ADCPIN PAGE 1

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

2 ;

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

4 ;

5 ; Date : October 2003

6 ;

7 ; File : ADCpin.asm

8 ;

9 ; Hardware : ADuC842/ADuC843

10 ;

11 ; Description : Performs hardware pin driven ADC conversions and

12 ; outputs results on the UART Continuously flashes

13 ; LED (independently of ADC routine) at approximately

14 ; 5Hz (assuming an 2.097152 MHz Mclk).

15 ;

16 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

17

18 $MOD842 ; Use 8052&ADuC842 predefined symbols

19

00B4 20 LED EQU P3.4 ; P3.4 drives red LED on eval board

0000 21 CHAN EQU 0 ; convert this ADC input channel..

22 ; ..chan values can be 0 thru 8

23 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

24 ; BEGINNING OF CODE

---- 25 CSEG

26

0000 27 ORG 0000h

28

0000 02004B 29 JMP MAIN ; jump to main program

30 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

31 ; INTERRUPT VECTOR SPACE

0033 32 ORG 0033H ; (ADC ISR)

0033 E5DA 33 MOV A,ADCDATAH

0035 120096 34 CALL SENDVAL

0038 E5D9 35 MOV A,ADCDATAL

003A 120096 36 CALL SENDVAL

003D 32 37 RETI

38

39 ;====================================================================

40 ; MAIN PROGRAM

004B 41 ORG 004Bh

42

004B 43 MAIN:

44

45 ; Set up UART

004B 759E83 46 MOV T3CON,#083h

004E 759D2D 47 MOV T3FD,#02DH

0051 759852 48 MOV SCON,#52H

49 ; PRECONFIGURE...

50

0054 75EFAC 51 MOV ADCCON1,#0ACh ; power up ADC

0057 75D800 52 MOV ADCCON2,#CHAN ; select channel to convert

53

54 ; LAUNCH CONTINUOUS CONVERSIONS...

55

005A D2AF 56 SETB EA ; enable interrupts

005C D2AE 57 SETB EADC ; enable ADC interrupt

005E 43EF01 58 ORL ADCCON1,#001h ; enable hardware CONVST pin

ADCPIN PAGE 2

59

60 ; CONTINUE WITH OTHER CODE...

61

0061 B2B4 62 AGAIN: CPL LED ; blink (complement) the LED

0063 740A 63 MOV A,#010 ; delay length

0065 12006A 64 CALL DELAY ; delay 100ms

0068 80F7 65 JMP AGAIN ; repeat

66

67 ; the micro is free to continue with other tasks (flashing the LED in

68 ; this case) while the ADC is converting, synchronously to the

69 ; external CONVST pin. results are being handled by the ADC

70 ; interrupt service routine.

71

72 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

73 ; SUBROUTINE

006A 74 DELAY: ; Delays by 10ms \* A

75 ; 10mSec based on 2.097152MHz

76 ; Core Clock

77 ;

78

006A F9 79 MOV R1,A ; Acc holds delay variable (1 clock)

006B 7A1B 80 DLY0: MOV R2,#01Bh ; Set up delay loop0 (2 clocks)

006D 7BFF 81 DLY1: MOV R3,#0FFh ; Set up delay loop1 (2 clocks)

006F DBFE 82 DJNZ R3,$ ; Dec R3 & Jump here until R3 is 0 (3 clocks)

0071 DAFA 83 DJNZ R2,DLY1 ; Dec R2 & Jump DLY1 until R2 is 0 (3 clocks)

0073 D9F6 84 DJNZ R1,DLY0 ; Dec R1 & Jump DLY0 until R1 is 0 (3 clocks)

0075 22 85 RET ; Return from subroutine

86

87 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

88

=1 89 $INCLUDE(UARTIO.ASM)

=1 90 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 91 ;

=1 92 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 93 ;

=1 94 ; Date : 12 October 1999

=1 95 ;

=1 96 ; File : UARTIO.hex

=1 97 ;

=1 98 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 99 ;

=1 100 ; Description : standard UART I/O subroutines. total size of this

=1 101 ; code when assembled is 155 bytes. routines for use

=1 102 ; external to this file are:

=1 103 ;

=1 104 ; SENDSTRING - sends a string of characters

=1 105 ; SENDCHAR - sends a single character

=1 106 ; SENDVAL - sends a byte as 2 ASCII characters

=1 107 ; HEX2ASCII - converts from HEX to ASCII

=1 108 ; ASCII2HEX - converts from ASCII to HEX

=1 109 ; GETCHAR - gets a single character

=1 110 ; GETVAL - gets a byte as 2 ASCII characters

=1 111 ;

=1 112 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 113

=1 114 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 115 ; SENDSTRING

=1 116

ADCPIN PAGE 3

0076 =1 117 SENDSTRING: ; sends ASCII string to UART starting at location

=1 118 ; DPTR and ending with a null (0) value

=1 119

0076 C0E0 =1 120 PUSH ACC

0078 C0F0 =1 121 PUSH B

007A E4 =1 122 CLR A

007B F5F0 =1 123 MOV B,A

007D E5F0 =1 124 IO0010: MOV A,B

007F 05F0 =1 125 INC B

0081 93 =1 126 MOVC A,@A+DPTR

0082 6005 =1 127 JZ IO0020

0084 12008E =1 128 CALL SENDCHAR

0087 80F4 =1 129 JMP IO0010

0089 D0F0 =1 130 IO0020: POP B

008B D0E0 =1 131 POP ACC

=1 132

008D 22 =1 133 RET

=1 134

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

=1 136 ; SENDCHAR

=1 137

008E =1 138 SENDCHAR: ; sends ASCII value contained in A to UART

=1 139

008E 3099FD =1 140 JNB TI,$ ; wait til present char gone

0091 C299 =1 141 CLR TI ; must clear TI

0093 F599 =1 142 MOV SBUF,A

=1 143

0095 22 =1 144 RET

=1 145

=1 146 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 147 ; SENDVAL

=1 148

0096 =1 149 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 150 ; and then spits these two characters up the UART.

=1 151 ; does not change the value of A.

=1 152

0096 C0E0 =1 153 PUSH ACC

0098 C4 =1 154 SWAP A

0099 1200AA =1 155 CALL HEX2ASCII

009C 118E =1 156 CALL SENDCHAR ; send high nibble

009E D0E0 =1 157 POP ACC

00A0 C0E0 =1 158 PUSH ACC

00A2 1200AA =1 159 CALL HEX2ASCII

00A5 118E =1 160 CALL SENDCHAR ; send low nibble

00A7 D0E0 =1 161 POP ACC

=1 162

00A9 22 =1 163 RET

=1 164

=1 165 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 166 ; HEX2ASCII

=1 167

00AA =1 168 HEX2ASCII: ; converts A into the hex character representing the

=1 169 ; value of A's least significant nibble

=1 170

00AA 540F =1 171 ANL A,#00Fh

00AC B40A00 =1 172 CJNE A,#00Ah,$+3

00AF 4002 =1 173 JC IO0030

00B1 2407 =1 174 ADD A,#007h

ADCPIN PAGE 4

00B3 2430 =1 175 IO0030: ADD A,#'0'

=1 176

00B5 22 =1 177 RET

=1 178

=1 179 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 180 ; ASCII2HEX

=1 181

00B6 =1 182 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 183 ; into the corresponding number (0-15). returns C=1

=1 184 ; when input is other than an ASCII digit,

=1 185 ; indicating invalid output (returned as 255).

=1 186

00B6 C3 =1 187 CLR C

00B7 9430 =1 188 SUBB A,#'0'

00B9 B40A00 =1 189 CJNE A,#10,$+3

00BC 401B =1 190 JC IO0050 ; if '0'<=char<='9', return OK

00BE B41100 =1 191 CJNE A,#17,$+3

00C1 4013 =1 192 JC IO0040 ; if '9'<char<'A', return FAIL

00C3 9407 =1 193 SUBB A,#7

00C5 B41000 =1 194 CJNE A,#10h,$+3

00C8 400F =1 195 JC IO0050 ; if 'A'<=char<='F', return OK

00CA B42A00 =1 196 CJNE A,#42,$+3

00CD 4007 =1 197 JC IO0040 ; if 'F'<char<'a', return FAIL

00CF 9420 =1 198 SUBB A,#20h

00D1 B41000 =1 199 CJNE A,#10h,$+3

00D4 4003 =1 200 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 201

00D6 C3 =1 202 IO0040: CLR C ; ..else return FAIL

00D7 74FF =1 203 MOV A,#0FFh

=1 204

00D9 B3 =1 205 IO0050: CPL C

00DA 22 =1 206 RET

=1 207

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

=1 209 ; GETCHAR

=1 210

00DB =1 211 GETCHAR: ; waits for a single ASCII character to be received

=1 212 ; by the UART. places this character into A.

=1 213

00DB 3098FD =1 214 JNB RI,$

00DE E599 =1 215 MOV A,SBUF

00E0 C298 =1 216 CLR RI

=1 217

00E2 22 =1 218 RET

=1 219

=1 220 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 221 ; GETVAL

=1 222

00E3 =1 223 GETVAL: ; waits for two ASCII hex digits to be received by

=1 224 ; the UART. returns the hex value in A.

=1 225

00E3 C0F0 =1 226 PUSH B

00E5 C000 =1 227 PUSH 0

00E7 C298 =1 228 IO0060: CLR RI

00E9 11DB =1 229 CALL GETCHAR ; first nibble

00EB F500 =1 230 MOV 0,A ; store received char

00ED 11B6 =1 231 CALL ASCII2HEX

00EF 40F6 =1 232 JC IO0060 ; if not '0' thru 'F', don't accept

ADCPIN PAGE 5

00F1 C4 =1 233 SWAP A ; swap nibbles

00F2 F5F0 =1 234 MOV B,A ; store nibble in B

00F4 E500 =1 235 MOV A,0 ; echo received char

00F6 118E =1 236 CALL SENDCHAR

00F8 C298 =1 237 IO0070: CLR RI

00FA 11DB =1 238 CALL GETCHAR ; second nibble

00FC F500 =1 239 MOV 0,A ; store received char

00FE 11B6 =1 240 CALL ASCII2HEX

0100 40F6 =1 241 JC IO0070 ; if not '0' thru 'F', don't accept

0102 45F0 =1 242 ORL A,B ; combine nibbles

0104 F5F0 =1 243 MOV B,A ; store results in B

0106 E500 =1 244 MOV A,0 ; echo received char

0108 118E =1 245 CALL SENDCHAR

010A E5F0 =1 246 MOV A,B ; final result

010C D000 =1 247 POP 0

010E D0F0 =1 248 POP B

=1 249

0110 22 =1 250 RET

=1 251

252 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

ADCPIN 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 0061H

ASCII2HEX. . . . . . . . . . . . C ADDR 00B6H

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

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

DELAY. . . . . . . . . . . . . . C ADDR 006AH

DLY0 . . . . . . . . . . . . . . C ADDR 006BH

DLY1 . . . . . . . . . . . . . . C ADDR 006DH

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

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

GETCHAR. . . . . . . . . . . . . C ADDR 00DBH

GETVAL . . . . . . . . . . . . . C ADDR 00E3H NOT USED

HEX2ASCII. . . . . . . . . . . . C ADDR 00AAH

IO0010 . . . . . . . . . . . . . C ADDR 007DH

IO0020 . . . . . . . . . . . . . C ADDR 0089H

IO0030 . . . . . . . . . . . . . C ADDR 00B3H

IO0040 . . . . . . . . . . . . . C ADDR 00D6H

IO0050 . . . . . . . . . . . . . C ADDR 00D9H

IO0060 . . . . . . . . . . . . . C ADDR 00E7H

IO0070 . . . . . . . . . . . . . C ADDR 00F8H

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

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

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

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

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

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

SENDCHAR . . . . . . . . . . . . C ADDR 008EH

SENDSTRING . . . . . . . . . . . C ADDR 0076H NOT USED

SENDVAL. . . . . . . . . . . . . C ADDR 0096H

T3CON. . . . . . . . . . . . . . D ADDR 009EH PREDEFINED

T3FD . . . . . . . . . . . . . . D ADDR 009DH PREDEFINED

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