836UART2 PAGE 1

1 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2 ;

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

4 ;

5 ; Date : 5 November 2001

6 ;

7 ; File : 836uart2.asm

8 ;

9 ; Hardware : ADuC836

10 ;

11 ; Description : sample program that performs ADC conversions in

12 ; continuous mode and sends results to a PC via the

13 ; UART. program also accepts commands from the PC

14 ; via incoming characters on the UART. an ASCII "0"

15 ; innitiates a zero-scale calibration, and an ASCII

16 ; "1" innitiates a full-scale calibration.

17 ;

18 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

19

20 $MOD836 ; Use 8052&ADuC836 predefined symbols

21

00B4 22 LED EQU P3.4 ; P3.4 drives red LED on eval board

23

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

25 ; BEGINNING OF CODE

---- 26 CSEG

27

0000 28 ORG 0000h

29

0000 020100 30 JMP MAIN

31

32 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

33 ; INTERRUPT VECTOR SPACE

34

0033 35 ORG 0033h ; (ADC ISR)

36

0033 B2B4 37 CPL LED

38

0035 900200 39 MOV DPTR,#SEPERATOR ; send linefeed+CR out UART

0038 120165 40 CALL SENDSTRING

41

003B E5DB 42 MOV A,ADC0H ; send ADC data via UART

003D 120185 43 CALL SENDVAL

0040 E5DA 44 MOV A,ADC0M

0042 120185 45 CALL SENDVAL

46

47

0045 C2DF 48 CLR RDY0

0047 32 49 RETI

50

51 ;====================================================================

52 ; MAIN PROGRAM

0100 53 ORG 0100h

54

0100 55 MAIN:

0100 75817F 56 MOV SP,#127

57

58 ; CONFIGURE UART....

836UART2 PAGE 2

59

0103 759E82 60 MOV T3CON,#82h

0106 759D12 61 MOV T3FD,#12h

0109 759852 62 MOV SCON,#52h

63

64 ; CONFIGURE ADC....

65

010C 75D4C8 66 MOV SF,#200 ; 6.8266667Hz ADC data rate

010F 75D245 67 MOV ADC0CON,#045h ; externalVref, bipolar, ±640mV

0112 D2AE 68 SETB EADC ; enable ADC interrupt (trig on RDY0)

69

70 ; START CONVERTING & WAIT FOR INTERRUPTS OR INCOMING UART COMS....

71

0114 D2AF 72 WAIT: SETB EA

0116 C2DF 73 CLR RDY0

0118 75D123 74 MOV ADCMODE,#023h ; continuous conversion mode

011B C298 75 CLR RI

011D 3098FD 76 JNB RI,$ ; wait here for UART command..

77 ; ..or ADC interrupt

78

79 ; WHEN UART COMMAND RECEIVED....

80

0120 C2AF 81 CLR EA ; disable interrupts

0122 75D121 82 MOV ADCMODE,#021h ; put ADC in idle mode

0125 E599 83 MOV A,SBUF

0127 B43005 84 CJNE A,#'0',NEXT01 ; if "0" received..

012A 120137 85 CALL CALZERO ; ..perform zero calibration

012D 80E5 86 JMP WAIT

012F B431E2 87 NEXT01: CJNE A,#'1',WAIT ; if "1" received..

0132 12014E 88 CALL CALGAIN ; ..perform gain calibration

0135 80DD 89 JMP WAIT

90

91 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

92 ; SUBROUTINES

93

0137 94 CALZERO: ; perform zero-scale ADC calibration

95

0137 900203 96 MOV DPTR,#CALZSMSG ; send char to indicate..

013A 120165 97 CALL SENDSTRING ; ..begin zero calibration

98

013D C2DF 99 CLR RDY0

013F 75D122 100 MOV ADCMODE,#034 ; zero-scale self cal

101 ; MOV ADCMODE,#036h ; zero-scale system cal

0142 30DFFD 102 JNB RDY0,$

0145 C2DF 103 CLR RDY0

104

0147 900207 105 MOV DPTR,#CALDONEMSG ; send char to indicate..

014A 120165 106 CALL SENDSTRING ; ..calibraion complete

107

014D 22 108 RET

109

110 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

111

014E 112 CALGAIN: ; perform full-scale ADC system calibration

113

014E 900205 114 MOV DPTR,#CALGNMSG ; send char to indicate..

0151 120165 115 CALL SENDSTRING ; ..begin gain calibration

116

836UART2 PAGE 3

0154 C2DF 117 CLR RDY0

0156 75D123 118 MOV ADCMODE,#035 ; full-scale self cal

119 ; MOV ADCMODE,#037h ; full-scale system cal

0159 30DFFD 120 JNB RDY0,$

015C C2DF 121 CLR RDY0

122

015E 900207 123 MOV DPTR,#CALDONEMSG ; send char to indicate..

0161 120165 124 CALL SENDSTRING ; ..calibration complete

125

0164 22 126 RET

127

128 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

129 ; SUBROUTINE INCLUDE FILE

130

=1 131 $INCLUDE(UARTIO.asm)

=1 132 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 133 ;

=1 134 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 135 ;

=1 136 ; Date : 12 October 1999

=1 137 ;

=1 138 ; File : UARTIO.hex

=1 139 ;

=1 140 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 141 ;

=1 142 ; Description : standard UART I/O subroutines. total size of this

=1 143 ; code when assembled is 155 bytes. routines for use

=1 144 ; external to this file are:

=1 145 ;

=1 146 ; SENDSTRING - sends a string of characters

=1 147 ; SENDCHAR - sends a single character

=1 148 ; SENDVAL - sends a byte as 2 ASCII characters

=1 149 ; HEX2ASCII - converts from HEX to ASCII

=1 150 ; ASCII2HEX - converts from ASCII to HEX

=1 151 ; GETCHAR - gets a single character

=1 152 ; GETVAL - gets a byte as 2 ASCII characters

=1 153 ;

=1 154 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 155

=1 156 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 157 ; SENDSTRING

=1 158

0165 =1 159 SENDSTRING: ; sends ASCII string to UART starting at location

=1 160 ; DPTR and ending with a null (0) value

=1 161

0165 C0E0 =1 162 PUSH ACC

0167 C0F0 =1 163 PUSH B

0169 E4 =1 164 CLR A

016A F5F0 =1 165 MOV B,A

016C E5F0 =1 166 IO0010: MOV A,B

016E 05F0 =1 167 INC B

0170 93 =1 168 MOVC A,@A+DPTR

0171 6005 =1 169 JZ IO0020

0173 12017D =1 170 CALL SENDCHAR

0176 80F4 =1 171 JMP IO0010

0178 D0F0 =1 172 IO0020: POP B

017A D0E0 =1 173 POP ACC

=1 174

836UART2 PAGE 4

017C 22 =1 175 RET

=1 176

=1 177 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 178 ; SENDCHAR

=1 179

017D =1 180 SENDCHAR: ; sends ASCII value contained in A to UART

=1 181

017D 3099FD =1 182 JNB TI,$ ; wait til present char gone

0180 C299 =1 183 CLR TI ; must clear TI

0182 F599 =1 184 MOV SBUF,A

=1 185

0184 22 =1 186 RET

=1 187

=1 188 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 189 ; SENDVAL

=1 190

0185 =1 191 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 192 ; and then spits these two characters up the UART.

=1 193 ; does not change the value of A.

=1 194

0185 C0E0 =1 195 PUSH ACC

0187 C4 =1 196 SWAP A

0188 120199 =1 197 CALL HEX2ASCII

018B 317D =1 198 CALL SENDCHAR ; send high nibble

018D D0E0 =1 199 POP ACC

018F C0E0 =1 200 PUSH ACC

0191 120199 =1 201 CALL HEX2ASCII

0194 317D =1 202 CALL SENDCHAR ; send low nibble

0196 D0E0 =1 203 POP ACC

=1 204

0198 22 =1 205 RET

=1 206

=1 207 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 208 ; HEX2ASCII

=1 209

0199 =1 210 HEX2ASCII: ; converts A into the hex character representing the

=1 211 ; value of A's least significant nibble

=1 212

0199 540F =1 213 ANL A,#00Fh

019B B40A00 =1 214 CJNE A,#00Ah,$+3

019E 4002 =1 215 JC IO0030

01A0 2407 =1 216 ADD A,#007h

01A2 2430 =1 217 IO0030: ADD A,#'0'

=1 218

01A4 22 =1 219 RET

=1 220

=1 221 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 222 ; ASCII2HEX

=1 223

01A5 =1 224 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 225 ; into the corresponding number (0-15). returns C=1

=1 226 ; when input is other than an ASCII digit,

=1 227 ; indicating invalid output (returned as 255).

=1 228

01A5 C3 =1 229 CLR C

01A6 9430 =1 230 SUBB A,#'0'

01A8 B40A00 =1 231 CJNE A,#10,$+3

01AB 401B =1 232 JC IO0050 ; if '0'<=char<='9', return OK

836UART2 PAGE 5

01AD B41100 =1 233 CJNE A,#17,$+3

01B0 4013 =1 234 JC IO0040 ; if '9'<char<'A', return FAIL

01B2 9407 =1 235 SUBB A,#7

01B4 B41000 =1 236 CJNE A,#10h,$+3

01B7 400F =1 237 JC IO0050 ; if 'A'<=char<='F', return OK

01B9 B42A00 =1 238 CJNE A,#42,$+3

01BC 4007 =1 239 JC IO0040 ; if 'F'<char<'a', return FAIL

01BE 9420 =1 240 SUBB A,#20h

01C0 B41000 =1 241 CJNE A,#10h,$+3

01C3 4003 =1 242 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 243

01C5 C3 =1 244 IO0040: CLR C ; ..else return FAIL

01C6 74FF =1 245 MOV A,#0FFh

=1 246

01C8 B3 =1 247 IO0050: CPL C

01C9 22 =1 248 RET

=1 249

=1 250 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 251 ; GETCHAR

=1 252

01CA =1 253 GETCHAR: ; waits for a single ASCII character to be received

=1 254 ; by the UART. places this character into A.

=1 255

01CA 3098FD =1 256 JNB RI,$

01CD E599 =1 257 MOV A,SBUF

01CF C298 =1 258 CLR RI

=1 259

01D1 22 =1 260 RET

=1 261

=1 262 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 263 ; GETVAL

=1 264

01D2 =1 265 GETVAL: ; waits for two ASCII hex digits to be received by

=1 266 ; the UART. returns the hex value in A.

=1 267

01D2 C0F0 =1 268 PUSH B

01D4 C000 =1 269 PUSH 0

01D6 C298 =1 270 IO0060: CLR RI

01D8 31CA =1 271 CALL GETCHAR ; first nibble

01DA F500 =1 272 MOV 0,A ; store received char

01DC 31A5 =1 273 CALL ASCII2HEX

01DE 40F6 =1 274 JC IO0060 ; if not '0' thru 'F', don't accept

01E0 C4 =1 275 SWAP A ; swap nibbles

01E1 F5F0 =1 276 MOV B,A ; store nibble in B

01E3 E500 =1 277 MOV A,0 ; echo received char

01E5 317D =1 278 CALL SENDCHAR

01E7 C298 =1 279 IO0070: CLR RI

01E9 31CA =1 280 CALL GETCHAR ; second nibble

01EB F500 =1 281 MOV 0,A ; store received char

01ED 31A5 =1 282 CALL ASCII2HEX

01EF 40F6 =1 283 JC IO0070 ; if not '0' thru 'F', don't accept

01F1 45F0 =1 284 ORL A,B ; combine nibbles

01F3 F5F0 =1 285 MOV B,A ; store results in B

01F5 E500 =1 286 MOV A,0 ; echo received char

01F7 317D =1 287 CALL SENDCHAR

01F9 E5F0 =1 288 MOV A,B ; final result

01FB D000 =1 289 POP 0

01FD D0F0 =1 290 POP B

836UART2 PAGE 6

=1 291

01FF 22 =1 292 RET

=1 293

294

295 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

296 ; TEXT DATA TABLES

297

0200 0A0D00 298 SEPERATOR: DB 10,13,0

0203 5A00 299 CALZSMSG: DB 'Z',0

0205 4700 300 CALGNMSG: DB 'G',0

0207 7800 301 CALDONEMSG: DB 'x',0

302

303 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

304

305 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

836UART2 PAGE 7

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

ADC0CON. . . . . . . . . . . . . D ADDR 00D2H PREDEFINED

ADC0H. . . . . . . . . . . . . . D ADDR 00DBH PREDEFINED

ADC0M. . . . . . . . . . . . . . D ADDR 00DAH PREDEFINED

ADCMODE. . . . . . . . . . . . . D ADDR 00D1H PREDEFINED

ASCII2HEX. . . . . . . . . . . . C ADDR 01A5H

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

CALDONEMSG . . . . . . . . . . . C ADDR 0207H

CALGAIN. . . . . . . . . . . . . C ADDR 014EH

CALGNMSG . . . . . . . . . . . . C ADDR 0205H

CALZERO. . . . . . . . . . . . . C ADDR 0137H

CALZSMSG . . . . . . . . . . . . C ADDR 0203H

EA . . . . . . . . . . . . . . . B ADDR 00AFH PREDEFINED

EADC . . . . . . . . . . . . . . B ADDR 00AEH PREDEFINED

GETCHAR. . . . . . . . . . . . . C ADDR 01CAH

GETVAL . . . . . . . . . . . . . C ADDR 01D2H NOT USED

HEX2ASCII. . . . . . . . . . . . C ADDR 0199H

IO0010 . . . . . . . . . . . . . C ADDR 016CH

IO0020 . . . . . . . . . . . . . C ADDR 0178H

IO0030 . . . . . . . . . . . . . C ADDR 01A2H

IO0040 . . . . . . . . . . . . . C ADDR 01C5H

IO0050 . . . . . . . . . . . . . C ADDR 01C8H

IO0060 . . . . . . . . . . . . . C ADDR 01D6H

IO0070 . . . . . . . . . . . . . C ADDR 01E7H

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

MAIN . . . . . . . . . . . . . . C ADDR 0100H

NEXT01 . . . . . . . . . . . . . C ADDR 012FH

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

RDY0 . . . . . . . . . . . . . . B ADDR 00DFH PREDEFINED

RI . . . . . . . . . . . . . . . B ADDR 0098H PREDEFINED

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

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

SENDCHAR . . . . . . . . . . . . C ADDR 017DH

SENDSTRING . . . . . . . . . . . C ADDR 0165H

SENDVAL. . . . . . . . . . . . . C ADDR 0185H

SEPERATOR. . . . . . . . . . . . C ADDR 0200H

SF . . . . . . . . . . . . . . . D ADDR 00D4H PREDEFINED

SP . . . . . . . . . . . . . . . D ADDR 0081H PREDEFINED

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

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

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

WAIT . . . . . . . . . . . . . . C ADDR 0114H