Engineering Bulletin

EB611/D
7/2003

HC11 and HC12 Families' Pulse Accumulator

By Darci Ernst
Systems Engineering
Austin, TX

Introduction and Background

The timer module is one of the peripherals found on Motorola's HC12 Family of microcontrollers. The HC12's enhanced capture timer (ECT) module is an updated version of the original Motorola HC11 Family timer. Like the original, the ECT's functionality is based around a 16-bit free-running timer/counter.

The HC12 timer has been enhanced with pulse accumulator capabilities, which can be configured to count a predetermined number of events (pulses) and instruct the microcontroller to execute a specific function. The synchronous operation of the logic of the dual-port pulse accumulator (versus the asynchronous inputs) makes it possible for events to be missed, unless the software engineer creates a simple safeguard to prevent execution errors and ensure proper operation of the application. This bulletin gives detailed explanations of the above situations and potential software workarounds.

Detailed Description

Configuring the pulse accumulator to count to a predetermined number of events is simple and reduces CPU overhead. First, configure the pulse accumulator for an overflow interrupt (PAOVI). In that configuration, an interrupt will be generated when the counter overflows from $FF to $00 (or $FFFF to $0000 if in 16-bit mode). Next, preload the pulse accumulator counter with a value corresponding to the desired number of counts. For example, if the controller needs to count 5 events (pulses) and then perform a specific action, load the pulse accumulator counter with $FB. (Calculation, $00 – $05 = $FB) Once the fifth count occurs, the counter will overflow from $FF to $00, an overflow interrupt would be generated, and the appropriate interrupt service routine would be executed.

However, if an event comes in when the new value is being written to the pulse accumulator counter, that event could be missed.

One solution to this problem is to use the timer input function available to the same pin to monitor the pulse accumulator input. If the timer interrupt flag is cleared prior to the PACNx write, it may then be read after writing the pulse accumulator to check whether an event occurred during the write to PACNx cycle.

Within this normal workaround, there are two potential situations which can occur:

- A timer interrupt flag (CxF in TFLG1) may read an incorrect value if a read of TFLG1 occurs one cycle after the event. Under all circumstances, the interrupt flag is asserted one full clock cycle after the event was detected. Then, one cycle after that, the flag can be read correctly.
- The HC12 (not the HCS12) pulse accumulator overflow flag may not be asserted at the appropriate time (HC12 errata AR_644 and AR_645, ECT: pulse accumulator overflow). This may occur when preloading the counter with a $FF (or $FFFF in 16-bit mode). Like the previous situation, this will only occur if an event transpires while the pulse accumulator counter is being written. However in this situation, it is the pulse accumulator overflow flag which is not being correctly asserted, even though the pulse accumulator overflows. This happens on the HC12 because the overflow flags are masked during a write to the pulse accumulator counter. Therefore, if the event occurs while writing to the counter, the counter will roll over, but the overflow flag will not be asserted.

If you are using a newer part, like one from the HCS12 Family, the second situation does not happen but the first situation can still occur.

Figure 1 shows a flowchart for a successful software check against missing events on the HC11 MCU during a write to the PAC.

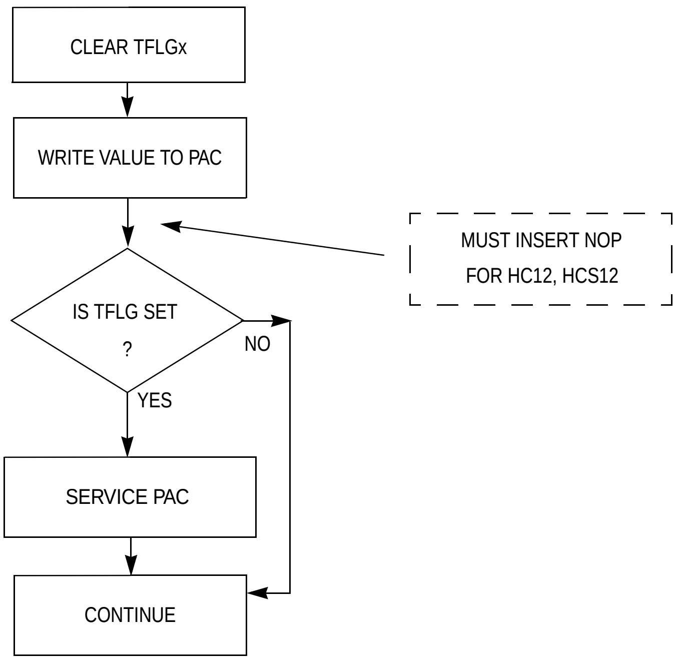

Figure 1. Flowchart for HC11 Sequence that Fails on HC12

There are two cases in which this flowchart must be modified in order for the flag to be correctly caught and the pulse accumulator updated:

- HC11 conversion issue
- Missing interrupt request errata for HC12 MCU's

Case #1 – HC11 Conversion Issue

The first case has to do with cycle count differences between HC11 and HC12 MCUs. On both families, there is a short delay from when an edge occurs on the pulse accumulator to when the actual accumulator is updated. The timer structure is the same as on older HC11 devices, but identical code running on an HC12 has been shown to occasionally miss events. The event is not really missed. Because the HC12 instruction set requires fewer cycles to execute certain instructions, the HC12 reads the result early, before the bit has been set. See **Figure 2** and note when the CxF bit is set in the TFLG1 register versus when it is read. The HC11 takes two cycles to read the TFLG1 register so the CxF bit is set by the end of the read.

The sequence detailed in **Figure 2** shows the required series of events that can cause a timer flag to be missed in the HC12 or HCS12 MCU. Notice at [A] the external PAC input signal arrives in the PAC one clock period later.

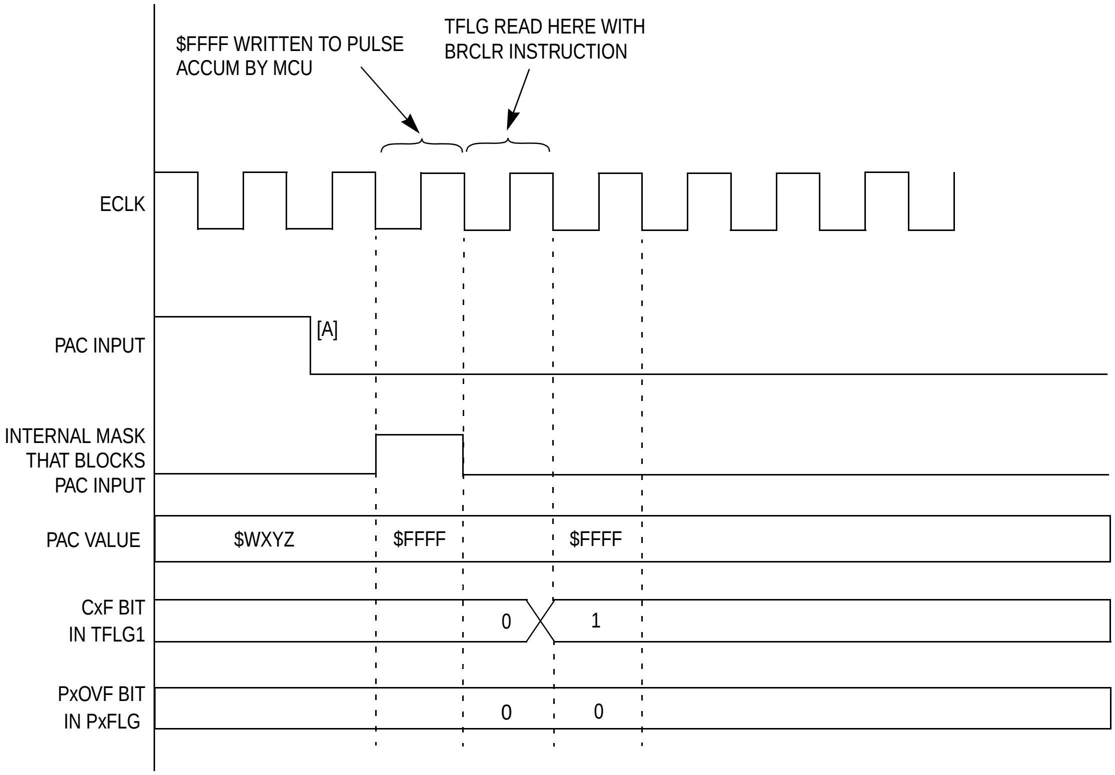

Figure 2. Sequence of Events Required for Missing the Timer Flag in an HC12 or HCS12 MCU(CxF Bit in TFLG1)

Figure 3 shows a NOP instruction allowing the TFLG1 assertion prior to being read by the CPU. Notice at [A] the external PAC input signal arrives in the PAC one clock period later.

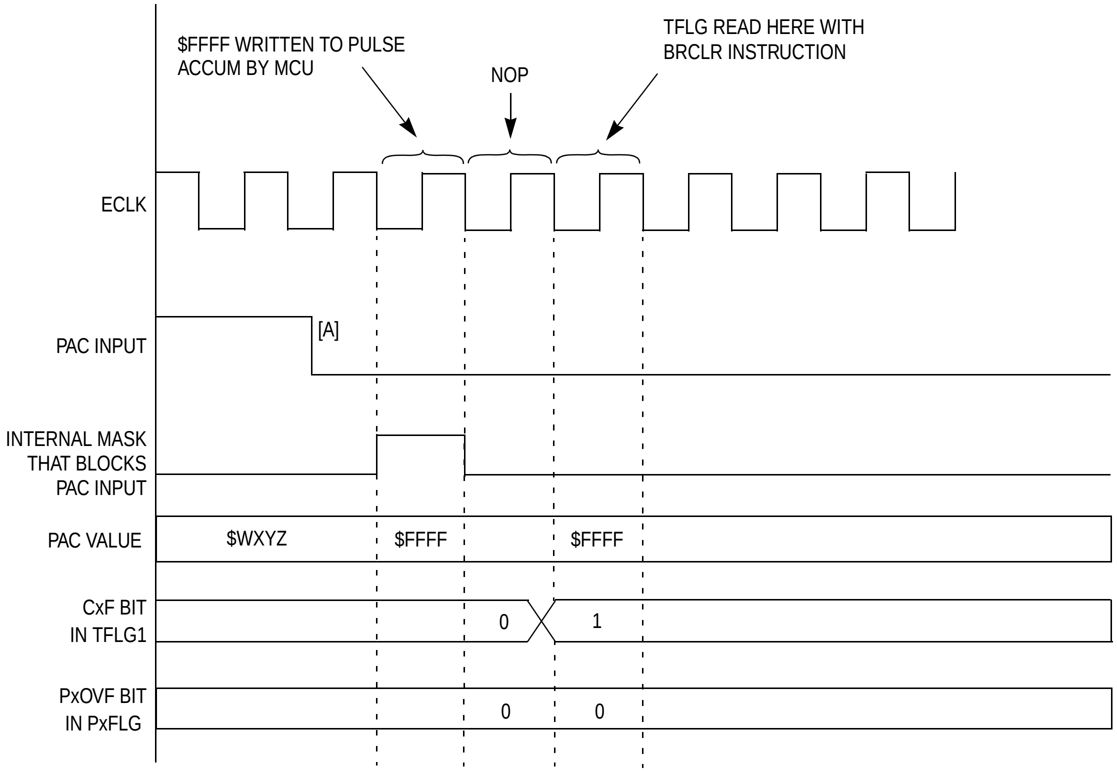
$FFFF WRITTEN TO PULSE ACCUM BY MCU
NOP
TFLG READ HERE WITH BRCLR INSTRUCTION
ECLK
PAC INPUT
[A]
INTERNAL MASK THAT BLOCKS PAC INPUT
PAC VALUE
$WXYZ
$FFFF
$FFFF
CxF BIT IN TFLG1
0
1
PxOVF BIT IN PxFLG
0
0

Figure 3. Solving Conversion Issue by Implementing NOP

Case #2 – Missing Interrupt Request Errata for HC12 MCUs

The second case (applicable only to the HC12, not the HCS12) is that a somewhat unlikely series of events can cause the pulse accumulator to not generate an overflow interrupt (PAOVI) when the counter goes from $FF to $00 (or $FFFF to $0000, depending upon mode of operation). This could cause the code to miss an overflow event. Refer to **Figure 4**; the PxOVF bit does not get set.

Figure 4 shows a NOP instruction allowing the TFLG1 assertion prior to being read by the CPU, but the overflow flag (PxOVF) is still missed. Notice at [A] the external PAC input signal arrives in the PAC one clock period later.

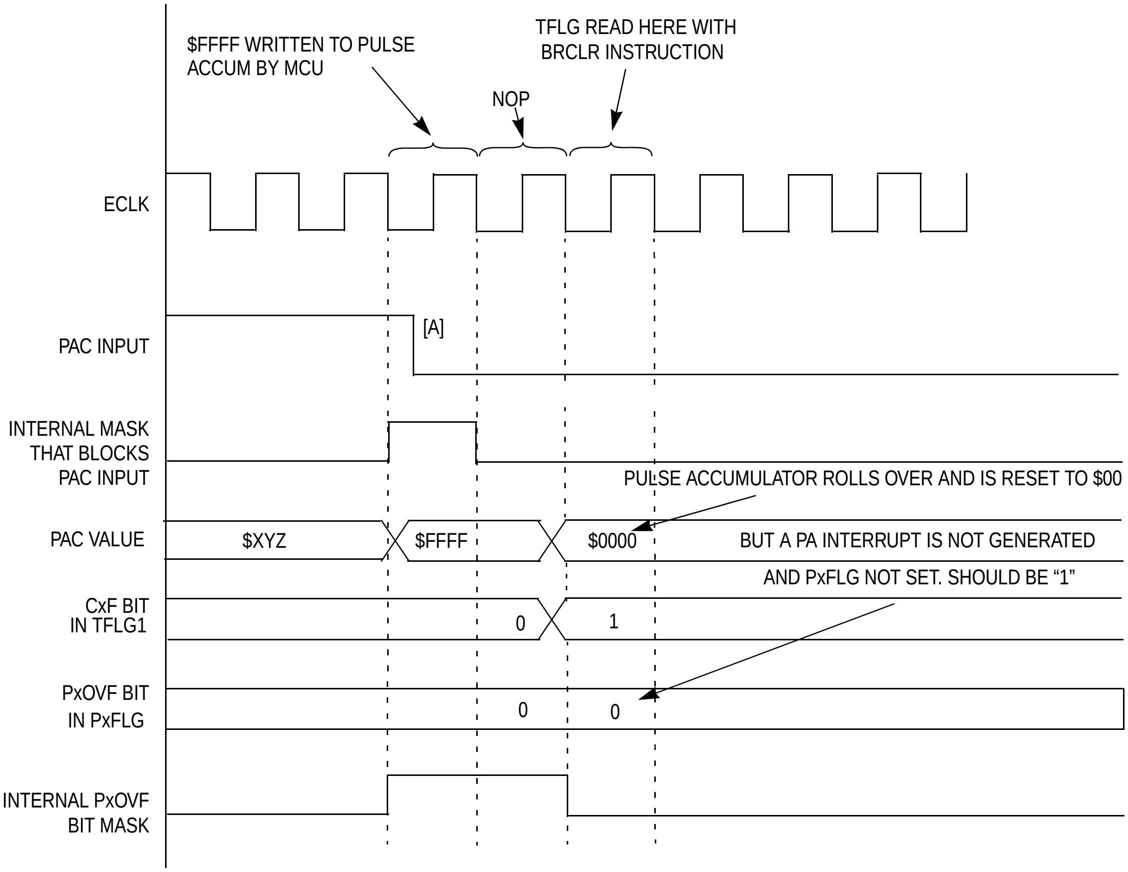
$FFFF WRITTEN TO PULSE ACCUM BY MCU
TFLG READ HERE WITH BRCLR INSTRUCTION
NOP
ECLK
PAC INPUT
[A]
INTERNAL MASK THAT BLOCKS PAC INPUT
PULSE ACCUMULATOR ROLLS OVER AND IS RESET TO $00
PAC VALUE
$XYZ
$FFFF
$0000
BUT A PA INTERRUPT IS NOT GENERATED
AND PxFLG NOT SET. SHOULD BE "1"
CxF BIT IN TFLG1
0
1
PxOVF BIT IN PxFLG
0
0
INTERNAL PxOVF BIT MASK

Figure 4. Sequence of Events Required for Missed PA Interrupt Request

Errata for HC12

The pulse accumulator logic could fail to generate an interrupt request and/or set the PAOVF when a trigger event occurs at the same time a $FF value is written to the PACx register (even though the register rolls over to $00).

NOTE: *The trigger event may be delayed from pin activity 256, 512, or 1024 module clocks if the DLYCT register is non-zero.*

Workaround for Both Asynchronous Write and Missing Pulse Accumulator Overflow Errata

Motorola recommends that the input capture function for the subject channel be enabled prior to writing a value to the PACx register. Write to the pulse accumulator register. Then execute one NOP (to allow the timer to update the input capture flag). Then read the input capture interrupt flag to see whether it is set. If yes, a check must be made for a missing pulse accumulator event. A flow chart of this is shown in **Figure 5**.

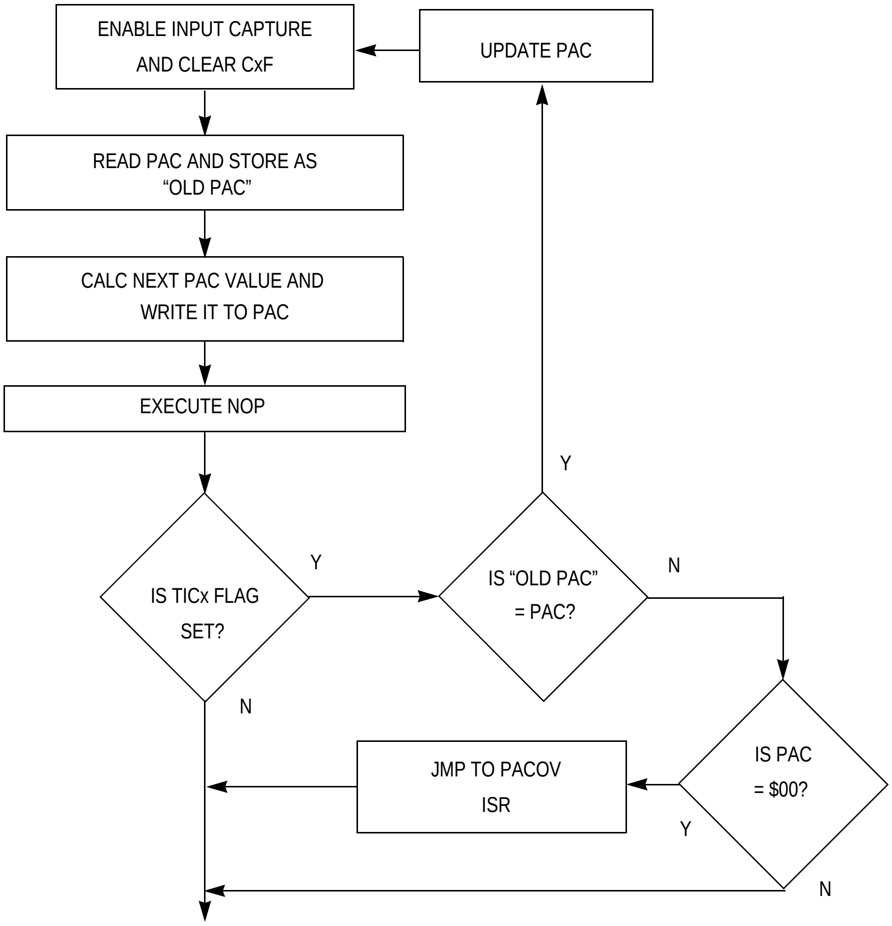
ENABLE INPUT CAPTURE AND CLEAR CxF
UPDATE PAC
READ PAC AND STORE AS "OLD PAC"
CALC NEXT PAC VALUE AND WRITE IT TO PAC
EXECUTE NOP
IS TICx FLAG SET?
Y
N
IS "OLD PAC" = PAC?
Y
N
IS PAC = $00?
Y
N
JMP TO PACOV ISR

Figure 5. Flowchart for Workaround on Both Asynchronous Write and Missing Pulse Accumulator Overflow Errata

Conclusion

In summary, the versatility of the enhanced capture timer module provided by the pulse accumulator function allows an engineer to easily perform event timing-based functions without the software overhead previously required. The pulse accumulator provides an internal hardware-based solution rather than software to monitor an input capture channel or poll a general-purpose input pin.

The exceptions described in this document would have little or no effect on most applications. For the select cases where this may pose a problem, a simple solution has been provided.

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:
Motorola Literature Distribution
P.O. Box 5405
Denver, Colorado 80217
1-800-521-6274 or 480-768-2130

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

HOME PAGE:
http://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