CALUART PAGE 1

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

2 ;

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

4 ;

5 ; Date : January 2001

6 ;

7 ; File : Caluart.asm

8 ;

9 ; Hardware : ADuC816

10 ;

11 ; Description : sample program that performs ADC conversions in

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

13 ; UART.

14 ; Program also accepts commands from the PC

15 ; via incoming characters on the UART. an ASCII '0'

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

17 ; '1' innitiates a full-scale calibration.

18 ;

19 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

20

21 $MOD816 ; Use 8052&ADuC816 predefined symbols

22

00B4 23 LED EQU P3.4 ; P3.4 drives red LED on eval board

24

25 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

26 ; BEGINNING OF CODE

---- 27 CSEG

28

0000 29 ORG 0000h

30

0000 020100 31 JMP MAIN

32

33 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

34 ; INTERRUPT VECTOR SPACE

35

0033 36 ORG 0033h ; (ADC ISR)

37

0033 B2B4 38 CPL LED

39

0035 900209 40 MOV DPTR,#SEPERATOR ; send linefeed+CR out UART

0038 12016E 41 CALL SENDSTRING

42

003B E5DB 43 MOV A,ADC0H ; send ADC data via UART

003D 12018E 44 CALL SENDVAL

0040 E5DA 45 MOV A,ADC0M

0042 12018E 46 CALL SENDVAL

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....

CALUART PAGE 2

59

0103 75CBFF 60 MOV RCAP2H,#0FFh ; config UART for 9830baud

0106 75CAFB 61 MOV RCAP2L,#-5 ; (close enough to 9600baud)

0109 75CDFF 62 MOV TH2,#0FFh

010C 75CCFB 63 MOV TL2,#-5

010F 759852 64 MOV SCON,#01010010b

0112 75C834 65 MOV T2CON,#00110100b

66

67 ; CONFIGURE ADC....

68

0115 75D4C8 69 MOV SF,#200 ; 6.8266667Hz ADC data rate

0118 75D245 70 MOV ADC0CON,#045h ; externalVref, bipolar, ±640mV

011B D2AE 71 SETB EADC ; enable ADC interrupt (trig on RDY0)

72

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

74

011D D2AF 75 WAIT: SETB EA

011F C2DF 76 CLR RDY0

0121 75D123 77 MOV ADCMODE,#023h ; continuous conversion mode

0124 C298 78 CLR RI

0126 3098FD 79 JNB RI,$ ; wait here for UART command..

80 ; ..or ADC interrupt

81

82 ; WHEN UART COMMAND RECEIVED....

83

0129 C2AF 84 CLR EA ; disable interrupts

012B 75D121 85 MOV ADCMODE,#021h ; put ADC in idle mode

012E E599 86 MOV A,SBUF

0130 B43005 87 CJNE A,#'0',NEXT01 ; if "0" received..

0133 120140 88 CALL CALZERO ; ..perform zero calibration

0136 80E5 89 JMP WAIT

0138 B431E2 90 NEXT01: CJNE A,#'1',WAIT ; if "1" received..

013B 120157 91 CALL CALGAIN ; ..perform gain calibration

013E 80DD 92 JMP WAIT

93

94 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

95 ; SUBROUTINES

96

0140 97 CALZERO: ; perform zero-scale ADC calibration

98

0140 90020C 99 MOV DPTR,#CALZSMSG ; send char to indicate..

0143 12016E 100 CALL SENDSTRING ; ..begin zero calibration

101

0146 C2DF 102 CLR RDY0

0148 75D122 103 MOV ADCMODE,#034 ; zero-scale self cal

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

014B 30DFFD 105 JNB RDY0,$

014E C2DF 106 CLR RDY0

107

0150 900210 108 MOV DPTR,#CALDONEMSG ; send char to indicate..

0153 12016E 109 CALL SENDSTRING ; ..calibraion complete

110

0156 22 111 RET

112

113 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

114

0157 115 CALGAIN: ; perform full-scale ADC system calibration

116

CALUART PAGE 3

0157 90020E 117 MOV DPTR,#CALGNMSG ; send char to indicate..

015A 12016E 118 CALL SENDSTRING ; ..begin gain calibration

119

015D C2DF 120 CLR RDY0

015F 75D123 121 MOV ADCMODE,#035 ; full-scale self cal

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

0162 30DFFD 123 JNB RDY0,$

0165 C2DF 124 CLR RDY0

125

0167 900210 126 MOV DPTR,#CALDONEMSG ; send char to indicate..

016A 12016E 127 CALL SENDSTRING ; ..calibration complete

128

016D 22 129 RET

130

131 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

132 ; SUBROUTINE INCLUDE FILE

133

=1 134 $INCLUDE(UARTIO.asm)

=1 135 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 136 ;

=1 137 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 138 ;

=1 139 ; Date : 12 October 1999

=1 140 ;

=1 141 ; File : UARTIO.hex

=1 142 ;

=1 143 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 144 ;

=1 145 ; Description : standard UART I/O subroutines. total size of this

=1 146 ; code when assembled is 155 bytes. routines for use

=1 147 ; external to this file are:

=1 148 ;

=1 149 ; SENDSTRING - sends a string of characters

=1 150 ; SENDCHAR - sends a single character

=1 151 ; SENDVAL - sends a byte as 2 ASCII characters

=1 152 ; HEX2ASCII - converts from HEX to ASCII

=1 153 ; ASCII2HEX - converts from ASCII to HEX

=1 154 ; GETCHAR - gets a single character

=1 155 ; GETVAL - gets a byte as 2 ASCII characters

=1 156 ;

=1 157 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 158

=1 159 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 160 ; SENDSTRING

=1 161

016E =1 162 SENDSTRING: ; sends ASCII string to UART starting at location

=1 163 ; DPTR and ending with a null (0) value

=1 164

016E C0E0 =1 165 PUSH ACC

0170 C0F0 =1 166 PUSH B

0172 E4 =1 167 CLR A

0173 F5F0 =1 168 MOV B,A

0175 E5F0 =1 169 IO0010: MOV A,B

0177 05F0 =1 170 INC B

0179 93 =1 171 MOVC A,@A+DPTR

017A 6005 =1 172 JZ IO0020

017C 120186 =1 173 CALL SENDCHAR

017F 80F4 =1 174 JMP IO0010

CALUART PAGE 4

0181 D0F0 =1 175 IO0020: POP B

0183 D0E0 =1 176 POP ACC

=1 177

0185 22 =1 178 RET

=1 179

=1 180 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 181 ; SENDCHAR

=1 182

0186 =1 183 SENDCHAR: ; sends ASCII value contained in A to UART

=1 184

0186 3099FD =1 185 JNB TI,$ ; wait til present char gone

0189 C299 =1 186 CLR TI ; must clear TI

018B F599 =1 187 MOV SBUF,A

=1 188

018D 22 =1 189 RET

=1 190

=1 191 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 192 ; SENDVAL

=1 193

018E =1 194 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 195 ; and then spits these two characters up the UART.

=1 196 ; does not change the value of A.

=1 197

018E C0E0 =1 198 PUSH ACC

0190 C4 =1 199 SWAP A

0191 1201A2 =1 200 CALL HEX2ASCII

0194 3186 =1 201 CALL SENDCHAR ; send high nibble

0196 D0E0 =1 202 POP ACC

0198 C0E0 =1 203 PUSH ACC

019A 1201A2 =1 204 CALL HEX2ASCII

019D 3186 =1 205 CALL SENDCHAR ; send low nibble

019F D0E0 =1 206 POP ACC

=1 207

01A1 22 =1 208 RET

=1 209

=1 210 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 211 ; HEX2ASCII

=1 212

01A2 =1 213 HEX2ASCII: ; converts A into the hex character representing the

=1 214 ; value of A's least significant nibble

=1 215

01A2 540F =1 216 ANL A,#00Fh

01A4 B40A00 =1 217 CJNE A,#00Ah,$+3

01A7 4002 =1 218 JC IO0030

01A9 2407 =1 219 ADD A,#007h

01AB 2430 =1 220 IO0030: ADD A,#'0'

=1 221

01AD 22 =1 222 RET

=1 223

=1 224 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 225 ; ASCII2HEX

=1 226

01AE =1 227 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 228 ; into the corresponding number (0-15). returns C=1

=1 229 ; when input is other than an ASCII digit,

=1 230 ; indicating invalid output (returned as 255).

=1 231

01AE C3 =1 232 CLR C

CALUART PAGE 5

01AF 9430 =1 233 SUBB A,#'0'

01B1 B40A00 =1 234 CJNE A,#10,$+3

01B4 401B =1 235 JC IO0050 ; if '0'<=char<='9', return OK

01B6 B41100 =1 236 CJNE A,#17,$+3

01B9 4013 =1 237 JC IO0040 ; if '9'<char<'A', return FAIL

01BB 9407 =1 238 SUBB A,#7

01BD B41000 =1 239 CJNE A,#10h,$+3

01C0 400F =1 240 JC IO0050 ; if 'A'<=char<='F', return OK

01C2 B42A00 =1 241 CJNE A,#42,$+3

01C5 4007 =1 242 JC IO0040 ; if 'F'<char<'a', return FAIL

01C7 9420 =1 243 SUBB A,#20h

01C9 B41000 =1 244 CJNE A,#10h,$+3

01CC 4003 =1 245 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 246

01CE C3 =1 247 IO0040: CLR C ; ..else return FAIL

01CF 74FF =1 248 MOV A,#0FFh

=1 249

01D1 B3 =1 250 IO0050: CPL C

01D2 22 =1 251 RET

=1 252

=1 253 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 254 ; GETCHAR

=1 255

01D3 =1 256 GETCHAR: ; waits for a single ASCII character to be received

=1 257 ; by the UART. places this character into A.

=1 258

01D3 3098FD =1 259 JNB RI,$

01D6 E599 =1 260 MOV A,SBUF

01D8 C298 =1 261 CLR RI

=1 262

01DA 22 =1 263 RET

=1 264

=1 265 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 266 ; GETVAL

=1 267

01DB =1 268 GETVAL: ; waits for two ASCII hex digits to be received by

=1 269 ; the UART. returns the hex value in A.

=1 270

01DB C0F0 =1 271 PUSH B

01DD C000 =1 272 PUSH 0

01DF C298 =1 273 IO0060: CLR RI

01E1 31D3 =1 274 CALL GETCHAR ; first nibble

01E3 F500 =1 275 MOV 0,A ; store received char

01E5 31AE =1 276 CALL ASCII2HEX

01E7 40F6 =1 277 JC IO0060 ; if not '0' thru 'F', don't accept

01E9 C4 =1 278 SWAP A ; swap nibbles

01EA F5F0 =1 279 MOV B,A ; store nibble in B

01EC E500 =1 280 MOV A,0 ; echo received char

01EE 3186 =1 281 CALL SENDCHAR

01F0 C298 =1 282 IO0070: CLR RI

01F2 31D3 =1 283 CALL GETCHAR ; second nibble

01F4 F500 =1 284 MOV 0,A ; store received char

01F6 31AE =1 285 CALL ASCII2HEX

01F8 40F6 =1 286 JC IO0070 ; if not '0' thru 'F', don't accept

01FA 45F0 =1 287 ORL A,B ; combine nibbles

01FC F5F0 =1 288 MOV B,A ; store results in B

01FE E500 =1 289 MOV A,0 ; echo received char

0200 3186 =1 290 CALL SENDCHAR

CALUART PAGE 6

0202 E5F0 =1 291 MOV A,B ; final result

0204 D000 =1 292 POP 0

0206 D0F0 =1 293 POP B

=1 294

0208 22 =1 295 RET

=1 296

297

298 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

299 ; TEXT DATA TABLES

300

0209 0A0D00 301 SEPERATOR: DB 10,13,0

020C 5A00 302 CALZSMSG: DB 'Z',0

020E 4700 303 CALGNMSG: DB 'G',0

0210 7800 304 CALDONEMSG: DB 'x',0

305

306 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

307

308 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

CALUART 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 01AEH

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

CALDONEMSG . . . . . . . . . . . C ADDR 0210H

CALGAIN. . . . . . . . . . . . . C ADDR 0157H

CALGNMSG . . . . . . . . . . . . C ADDR 020EH

CALZERO. . . . . . . . . . . . . C ADDR 0140H

CALZSMSG . . . . . . . . . . . . C ADDR 020CH

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

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

GETCHAR. . . . . . . . . . . . . C ADDR 01D3H

GETVAL . . . . . . . . . . . . . C ADDR 01DBH NOT USED

HEX2ASCII. . . . . . . . . . . . C ADDR 01A2H

IO0010 . . . . . . . . . . . . . C ADDR 0175H

IO0020 . . . . . . . . . . . . . C ADDR 0181H

IO0030 . . . . . . . . . . . . . C ADDR 01ABH

IO0040 . . . . . . . . . . . . . C ADDR 01CEH

IO0050 . . . . . . . . . . . . . C ADDR 01D1H

IO0060 . . . . . . . . . . . . . C ADDR 01DFH

IO0070 . . . . . . . . . . . . . C ADDR 01F0H

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

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

NEXT01 . . . . . . . . . . . . . C ADDR 0138H

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

RCAP2H . . . . . . . . . . . . . D ADDR 00CBH PREDEFINED

RCAP2L . . . . . . . . . . . . . D ADDR 00CAH PREDEFINED

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

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

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

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

SENDCHAR . . . . . . . . . . . . C ADDR 0186H

SENDSTRING . . . . . . . . . . . C ADDR 016EH

SENDVAL. . . . . . . . . . . . . C ADDR 018EH

SEPERATOR. . . . . . . . . . . . C ADDR 0209H

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

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

T2CON. . . . . . . . . . . . . . D ADDR 00C8H PREDEFINED

TH2. . . . . . . . . . . . . . . D ADDR 00CDH PREDEFINED

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

TL2. . . . . . . . . . . . . . . D ADDR 00CCH PREDEFINED

WAIT . . . . . . . . . . . . . . C ADDR 011DH