ADCUART PAGE 1

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

2 ;

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

4 ;

5 ; Date : January 2001

6 ;

7 ; File : ADCuart.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 ;

15 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

16

17 $MOD816 ; Use 8052&ADuC816 predefined symbols

18

00B4 19 LED EQU P3.4 ; P3.4 drives red LED on eval board

20

21 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

22 ; BEGINNING OF CODE

---- 23 CSEG

24

0000 25 ORG 0000h

26

0000 020100 27 JMP MAIN

28

29 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

30 ; ADC ISR

31

0033 32 ORG 0033h ; (ADC ISR)

33

0033 B2B4 34 CPL LED

35

0035 9001BF 36 MOV DPTR,#SEPERATOR ; send linefeed+CR out UART

0038 120124 37 CALL SENDSTRING

38

003B E5DB 39 MOV A,ADC0H ; send ADC data via UART

003D 120144 40 CALL SENDVAL

0040 E5DA 41 MOV A,ADC0M

0042 120144 42 CALL SENDVAL

43

0045 C2DF 44 CLR RDY0

0047 32 45 RETI

46

47 ;====================================================================

48 ; MAIN PROGRAM

0100 49 ORG 0100h

50

0100 51 MAIN:

0100 75817F 52 MOV SP,#127

53

54 ; CONFIGURE UART....

55

0103 75CBFF 56 MOV RCAP2H,#0FFh ; config UART for 9830baud

0106 75CAFB 57 MOV RCAP2L,#-5 ; (close enough to 9600baud)

0109 75CDFF 58 MOV TH2,#0FFh

ADCUART PAGE 2

010C 75CCFB 59 MOV TL2,#-5

010F 759852 60 MOV SCON,#01010010b

0112 75C834 61 MOV T2CON,#00110100b

62

63 ; CONFIGURE ADC AND START CONVERTING....

64

0115 75D4C8 65 MOV SF,#200 ; 6.8266667Hz ADC data rate

0118 75D245 66 MOV ADC0CON,#45h ; externalVref, bipolar, ±640mV

011B D2AE 67 SETB EADC ; enable ADC interrupt (trig on RDY0)

011D D2AF 68 SETB EA ; enable global interrupts

011F 75D123 69 MOV ADCMODE,#23h ; continuous conversion mode

70

71 ; WAIT FOR INTERRUPTS....

72

0122 80FE 73 JMP $ ; endless loop

74

75 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

76 ; SUBROUTINE INCLUDE FILE

77

=1 78 $INCLUDE(UARTIO.asm)

=1 79 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 80 ;

=1 81 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 82 ;

=1 83 ; Date : 12 October 1999

=1 84 ;

=1 85 ; File : UARTIO.hex

=1 86 ;

=1 87 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 88 ;

=1 89 ; Description : standard UART I/O subroutines. total size of this

=1 90 ; code when assembled is 155 bytes. routines for use

=1 91 ; external to this file are:

=1 92 ;

=1 93 ; SENDSTRING - sends a string of characters

=1 94 ; SENDCHAR - sends a single character

=1 95 ; SENDVAL - sends a byte as 2 ASCII characters

=1 96 ; HEX2ASCII - converts from HEX to ASCII

=1 97 ; ASCII2HEX - converts from ASCII to HEX

=1 98 ; GETCHAR - gets a single character

=1 99 ; GETVAL - gets a byte as 2 ASCII characters

=1 100 ;

=1 101 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 102

=1 103 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 104 ; SENDSTRING

=1 105

0124 =1 106 SENDSTRING: ; sends ASCII string to UART starting at location

=1 107 ; DPTR and ending with a null (0) value

=1 108

0124 C0E0 =1 109 PUSH ACC

0126 C0F0 =1 110 PUSH B

0128 E4 =1 111 CLR A

0129 F5F0 =1 112 MOV B,A

012B E5F0 =1 113 IO0010: MOV A,B

012D 05F0 =1 114 INC B

012F 93 =1 115 MOVC A,@A+DPTR

0130 6005 =1 116 JZ IO0020

ADCUART PAGE 3

0132 12013C =1 117 CALL SENDCHAR

0135 80F4 =1 118 JMP IO0010

0137 D0F0 =1 119 IO0020: POP B

0139 D0E0 =1 120 POP ACC

=1 121

013B 22 =1 122 RET

=1 123

=1 124 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 125 ; SENDCHAR

=1 126

013C =1 127 SENDCHAR: ; sends ASCII value contained in A to UART

=1 128

013C 3099FD =1 129 JNB TI,$ ; wait til present char gone

013F C299 =1 130 CLR TI ; must clear TI

0141 F599 =1 131 MOV SBUF,A

=1 132

0143 22 =1 133 RET

=1 134

=1 135 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 136 ; SENDVAL

=1 137

0144 =1 138 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 139 ; and then spits these two characters up the UART.

=1 140 ; does not change the value of A.

=1 141

0144 C0E0 =1 142 PUSH ACC

0146 C4 =1 143 SWAP A

0147 120158 =1 144 CALL HEX2ASCII

014A 313C =1 145 CALL SENDCHAR ; send high nibble

014C D0E0 =1 146 POP ACC

014E C0E0 =1 147 PUSH ACC

0150 120158 =1 148 CALL HEX2ASCII

0153 313C =1 149 CALL SENDCHAR ; send low nibble

0155 D0E0 =1 150 POP ACC

=1 151

0157 22 =1 152 RET

=1 153

=1 154 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 155 ; HEX2ASCII

=1 156

0158 =1 157 HEX2ASCII: ; converts A into the hex character representing the

=1 158 ; value of A's least significant nibble

=1 159

0158 540F =1 160 ANL A,#00Fh

015A B40A00 =1 161 CJNE A,#00Ah,$+3

015D 4002 =1 162 JC IO0030

015F 2407 =1 163 ADD A,#007h

0161 2430 =1 164 IO0030: ADD A,#'0'

=1 165

0163 22 =1 166 RET

=1 167

=1 168 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 169 ; ASCII2HEX

=1 170

0164 =1 171 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 172 ; into the corresponding number (0-15). returns C=1

=1 173 ; when input is other than an ASCII digit,

=1 174 ; indicating invalid output (returned as 255).

ADCUART PAGE 4

=1 175

0164 C3 =1 176 CLR C

0165 9430 =1 177 SUBB A,#'0'

0167 B40A00 =1 178 CJNE A,#10,$+3

016A 401B =1 179 JC IO0050 ; if '0'<=char<='9', return OK

016C B41100 =1 180 CJNE A,#17,$+3

016F 4013 =1 181 JC IO0040 ; if '9'<char<'A', return FAIL

0171 9407 =1 182 SUBB A,#7

0173 B41000 =1 183 CJNE A,#10h,$+3

0176 400F =1 184 JC IO0050 ; if 'A'<=char<='F', return OK

0178 B42A00 =1 185 CJNE A,#42,$+3

017B 4007 =1 186 JC IO0040 ; if 'F'<char<'a', return FAIL

017D 9420 =1 187 SUBB A,#20h

017F B41000 =1 188 CJNE A,#10h,$+3

0182 4003 =1 189 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 190

0184 C3 =1 191 IO0040: CLR C ; ..else return FAIL

0185 74FF =1 192 MOV A,#0FFh

=1 193

0187 B3 =1 194 IO0050: CPL C

0188 22 =1 195 RET

=1 196

=1 197 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 198 ; GETCHAR

=1 199

0189 =1 200 GETCHAR: ; waits for a single ASCII character to be received

=1 201 ; by the UART. places this character into A.

=1 202

0189 3098FD =1 203 JNB RI,$

018C E599 =1 204 MOV A,SBUF

018E C298 =1 205 CLR RI

=1 206

0190 22 =1 207 RET

=1 208

=1 209 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 210 ; GETVAL

=1 211

0191 =1 212 GETVAL: ; waits for two ASCII hex digits to be received by

=1 213 ; the UART. returns the hex value in A.

=1 214

0191 C0F0 =1 215 PUSH B

0193 C000 =1 216 PUSH 0

0195 C298 =1 217 IO0060: CLR RI

0197 3189 =1 218 CALL GETCHAR ; first nibble

0199 F500 =1 219 MOV 0,A ; store received char

019B 3164 =1 220 CALL ASCII2HEX

019D 40F6 =1 221 JC IO0060 ; if not '0' thru 'F', don't accept

019F C4 =1 222 SWAP A ; swap nibbles

01A0 F5F0 =1 223 MOV B,A ; store nibble in B

01A2 E500 =1 224 MOV A,0 ; echo received char

01A4 313C =1 225 CALL SENDCHAR

01A6 C298 =1 226 IO0070: CLR RI

01A8 3189 =1 227 CALL GETCHAR ; second nibble

01AA F500 =1 228 MOV 0,A ; store received char

01AC 3164 =1 229 CALL ASCII2HEX

01AE 40F6 =1 230 JC IO0070 ; if not '0' thru 'F', don't accept

01B0 45F0 =1 231 ORL A,B ; combine nibbles

01B2 F5F0 =1 232 MOV B,A ; store results in B

ADCUART PAGE 5

01B4 E500 =1 233 MOV A,0 ; echo received char

01B6 313C =1 234 CALL SENDCHAR

01B8 E5F0 =1 235 MOV A,B ; final result

01BA D000 =1 236 POP 0

01BC D0F0 =1 237 POP B

=1 238

01BE 22 =1 239 RET

=1 240

241

242 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

243 ; TEXT DATA TABLES

244

01BF 0A0D00 245 SEPERATOR: DB 10,13,0

246

247 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

248

249 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

ADCUART PAGE 6

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 0164H

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

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

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

GETCHAR. . . . . . . . . . . . . C ADDR 0189H

GETVAL . . . . . . . . . . . . . C ADDR 0191H NOT USED

HEX2ASCII. . . . . . . . . . . . C ADDR 0158H

IO0010 . . . . . . . . . . . . . C ADDR 012BH

IO0020 . . . . . . . . . . . . . C ADDR 0137H

IO0030 . . . . . . . . . . . . . C ADDR 0161H

IO0040 . . . . . . . . . . . . . C ADDR 0184H

IO0050 . . . . . . . . . . . . . C ADDR 0187H

IO0060 . . . . . . . . . . . . . C ADDR 0195H

IO0070 . . . . . . . . . . . . . C ADDR 01A6H

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

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

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 013CH

SENDSTRING . . . . . . . . . . . C ADDR 0124H

SENDVAL. . . . . . . . . . . . . C ADDR 0144H

SEPERATOR. . . . . . . . . . . . C ADDR 01BFH

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