```
/**
  Generated Pin Manager File

  Company:
    Microchip Technology Inc.

  File Name:
    pin_manager.c

  Summary:
    This is the Pin Manager file generated using PIC10 / PIC12 / PIC16 / PIC18
MCUs

  Description:
    This header file provides implementations for pin APIs for all pins selected
in the GUI.
    Generation Information :
        Product Revision  :  PIC10 / PIC12 / PIC16 / PIC18 MCUs - 1.81.8
        Device            :  PIC16F1619
        Driver Version    :  2.11
    The generated drivers are tested against the following:
        Compiler          :  XC8 2.36 and above
        MPLAB             :  MPLAB X 6.00

    Copyright (c) 2013 - 2015 released Microchip Technology Inc.  All rights
reserved.
*/

/*
    (c) 2018 Microchip Technology Inc. and its subsidiaries.

    Subject to your compliance with these terms, you may use Microchip software
and any
    derivatives exclusively with Microchip products. It is your responsibility
to comply with third party
    license terms applicable to your use of third party software (including open
source software) that
    may accompany Microchip software.

    THIS SOFTWARE IS SUPPLIED BY MICROCHIP "AS IS". NO WARRANTIES, WHETHER
    EXPRESS, IMPLIED OR STATUTORY, APPLY TO THIS SOFTWARE, INCLUDING ANY
    IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS
    FOR A PARTICULAR PURPOSE.

    IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE,
    INCIDENTAL OR CONSEQUENTIAL LOSS, DAMAGE, COST OR EXPENSE OF ANY KIND
    WHATSOEVER RELATED TO THE SOFTWARE, HOWEVER CAUSED, EVEN IF MICROCHIP
    HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO
    THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP'S TOTAL LIABILITY ON ALL
    CLAIMS IN ANY WAY RELATED TO THIS SOFTWARE WILL NOT EXCEED THE AMOUNT
    OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THIS
    SOFTWARE.
*/
```

```
#include "pin_manager.h"

#include <xc.h>

#include "mcc.h"

#define Rx_out PORTAbits.RA2

void (*IOCCF3_InterruptHandler)(void);


void PIN_MANAGER_Initialize(void)
{
    /**
    LATx registers
    */
    LATA = 0x00;
    LATB = 0x00;
    LATC = 0x00;

    /**
    TRISx registers
    */
    TRISA = 0x31;
    TRISB = 0x30;
    TRISC = 0x2D;

    /**
    ANSELx registers
    */
    ANSELC = 0x05;
    ANSELB = 0x30;
    ANSELA = 0x11;

    /**
    WPUx registers
    */
    WPUB = 0x00;
    WPUA = 0x00;
    WPUC = 0x00;
    OPTION_REGbits.nWPUEN = 1;

    /**
    ODx registers
    */
    ODCONA = 0x00;
    ODCONB = 0x00;
    ODCONC = 0x00;

    /**
    SLRCONx registers
    */
    SLRCONA = 0x37;
    SLRCONB = 0xF0;
```

```
    SLRCONC = 0xFF;

    /**
    INLVLx registers
    */
    INLVLA = 0x3F;
    INLVLB = 0xF0;
    INLVLC = 0xFF;


    /**
    IOCx registers
    */
    //interrupt on change for group IOCCF - flag
    IOCCFbits.IOCCF3 = 0;
    //interrupt on change for group IOCCN - negative
    IOCCNbits.IOCCN3 = 1;
    //interrupt on change for group IOCCP - positive
    IOCCPbits.IOCCP3 = 0;



    // register default IOC callback functions at runtime; use these methods to
register a custom function
    IOCCF3_SetInterruptHandler(IOCCF3_DefaultInterruptHandler);

    // Enable IOCI interrupt
    INTCONbits.IOCIE = 1;

}

void PIN_MANAGER_IOC(void)
{
        // interrupt on change for pin IOCCF3
    if(IOCCFbits.IOCCF3 == 1)
    {
        IOCCF3_ISR();
    }
}

/**
   IOCCF3 Interrupt Service Routine
*/
void IOCCF3_ISR(void) {

    // Add custom IOCCF3 code

    // The DTR/RTS signal has just gone low, so switch to Rx immediately

                // switch transceiver to Rx mode
                Rx_out = 1;
                __delay_ms(50);
                Rx_out = 0;
```

```
    // Call the interrupt handler for the callback registered at runtime
    if(IOCCF3_InterruptHandler)
    {
        IOCCF3_InterruptHandler();
    }
    IOCCFbits.IOCCF3 = 0;
}

/**
  Allows selecting an interrupt handler for IOCCF3 at application runtime
*/
void IOCCF3_SetInterruptHandler(void (* InterruptHandler)(void)){
    IOCCF3_InterruptHandler = InterruptHandler;
}

/**
  Default interrupt handler for IOCCF3
*/
void IOCCF3_DefaultInterruptHandler(void){
    // add your IOCCF3 interrupt custom code
    // or set custom function using IOCCF3_SetInterruptHandler()


}

/**
 End of File
*/
```