,channel_1,toggle);//Function,Channel number,action
TIMER_General_Init(ON,channel_1,external_clock,ON);  // Timer ON,
}// end of init
```

Timer Init Routine

The function call on `TIMER_Init_ICOC(OC,channel_1,toggle)` has three parameters. The first one selects an output compare or an input capture. The second parameter provides a way to specify which timer channel. For example, if a value of 1 is passed to the second parameter, channel 0 is selected. The value of 0 is left to disable all channels. The example below gives the corresponding channel numbers and parameter 2 values.

```
Timer pin   channel        parameter needed
  PT0         timer 0        channel_1
  PT1         timer 1        channel_2
  PT2         timer 2        channel_3
  PT3         timer 3        channel_4
  PT4         timer 4        channel_5
  PT5         timer 5        channel_6
  PT6         timer 6        channel_7
  PT7         timer 7        channel_8

#define channel_1 1
#define channel_2 2
#define channel_3 3
#define channel_4 4
#define channel_5 5
#define channel_6 6
#define channel_7 7
#define channel_8 8

definitions for prescaler values, where the input clock to the
timer counter which is derived from the core bus speed divided
by the prescaler settings.

#define div_by_one          0
#define div_by_two          1
#define div_by_four         2
#define div_by_eight        3
#define div_by_sixteen      4
#define div_by_thirtytwo    5
#define div_by_sixtyfour    6
#define external_clock      7

#define buffered   1
#define unbuffered 0
```

The third parameter allows the user to set the pin action. When a string toggle which is defined as a 1 is passed to the third parameter, the action toggle is selected. The strings rise and fall with defined values 2 and 3 can be used to make the pin rise or fall (as shown below).

```
ACTION TAKEN AT
Output compare          Parameter
  nothing                 Disconnect
  toggle                  toggle
  fall                    fall
  rise                    rise


#define nothing 0
#define toggle  1
#define fall    2
#define rise    3
```

Timer Initialize Routine

TIMER_Init_ICOC(char IC_OC,char Chan_Numb,char EDGE_ACTION) routine code checks for an IC or OC string to be provided. In this exercise, an output compare is used. As soon as the OC condition is met, Buffer_Flag is used by an if statement to set a buffered or unbuffered output compare. A switch statement is used to choose the channel value provided in Chan_Numb which is passed by a channel 1 string, which is defined with a value of 1 (in the example below) when the function is called by the main program. As shown in the example below, a 1 is required for the case code 1 statement to be executed. Setting the pin action condition to toggle by defining the string toggle to a value of 1 and through an if statement configures the appropriate edge configuration ELSXA bits.

```
void  TIMER_Init_ICOC (char IC_OC,char Chan_Numb,char EDGE_ACTION,char
Buffer_Flag)
{
if(IC_OC==OC)
    {
     switch(Chan_Numb)
        {
         case nothing:
         break;
         case channel 1:// T1SC0
         if( Buffer_Flag==Buffered)
          MS0A_1=1; MS0B_1=1; //Configured as buffered
              MS0A_1=0; MS0B_1=1; //Configured as unbuffered

           if( EDGE_ACTION==nothing)  /* Pin under port control*/
             {
             ELS0A_1=0;ELS0B_1=0;
             }
           if( EDGE_ACTION==toggle) /*config to toggle*/
             {
             ELS0A_1=1;ELS0B_1=0;
             }
           if( EDGE_ACTION==fall) /* config to fall */
             {
             ELS0A_1=0;ELS0B_1=1;
             }
           if( EDGE_ACTION==rise) /*config to rise  */
             {
             ELS0A_1=1;ELS0B_1=1;
             }
           break;
        }// end of if(IC_OC==OC)
}
```

Output Seen on PTE2

Channel 1 of **Figure 1** shows a 32 kHz 50% duty cycle signal generated by another development board. The 32 kHz frequency is then fed into PTD6 on the MC68HC908AB32. At initialization, the prescaler bits for the TIM08 module are set to a value of 1 to enable external timer count clocking. At 32 kHz, the period is 29 µs. If a value of 20 is added to the output compare value during the interrupt service routine, then a match should occur at every 580 µs.

As seen in **Figure 1**, channel 3 displays a delta change of 578 µs. Delta time shown from one output compare match event to the next is determined by the value loaded to the timer compare register in the ISR. **Figure 1** also shows that it takes 20 timer clock rising edges to the next timer match event.

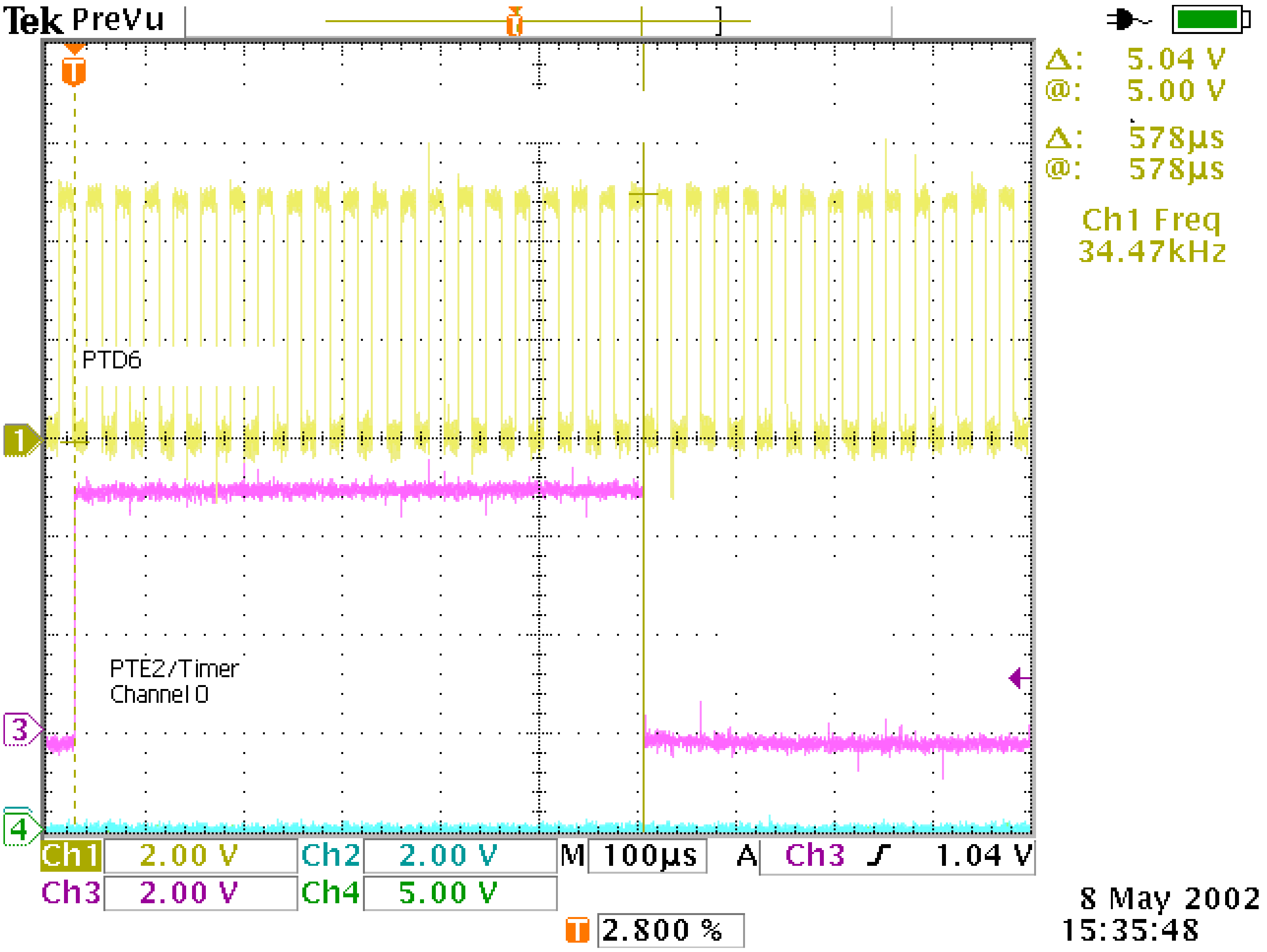

Figure 1. PTE2 Timer Response

This page intentionally left blank.

This page intentionally left blank.

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217
1-303-675-2140 or 1-800-441-2447

JAPAN:

Motorola Japan Ltd.; SPS, Technical Information Center,
3-20-1, Minami-Azabu Minato-ku, Tokyo 106-8573 Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.;
Silicon Harbour Centre, 2 Dai King Street,
Tai Po Industrial Estate, Tai Po, N.T., Hong Kong
852-26668334

TECHNICAL INFORMATION CENTER:

1-800-521-6274

HOME PAGE:

http://www.motorola.com/semiconductors

Information in this document is provided solely to enable system and software implementers to use Motorola products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

MOTOROLA

Motorola and the Stylized M Logo are registered in the U.S. Patent and Trademark Office. digital dna is a trademark of Motorola, Inc. All other product or service names are the property of their respective owners. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

© Motorola, Inc. 2002