```c
/*******************************************************************************
* File Name: Change_I2C_ADD.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 <Change_I2C_ADD.H>



/*******************************************************************************
*  Place your includes, defines and code here
********************************************************************************/
/* `#START Change_I2C_ADD_intc` */
#include <device.h>
#define default_add 0b00000000
uint8 Address = default_add;
/* `#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: Change_I2C_ADD_Start
********************************************************************************
* Summary:
*   Set up the interrupt and enable it.
*
* Parameters:
*   void.
*
*
* Return:
*   void.
*
*******************************************************************************/
```

```c
void Change_I2C_ADD_Start(void)
{
    /* For all we know the interrupt is active. */
    Change_I2C_ADD_Disable();

    /* Set the ISR to point to the Change_I2C_ADD Interrupt. */
    Change_I2C_ADD_SetVector(Change_I2C_ADD_Interrupt);

    /* Set the priority. */
    Change_I2C_ADD_SetPriority(Change_I2C_ADD_INTC_PRIOR_NUMBER);

    /* Enable it. */
    Change_I2C_ADD_Enable();
}


/*******************************************************************************
* Function Name: Change_I2C_ADD_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 Change_I2C_ADD_StartEx(cyisraddress address)
{
    /* For all we know the interrupt is active. */
    Change_I2C_ADD_Disable();

    /* Set the ISR to point to the Change_I2C_ADD Interrupt. */
    Change_I2C_ADD_SetVector(address);

    /* Set the priority. */
    Change_I2C_ADD_SetPriority(Change_I2C_ADD_INTC_PRIOR_NUMBER);

    /* Enable it. */
    Change_I2C_ADD_Enable();
}


/*******************************************************************************
* Function Name: Change_I2C_ADD_Stop
********************************************************************************
* Summary:
*   Disables and removes the interrupt.
*
* Parameters:
*
*
* Return:
*   void.
*
*******************************************************************************/
```

```c
void Change_I2C_ADD_Stop(void)
{
    /* Disable this interrupt. */
    Change_I2C_ADD_Disable();

    /* Set the ISR to point to the passive one. */
    Change_I2C_ADD_SetVector(IntDefaultHandler);
}


/*******************************************************************************
* Function Name: Change_I2C_ADD_Interrupt
********************************************************************************
* Summary:
*   The default Interrupt Service Routine for Change_I2C_ADD.
*
*   Add custom code between the coments to keep the next version of this file
*   from over writting your code.
*
*
*
* Parameters:
*
*
* Return:
*   void.
*
*******************************************************************************/
CY_ISR(Change_I2C_ADD_Interrupt)
{
    /*  Place your Interrupt code here. */
    /* `#START Change_I2C_ADD_Interrupt` */
    Address = default_add;
    uint8 LSB = I2C_add_LSB_Read();
    uint8 LSB_1 = I2C_add_LSB_1_Read();
    if(LSB)
        Address = 1;
    if(LSB_1)
        Address = 2;
    if(LSB & LSB_1)
        Address = 3;

    I2C_1_SlaveSetAddress(Address);



    /* `#END` */
}


/*******************************************************************************
* Function Name: Change_I2C_ADD_SetVector
********************************************************************************
* Summary:
*   Change the ISR vector for the Interrupt. Note calling Change_I2C_ADD_Start
*   will override any effect this method would have had. To set the vector before
*   the component has been started use Change_I2C_ADD_StartEx instead.
*
*
* Parameters:
```

```
*    address: Address of the ISR to set in the interrupt vector table.
*
*
* Return:
*    void.
*
*
*******************************************************************************/
void Change_I2C_ADD_SetVector(cyisraddress address)
{
    cyisraddress * ramVectorTable;

    ramVectorTable = (cyisraddress *) *CYINT_VECT_TABLE;

    ramVectorTable[CYINT_IRQ_BASE + Change_I2C_ADD__INTC_NUMBER] = address;
}


/*******************************************************************************
* Function Name: Change_I2C_ADD_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 Change_I2C_ADD_GetVector(void)
{
    cyisraddress * ramVectorTable;

    ramVectorTable = (cyisraddress *) *CYINT_VECT_TABLE;

    return ramVectorTable[CYINT_IRQ_BASE + Change_I2C_ADD__INTC_NUMBER];
}


/*******************************************************************************
* Function Name: Change_I2C_ADD_SetPriority
********************************************************************************
* Summary:
*    Sets the Priority of the Interrupt. Note calling Change_I2C_ADD_Start
*    or Change_I2C_ADD_StartEx will override any effect this method would have had.
*    This method should only be called after Change_I2C_ADD_Start or
*    Change_I2C_ADD_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.
*
*
```

```c
 * Return:
 *    void.
 *
 *
 *******************************************************************/
void Change_I2C_ADD_SetPriority(uint8 priority)
{
    *Change_I2C_ADD_INTC_PRIOR = priority << 5;
}


/*******************************************************************
 * Function Name: Change_I2C_ADD_GetPriority
 *******************************************************************
 * Summary:
 *    Gets the Priority of the Interrupt.
 *
 *
 * Parameters:
 *    void.
 *
 *
 * Return:
 *    Priority of the interrupt. 0 - 7, 0 being the highest.
 *
 *
 *******************************************************************/
uint8 Change_I2C_ADD_GetPriority(void)
{
    uint8 priority;


    priority = *Change_I2C_ADD_INTC_PRIOR >> 5;

    return priority;
}


/*******************************************************************
 * Function Name: Change_I2C_ADD_Enable
 *******************************************************************
 * Summary:
 *    Enables the interrupt.
 *
 *
 * Parameters:
 *    void.
 *
 *
 * Return:
 *    void.
 *
 *
 *******************************************************************/
void Change_I2C_ADD_Enable(void)
{
    /* Enable the general interrupt. */
    *Change_I2C_ADD_INTC_SET_EN = Change_I2C_ADD__INTC_MASK;
}
```

```c
/*******************************************************************************
* Function Name: Change_I2C_ADD_GetState
********************************************************************************
* Summary:
*   Gets the state (enabled, disabled) of the Interrupt.
*
*
* Parameters:
*   void.
*
*
* Return:
*   1 if enabled, 0 if disabled.
*
*
*******************************************************************************/
uint8 Change_I2C_ADD_GetState(void)
{
    /* Get the state of the general interrupt. */
    return (*Change_I2C_ADD_INTC_SET_EN & Change_I2C_ADD__INTC_MASK) ? 1:0;
}


/*******************************************************************************
* Function Name: Change_I2C_ADD_Disable
********************************************************************************
* Summary:
*   Disables the Interrupt.
*
*
* Parameters:
*   void.
*
*
* Return:
*   void.
*
*
*******************************************************************************/
void Change_I2C_ADD_Disable(void)
{
    /* Disable the general interrupt. */
    *Change_I2C_ADD_INTC_CLR_EN = Change_I2C_ADD__INTC_MASK;
}


/*******************************************************************************
* Function Name: Change_I2C_ADD_SetPending
********************************************************************************
* Summary:
*   Causes the Interrupt to enter the pending state, a software method of
*   generating the interrupt.
*
*
* Parameters:
*   void.
*
*
```

```c
 * Return:
 *   void.
 *
 *
 *******************************************************************************/
void Change_I2C_ADD_SetPending(void)
{
    *Change_I2C_ADD_INTC_SET_PD = Change_I2C_ADD__INTC_MASK;
}


/*******************************************************************************
 * Function Name: Change_I2C_ADD_ClearPending
 *******************************************************************************
 * Summary:
 *   Clears a pending interrupt.
 *
 * Parameters:
 *   void.
 *
 *
 * Return:
 *   void.
 *
 *
 *******************************************************************************/
void Change_I2C_ADD_ClearPending(void)
{
    *Change_I2C_ADD_INTC_CLR_PD = Change_I2C_ADD__INTC_MASK;
}
```