UART2 PAGE 1

1 ;====================================================================

2 ;

3 ; Author : ADI - Apps

4 ;

5 ; Date : November 2001

6 ;

7 ; File : UART.asm

8 ;

9 ; Hardware : ADuC836

10 ;

11 ; Description : This Program transmits the numbers 0->7F (starting

12 ; with 0) down the UART in ASCII form to the PC where

13 ; they can be viewed using the preconfigured

14 ; Hyperterminal program. (c:\ADuC\9600com1.ht)

15 ; After the transmission of the 16 bytes a 5 second

16 ; delay is called and the process is repeated.

17 ;

18 ;====================================================================

19 ;

20 $MOD836 ;Use 8052 predefined Symbols

21

00B4 22 LED EQU P3.4

23

24 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

25 ; BEGINNING OF CODE

---- 26 CSEG

0000 27 ORG 0000H

28

0000 020060 29 JMP MAIN

30

0060 31 ORG 0060H ; Start code at address above interrupts

32

33

0060 34 MAIN: ; Main program

35

0060 759E82 36 MOV T3CON,#82h

0063 759D12 37 MOV T3FD,#12h

0066 759852 38 MOV SCON,#52h

39

0069 40 START:

0069 B2B4 41 CPL LED ; CPL LED with each transmission

006B 9000EB 42 MOV DPTR, #TITLE

006E 12009F 43 CALL SENDSTRING ; write title block on screen

44

0071 7800 45 MOV R0, #00H ; Start transmissions from 0

0073 7908 46 MOV R1, #08H ; Start a new line after 8 transmissions

47

0075 48 LOOP1: ; Every eight transmissions start on a

49 ; new line

0075 740A 50 MOV A, #10 ; Transmit a linefeed

0077 1200B7 51 CALL SENDCHAR

007A 740D 52 MOV A, #13 ; Transmit a carriage return

007C 1200B7 53 CALL SENDCHAR

54

007F 7908 55 MOV R1, #08H

56

0081 57 LOOP2:

0081 7420 58 MOV A, #20H ; Transmit a SPACE (=ASCII 20) between

UART2 PAGE 2

59 ; transmissions on same line

0083 1200B7 60 CALL SENDCHAR

61

0086 E8 62 MOV A, R0 ; Transmit R0 = present data

0087 1200BF 63 CALL SENDVAL

008A 08 64 INC R0 ; increment data

65

008B B47F03 66 CJNE A, #7FH, CONT ; check if data =7F, if no continue

008E 020098 67 JMP WAIT5S ; if = 7F wait 5s and repeat

68

0091 19 69 CONT: DEC R1 ; decrement R1....

0092 E9 70 MOV A, R1

0093 B400EB 71 CJNE A, #00H, LOOP2 ; and check if new line is required

72 ; jump to loop 2 for a space

0096 80DD 73 JMP LOOP1 ; jump to loop 1 for a new line

74

75

0098 7432 76 WAIT5S: MOV A, #50 ; wait 5s

009A 1200DF 77 CALL DELAY

009D 80CA 78 JMP START ; start transmissions again

79

80

81 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

82 ; SENDSTRING

83

009F 84 SENDSTRING: ; sends ASCII string to UART starting at location

85 ; DPTR and ending with a null (0) value

86

009F C0E0 87 PUSH ACC

00A1 C0F0 88 PUSH B

00A3 E4 89 CLR A

00A4 F5F0 90 MOV B,A

00A6 E5F0 91 IO0010: MOV A,B

00A8 05F0 92 INC B

00AA 93 93 MOVC A,@A+DPTR

00AB 6005 94 JZ IO0020

00AD 1200B7 95 CALL SENDCHAR

00B0 80F4 96 JMP IO0010

00B2 D0F0 97 IO0020: POP B

00B4 D0E0 98 POP ACC

99

00B6 22 100 RET

101

102 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

103 ; SENDCHAR

104

00B7 105 SENDCHAR: ; sends ASCII value contained in A to UART

106

00B7 3099FD 107 JNB TI,$ ; wait til present char gone

00BA C299 108 CLR TI ; must clear TI

00BC F599 109 MOV SBUF,A

110

00BE 22 111 RET

112

113 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

114 ; SENDVAL

115

00BF 116 SENDVAL: ; converts the hex value of A into two ASCII chars,

UART2 PAGE 3

117 ; and then spits these two characters up the UART.

118 ; does not change the value of A.

119

00BF C0E0 120 PUSH ACC

00C1 C4 121 SWAP A

00C2 1200D3 122 CALL HEX2ASCII

00C5 11B7 123 CALL SENDCHAR ; send high nibble

00C7 D0E0 124 POP ACC

00C9 C0E0 125 PUSH ACC

00CB 1200D3 126 CALL HEX2ASCII

00CE 11B7 127 CALL SENDCHAR ; send low nibble

00D0 D0E0 128 POP ACC

129

00D2 22 130 RET

131

132

133 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

134 ; HEX2ASCII

135

00D3 136 HEX2ASCII: ; converts A into the hex character representing the

137 ; value of A's least significant nibble

138

00D3 540F 139 ANL A,#00Fh

00D5 B40A00 140 CJNE A,#00Ah,$+3

00D8 4002 141 JC IO0030

00DA 2407 142 ADD A,#007h

00DC 2430 143 IO0030: ADD A,#'0'

144

00DE 22 145 RET

146

147 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

148 ; DELAY

00DF 149 DELAY: ; Delays by 100ms \* A

150 ; 100mSec based on 1.573MHZ Core Clock

151

152

00DF FA 153 MOV R2,A ; Acc holds delay variable

00E0 7B32 154 DLY0: MOV R3,#50 ; Set up delay loop0

00E2 7C83 155 DLY1: MOV R4,#131 ; Set up delay loop1

00E4 DCFE 156 DJNZ R4,$ ; Dec R4 & Jump here until R4 is 0

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

00E6 DBFA 158 DJNZ R3,DLY1 ; Dec R3 & Jump DLY1 until R3 is 0

159 ; Wait for 50\*2ms

00E8 DAF6 160 DJNZ R2,DLY0 ; Dec R2 & Jump DLY0 until R2 is 0

161 ; wait for ACC\*100ms

00EA 22 162 RET ; Return from subroutine

163

164 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

165

166

00EB 0A0A0D5F 167 TITLE: DB 10,10,13,'\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_',10,13

00EF 5F5F5F5F

00F3 5F5F5F5F

00F7 5F5F5F5F

00FB 5F5F5F5F

00FF 5F5F5F5F

0103 5F5F5F5F

0107 5F5F5F5F

UART2 PAGE 4

010B 5F5F5F5F

010F 5F5F5F0A

0113 0D

0114 416E616C 168 DB 'Analog Devices MicroConverter ADuC836',10,13

0118 6F672044

011C 65766963

0120 6573204D

0124 6963726F

0128 436F6E76

012C 65727465

0130 72204144

0134 75433833

0138 360A0D

013B 20202020 169 DB ' UART Demo Routine',10,13

013F 20202020

0143 20554152

0147 54204465

014B 6D6F2052

014F 6F757469

0153 6E650A0D

0157 20205472 170 DB ' Transmission of Data from 0 to 7F',10,13,0

015B 616E736D

015F 69737369

0163 6F6E206F

0167 66204461

016B 74612066

016F 726F6D20

0173 3020746F

0177 2037460A

017B 0D00

171

172 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

UART2 PAGE 5

ACC. . . . . . . . . . . . . . . D ADDR 00E0H PREDEFINED

B. . . . . . . . . . . . . . . . D ADDR 00F0H PREDEFINED

CONT . . . . . . . . . . . . . . C ADDR 0091H

DELAY. . . . . . . . . . . . . . C ADDR 00DFH

DLY0 . . . . . . . . . . . . . . C ADDR 00E0H

DLY1 . . . . . . . . . . . . . . C ADDR 00E2H

HEX2ASCII. . . . . . . . . . . . C ADDR 00D3H

IO0010 . . . . . . . . . . . . . C ADDR 00A6H

IO0020 . . . . . . . . . . . . . C ADDR 00B2H

IO0030 . . . . . . . . . . . . . C ADDR 00DCH

LED. . . . . . . . . . . . . . . NUMB 00B4H

LOOP1. . . . . . . . . . . . . . C ADDR 0075H

LOOP2. . . . . . . . . . . . . . C ADDR 0081H

MAIN . . . . . . . . . . . . . . C ADDR 0060H

P3 . . . . . . . . . . . . . . . D ADDR 00B0H PREDEFINED

SBUF . . . . . . . . . . . . . . D ADDR 0099H PREDEFINED

SCON . . . . . . . . . . . . . . D ADDR 0098H PREDEFINED

SENDCHAR . . . . . . . . . . . . C ADDR 00B7H

SENDSTRING . . . . . . . . . . . C ADDR 009FH

SENDVAL. . . . . . . . . . . . . C ADDR 00BFH

START. . . . . . . . . . . . . . C ADDR 0069H

T3CON. . . . . . . . . . . . . . D ADDR 009EH PREDEFINED

T3FD . . . . . . . . . . . . . . D ADDR 009DH PREDEFINED

TI . . . . . . . . . . . . . . . B ADDR 0099H PREDEFINED

TITLE. . . . . . . . . . . . . . C ADDR 00EBH

WAIT5S . . . . . . . . . . . . . C ADDR 0098H