Application Report
SPRA499

# Handling Different Interrupts of the TMS320F241, TMS320F243, and TMS320C242 DSPs

*Claire Monnet* *Digital Signal Processing Solutions*

## Abstract

The Texas Instruments (TI™) TMS320F241, TMS320F243, TMS320C242 devices contain many on-chip peripherals, each of them capable of generating multiple interrupts in response to various events. The C2xLP core can support one non-maskable interrupt and six maskable interrupt requests. It does not handle directly all of the peripheral interrupt requests. That is why a centralized interrupt controller is needed to arbitrate them. On the TMS320F241, TMS320F243, and TMS320C242 devices, this controller is called the Peripheral Interrupt Expansion (PIE).

This document explains the general organization of the PIE. An Interrupt Service Routine (ISR) programming example is also provided.

### Contents



### Figures

# Design Problem

The TI TMS320F241, TMS320F243, TMS320C242 devices contain many on-chip peripherals, each of them capable of generating multiple interrupts in response to various events. The C2xLP core can support one non-maskable interrupt and six maskable interrupt requests. It does not handle directly all of the peripheral interrupt requests. That is why a centralized interrupt controller is needed to arbitrate them. On the TMS320F241, TMS320F243, and TMS320C242 devices, this controller is called the Peripheral Interrupt Expansion (PIE).

This document explains the general organization of the PIE. An Interrupt Service Routine (ISR) programming example is also provided.

# Solution

## *1. Peripheral Interrupt Expansion Organization*

The PIE centralized interrupt controller handles all peripheral interrupt requests. It is connected to the peripherals, the CPU, and to the data and address buses.

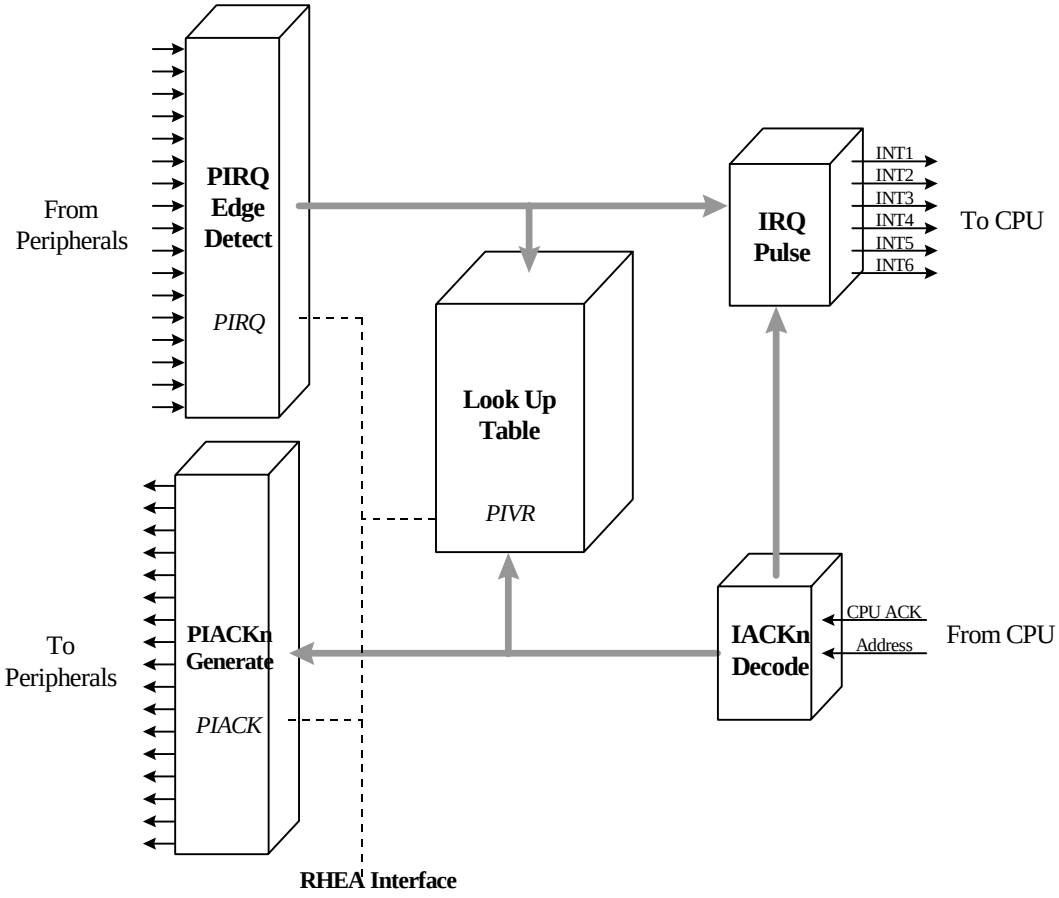
The PIE contains three registers.

**PIRQ** Peripheral Interrupt Request. Detects and records any interrupt request from peripherals

**PIVR** Peripheral Interrupt Vector Register. Contains the interrupt vector of the most recently acknowledged peripheral interrupt

**PIACK** Peripheral Interrupt Acknowledge. Sends the Interrupt Acknowledge to peripherals

*Figure 1. Peripheral Interrupt Expansion Block Diagram*




*PIRQ, PIACK, PIVR: interrupt registers*

- Interrupt Priority Level

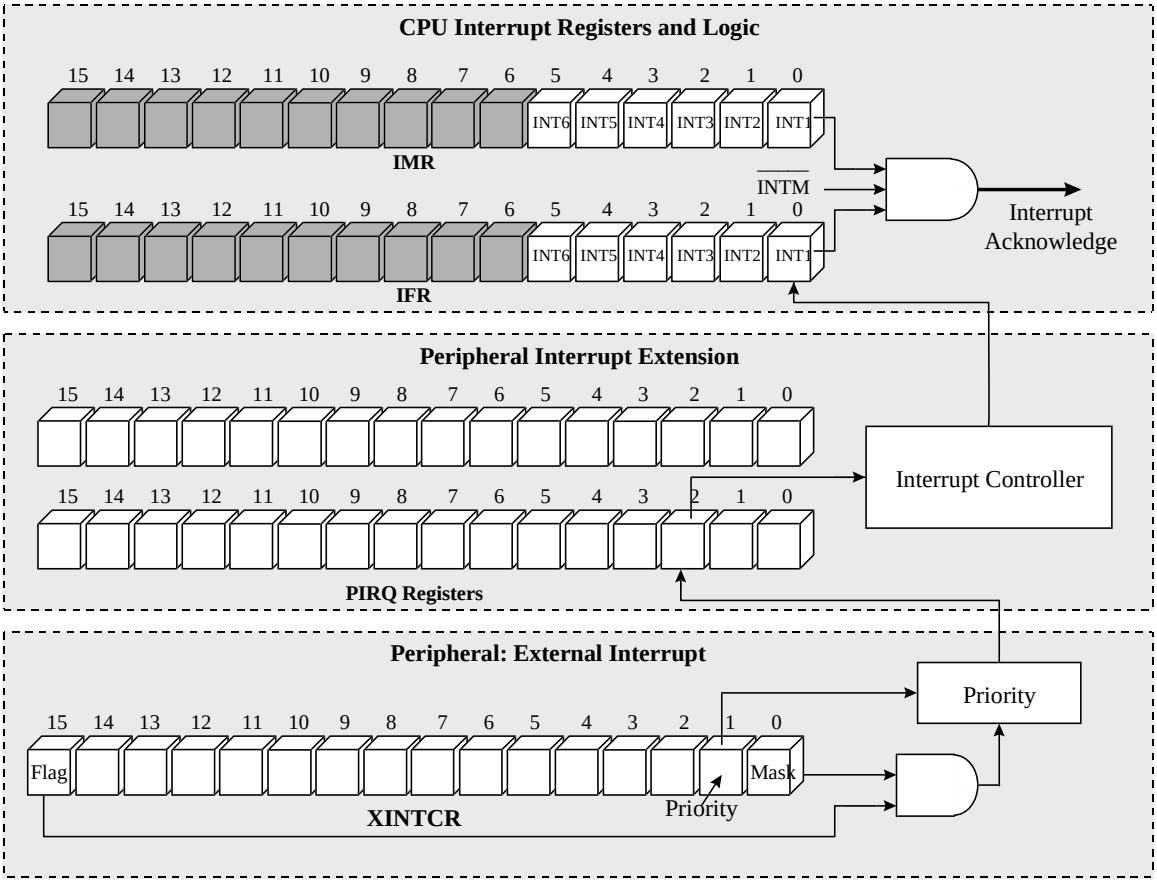
  Some peripheral has two different priority levels for the interrupt requests. ADC, the external interrupts, SCI, SPI and CAN may be configured to generate either low priority request and/or high priority request to the PIE controller.

- Interrupt Request

  In each peripheral, an interrupt flag bit and an interrupt enable bit will cause a PIRQ on the PIE module. This interrupt Request simply reflects the status of the peripheral. The request can be cleared by an interrupt acknowledge or by software (that is, clearing the interrupt flag by writing a 1 in the flag bit inside the peripheral).

  If no unacknowledged CPU interrupt request of the same priority level has previously been sent, an INT pulse is generated and sent to the CPU when PIRQ is active.

*Figure 2. Interrupt Request Example*




The interrupt request to the CPU will set the corresponding flag in the CPU interrupt flag register IFR. If the CPU interrupt has been enabled by setting the appropriate bit in the CPU interrupt mask register IMR and if the INTM bit is cleared, the CPU will acknowledge the interrupt.

- Interrupt Acknowledge

  After accepting the interrupt, the CPU sends to the PIE module an interrupt acknowledge and a value in the Program Address Bus (PAB), which corresponds to the interrupt priority level being responded to. The PIE decodes this value and recognizes which interrupt request has been acknowledged. The PIE sends to the peripherals a PIACK. This signal will be used by the peripheral to clear the interrupt flag.

- Execute Interrupt Routine

  When an interrupt request is acknowledged, the PIE controller loads the peripheral interrupt vector into the Peripheral Interrupt Vector Register from the table stored in the PIE module. Then the General Interrupt Service Routine (GISR) saves the necessary context and reads the PIVR. It executes the appropriate branch to the Specific Interrupt Routine Service (SISR) using the interrupt vector as a target address.

## 2. Interrupt Service Routine (ISR) Programming

There is no noticeable difference between the ISR programming in the X241/2/3 and the ISR programming in the X240 device.

To use interrupts:

- PART 1: Interrupt Branch
  - Branch all interrupts to a corresponding ISR

    INT1 B ISR1
- PART 2 : Main program
  - Disable all core interrupt before initialization:

    SETC INTM
  - Mask the non-used core interrupt

    SPLK #0000000000000001b, IMR => Mask all core interrupts except INT1
  - Reset all core interrupt flags

    SPLK #0000000000111111b,IFR
  - If only one peripheral interrupt by interrupt level: Mask all other peripheral interrupts
  - Unmasked the chosen interrupt
  - Clear the interrupt flags
  - Enable unmasked core interrupts.

    CLRC INTM
- PART 3: Interrupt Service Routine
  - Label correctly each interrupt
  - Write the code
  - Enable core interrupt

    CLRC INTM
  - Return to the main program.

    RET

Example 1 shows a programming example using the external interrupts (XINT1 & XINT2) and the Timer 1 period interrupt.

*Example 1. External Interrupt and Timer Period Interrupt*

```
.sect  ".vectors"

RSVECT          B      START           ; Reset Vector
INT1            B      EXTERN_ISR      ; Branch to EXTERN_ISR
INT2            B      PERIOD_ISR      ; Branch to PERIOD_ISR
```

```
INT3         B      PHANTOM          ; Branch to PHANTOM
INT4         B      ...
...                 .      ...
;=================================================================
;                                M A I N   C O D E
;=================================================================
      .text
START:

   SETC          INTM               ; Disable interrupts
   ...           ...                ; Initialization

   LDP              #0E1h           ; data pointer on 7080h
   SPLK          #0FFFFH,OCRA        ; Configure the Shared Pins
   SPLK          #003FFh,OCRB

;*************** External Interrupt Setting *********************

   LDP       #0E0h                      ; data pointer on 7000h
   SPLK      #1000000000000001b,XINT1CR ; Configure External
;             ||||||||||||||||           interrupt register 1
;             FEDCBA9876543210
; bit 15:    Interrupt Flag.
; bit 14-3:  Reserved
; bit 2:     0: Interrupt generated by a falling edge
; bit 1:     0: High priority
; bit 0:     1: Enable interrupt

   SPLK      #1000000000000001b,XINT2CR ; Configure External
                                        ; Interrupt register 2
; Enable XINT2, high priority generated on falling edge
;******************* Event Manager Setting ***********************

   LDP           #0E8h            ; Load Event Manager data page
   SPLK          #00080h, EVIMRA  ; Mask all group A interrupt except
                                  ; GP Timer 1 period interrupt
   SPLK          #00000h, EVIMRB  ; Mask all group B interrupt
   SPLK          #00000h, EVIMRC  ; Mask all group C interrupt
   SPLK          #0FFFFh, EVIVRA  ; Clear all group A interrupt flag
```

```
   SPLK         #0FFFFh, EVIVRB  ; Clear all group B interrupt flag
   SPLK         #0FFFFh, EVIVRC  ; Clear all group C interrupt flag

   SPLK         #0000h,T1CNT     ; Initialize GP Timer 1 counter
   SPLK         #1000h,T1PR      ; Initialize GP Timer 1 period
   SPLK         #0000h,GPTCON    ; Initialize GP Timer control


;************  Mask/Unmask and enable core interrupt ************;

   LDP          #0
   SPLK         #0000000000000011b, IMR; Mask all interrupt except
                                       ; INT1 and INT2
   SPLK         #000ffh,IFR            ; Clear all core interrupt
                                       ; flags
   CLRC         INTM                   ; Enable  core interrupt

;*****************     Start GP Timer 1    **********************

   LDP             #0E8h                  ; Load Event Manager data page
   SPLK         #0001000001000000, T1CON
;                FEDCBA9876543210

; bit 12-11  10: continuous-up count mode
; bit 10-8   000: prescaler = 1
; bit 6         1: Timer enabled
; bit 5-4       00: Internal clock selected

...    ...      ...




;=====================================================================
;                     Interrupt Service Routines
;=====================================================================

EXTERN_ISR                             ; ISR started by an external
                                       ; Interrupt
```

```
        LDP     #00E0h
        LACL    PIVR            ; Load peripheral INT vector address
        SUB     #0001h          ; Subtract XINT1 offset from above
        BCND    XINT1_ISR,EQ    ; If ACC=0, execute XINT1 SISR

        LACL    PIVR            ; Load peripheral INT vector address
        SUB     #0011h          ; Subtract XINT2 offset from above
        BCND    XINT2_ISR,EQ    ; If ACC=0, execute XINT2 SISR

        CLRC    INTM
        RET


;============================   XINT1   ============================
XINT1_ISR:
        LDP     #00E0h
        LACL    XINT1CR             ;Clear XINT1 flag
        SACL    XINT1CR

        CLRC    INTM                ; Enable Core Interrupt
        RET                         ; Return from ISR


;=========================    XINT2    ============================
XINT2_ISR:
        LDP     #00E0h
        LACL    XINT2CR             ;Clear XINT2 flag
        SACL    XINT2CR

        CLRC    INTM                ; Enable Core Interrupt
        RET                         ; Return from ISR




PERIOD_ISR                          ; ISR started by timer 1
                                    ; period interrupt
        ...     ...
```

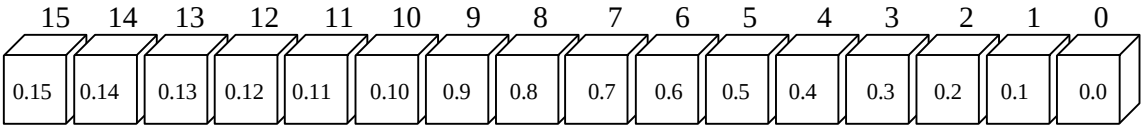
```
        CLRC   INTM                     ; Enable Core Interrupt
        RET                             ; Return from ISR


PHANTOM                                 ; Phantom interrupt

        RET                             ; Return from ISR
```

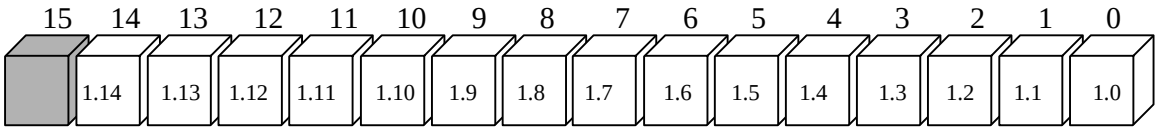
# Appendix A. Peripheral Interrupt Request Registers and Peripheral Interrupt Acknowledge Registers

*PIRQR0 or PIACKR0:*




| Bit position | *Interrupt* | *Interrupt Description* | Interrupt Level |
|---|---|---|---|
| 0.0 | PDPINT | Power Device Protection interrupt pin. | INT 1 |
| 0.1 | ADCINT | ADC Interrupt. High priority | INT1 |
| 0.2 | XINT1 | External Interrupt pin 1. High priority | INT1 |
| 0.3 | XINT2 | External Interrupt pin 2. High priority | INT1 |
| 0.4 | SPIINT | SPI interrupt. High priority | INT1 |
| 0.5 | RXINT | SCI receiver interrupt. High priority | INT1 |
| 0.6 | TXINT | SCI transmitter interrupt. High priority | INT1 |
| 0.7 | CANMBINT | CAN mailbox interrupt. High priority | INT1 |
| 0.8 | CANERINT | CAN error interrupt. High priority | INT1 |
| 0.9 | CMP1INT | Compare 1 interrupt | INT2 |
| 0.10 | CMP2INT | Compare 2 interrupt | INT2 |
| 0.11 | CMP3INT | Compare 3 interrupt | INT2 |
| 0.12 | TPINT1 | Timer 1 period interrupt | INT2 |
| 0.13 | TCINT1 | Timer 1 compare interrupt | INT2 |
| 0.14 | TUFINT1 | Timer 1 underflow interrupt | INT2 |
| 0.15 | TOFINT1 | Timer 1 overflow interrupt | INT2 |

## *PIRQR1 or PIACKR1:*




| Bit position | Interrupt | Interrupt Description | Interrupt Level |
|---|---|---|---|
| 1.0 | TPINT2 | Timer 2 period interrupt | INT3 |
| 1.1 | TCINT2 | Timer 2 compare interrupt | INT3 |
| 1.2 | TUFINT2 | Timer 2 underflow interrupt | INT3 |
| 1.3 | TOFINT2 | Timer 2 overflow interrupt | INT3 |
| 1.4 | CAPINT1 | Capture 1 interrupt | INT4 |
| 1.5 | CAPINT2 | Capture 2 interrupt | INT4 |
| 1.6 | CAPINT3 | Capture 3 interrupt | INT4 |
| 1.7 | SPIINT | SPI interrupt. Low priority | INT5 |
| 1.8 | RXINT | SCI receiver interrupt. Low priority | INT5 |
| 1.9 | TXINT | SCI transmitter interrupt. Low priority | INT5 |
| 1.10 | CANMBINT | CAN mailbox interrupt. Low priority | INT5 |
| 1.11 | CANERINT | CAN error interrupt. Low priority | INT5 |
| 1.12 | ADCINT | ADC Interrupt. Low priority | INT6 |
| 1.13 | XINT1 | External Interrupt pin 1. Low priority | INT6 |
| 1.14 | XINT2 | External Interrupt pin 2. Low priority | INT6 |