ADCSINGL PAGE 1

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

2 ;

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

4 ;

5 ; Date : March 2001

6 ;

7 ; File : ADCsingl.asm

8 ;

9 ; Hardware : ADuC832

10 ;

11 ; Description : Performs repeated single ADC conversions and moves

12 ; results to UART. Sets the red LED on the eval

13 ; board upon completion of each conversion. A new

14 ; conversion is innitiated every 200ms.

15 ; All rate calculations assume an 2.097152MHz Mclk.

16 ;

17 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

18

19 $MOD832 ; Use 8052&ADuC832 predefined symbols

20

00B4 21 LED EQU P3.4 ; P3.4 drives red LED on eval board

0000 22 CHAN EQU 0 ; convert this ADC input channel..

23 ; ..chan values can be 0 thru 6

24 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

25 ; BEGINNING OF CODE

---- 26 CSEG

27

0000 28 ORG 0000h

29

0000 02004B 30 JMP MAIN ; jump to main program

31 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

32

0033 33 ORG 0033H ; INTERRUPT VECTOR SPACE

0033 D2B4 34 SETB LED

0035 32 35 RETI

36

37

38 ;====================================================================

39 ; MAIN PROGRAM

004B 40 ORG 004Bh

41

004B 42 MAIN:

43

44 ; Set up UART

004B 75CBFF 45 MOV RCAP2H,#0FFh ; config UART for 9600 baud

004E 75CAF9 46 MOV RCAP2L,#-7 ;

0051 75CDFF 47 MOV TH2,#0FFh

0054 75CCF9 48 MOV TL2,#-7

0057 759852 49 MOV SCON,#52h

005A 75C834 50 MOV T2CON,#34h

51 ; PRECONFIGURE...

52

005D 75EF80 53 MOV ADCCON1,#080h ; power up ADC

0060 75D800 54 MOV ADCCON2,#CHAN ; select channel to convert

0063 D2AF 55 SETB EA ; enable interrupts

0065 D2AE 56 SETB EADC ; enable ADC interrupt

57

58 ; PERFORM REPEATED SINGLE CONVERSIONS...

ADCSINGL PAGE 2

59

0067 C2B4 60 AGAIN: CLR LED ; turn the LED on

0069 7401 61 MOV A,#01H ; Delay length

006B 120084 62 CALL DELAY ; delay 100ms

006E D2DC 63 SETB SCONV ; innitiate single ADC conversion

64 ; ADC ISR is called upon completion

0070 30B4FD 65 JNB LED,$

0073 E5DA 66 MOV A,ADCDATAH

0075 1200B0 67 CALL SENDVAL

0078 E5D9 68 MOV A,ADCDATAL

007A 1200B0 69 CALL SENDVAL

70

007D 7401 71 MOV A,#01H ; Delay length

007F 120084 72 CALL DELAY ; delay 100ms

0082 80E3 73 JMP AGAIN ; repeat

74

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

76 ; SUBROUTINE

0084 77 DELAY: ; Delays by 100ms \* A

78 ; 100mSec based on 2.097152MHZ

79 ; Core Clock

80 ; i.e. default ADuC812S Clock

81

0084 F9 82 MOV R1,A ; Acc holds delay variable

0085 7A22 83 DLY0: MOV R2,#022h ; Set up delay loop0

0087 7BFF 84 DLY1: MOV R3,#0FFh ; Set up delay loop1

0089 DBFE 85 DJNZ R3,$ ; Dec R3 & Jump here until R3 is 0

008B DAFA 86 DJNZ R2,DLY1 ; Dec R2 & Jump DLY1 until R2 is 0

008D D9F6 87 DJNZ R1,DLY0 ; Dec R1 & Jump DLY0 until R1 is 0

008F 22 88 RET ; Return from subroutine

89

90

91 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

92

93

=1 94 $INCLUDE(UARTIO.ASM)

=1 95 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 96 ;

=1 97 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 98 ;

=1 99 ; Date : 12 October 1999

=1 100 ;

=1 101 ; File : UARTIO.hex

=1 102 ;

=1 103 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 104 ;

=1 105 ; Description : standard UART I/O subroutines. total size of this

=1 106 ; code when assembled is 155 bytes. routines for use

=1 107 ; external to this file are:

=1 108 ;

=1 109 ; SENDSTRING - sends a string of characters

=1 110 ; SENDCHAR - sends a single character

=1 111 ; SENDVAL - sends a byte as 2 ASCII characters

=1 112 ; HEX2ASCII - converts from HEX to ASCII

=1 113 ; ASCII2HEX - converts from ASCII to HEX

=1 114 ; GETCHAR - gets a single character

=1 115 ; GETVAL - gets a byte as 2 ASCII characters

=1 116 ;

ADCSINGL PAGE 3

=1 117 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 118

=1 119 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 120 ; SENDSTRING

=1 121

0090 =1 122 SENDSTRING: ; sends ASCII string to UART starting at location

=1 123 ; DPTR and ending with a null (0) value

=1 124

0090 C0E0 =1 125 PUSH ACC

0092 C0F0 =1 126 PUSH B

0094 E4 =1 127 CLR A

0095 F5F0 =1 128 MOV B,A

0097 E5F0 =1 129 IO0010: MOV A,B

0099 05F0 =1 130 INC B

009B 93 =1 131 MOVC A,@A+DPTR

009C 6005 =1 132 JZ IO0020

009E 1200A8 =1 133 CALL SENDCHAR

00A1 80F4 =1 134 JMP IO0010

00A3 D0F0 =1 135 IO0020: POP B

00A5 D0E0 =1 136 POP ACC

=1 137

00A7 22 =1 138 RET

=1 139

=1 140 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 141 ; SENDCHAR

=1 142

00A8 =1 143 SENDCHAR: ; sends ASCII value contained in A to UART

=1 144

00A8 3099FD =1 145 JNB TI,$ ; wait til present char gone

00AB C299 =1 146 CLR TI ; must clear TI

00AD F599 =1 147 MOV SBUF,A

=1 148

00AF 22 =1 149 RET

=1 150

=1 151 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 152 ; SENDVAL

=1 153

00B0 =1 154 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 155 ; and then spits these two characters up the UART.

=1 156 ; does not change the value of A.

=1 157

00B0 C0E0 =1 158 PUSH ACC

00B2 C4 =1 159 SWAP A

00B3 1200C4 =1 160 CALL HEX2ASCII

00B6 11A8 =1 161 CALL SENDCHAR ; send high nibble

00B8 D0E0 =1 162 POP ACC

00BA C0E0 =1 163 PUSH ACC

00BC 1200C4 =1 164 CALL HEX2ASCII

00BF 11A8 =1 165 CALL SENDCHAR ; send low nibble

00C1 D0E0 =1 166 POP ACC

=1 167

00C3 22 =1 168 RET

=1 169

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

=1 171 ; HEX2ASCII

=1 172

00C4 =1 173 HEX2ASCII: ; converts A into the hex character representing the

=1 174 ; value of A's least significant nibble

ADCSINGL PAGE 4

=1 175

00C4 540F =1 176 ANL A,#00Fh

00C6 B40A00 =1 177 CJNE A,#00Ah,$+3

00C9 4002 =1 178 JC IO0030

00CB 2407 =1 179 ADD A,#007h

00CD 2430 =1 180 IO0030: ADD A,#'0'

=1 181

00CF 22 =1 182 RET

=1 183

=1 184 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 185 ; ASCII2HEX

=1 186

00D0 =1 187 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 188 ; into the corresponding number (0-15). returns C=1

=1 189 ; when input is other than an ASCII digit,

=1 190 ; indicating invalid output (returned as 255).

=1 191

00D0 C3 =1 192 CLR C

00D1 9430 =1 193 SUBB A,#'0'

00D3 B40A00 =1 194 CJNE A,#10,$+3

00D6 401B =1 195 JC IO0050 ; if '0'<=char<='9', return OK

00D8 B41100 =1 196 CJNE A,#17,$+3

00DB 4013 =1 197 JC IO0040 ; if '9'<char<'A', return FAIL

00DD 9407 =1 198 SUBB A,#7

00DF B41000 =1 199 CJNE A,#10h,$+3

00E2 400F =1 200 JC IO0050 ; if 'A'<=char<='F', return OK

00E4 B42A00 =1 201 CJNE A,#42,$+3

00E7 4007 =1 202 JC IO0040 ; if 'F'<char<'a', return FAIL

00E9 9420 =1 203 SUBB A,#20h

00EB B41000 =1 204 CJNE A,#10h,$+3

00EE 4003 =1 205 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 206

00F0 C3 =1 207 IO0040: CLR C ; ..else return FAIL

00F1 74FF =1 208 MOV A,#0FFh

=1 209

00F3 B3 =1 210 IO0050: CPL C

00F4 22 =1 211 RET

=1 212

=1 213 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 214 ; GETCHAR

=1 215

00F5 =1 216 GETCHAR: ; waits for a single ASCII character to be received

=1 217 ; by the UART. places this character into A.

=1 218

00F5 3098FD =1 219 JNB RI,$

00F8 E599 =1 220 MOV A,SBUF

00FA C298 =1 221 CLR RI

=1 222

00FC 22 =1 223 RET

=1 224

=1 225 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 226 ; GETVAL

=1 227

00FD =1 228 GETVAL: ; waits for two ASCII hex digits to be received by

=1 229 ; the UART. returns the hex value in A.

=1 230

00FD C0F0 =1 231 PUSH B

00FF C000 =1 232 PUSH 0

ADCSINGL PAGE 5

0101 C298 =1 233 IO0060: CLR RI

0103 11F5 =1 234 CALL GETCHAR ; first nibble

0105 F500 =1 235 MOV 0,A ; store received char

0107 11D0 =1 236 CALL ASCII2HEX

0109 40F6 =1 237 JC IO0060 ; if not '0' thru 'F', don't accept

010B C4 =1 238 SWAP A ; swap nibbles

010C F5F0 =1 239 MOV B,A ; store nibble in B

010E E500 =1 240 MOV A,0 ; echo received char

0110 11A8 =1 241 CALL SENDCHAR

0112 C298 =1 242 IO0070: CLR RI

0114 11F5 =1 243 CALL GETCHAR ; second nibble

0116 F500 =1 244 MOV 0,A ; store received char

0118 11D0 =1 245 CALL ASCII2HEX

011A 40F6 =1 246 JC IO0070 ; if not '0' thru 'F', don't accept

011C 45F0 =1 247 ORL A,B ; combine nibbles

011E F5F0 =1 248 MOV B,A ; store results in B

0120 E500 =1 249 MOV A,0 ; echo received char

0122 11A8 =1 250 CALL SENDCHAR

0124 E5F0 =1 251 MOV A,B ; final result

0126 D000 =1 252 POP 0

0128 D0F0 =1 253 POP B

=1 254

012A 22 =1 255 RET

=1 256

257 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

ADCSINGL PAGE 6

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

ADCCON1. . . . . . . . . . . . . D ADDR 00EFH PREDEFINED

ADCCON2. . . . . . . . . . . . . D ADDR 00D8H PREDEFINED

ADCDATAH . . . . . . . . . . . . D ADDR 00DAH PREDEFINED

ADCDATAL . . . . . . . . . . . . D ADDR 00D9H PREDEFINED

AGAIN. . . . . . . . . . . . . . C ADDR 0067H

ASCII2HEX. . . . . . . . . . . . C ADDR 00D0H

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

CHAN . . . . . . . . . . . . . . NUMB 0000H

DELAY. . . . . . . . . . . . . . C ADDR 0084H

DLY0 . . . . . . . . . . . . . . C ADDR 0085H

DLY1 . . . . . . . . . . . . . . C ADDR 0087H

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

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

GETCHAR. . . . . . . . . . . . . C ADDR 00F5H

GETVAL . . . . . . . . . . . . . C ADDR 00FDH NOT USED

HEX2ASCII. . . . . . . . . . . . C ADDR 00C4H

IO0010 . . . . . . . . . . . . . C ADDR 0097H

IO0020 . . . . . . . . . . . . . C ADDR 00A3H

IO0030 . . . . . . . . . . . . . C ADDR 00CDH

IO0040 . . . . . . . . . . . . . C ADDR 00F0H

IO0050 . . . . . . . . . . . . . C ADDR 00F3H

IO0060 . . . . . . . . . . . . . C ADDR 0101H

IO0070 . . . . . . . . . . . . . C ADDR 0112H

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

MAIN . . . . . . . . . . . . . . C ADDR 004BH

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

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

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

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

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

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

SCONV. . . . . . . . . . . . . . B ADDR 00DCH PREDEFINED

SENDCHAR . . . . . . . . . . . . C ADDR 00A8H

SENDSTRING . . . . . . . . . . . C ADDR 0090H NOT USED

SENDVAL. . . . . . . . . . . . . C ADDR 00B0H

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

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

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

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