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

;

; Author : ADI - Apps

;

; Date : October 2003

;

; File : Tx1to10.asm

;

; Hardware : ADuC842/ADuC843

;

; Description : This Program transmits the numbers 1-10 in binary

; form continuously down the spi port.

; After the transmission of each byte the incoming

; byte is saved in order between internal RAM

; addresses 40h and 50h.

;

; An SPI slave program can be run on a second ADuC832

; to communicate with this master code.

; The Slave program (spislav.asm in the SPI\SLAVE

; directory) should be started after the master

; program (spimast.asm) but within the time delay

; of 5s in order that the slave program is

; synchronised by the first outputted clock of the

; master.

;

; The clock is outputted at sclock (pin 26)

; The data is outputted at sdata/MOSI (pin 27)

; The data is inputted at MISO (pin 14)

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

;

$MOD842 ;Use 8052 predefined Symbols

LED EQU P3.4

FLAG BIT 00H

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

; BEGINNING OF CODE

CSEG

ORG 0000H

JMP MAIN

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

; SPI INTERRUPT ROUTINE

ORG 003BH

SETB P3.5 ; set the SS bit after transmission

CLR FLAG ; Clear flag to leave loop

MOV @R1, SPIDAT ; move input into memory

INC R1 ; increment memory location so new

; data is stored in new address

CJNE R1, #50H, CONT ; check if memory location =50h.

; if not continue

MOV R1, #40H ; if so reset address to 40h to store

; the next 16 bytes to the same

; memory locations

CONT: RETI

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

ORG 0060H ; Start code at address above interrupts

MAIN: ; Main program

MOV SPICON,#037h ; Initialise SPICON to have

; -bitrate=fosc/64

; -CPHA=1

; -CPOL=0, sclk idling low

; -master mode select

; -Enable SPI serial port

MOV IEIP2, #01h ; Enable SPI interrrupt

SETB EA ; Enable Global Interrupts

MOV R1, #40h ; initialise R1 to 40 to store the

; input data from memory location 40

MOV R0, #00H ; initialise R0 to 0 to start

; transmissions from 1

; Delay the output of data by 2.0s in order that the slave program

; can be easily synchronised with the master program.

MOV A, #200

CALL DELAY

TRNSMT:

CLR P3.5 ; clear the SS bit during transmission

INC R0

MOV SPIDAT, R0 ; transmit the current value on R0

SETB FLAG ; set flag so that we wait here until

; the spi interrupt routine clears

; the FLAG

JB FLAG, $ ; stay here until flag is cleared

; by interrupt

; check if R0 is equal to 10. If so the number 10 has been

; transmitted and we should reset R0 to 0 to start transmission

; from 1 again

MOV A, R0

CJNE A, #0AH, TRNSMT ; if R0 is not 10, jump to TRNSMT

MOV R0, #00H ; if R0=10 make R0=0 & jump to TRNSMT

JMP TRNSMT

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

DELAY: ; Delays by 100ms \* A

; 10mSec based on 2.097152MHZ Core Clock

MOV R2,A ; Acc holds delay variable

DLY0: MOV R3,#01Bh ; Set up delay loop0

DLY1: MOV R4,#0FFh ; Set up delay loop1

DJNZ R4,$ ; Dec R4 & Jump here until R4 is 0

; wait here for

DJNZ R3,DLY1 ; Dec R3 & Jump DLY1 until R3 is 0

; Wait for

DJNZ R2,DLY0 ; Dec R2 & Jump DLY0 until R2 is 0

; wait for ACC\*100ms

RET ; Return from subroutine

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

END