```
/*
 * Ejemplo_SPI.c
 *
 * Este ejemplo configura el puerto SPI-A
 * los pines empleados son:
 *
 * SPISOMI:   GPIO17
 * SPISIMO:   GPIO16
 * SPISTE:    GPIO18
 * SPICLK:    GPIO19
 *
 * El pin CS se encuentra negado. Si se
 * conectan entre si los pines MISO y
 * MOSI se realiza un "eco" de los datos
 * enviados los cuales se pueden ver en la
 * variable "data"
 *
 * La configuración del puerto es:
 *
 * Bits: 16
 * SCLK: 12.5 MHz
 * Modo: Maestro
*/

#include "F28x_Project.h"

bool flag;
Uint16 i;
Uint16 data;

interrupt void SPIisr(void){

        flag = true;
        data = SpiaRegs.SPIRXBUF;

        // Limpia la bandera de OverFlow
        SpiaRegs.SPIFFRX.bit.RXFFOVFCLR=1;
        // Limpia la bandera de interrupcion
        SpiaRegs.SPIFFRX.bit.RXFFINTCLR=1;
        // Issue PIE ack
        PieCtrlRegs.PIEACK.all|=0x20;
        __asm("_NOP");
}

void main(){
        InitSysCtrl();
        InitPieCtrl();
        DINT;
        IER = 0x0000;
        IFR = 0x0000;

        InitPieVectTable();

        // Configura los GPIO
        InitSpiGpio();

        EALLOW;
        PieVectTable.SPIA_RX_INT = &SPIisr;
        EDIS;

        // Desactiva el módulo SPI
        SpiaRegs.SPICCR.bit.SPISWRESET = 0;
        // Selecciona Polaridad
        SpiaRegs.SPICCR.bit.CLKPOLARITY = 0;
        // Desactiva el modo de alta velocidad
        SpiaRegs.SPICCR.bit.HS_MODE = 0;
```

```
        // Deshabilita el módo LoopBack
        SpiaRegs.SPICCR.bit.SPILBK = 0;
        // Define la longitud de palabra en 16 bits
        SpiaRegs.SPICCR.bit.SPICHAR = 0xF;

        SpiaRegs.SPICTL.bit.OVERRUNINTENA = 0;
        // Selecciona el tipo de fase del clk
        SpiaRegs.SPICTL.bit.CLK_PHASE = 0;
        // Define al DSP como maestro
        SpiaRegs.SPICTL.bit.MASTER_SLAVE = 1;
        // Habilita la transmisión
        SpiaRegs.SPICTL.bit.TALK = 1;
        // Habilita la interrupción
        SpiaRegs.SPICTL.bit.SPIINTENA = 1;

        SpiaRegs.SPISTS.all = 0x00;
        //LSPCLK/(SPIBRR+1)=100M/(7+1)=12.5M
        SpiaRegs.SPIBRR.all = 0x07;

        SpiaRegs.SPIFFTX.bit.SPIRST = 1;
        SpiaRegs.SPIFFTX.bit.SPIFFENA = 0;
        // Habilita la FIFO (Tx)
        SpiaRegs.SPIFFTX.bit.TXFFIENA = 1;
        // Genera la interrupción cuando ya no
        // hay palabras en la FIFO
        SpiaRegs.SPIFFTX.bit.TXFFIL = 0;

        // Habilita la FIFO (Rx)
        SpiaRegs.SPIFFRX.bit.RXFFIENA =1;
        // Genera la interrupción cuando
        // se recibió una palabra
        SpiaRegs.SPIFFRX.bit.RXFFIL = 1;

        // Habilita el SPI
        SpiaRegs.SPICCR.bit.SPISWRESET = 1;


        SpiaRegs.SPIFFTX.bit.TXFIFO=1;
        SpiaRegs.SPIFFRX.bit.RXFIFORESET=1;

        // Habilita el bloque de PIE
        PieCtrlRegs.PIECTRL.bit.ENPIE = 1;
        //Habilita la interrupción 1 del grupo 6
        PieCtrlRegs.PIEIER6.bit.INTx1=1;
        //Habilita la interrupción 6 del CPU
        IER = 0x20;
        EINT;          // Habilita interrupciones

        i = 0;
        while(1){
                // Dato a enviar
                SpiaRegs.SPITXBUF = i;
                while(!flag); // Espera la interrupción
                flag = false;
                i++;          // Incrementa el dato a enviar
        }

}
```