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

; Author : ADI - Apps

; Date : May 2004

; File : intxram.asm

; Description : Writes 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 to SRAM and reads it back

; Resulting read back is transmitted via the UART. Can be viewed using Hyperterminal...

; 9600,8,none,1,hardware

;

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

$MOD847 ; Use 8052 predefined Symbols

CSEG ; Defines the following as a segment of code

ORG 0000H ; Load Code at '0'

JMP MAIN ; Jump to MAIN

ORG 0060h ;

MAIN: ;(main program)

MOV PLLCON, #05h

MOV T3CON,#81h ;configure UART for 9600 using Timer3

MOV T3FD,#12h

MOV SCON,#52h

MOV CFG847, #01h ;Configure Internal XRAM if LSB = 1. Otherwise External XRAM selected

MOV EWAIT, #07h ;Memory wait states.

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

;WRITE DATA TO XRAM (INT or EXTERNAL DEPENDING ON THE STATE OF CFG847.0)

MOV DPP, #00h ;Configure Page

MOV DPTR, #0000h ;Configure Addressing

; MOV DPH, #00h

; MOV DPL, #00h

count: MOV R0, #10 ;Count 10 writes before switching over to the read operation

MOV R1, #00h ;Initialise R1 as address counter instead of DPTR address counter

MOV A, #00h

loop:

MOVX @DPTR, A ;Use DPTR as the memory address register

INC A ;Increment the data

inc DPTR ;Increment the Address by incrementing the full DPTR

; INC DPL ;Increment just the low byte of the DPTR

DJNZ R0, Loop ;Continue until 10 bits of data have been written

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

;READ BACK DATA AND TRANSMIT TO HYPERTERMINAL ON A PC

MOV DPP, #00h ;Initialise page

MOV DPTR, #0000h ;Initialise ram address counter

; MOV DPH, #00h

; MOV DPL, #00h

MOV R0, #10 ;RAM Read counter Count 10 before stopping program

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

loop2:

MOVX A, @DPTR ;Use @R1 as address to move data from into Acc

PUSH DPP ;Save contents of DPTR (memory address) to stack

PUSH DPH

PUSH DPL

CALL sendval ;Transmit data via UART

MOV DPTR, #seperator ;Carriage return following data write to UART

CALL SENDSTRING

POP DPL ;Recover original contents of DPTR (memory address)

POP DPH

POP DPP

INC DPTR ;Increment address

; INC DPL ;Increment low byte of DPTR

DJNZ R0, LOOP2 ;Repeat until 10 bits of data have been read

complete:

JMP $ ;Wait here

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

;SUBROUTINES

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

SENDCHAR: ; sends ASCII value contained in A to UART

JNB TI,$ ; wait til present char gone

CLR TI ; must clear TI

MOV SBUF,A

RET

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

SENDVAL: ; converts the hex value of A into two ASCII chars,

; and then spits these two characters up the UART.

; does not change the value of A.

PUSH ACC

SWAP A

CALL HEX2ASCII

CALL SENDCHAR ; send high nibble

POP ACC

PUSH ACC

CALL HEX2ASCII

CALL SENDCHAR ; send low nibble

POP ACC

RET

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

HEX2ASCII: ; converts A into the hex character representing the

; value of A's least significant nibble

ANL A,#00Fh

CJNE A,#00Ah,$+3

JC IO0030

ADD A,#007h

IO0030: ADD A,#'0'

RET

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

SENDSTRING: ; sends ASCII string to UART starting at location

; DPTR and ending with a null (0) value

PUSH ACC

PUSH B

CLR A

MOV B,A

IO0010: MOV A,B

INC B

MOVC A,@A+DPTR

JZ IO0020

CALL SENDCHAR

JMP IO0010

IO0020: POP B

POP ACC

RET

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

SEPERATOR: DB 10,13,0

END