;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;

; Author : ADI - Apps www.analog.com/MicroConverter

;

; Date : 7 March 2000

;

; File : SPImstr.asm

;

; Hardware : ADuC812

;

; Include File : UARTIO.asm - serial I/O routines

;

; Description : Demonstrates an example master mode SPI interface.

; Code is intended for use with companion code file

; 'SPIslave.asm' running on a second MicroConverter

; chip. Chips must have SCLK, MOSI, MISO, & GND pins

; connected together, and P3.5 pin on master must

; connect to SS pin on slave.

;

; If using the ADuC812 eval board, you can simply

; connect the 10-pin J5 header (SPI/I2C) directly to

; that of the slave board. However, on the master

; board you must also ensure that LK5 is REMOVED, and

; you'll have to manually jumper P3.5 to J3/pin7 (the

; SS pin of the SPI/I2C header). To configure the

; slave board, refer to 'SPIslave.asm'.

;

; Once hardware is connected, download code to both

; master & slave devices ('SPImstr' to the master,

; 'SPIslave' to the slave). Reset the slave first,

; and then the master. The slave will sit with the

; LED off until the master starts exchanging data

; with it at which time its LED will start blinking

; in sync (or 180°out of phase) with that of the

; master. When first launched, both master and slave

; are transmitting zeros repeatedly on the SPI port.

; Pressing the INT0 button on either master or slave

; increments the value it is transmitting. Received

; SPI data is relayed out the UART and can be viewed

; on any VT100 terminal or terminal emulator at

; 9600baud/8bits/noparity/1stopbit. Characters sent

; from the terminal to the MicroConverter will update

; the value being transmitted by SPI.

;

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

$MOD812 ; Use 8052 & ADuC812 predefined symbols

LED EQU P3.4 ; P3.4 drives red LED on eval board

SS EQU P3.5 ; P3.5 drives slave device's SS pin

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

; DEFINE VARIABLES IN INTERNAL RAM

DSEG

ORG 0060h

INPUT: DS 1 ; data byte received by SPI

OUTPUT: DS 1 ; data byte to send by SPI

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

; BEGINNING OF CODE

CSEG

ORG 0000h

JMP MAIN ; jump to main program

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

; INTERRUPT VECTOR SPACE

ORG 0003h ; (.................... INT0 ISR)

INC OUTPUT

RETI

ORG 003Bh ; (.................... SPI ISR)

SETB SS ; pull slave's SS pin high

MOV INPUT,SPIDAT

RETI

;====================================================================

; MAIN PROGRAM

ORG 004Bh

MAIN:

MOV SP,#007h

; CONFIGURE UART...

MOV SCON,#52h ; configure UART for 9600baud..

MOV TMOD,#20h ; ..assuming 11.0592MHz crystal

MOV TH1,#-3

SETB TR1

; CONFIGURE SPI...

MOV SPICON,#037h ; configure SPI port for:

; Fosc/64, CPHA=1, CPOL=0, master

;==> NOTE: it is important that CPHA and CPOL be the same for both

; the master and all slave devices. otherwise, data will

; be transitioning at the same time as it's being latched.

MOV IE2,#1 ; enable SPI interrupt

; CONFIGURE INTERRUPT 0...

SETB IT0 ; INT0 edge triggered

SETB EX0 ; enable INT0 interrupt

; ENABLE INTERRUPTS & ENTER MAIN LOOP...

MOV OUTPUT,#0 ; set initial value for output byte

SETB EA ; enable inturrupts

LOOP: CPL LED ; flash the LED on the eval board

MOV A,OUTPUT ; byte to send via SPI into ACC

CALL SENDSPI ; trigger SPI send/receive transfer

CALL DELAY ; pause 100ms

MOV A,INPUT ; send value received by SPI..

CALL SENDVAL ; ..out the UART as 2 hex chars

MOV DPTR,#SEPERATOR ; send line-feed & crdg-return..

CALL SENDSTRING ; ..out the UART

JNB RI,LOOP ; repeat (unless UART data received)

; WHEN UART DATA RECEIVED, MOVE DATA TO SPI OUTPUT...

MOV OUTPUT,SBUF ; update OUTPUT byte to new value

CLR RI ; must clear RI

JMP LOOP ; back to main loop

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

; SUBROUTINES

SENDSPI: ; sends the value in ACC out the SPI port. also

; receives simultaneously into SPIDAT. SPI interrupt

; is triggered when transfer is complete.

CLR SS ; must pull slave's SS pin low first

MOV SPIDAT,OUTPUT ; trigger data transfer

RET

; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

DELAY: ; delays approximately 100ms

PUSH ACC

PUSH B

MOV A,#200 ; 200 \* 500us = 100ms

DLY1: MOV B,#229 ; 229 \* 2.17us = 500us

DJNZ B,$ ; sit here for 500us

DJNZ ACC,DLY1 ; repeat 200 times (100ms delay)

POP B

POP ACC

RET

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

; SUBROUTINE INCLUDE FILE

$INCLUDE(UARTIO.asm)

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

; TEXT DATA TABLES

SEPERATOR: DB 10,13,0

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

END