834UART2 PAGE 1

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

2 ;

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

4 ;

5 ; Date : 5 November 2001

6 ;

7 ; File : 834uart2.asm

8 ;

9 ; Hardware : ADuC834

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 $MOD834 ; Use 8052&ADuC834 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

0045 E5D9 46 MOV A,ADC0L

0047 120185 47 CALL SENDVAL

48

004A C2DF 49 CLR RDY0

004C 32 50 RETI

51

52 ;====================================================================

53 ; MAIN PROGRAM

0100 54 ORG 0100h

55

0100 56 MAIN:

0100 75817F 57 MOV SP,#127

58

834UART2 PAGE 2

59 ; CONFIGURE UART....

60

0103 759E82 61 MOV T3CON,#82h

0106 759D12 62 MOV T3FD,#12h

0109 759852 63 MOV SCON,#52h

64

65 ; CONFIGURE ADC....

66

010C 75D4C8 67 MOV SF,#200 ; 6.8266667Hz ADC data rate

010F 75D245 68 MOV ADC0CON,#045h ; externalVref, bipolar, ±640mV

0112 D2AE 69 SETB EADC ; enable ADC interrupt (trig on RDY0)

70

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

72

0114 D2AF 73 WAIT: SETB EA

0116 C2DF 74 CLR RDY0

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

011B C298 76 CLR RI

011D 3098FD 77 JNB RI,$ ; wait here for UART command..

78 ; ..or ADC interrupt

79

80 ; WHEN UART COMMAND RECEIVED....

81

0120 C2AF 82 CLR EA ; disable interrupts

0122 75D121 83 MOV ADCMODE,#021h ; put ADC in idle mode

0125 E599 84 MOV A,SBUF

0127 B43005 85 CJNE A,#'0',NEXT01 ; if "0" received..

012A 120137 86 CALL CALZERO ; ..perform zero calibration

012D 80E5 87 JMP WAIT

012F B431E2 88 NEXT01: CJNE A,#'1',WAIT ; if "1" received..

0132 12014E 89 CALL CALGAIN ; ..perform gain calibration

0135 80DD 90 JMP WAIT

91

92 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

93 ; SUBROUTINES

94

0137 95 CALZERO: ; perform zero-scale ADC calibration

96

0137 900203 97 MOV DPTR,#CALZSMSG ; send char to indicate..

013A 120165 98 CALL SENDSTRING ; ..begin zero calibration

99

013D C2DF 100 CLR RDY0

013F 75D122 101 MOV ADCMODE,#034 ; zero-scale self cal

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

0142 30DFFD 103 JNB RDY0,$

0145 C2DF 104 CLR RDY0

105

0147 900207 106 MOV DPTR,#CALDONEMSG ; send char to indicate..

014A 120165 107 CALL SENDSTRING ; ..calibraion complete

108

014D 22 109 RET

110

111 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

112

014E 113 CALGAIN: ; perform full-scale ADC system calibration

114

014E 900205 115 MOV DPTR,#CALGNMSG ; send char to indicate..

0151 120165 116 CALL SENDSTRING ; ..begin gain calibration

834UART2 PAGE 3

117

0154 C2DF 118 CLR RDY0

0156 75D123 119 MOV ADCMODE,#035 ; full-scale self cal

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

0159 30DFFD 121 JNB RDY0,$

015C C2DF 122 CLR RDY0

123

015E 900207 124 MOV DPTR,#CALDONEMSG ; send char to indicate..

0161 120165 125 CALL SENDSTRING ; ..calibration complete

126

0164 22 127 RET

128

129 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

130 ; SUBROUTINE INCLUDE FILE

131

=1 132 $INCLUDE(UARTIO.asm)

=1 133 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 134 ;

=1 135 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 136 ;

=1 137 ; Date : 12 October 1999

=1 138 ;

=1 139 ; File : UARTIO.hex

=1 140 ;

=1 141 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 142 ;

=1 143 ; Description : standard UART I/O subroutines. total size of this

=1 144 ; code when assembled is 155 bytes. routines for use

=1 145 ; external to this file are:

=1 146 ;

=1 147 ; SENDSTRING - sends a string of characters

=1 148 ; SENDCHAR - sends a single character

=1 149 ; SENDVAL - sends a byte as 2 ASCII characters

=1 150 ; HEX2ASCII - converts from HEX to ASCII

=1 151 ; ASCII2HEX - converts from ASCII to HEX

=1 152 ; GETCHAR - gets a single character

=1 153 ; GETVAL - gets a byte as 2 ASCII characters

=1 154 ;

=1 155 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 156

=1 157 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 158 ; SENDSTRING

=1 159

0165 =1 160 SENDSTRING: ; sends ASCII string to UART starting at location

=1 161 ; DPTR and ending with a null (0) value

=1 162

0165 C0E0 =1 163 PUSH ACC

0167 C0F0 =1 164 PUSH B

0169 E4 =1 165 CLR A

016A F5F0 =1 166 MOV B,A

016C E5F0 =1 167 IO0010: MOV A,B

016E 05F0 =1 168 INC B

0170 93 =1 169 MOVC A,@A+DPTR

0171 6005 =1 170 JZ IO0020

0173 12017D =1 171 CALL SENDCHAR

0176 80F4 =1 172 JMP IO0010

0178 D0F0 =1 173 IO0020: POP B

017A D0E0 =1 174 POP ACC

834UART2 PAGE 4

=1 175

017C 22 =1 176 RET

=1 177

=1 178 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 179 ; SENDCHAR

=1 180

017D =1 181 SENDCHAR: ; sends ASCII value contained in A to UART

=1 182

017D 3099FD =1 183 JNB TI,$ ; wait til present char gone

0180 C299 =1 184 CLR TI ; must clear TI

0182 F599 =1 185 MOV SBUF,A

=1 186

0184 22 =1 187 RET

=1 188

=1 189 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 190 ; SENDVAL

=1 191

0185 =1 192 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 193 ; and then spits these two characters up the UART.

=1 194 ; does not change the value of A.

=1 195

0185 C0E0 =1 196 PUSH ACC

0187 C4 =1 197 SWAP A

0188 120199 =1 198 CALL HEX2ASCII

018B 317D =1 199 CALL SENDCHAR ; send high nibble

018D D0E0 =1 200 POP ACC

018F C0E0 =1 201 PUSH ACC

0191 120199 =1 202 CALL HEX2ASCII

0194 317D =1 203 CALL SENDCHAR ; send low nibble

0196 D0E0 =1 204 POP ACC

=1 205

0198 22 =1 206 RET

=1 207

=1 208 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 209 ; HEX2ASCII

=1 210

0199 =1 211 HEX2ASCII: ; converts A into the hex character representing the

=1 212 ; value of A's least significant nibble

=1 213

0199 540F =1 214 ANL A,#00Fh

019B B40A00 =1 215 CJNE A,#00Ah,$+3

019E 4002 =1 216 JC IO0030

01A0 2407 =1 217 ADD A,#007h

01A2 2430 =1 218 IO0030: ADD A,#'0'

=1 219

01A4 22 =1 220 RET

=1 221

=1 222 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 223 ; ASCII2HEX

=1 224

01A5 =1 225 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 226 ; into the corresponding number (0-15). returns C=1

=1 227 ; when input is other than an ASCII digit,

=1 228 ; indicating invalid output (returned as 255).

=1 229

01A5 C3 =1 230 CLR C

01A6 9430 =1 231 SUBB A,#'0'

01A8 B40A00 =1 232 CJNE A,#10,$+3

834UART2 PAGE 5

01AB 401B =1 233 JC IO0050 ; if '0'<=char<='9', return OK

01AD B41100 =1 234 CJNE A,#17,$+3

01B0 4013 =1 235 JC IO0040 ; if '9'<char<'A', return FAIL

01B2 9407 =1 236 SUBB A,#7

01B4 B41000 =1 237 CJNE A,#10h,$+3

01B7 400F =1 238 JC IO0050 ; if 'A'<=char<='F', return OK

01B9 B42A00 =1 239 CJNE A,#42,$+3

01BC 4007 =1 240 JC IO0040 ; if 'F'<char<'a', return FAIL

01BE 9420 =1 241 SUBB A,#20h

01C0 B41000 =1 242 CJNE A,#10h,$+3

01C3 4003 =1 243 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 244

01C5 C3 =1 245 IO0040: CLR C ; ..else return FAIL

01C6 74FF =1 246 MOV A,#0FFh

=1 247

01C8 B3 =1 248 IO0050: CPL C

01C9 22 =1 249 RET

=1 250

=1 251 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 252 ; GETCHAR

=1 253

01CA =1 254 GETCHAR: ; waits for a single ASCII character to be received

=1 255 ; by the UART. places this character into A.

=1 256

01CA 3098FD =1 257 JNB RI,$

01CD E599 =1 258 MOV A,SBUF

01CF C298 =1 259 CLR RI

=1 260

01D1 22 =1 261 RET

=1 262

=1 263 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 264 ; GETVAL

=1 265

01D2 =1 266 GETVAL: ; waits for two ASCII hex digits to be received by

=1 267 ; the UART. returns the hex value in A.

=1 268

01D2 C0F0 =1 269 PUSH B

01D4 C000 =1 270 PUSH 0

01D6 C298 =1 271 IO0060: CLR RI

01D8 31CA =1 272 CALL GETCHAR ; first nibble

01DA F500 =1 273 MOV 0,A ; store received char

01DC 31A5 =1 274 CALL ASCII2HEX

01DE 40F6 =1 275 JC IO0060 ; if not '0' thru 'F', don't accept

01E0 C4 =1 276 SWAP A ; swap nibbles

01E1 F5F0 =1 277 MOV B,A ; store nibble in B

01E3 E500 =1 278 MOV A,0 ; echo received char

01E5 317D =1 279 CALL SENDCHAR

01E7 C298 =1 280 IO0070: CLR RI

01E9 31CA =1 281 CALL GETCHAR ; second nibble

01EB F500 =1 282 MOV 0,A ; store received char

01ED 31A5 =1 283 CALL ASCII2HEX

01EF 40F6 =1 284 JC IO0070 ; if not '0' thru 'F', don't accept

01F1 45F0 =1 285 ORL A,B ; combine nibbles

01F3 F5F0 =1 286 MOV B,A ; store results in B

01F5 E500 =1 287 MOV A,0 ; echo received char

01F7 317D =1 288 CALL SENDCHAR

01F9 E5F0 =1 289 MOV A,B ; final result

01FB D000 =1 290 POP 0

834UART2 PAGE 6

01FD D0F0 =1 291 POP B

=1 292

01FF 22 =1 293 RET

=1 294

295

296 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

297 ; TEXT DATA TABLES

298

0200 0A0D00 299 SEPERATOR: DB 10,13,0

0203 5A00 300 CALZSMSG: DB 'Z',0

0205 4700 301 CALGNMSG: DB 'G',0

0207 7800 302 CALDONEMSG: DB 'x',0

303

304 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

305

306 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

834UART2 PAGE 7

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

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

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

ADC0L. . . . . . . . . . . . . . D ADDR 00D9H 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