```
/*******************************************************************************
* File Name: isr_dist.c
* Version 1.60
*
*  Description:
*   API for controlling the state of an interrupt.
*
*
*  Note:
*
********************************************************************************
* Copyright 2008-2010, Cypress Semiconductor Corporation.  All rights reserved.
* You may use this file only in accordance with the license, terms, conditions,
* disclaimers, and limitations in the end user license agreement accompanying
* the software package with which this file was provided.
*******************************************************************************/



#include <CYDEVICE.H>
#include <CYDEVICE_TRM.H>
#include <CYLIB.H>
#include <isr_dist.H>


/*******************************************************************************
*  Place your includes, defines and code here
********************************************************************************/
/* `#START isr_dist_intc` */
#include <device.h>
/* `#END` */

#ifndef CYINT_IRQ_BASE
#define CYINT_IRQ_BASE  16
#endif
#ifndef CYINT_VECT_TABLE
#define CYINT_VECT_TABLE    ((cyisraddress **) CYREG_NVIC_VECT_OFFSET)
#endif

/* Declared in startup, used to set unused interrupts to. */
CY_ISR_PROTO(IntDefaultHandler);

/*******************************************************************************
* Function Name: isr_dist_Start
********************************************************************************
* Summary:
*  Set up the interrupt and enable it.
*
* Parameters:
*   void.
*
*
* Return:
*   void.
*
*******************************************************************************/
void isr_dist_Start(void)
{
```

```
    /* For all we know the interrupt is active. */
    isr_dist_Disable();

    /* Set the ISR to point to the isr_dist Interrupt. */
    isr_dist_SetVector(isr_dist_Interrupt);

    /* Set the priority. */
    isr_dist_SetPriority(isr_dist_INTC_PRIOR_NUMBER);

    /* Enable it. */
    isr_dist_Enable();
}

/*******************************************************************************
* Function Name: isr_dist_StartEx
********************************************************************************
* Summary:
*  Set up the interrupt and enable it.
*
* Parameters:
*   address: Address of the ISR to set in the interrupt vector table.
*
*
* Return:
*   void.
*
*******************************************************************************/
void isr_dist_StartEx(cyisraddress address)
{
    /* For all we know the interrupt is active. */
    isr_dist_Disable();

    /* Set the ISR to point to the isr_dist Interrupt. */
    isr_dist_SetVector(address);

    /* Set the priority. */
    isr_dist_SetPriority(isr_dist_INTC_PRIOR_NUMBER);

    /* Enable it. */
    isr_dist_Enable();
}

/*******************************************************************************
* Function Name: isr_dist_Stop
********************************************************************************
* Summary:
*   Disables and removes the interrupt.
*
* Parameters:
*
*
* Return:
*   void.
*
*******************************************************************************/
void isr_dist_Stop(void)
{
```

```
    /* Disable this interrupt. */
    isr_dist_Disable();

    /* Set the ISR to point to the passive one. */
    isr_dist_SetVector(IntDefaultHandler);
}

/*******************************************************************************
* Function Name: isr_dist_Interrupt
********************************************************************************
* Summary:
*   The default Interrupt Service Routine for isr_dist.
*
*   Add custom code between the coments to keep the next version of this file
*   from over writting your code.
*
*
*
* Parameters:
*
*
* Return:
*   void.
*
*******************************************************************************/
CY_ISR(isr_dist_Interrupt)
{
    /*  Place your Interrupt code here. */
    /* `#START isr_dist_Interrupt` */
    WaitBurstVar++;
    /* `#END` */
}

/*******************************************************************************
* Function Name: isr_dist_SetVector
********************************************************************************
* Summary:
*   Change the ISR vector for the Interrupt. Note calling isr_dist_Start
*   will override any effect this method would have had. To set the vector before
*   the component has been started use isr_dist_StartEx instead.
*
*
* Parameters:
*   address: Address of the ISR to set in the interrupt vector table.
*
*
* Return:
*   void.
*
*
*******************************************************************************/
void isr_dist_SetVector(cyisraddress address)
{
    cyisraddress * ramVectorTable;

    ramVectorTable = (cyisraddress *) *CYINT_VECT_TABLE;
```

```
    ramVectorTable[CYINT_IRQ_BASE + isr_dist__INTC_NUMBER] = address;
}

/*******************************************************************************
* Function Name: isr_dist_GetVector
********************************************************************************
* Summary:
*   Gets the "address" of the current ISR vector for the Interrupt.
*
*
* Parameters:
*   void.
*
*
* Return:
*   Address of the ISR in the interrupt vector table.
*
*
*******************************************************************************/
cyisraddress isr_dist_GetVector(void)
{
    cyisraddress * ramVectorTable;

    ramVectorTable = (cyisraddress *) *CYINT_VECT_TABLE;

    return ramVectorTable[CYINT_IRQ_BASE + isr_dist__INTC_NUMBER];
}

/*******************************************************************************
* Function Name: isr_dist_SetPriority
********************************************************************************
* Summary:
*   Sets the Priority of the Interrupt. Note calling isr_dist_Start
*   or isr_dist_StartEx will override any effect this method would have had.
*   This method should only be called after isr_dist_Start or
*   isr_dist_StartEx has been called. To set the initial
*   priority for the component use the cydwr file in the tool.
*
*
* Parameters:
*   priority: Priority of the interrupt. 0 - 7, 0 being the highest.
*
*
* Return:
*   void.
*
*
*******************************************************************************/
void isr_dist_SetPriority(uint8 priority)
{
    *isr_dist_INTC_PRIOR = priority << 5;
}

/*******************************************************************************
* Function Name: isr_dist_GetPriority
********************************************************************************
* Summary:
```

```
*   Gets the Priority of the Interrupt.
*
*
* Parameters:
*   void.
*
*
* Return:
*   Priority of the interrupt. 0 - 7, 0 being the highest.
*
*
********************************************************************************/
uint8 isr_dist_GetPriority(void)
{
    uint8 priority;


    priority = *isr_dist_INTC_PRIOR >> 5;

    return priority;
}

/*******************************************************************************
* Function Name: isr_dist_Enable
********************************************************************************
* Summary:
*   Enables the interrupt.
*
*
* Parameters:
*   void.
*
*
* Return:
*   void.
*
*
********************************************************************************/
void isr_dist_Enable(void)
{
    /* Enable the general interrupt. */
    *isr_dist_INTC_SET_EN = isr_dist__INTC_MASK;
}

/*******************************************************************************
* Function Name: isr_dist_GetState
********************************************************************************
* Summary:
*   Gets the state (enabled, disabled) of the Interrupt.
*
*
* Parameters:
*   void.
*
*
* Return:
*   1 if enabled, 0 if disabled.
```

```
*
*
********************************************************************************/
uint8 isr_dist_GetState(void)
{
    /* Get the state of the general interrupt. */
    return (*isr_dist_INTC_SET_EN & isr_dist__INTC_MASK) ? 1:0;
}

/*******************************************************************************
* Function Name: isr_dist_Disable
********************************************************************************
* Summary:
*   Disables the Interrupt.
*
*
* Parameters:
*   void.
*
*
* Return:
*   void.
*
*
*******************************************************************************/
void isr_dist_Disable(void)
{
    /* Disable the general interrupt. */
    *isr_dist_INTC_CLR_EN = isr_dist__INTC_MASK;
}

/*******************************************************************************
* Function Name: isr_dist_SetPending
********************************************************************************
* Summary:
*   Causes the Interrupt to enter the pending state, a software method of
*   generating the interrupt.
*
*
* Parameters:
*   void.
*
*
* Return:
*   void.
*
*
*******************************************************************************/
void isr_dist_SetPending(void)
{
    *isr_dist_INTC_SET_PD = isr_dist__INTC_MASK;
}

/*******************************************************************************
* Function Name: isr_dist_ClearPending
********************************************************************************
* Summary:
```

```
*   Clears a pending interrupt.
*
* Parameters:
*   void.
*
*
* Return:
*   void.
*
*
*******************************************************************************/
void isr_dist_ClearPending(void)
{
    *isr_dist_INTC_CLR_PD = isr_dist__INTC_MASK;
}
```