824UART PAGE 1

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

2 ;

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

4 ;

5 ; Date : 22 September 1999

6 ;

7 ; File : 824uart.asm

8 ;

9 ; Hardware : ADuC824

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 $MOD824 ; Use 8052&ADuC824 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 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

0045 E5D9 43 MOV A,ADC0L

0047 120144 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 75CBFF 58 MOV RCAP2H,#0FFh ; config UART for 9830baud

824UART PAGE 2

0106 75CAFB 59 MOV RCAP2L,#-5 ; (close enough to 9600baud)

0109 75CDFF 60 MOV TH2,#0FFh

010C 75CCFB 61 MOV TL2,#-5

010F 759852 62 MOV SCON,#01010010b

0112 75C834 63 MOV T2CON,#00110100b

64

65 ; CONFIGURE ADC AND START CONVERTING....

66

0115 75D4C8 67 MOV SF,#200 ; 6.8266667Hz ADC data rate

0118 75D245 68 MOV ADC0CON,#045h ; externalVref, bipolar, ±640mV

011B D2AE 69 SETB EADC ; enable ADC interrupt (trig on RDY0)

011D D2AF 70 SETB EA ; enable global interrupts

011F 75D123 71 MOV ADCMODE,#023h ; continuous conversion mode

72

73 ; WAIT FOR INTERRUPTS....

74

0122 80FE 75 JMP $ ; endless loop

76

77 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

78 ; SUBROUTINE INCLUDE FILE

79

=1 80 $INCLUDE(UARTIO.asm)

=1 81 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 82 ;

=1 83 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 84 ;

=1 85 ; Date : 12 October 1999

=1 86 ;

=1 87 ; File : UARTIO.hex

=1 88 ;

=1 89 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 90 ;

=1 91 ; Description : standard UART I/O subroutines. total size of this

=1 92 ; code when assembled is 155 bytes. routines for use

=1 93 ; external to this file are:

=1 94 ;

=1 95 ; SENDSTRING - sends a string of characters

=1 96 ; SENDCHAR - sends a single character

=1 97 ; SENDVAL - sends a byte as 2 ASCII characters

=1 98 ; HEX2ASCII - converts from HEX to ASCII

=1 99 ; ASCII2HEX - converts from ASCII to HEX

=1 100 ; GETCHAR - gets a single character

=1 101 ; GETVAL - gets a byte as 2 ASCII characters

=1 102 ;

=1 103 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 104

=1 105 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 106 ; SENDSTRING

=1 107

0124 =1 108 SENDSTRING: ; sends ASCII string to UART starting at location

=1 109 ; DPTR and ending with a null (0) value

=1 110

0124 C0E0 =1 111 PUSH ACC

0126 C0F0 =1 112 PUSH B

0128 E4 =1 113 CLR A

0129 F5F0 =1 114 MOV B,A

012B E5F0 =1 115 IO0010: MOV A,B

012D 05F0 =1 116 INC B

824UART PAGE 3

012F 93 =1 117 MOVC A,@A+DPTR

0130 6005 =1 118 JZ IO0020

0132 12013C =1 119 CALL SENDCHAR

0135 80F4 =1 120 JMP IO0010

0137 D0F0 =1 121 IO0020: POP B

0139 D0E0 =1 122 POP ACC

=1 123

013B 22 =1 124 RET

=1 125

=1 126 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 127 ; SENDCHAR

=1 128

013C =1 129 SENDCHAR: ; sends ASCII value contained in A to UART

=1 130

013C 3099FD =1 131 JNB TI,$ ; wait til present char gone

013F C299 =1 132 CLR TI ; must clear TI

0141 F599 =1 133 MOV SBUF,A

=1 134

0143 22 =1 135 RET

=1 136

=1 137 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 138 ; SENDVAL

=1 139

0144 =1 140 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 141 ; and then spits these two characters up the UART.

=1 142 ; does not change the value of A.

=1 143

0144 C0E0 =1 144 PUSH ACC

0146 C4 =1 145 SWAP A

0147 120158 =1 146 CALL HEX2ASCII

014A 313C =1 147 CALL SENDCHAR ; send high nibble

014C D0E0 =1 148 POP ACC

014E C0E0 =1 149 PUSH ACC

0150 120158 =1 150 CALL HEX2ASCII

0153 313C =1 151 CALL SENDCHAR ; send low nibble

0155 D0E0 =1 152 POP ACC

=1 153

0157 22 =1 154 RET

=1 155

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

=1 157 ; HEX2ASCII

=1 158

0158 =1 159 HEX2ASCII: ; converts A into the hex character representing the

=1 160 ; value of A's least significant nibble

=1 161

0158 540F =1 162 ANL A,#00Fh

015A B40A00 =1 163 CJNE A,#00Ah,$+3

015D 4002 =1 164 JC IO0030

015F 2407 =1 165 ADD A,#007h

0161 2430 =1 166 IO0030: ADD A,#'0'

=1 167

0163 22 =1 168 RET

=1 169

=1 170 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 171 ; ASCII2HEX

=1 172

0164 =1 173 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 174 ; into the corresponding number (0-15). returns C=1

824UART PAGE 4

=1 175 ; when input is other than an ASCII digit,

=1 176 ; indicating invalid output (returned as 255).

=1 177

0164 C3 =1 178 CLR C

0165 9430 =1 179 SUBB A,#'0'

0167 B40A00 =1 180 CJNE A,#10,$+3

016A 401B =1 181 JC IO0050 ; if '0'<=char<='9', return OK

016C B41100 =1 182 CJNE A,#17,$+3

016F 4013 =1 183 JC IO0040 ; if '9'<char<'A', return FAIL

0171 9407 =1 184 SUBB A,#7

0173 B41000 =1 185 CJNE A,#10h,$+3

0176 400F =1 186 JC IO0050 ; if 'A'<=char<='F', return OK

0178 B42A00 =1 187 CJNE A,#42,$+3

017B 4007 =1 188 JC IO0040 ; if 'F'<char<'a', return FAIL

017D 9420 =1 189 SUBB A,#20h

017F B41000 =1 190 CJNE A,#10h,$+3

0182 4003 =1 191 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 192

0184 C3 =1 193 IO0040: CLR C ; ..else return FAIL

0185 74FF =1 194 MOV A,#0FFh

=1 195

0187 B3 =1 196 IO0050: CPL C

0188 22 =1 197 RET

=1 198

=1 199 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 200 ; GETCHAR

=1 201

0189 =1 202 GETCHAR: ; waits for a single ASCII character to be received

=1 203 ; by the UART. places this character into A.

=1 204

0189 3098FD =1 205 JNB RI,$

018C E599 =1 206 MOV A,SBUF

018E C298 =1 207 CLR RI

=1 208

0190 22 =1 209 RET

=1 210

=1 211 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 212 ; GETVAL

=1 213

0191 =1 214 GETVAL: ; waits for two ASCII hex digits to be received by

=1 215 ; the UART. returns the hex value in A.

=1 216

0191 C0F0 =1 217 PUSH B

0193 C000 =1 218 PUSH 0

0195 C298 =1 219 IO0060: CLR RI

0197 3189 =1 220 CALL GETCHAR ; first nibble

0199 F500 =1 221 MOV 0,A ; store received char

019B 3164 =1 222 CALL ASCII2HEX

019D 40F6 =1 223 JC IO0060 ; if not '0' thru 'F', don't accept

019F C4 =1 224 SWAP A ; swap nibbles

01A0 F5F0 =1 225 MOV B,A ; store nibble in B

01A2 E500 =1 226 MOV A,0 ; echo received char

01A4 313C =1 227 CALL SENDCHAR

01A6 C298 =1 228 IO0070: CLR RI

01A8 3189 =1 229 CALL GETCHAR ; second nibble

01AA F500 =1 230 MOV 0,A ; store received char

01AC 3164 =1 231 CALL ASCII2HEX

01AE 40F6 =1 232 JC IO0070 ; if not '0' thru 'F', don't accept

824UART PAGE 5

01B0 45F0 =1 233 ORL A,B ; combine nibbles

01B2 F5F0 =1 234 MOV B,A ; store results in B

01B4 E500 =1 235 MOV A,0 ; echo received char

01B6 313C =1 236 CALL SENDCHAR

01B8 E5F0 =1 237 MOV A,B ; final result

01BA D000 =1 238 POP 0

01BC D0F0 =1 239 POP B

=1 240

01BE 22 =1 241 RET

=1 242

243

244 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

245 ; TEXT DATA TABLES

246

01BF 0A0D00 247 SEPERATOR: DB 10,13,0

248

249 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

250

251 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

824UART 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 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