Order this document
by EB276/D

Motorola Semiconductor Engineering Bulletin

EB276

Using the ITC Function on the Time Processor Unit A

By Sharon Darley
Austin, Texas

Introduction

The ITC function counts input transitions and time stamps the last two. The user specifies the number of transitions to be counted via the parameter MAX_COUNT. Each time the TPU (time processor unit) counts an input transition, it increments the parameter TRANS_COUNT and compares it with MAX_COUNT.

The ITC function has two main modes of operation:

- Continuous mode
- Single-shot mode

In continuous mode, the ITC function will repeatedly count the number of transitions programmed in MAX_COUNT. Each time TRANS_COUNT reaches the value in MAX_COUNT, TRANS_COUNT resets to 0. If BANK_ADDRESS points to a valid parameter address, then the value in the high byte of that address is incremented by 1. If interrupts are enabled, then an interrupt request will be made. Finally, if the continual with links mode has been selected with the host sequence field bits, then a link will be generated to the channel specified by START_LINK_CHANNEL.

The single-shot mode works exactly the same way as the continuous mode except that the ITC function counts the number of transitions specified in MAX_COUNT only once, and then it ignores all further transitions.

The ITC function is not designed to work as a free-running counter. It will always count at least one transition before generating an interrupt, even if the value in MAX_COUNT is 0.

Example Program

This program uses single-shot with links mode to count input pulses and generate a link when MAX_COUNT reaches a specified value. In single-shot mode with links, the ITC function counts the number of transitions programmed in MAX_COUNT once. When TRANS_COUNT reaches the value in MAX_COUNT, a link is generated to the channel specified by START_LINK_CHANNEL, and the value in the high byte of the parameter pointed to by BANK_ADDRESS is incremented by 1. In this example, BANK_ADDRESS points to an unimplemented RAM location so that it does not affect operation of other channels.

In this program, the ITC function on channel 1 counts input pulses from the PWM function on channel 0. When the ITC function counts seven pulses, it generates a link to channel 2, which is set up to run the SPWM function. This simply means that channel 1 issues a service request to channel 2. To see when the link is generated, the SPWM square wave is programmed to be out of phase with the PWM square wave. The rising edge of the SPWM wave will begin at the falling edge of the PWM wave.

Channel 0 is set up to run the PWM function, channel 1 is set up to run the ITC function, and channel 2 is set up to run the SPWM function.

Program Code for CPU32-Based Microcontrollers

This program was assembled using the IASM32 assembler, available from P&E Microcomputer Systems, Inc. with the M68332 in-circuit debugger.

```
Initialization
TPUMCR    equ    $fffe00
TICR      equ    $fffe08
CIER      equ    $fffe0a
CFSR0     equ    $fffe0c
CFSR1     equ    $fffe0e
CFSR2     equ    $fffe10
CFSR3     equ    $fffe12
HSQR0     equ    $fffe14
HSQR1     equ    $fffe16
HSRR0     equ    $fffe18
HSRR1     equ    $fffe1a
CPR0      equ    $fffe1c
CPR1      equ    $fffe1e
PRAM0_0   equ    $ffff00
PRAM0_1   equ    $ffff02
PRAM0_2   equ    $ffff04
PRAM0_3   equ    $ffff06
PRAM0_4   equ    $ffff08
PRAM0_5   equ    $ffff0A
PRAM0_6   equ    $ffff0C
PRAM0_7   equ    $ffff0E
PRAM1_0   equ    $ffff10
PRAM1_1   equ    $ffff12
PRAM1_2   equ    $ffff14
PRAM1_3   equ    $ffff16
PRAM1_4   equ    $ffff18
PRAM1_5   equ    $ffff1A
PRAM1_6   equ    $ffff1C
PRAM1_7   equ    $ffff1E
PRAM2_0   equ    $ffff20
PRAM2_1   equ    $ffff22
PRAM2_2   equ    $ffff24
PRAM2_3   equ    $ffff26
PRAM2_4   equ    $ffff28
PRAM2_5   equ    $ffff2A
PRAM2_6   equ    $ffff2C
PRAM2_7   equ    $ffff2E
PRAM4_0   equ    $ffff40
PRAM4_1   equ    $ffff42
PRAM4_2   equ    $ffff44
PRAM4_3   equ    $ffff46
PRAM4_4   equ    $ffff48
PRAM4_5   equ    $ffff4a
PRAM5_0   equ    $ffff50
PRAM5_1   equ    $ffff52
```

```
PRAM5_2   equu    $ffff54
```

```
PRAM5_2   equ    $ffff54
PRAM5_3   equ    $ffff56
PRAM5_4   equ    $ffff58
PRAM5_5   equ    $ffff5a
       org    $4000                    ; begin at memory location $4000
       move.w #$07A9,(CFSR3).L         ; Channel Function Select Field
                                       ; (channel numbers may
                                       ; vary for different mask sets)
       move.w #$00FF,(CPR1).L          ; Channel Priority Field, high priority
       move.w  #$0008,(HSQR1).L        ; ITC mode = single shot with links
                                       ; SPWM = mode 0
```

PWM Initialization for Channel 0

This PWM wave will have a pulse period of $1000 and a pulse hightime of $500. The ITC function on channel 1 will count the rising edges.

```
       move.w #$0092,(PRAM0_0).L       ; Channel Control, use TCR1
       move.w #$0500,(PRAM0_2).L       ; pulse hightime = $500
       move.w #$1000,(PRAM0_3).L       ; pulse period =  $1000
```

ITC Initialization for Channel 1

In this example, the ITC function only links to channel 2. Thus, START_LINK_CHANNEL = 2, and LINK_CHANNEL_COUNT = 1. As required, LINK_CHANNEL_COUNT is a value greater than zero and less than or equal to eight.

Since this program does not need to increment a parameter in another memory location when the number of transitions specified in MAX_COUNT has been counted, BANK_ADDRESS points to an unimplemented memory location.

```
       move.w #$0007,(PRAM1_0).L       ; Channel control, detect rising edge,
                                       ; use TCR1
       move.w #$210E,(PRAM1_1).L       ; START_LINK_CHANNEL = 2,
                                       ; LINK_CHANNEL_COUNT = 1,
                                       ; BANK_ADDRESS points to unimplemented
                                       ; RAM
       move.w #$0007,(PRAM1_2).L       ; MAX_COUNT = 7
```

SPWM Initialization for Channel 2 in Mode 0

The SPWM is set up in mode 0 so that it can receive links from another channel. It is initialized with a pulse hightime of $500 and a period of $1000. REF_ADDR1 points to a reference value to which DELAY and PERIOD are added to form the rising transition time. Here, it points to FINAL_TRANS_TIME on the ITC channel. FINAL_TRANS_TIME contains the TCR time of the final transition when MAX_COUNT is reached.

```
        move.w #$92,(PRAM2_0).L           ; Channel Control
        move.w #$500,(PRAM2_2).L          ; HIGH_TIME = $500
        move.w #$1000,(PRAM2_3).L         ; PERIOD = $1000
        move.w #$0018,(PRAM2_4).L         ; REF_ADDR1 = $18
        move.w #$0500,(PRAM2_5).L         ; DELAY = $500

Service Initialization Request

        move.w #$0026,(HSRR1).L           ; Initialization for ch 0, 1, 2
finish  bra    finish
```

Program Code for CPU16-Based Microcontrollers

This program was assembled using the IASM16 assembler available with the ICD16 in-circuit debugger from P&E Microcomputer Systems.

```
Initialization
TPUMCR    equu    $fffe00
```

```
PRAM1_2   equ    $ffff14
PRAM1_3   equ    $ffff16
PRAM1_4   equ    $ffff18
PRAM1_5   equ    $ffff1A
PRAM1_6   equ    $ffff1C
PRAM1_7   equ    $ffff1E
PRAM2_0   equ    $ffff20
PRAM2_1   equ    $ffff22
PRAM2_2   equ    $ffff24
PRAM2_3   equ    $ffff26
PRAM2_4   equ    $ffff28
PRAM2_5   equ    $ffff2A
PRAM2_6   equ    $ffff2C
PRAM2_7   equ    $ffff2E

**** MAIN PROGRAM ****

        org    $400
        ldab   #$0F                        ; use bank $0f for parameter RAM
        tbek
        ldd    #$07A9
        std    CFSR3                       ; Channel Function Select Field (Note:
                                           ; function numbers
        ldd    #$00FF                      ; may vary for different mask sets)
        std    CPR1                        ; Channel Priority Field, high priority
        ldd    #$0008
        std    HSQR1                       ; ITC mode = single with links, SPWM=mode0
```

PWM Initialization for Channel 0

This PWM wave will have a pulse period of $1000 and a pulse hightime of $500. The ITC function on channel 1 will count the rising edges.

```
       ldd    #$0092
       std    PRAM0_0                      ; Channel Control, use TCR1
       ldd    #$0500
       std    PRAM0_2                      ; pulse hightime = 500
       ldd    #$1000
       std    PRAM0_3                      ; pulse period = 1000
```

ITC Initialization for Channel 1

In this example, the ITC function only links to channel 2. Thus, START_LINK_CHANNEL = 2, and LINK_CHANNEL_COUNT = 1. As required, LINK_CHANNEL_COUNT is a value greater than zero and less than or equal to eight. Since this program does not need to increment a parameter in another memory location when the number of transitions specified in MAX_COUNT has been counted, BANK_ADDRESS points to an unimplemented memory location.

```
        ldd    #$0007
        std    PRAM1_0                  ; Channel control, detect rising edge, use
                                        ; TCR1
        ldd    #$210E
        std    PRAM1_1                  ; START_LINK_CHANNEL = 2,
                                        ; LINK_CHANNEL_COUNT = 1,
                                        ; BANK_ADDRESS points to unimplemented RAM
        ldd    #$0007
        std    PRAM1_2                  ; MAX_COUNT = 7
```

SPWM Initialization for Channel 2 in Mode 0

The SPWM is set up in mode 0 so that it can receive links from another channel. It is initialized with a pulse hightime of $500 and a period of $1000. REF_ADDR1 points to a reference value to which DELAY and PERIOD are added to form the rising transition time. Here, it points to FINAL_TRANS_TIME on the ITC channel. FINAL_TRANS_TIME contains the TCR time of the final transition when MAX_COUNT is reached. This waveform will be delayed from the PWM waveform. Its rising edge will occur at the falling edge of PWM.

```
        ldd    #$92
        ldd    #$500
        std    PRAM2_2                ; HIGH_TIME = $500
        ldd    #$1000
        std    PRAM2_3                ; PERIOD = $1000
        ldd    #$0018
        std    PRAM2_4                ; REF_ADDR1=$18
        ldd    #$0500
        std    PRAM2_5                ; DELAY = $500

Service Initialization Request

        ldd    #$0026
        std    HSRR1                    ; Initialization for ch 0, 1, 2
finish  bra      finish
```

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 and Ⓜ are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

How to reach us:
USA/EUROPE/Locations Not Listed: Motorola Literature Distribution, P.O. Box 5405, Denver, Colorado 80217.
1-800-441-2447 or 1-303-675-2140. Customer Focus Center, 1-800-521-6274
JAPAN: Motorola Japan Ltd.: SPD, Strategic Planning Office, 141, 4-32-1 Nishi-Gotanda, Shinagawa-Ku, Tokyo, Japan, 03-5487-8488
ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd., Silicon Harbour Centre, 2 Dai King Street, Tai Po Industrial Estate,
Tai Po, New Territories, Hong Kong, 852-26629298
Mfax™, Motorola Fax Back System: RMFAX0@email.sps.mot.com; http://sps.motorola.com/mfax/;
TOUCHTONE, 1-602-244-6609; US and Canada ONLY, 1-800-774-1848
HOME PAGE: http://motorola.com/sps/

MOTOROLA