834UART PAGE 1

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

2 ;

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

4 ;

5 ; Date : 5 November 2001

6 ;

7 ; File : 834uart.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.

14 ;

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

16

17 $MOD834 ; Use 8052&ADuC834 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 ; INTERRUPT VECTOR SPACE

31

0033 32 ORG 0033h ; (ADC ISR)

33

0033 B2B4 34 CPL LED

35

0035 9001B6 36 MOV DPTR,#SEPERATOR ; send linefeed+CR out UART

0038 12011B 37 CALL SENDSTRING

38

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

003D 12013B 40 CALL SENDVAL

0040 E5DA 41 MOV A,ADC0M

0042 12013B 42 CALL SENDVAL

0045 E5D9 43 MOV A,ADC0L

0047 12013B 44 CALL SENDVAL

45

004A C2DF 46 CLR RDY0

004C 32 47 RETI

48

49 ;====================================================================

50 ; MAIN PROGRAM

0100 51 ORG 0100h

52

0100 53 MAIN:

0100 75817F 54 MOV SP,#127

55

56 ; CONFIGURE UART....

57

0103 759E82 58 MOV T3CON,#82h

834UART PAGE 2

0106 759D12 59 MOV T3FD,#12h

0109 759852 60 MOV SCON,#52h

61

62 ; CONFIGURE ADC AND START CONVERTING....

63

010C 75D4C8 64 MOV SF,#200 ; 6.8266667Hz ADC data rate

010F 75D245 65 MOV ADC0CON,#045h ; externalVref, bipolar, ±640mV

0112 D2AE 66 SETB EADC ; enable ADC interrupt (trig on RDY0)

0114 D2AF 67 SETB EA ; enable global interrupts

0116 75D123 68 MOV ADCMODE,#023h ; continuous conversion mode

69

70 ; WAIT FOR INTERRUPTS....

71

0119 80FE 72 JMP $ ; endless loop

73

74 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

75 ; SUBROUTINE INCLUDE FILE

76

=1 77 $INCLUDE(UARTIO.asm)

=1 78 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 79 ;

=1 80 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 81 ;

=1 82 ; Date : 12 October 1999

=1 83 ;

=1 84 ; File : UARTIO.hex

=1 85 ;

=1 86 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 87 ;

=1 88 ; Description : standard UART I/O subroutines. total size of this

=1 89 ; code when assembled is 155 bytes. routines for use

=1 90 ; external to this file are:

=1 91 ;

=1 92 ; SENDSTRING - sends a string of characters

=1 93 ; SENDCHAR - sends a single character

=1 94 ; SENDVAL - sends a byte as 2 ASCII characters

=1 95 ; HEX2ASCII - converts from HEX to ASCII

=1 96 ; ASCII2HEX - converts from ASCII to HEX

=1 97 ; GETCHAR - gets a single character

=1 98 ; GETVAL - gets a byte as 2 ASCII characters

=1 99 ;

=1 100 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 101

=1 102 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 103 ; SENDSTRING

=1 104

011B =1 105 SENDSTRING: ; sends ASCII string to UART starting at location

=1 106 ; DPTR and ending with a null (0) value

=1 107

011B C0E0 =1 108 PUSH ACC

011D C0F0 =1 109 PUSH B

011F E4 =1 110 CLR A

0120 F5F0 =1 111 MOV B,A

0122 E5F0 =1 112 IO0010: MOV A,B

0124 05F0 =1 113 INC B

0126 93 =1 114 MOVC A,@A+DPTR

0127 6005 =1 115 JZ IO0020

0129 120133 =1 116 CALL SENDCHAR

834UART PAGE 3

012C 80F4 =1 117 JMP IO0010

012E D0F0 =1 118 IO0020: POP B

0130 D0E0 =1 119 POP ACC

=1 120

0132 22 =1 121 RET

=1 122

=1 123 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 124 ; SENDCHAR

=1 125

0133 =1 126 SENDCHAR: ; sends ASCII value contained in A to UART

=1 127

0133 3099FD =1 128 JNB TI,$ ; wait til present char gone

0136 C299 =1 129 CLR TI ; must clear TI

0138 F599 =1 130 MOV SBUF,A

=1 131

013A 22 =1 132 RET

=1 133

=1 134 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 135 ; SENDVAL

=1 136

013B =1 137 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 138 ; and then spits these two characters up the UART.

=1 139 ; does not change the value of A.

=1 140

013B C0E0 =1 141 PUSH ACC

013D C4 =1 142 SWAP A

013E 12014F =1 143 CALL HEX2ASCII

0141 3133 =1 144 CALL SENDCHAR ; send high nibble

0143 D0E0 =1 145 POP ACC

0145 C0E0 =1 146 PUSH ACC

0147 12014F =1 147 CALL HEX2ASCII

014A 3133 =1 148 CALL SENDCHAR ; send low nibble

014C D0E0 =1 149 POP ACC

=1 150

014E 22 =1 151 RET

=1 152

=1 153 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 154 ; HEX2ASCII

=1 155

014F =1 156 HEX2ASCII: ; converts A into the hex character representing the

=1 157 ; value of A's least significant nibble

=1 158

014F 540F =1 159 ANL A,#00Fh

0151 B40A00 =1 160 CJNE A,#00Ah,$+3

0154 4002 =1 161 JC IO0030

0156 2407 =1 162 ADD A,#007h

0158 2430 =1 163 IO0030: ADD A,#'0'

=1 164

015A 22 =1 165 RET

=1 166

=1 167 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 168 ; ASCII2HEX

=1 169

015B =1 170 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 171 ; into the corresponding number (0-15). returns C=1

=1 172 ; when input is other than an ASCII digit,

=1 173 ; indicating invalid output (returned as 255).

=1 174

834UART PAGE 4

015B C3 =1 175 CLR C

015C 9430 =1 176 SUBB A,#'0'

015E B40A00 =1 177 CJNE A,#10,$+3

0161 401B =1 178 JC IO0050 ; if '0'<=char<='9', return OK

0163 B41100 =1 179 CJNE A,#17,$+3

0166 4013 =1 180 JC IO0040 ; if '9'<char<'A', return FAIL

0168 9407 =1 181 SUBB A,#7

016A B41000 =1 182 CJNE A,#10h,$+3

016D 400F =1 183 JC IO0050 ; if 'A'<=char<='F', return OK

016F B42A00 =1 184 CJNE A,#42,$+3

0172 4007 =1 185 JC IO0040 ; if 'F'<char<'a', return FAIL

0174 9420 =1 186 SUBB A,#20h

0176 B41000 =1 187 CJNE A,#10h,$+3

0179 4003 =1 188 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 189

017B C3 =1 190 IO0040: CLR C ; ..else return FAIL

017C 74FF =1 191 MOV A,#0FFh

=1 192

017E B3 =1 193 IO0050: CPL C

017F 22 =1 194 RET

=1 195

=1 196 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 197 ; GETCHAR

=1 198

0180 =1 199 GETCHAR: ; waits for a single ASCII character to be received

=1 200 ; by the UART. places this character into A.

=1 201

0180 3098FD =1 202 JNB RI,$

0183 E599 =1 203 MOV A,SBUF

0185 C298 =1 204 CLR RI

=1 205

0187 22 =1 206 RET

=1 207

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

=1 209 ; GETVAL

=1 210

0188 =1 211 GETVAL: ; waits for two ASCII hex digits to be received by

=1 212 ; the UART. returns the hex value in A.

=1 213

0188 C0F0 =1 214 PUSH B

018A C000 =1 215 PUSH 0

018C C298 =1 216 IO0060: CLR RI

018E 3180 =1 217 CALL GETCHAR ; first nibble

0190 F500 =1 218 MOV 0,A ; store received char

0192 315B =1 219 CALL ASCII2HEX

0194 40F6 =1 220 JC IO0060 ; if not '0' thru 'F', don't accept

0196 C4 =1 221 SWAP A ; swap nibbles

0197 F5F0 =1 222 MOV B,A ; store nibble in B

0199 E500 =1 223 MOV A,0 ; echo received char

019B 3133 =1 224 CALL SENDCHAR

019D C298 =1 225 IO0070: CLR RI

019F 3180 =1 226 CALL GETCHAR ; second nibble

01A1 F500 =1 227 MOV 0,A ; store received char

01A3 315B =1 228 CALL ASCII2HEX

01A5 40F6 =1 229 JC IO0070 ; if not '0' thru 'F', don't accept

01A7 45F0 =1 230 ORL A,B ; combine nibbles

01A9 F5F0 =1 231 MOV B,A ; store results in B

01AB E500 =1 232 MOV A,0 ; echo received char

834UART PAGE 5

01AD 3133 =1 233 CALL SENDCHAR

01AF E5F0 =1 234 MOV A,B ; final result

01B1 D000 =1 235 POP 0

01B3 D0F0 =1 236 POP B

=1 237

01B5 22 =1 238 RET

=1 239

240

241 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

242 ; TEXT DATA TABLES

243

01B6 0A0D00 244 SEPERATOR: DB 10,13,0

245

246 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

247

248 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

834UART PAGE 6

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 015BH

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

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

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

GETCHAR. . . . . . . . . . . . . C ADDR 0180H

GETVAL . . . . . . . . . . . . . C ADDR 0188H NOT USED

HEX2ASCII. . . . . . . . . . . . C ADDR 014FH

IO0010 . . . . . . . . . . . . . C ADDR 0122H

IO0020 . . . . . . . . . . . . . C ADDR 012EH

IO0030 . . . . . . . . . . . . . C ADDR 0158H

IO0040 . . . . . . . . . . . . . C ADDR 017BH

IO0050 . . . . . . . . . . . . . C ADDR 017EH

IO0060 . . . . . . . . . . . . . C ADDR 018CH

IO0070 . . . . . . . . . . . . . C ADDR 019DH

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

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

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 0133H

SENDSTRING . . . . . . . . . . . C ADDR 011BH

SENDVAL. . . . . . . . . . . . . C ADDR 013BH

SEPERATOR. . . . . . . . . . . . C ADDR 01B6H

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