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

;

; Author : ADI - Apps

;

; Date : 11/17/99

;

; File : UART.asm

;

; Hardware : ADuC824

;

; Description : This Program transmits a number (starting at 0)

; down the UART every second. Pressing the INT0

; button increases the number being outputted.

; Pressing a key on the keyboard causes the ASCII

; char to be transmitted. Eg Pressing the 'A' button

; causes the number '41h' to appear on the

; hyperterminal program.

;

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

;

$MOD824 ;Use 8052 predefined Symbols

LED EQU P3.4

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

; 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

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

; INTERRUPT VECTOR SPACE

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

INC OUTPUT

RETI

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

ORG 0060H ; Start code at address above interrupts

MAIN: ; Main program

MOV RCAP2H,#0FFh ; config UART for 9830baud

MOV RCAP2L,#-5 ; (close enough to 9600baud)

MOV TH2,#0FFh

MOV TL2,#-5

MOV SCON,#52h

MOV T2CON,#34h

; 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

TXDATA:

CPL LED ; CPL LED with each transmission

MOV A, OUTPUT ; output value

CALL SENDVAL

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

CALL SENDSTRING ; ..out the UART

MOV A, #0Ah

CALL DELAY ; delay for 1s

JNB RI, TXDATA ; check if data is received from keyboard

MOV OUTPUT, SBUF

CLR RI

JMP TXDATA

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

; SENDSTRING

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

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

; SENDCHAR

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

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

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

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

; DELAY

DELAY: ; Delays by 100ms \* A

; 100mSec based on 1.573MHZ Core Clock

MOV R2,A ; Acc holds delay variable

DLY0: MOV R3,#50 ; Set up delay loop0

DLY1: MOV R4,#131 ; Set up delay loop1

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

; wait here for 131\*15.3us=2ms

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

; Wait for 50\*2ms

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

; wait for ACC\*100ms

RET ; Return from subroutine

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

SEPERATOR: DB 10,13,0

END