ADCPIN PAGE 1

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

2 ;

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

4 ;

5 ; Date : March 2001

6 ;

7 ; File : ADCpin.asm

8 ;

9 ; Hardware : ADuC814

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 $MOD814 ; Use 8052&ADuC814 predefined symbols

19

00B3 20 LED EQU P3.3 ; P3.3 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 12009F 34 CALL SENDVAL

0038 E5D9 35 MOV A,ADCDATAL

003A 12009F 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 75CBFF 46 MOV RCAP2H,#0FFh ; config UART for 9600 baud

004E 75CAF9 47 MOV RCAP2L,#-7 ;

0051 75CDFF 48 MOV TH2,#0FFh

0054 75CCF9 49 MOV TL2,#-7

0057 759852 50 MOV SCON,#52h

005A 75C834 51 MOV T2CON,#34h

52 ; PRECONFIGURE...

53

005D 75EF80 54 MOV ADCCON1,#080h ; power up ADC

0060 75D800 55 MOV ADCCON2,#CHAN ; select channel to convert

56

57 ; LAUNCH CONTINUOUS CONVERSIONS...

58

ADCPIN PAGE 2

0063 D2AF 59 SETB EA ; enable interrupts

0065 D2AE 60 SETB EADC ; enable ADC interrupt

0067 43EF01 61 ORL ADCCON1,#001h ; enable hardware CONVST pin

62

63 ; CONTINUE WITH OTHER CODE...

64

006A B2B3 65 AGAIN: CPL LED ; blink (complement) the LED

006C 7401 66 MOV A,#01H ; delay length

006E 120073 67 CALL DELAY ; delay 100ms

0071 80F7 68 JMP AGAIN ; repeat

69

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

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

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

73 ; interrupt service routine.

74

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

76 ; SUBROUTINE

0073 77 DELAY: ; Delays by 100ms \* A

78 ; 100mSec based on 2.097152MHZ

79 ; Core Clock

80 ; i.e. default ADuC814 Clock

81

0073 F9 82 MOV R1,A ; Acc holds delay variable

0074 7A22 83 DLY0: MOV R2,#022h ; Set up delay loop0

0076 7BFF 84 DLY1: MOV R3,#0FFh ; Set up delay loop1

0078 DBFE 85 DJNZ R3,$ ; Dec R3 & Jump here until R3 is 0

007A DAFA 86 DJNZ R2,DLY1 ; Dec R2 & Jump DLY1 until R2 is 0

007C D9F6 87 DJNZ R1,DLY0 ; Dec R1 & Jump DLY0 until R1 is 0

007E 22 88 RET ; Return from subroutine

89

90 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

91

=1 92 $INCLUDE(UARTIO.ASM)

=1 93 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 94 ;

=1 95 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 96 ;

=1 97 ; Date : 12 October 1999

=1 98 ;

=1 99 ; File : UARTIO.hex

=1 100 ;

=1 101 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 102 ;

=1 103 ; Description : standard UART I/O subroutines. total size of this

=1 104 ; code when assembled is 155 bytes. routines for use

=1 105 ; external to this file are:

=1 106 ;

=1 107 ; SENDSTRING - sends a string of characters

=1 108 ; SENDCHAR - sends a single character

=1 109 ; SENDVAL - sends a byte as 2 ASCII characters

=1 110 ; HEX2ASCII - converts from HEX to ASCII

=1 111 ; ASCII2HEX - converts from ASCII to HEX

=1 112 ; GETCHAR - gets a single character

=1 113 ; GETVAL - gets a byte as 2 ASCII characters

=1 114 ;

=1 115 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 116

ADCPIN PAGE 3

=1 117 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 118 ; SENDSTRING

=1 119

007F =1 120 SENDSTRING: ; sends ASCII string to UART starting at location

=1 121 ; DPTR and ending with a null (0) value

=1 122

007F C0E0 =1 123 PUSH ACC

0081 C0F0 =1 124 PUSH B

0083 E4 =1 125 CLR A

0084 F5F0 =1 126 MOV B,A

0086 E5F0 =1 127 IO0010: MOV A,B

0088 05F0 =1 128 INC B

008A 93 =1 129 MOVC A,@A+DPTR

008B 6005 =1 130 JZ IO0020

008D 120097 =1 131 CALL SENDCHAR

0090 80F4 =1 132 JMP IO0010

0092 D0F0 =1 133 IO0020: POP B

0094 D0E0 =1 134 POP ACC

=1 135

0096 22 =1 136 RET

=1 137

=1 138 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 139 ; SENDCHAR

=1 140

0097 =1 141 SENDCHAR: ; sends ASCII value contained in A to UART

=1 142

0097 3099FD =1 143 JNB TI,$ ; wait til present char gone

009A C299 =1 144 CLR TI ; must clear TI

009C F599 =1 145 MOV SBUF,A

=1 146

009E 22 =1 147 RET

=1 148

=1 149 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 150 ; SENDVAL

=1 151

009F =1 152 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 153 ; and then spits these two characters up the UART.

=1 154 ; does not change the value of A.

=1 155

009F C0E0 =1 156 PUSH ACC

00A1 C4 =1 157 SWAP A

00A2 1200B3 =1 158 CALL HEX2ASCII

00A5 1197 =1 159 CALL SENDCHAR ; send high nibble

00A7 D0E0 =1 160 POP ACC

00A9 C0E0 =1 161 PUSH ACC

00AB 1200B3 =1 162 CALL HEX2ASCII

00AE 1197 =1 163 CALL SENDCHAR ; send low nibble

00B0 D0E0 =1 164 POP ACC

=1 165

00B2 22 =1 166 RET

=1 167

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

=1 169 ; HEX2ASCII

=1 170

00B3 =1 171 HEX2ASCII: ; converts A into the hex character representing the

=1 172 ; value of A's least significant nibble

=1 173

00B3 540F =1 174 ANL A,#00Fh

ADCPIN PAGE 4

00B5 B40A00 =1 175 CJNE A,#00Ah,$+3

00B8 4002 =1 176 JC IO0030

00BA 2407 =1 177 ADD A,#007h

00BC 2430 =1 178 IO0030: ADD A,#'0'

=1 179

00BE 22 =1 180 RET

=1 181

=1 182 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 183 ; ASCII2HEX

=1 184

00BF =1 185 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 186 ; into the corresponding number (0-15). returns C=1

=1 187 ; when input is other than an ASCII digit,

=1 188 ; indicating invalid output (returned as 255).

=1 189

00BF C3 =1 190 CLR C

00C0 9430 =1 191 SUBB A,#'0'

00C2 B40A00 =1 192 CJNE A,#10,$+3

00C5 401B =1 193 JC IO0050 ; if '0'<=char<='9', return OK

00C7 B41100 =1 194 CJNE A,#17,$+3

00CA 4013 =1 195 JC IO0040 ; if '9'<char<'A', return FAIL

00CC 9407 =1 196 SUBB A,#7

00CE B41000 =1 197 CJNE A,#10h,$+3

00D1 400F =1 198 JC IO0050 ; if 'A'<=char<='F', return OK

00D3 B42A00 =1 199 CJNE A,#42,$+3

00D6 4007 =1 200 JC IO0040 ; if 'F'<char<'a', return FAIL

00D8 9420 =1 201 SUBB A,#20h

00DA B41000 =1 202 CJNE A,#10h,$+3

00DD 4003 =1 203 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 204

00DF C3 =1 205 IO0040: CLR C ; ..else return FAIL

00E0 74FF =1 206 MOV A,#0FFh

=1 207

00E2 B3 =1 208 IO0050: CPL C

00E3 22 =1 209 RET

=1 210

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

=1 212 ; GETCHAR

=1 213

00E4 =1 214 GETCHAR: ; waits for a single ASCII character to be received

=1 215 ; by the UART. places this character into A.

=1 216

00E4 3098FD =1 217 JNB RI,$

00E7 E599 =1 218 MOV A,SBUF

00E9 C298 =1 219 CLR RI

=1 220

00EB 22 =1 221 RET

=1 222

=1 223 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 224 ; GETVAL

=1 225

00EC =1 226 GETVAL: ; waits for two ASCII hex digits to be received by

=1 227 ; the UART. returns the hex value in A.

=1 228

00EC C0F0 =1 229 PUSH B

00EE C000 =1 230 PUSH 0

00F0 C298 =1 231 IO0060: CLR RI

00F2 11E4 =1 232 CALL GETCHAR ; first nibble

ADCPIN PAGE 5

00F4 F500 =1 233 MOV 0,A ; store received char

00F6 11BF =1 234 CALL ASCII2HEX

00F8 40F6 =1 235 JC IO0060 ; if not '0' thru 'F', don't accept

00FA C4 =1 236 SWAP A ; swap nibbles

00FB F5F0 =1 237 MOV B,A ; store nibble in B

00FD E500 =1 238 MOV A,0 ; echo received char

00FF 1197 =1 239 CALL SENDCHAR

0101 C298 =1 240 IO0070: CLR RI

0103 11E4 =1 241 CALL GETCHAR ; second nibble

0105 F500 =1 242 MOV 0,A ; store received char

0107 11BF =1 243 CALL ASCII2HEX

0109 40F6 =1 244 JC IO0070 ; if not '0' thru 'F', don't accept

010B 45F0 =1 245 ORL A,B ; combine nibbles

010D F5F0 =1 246 MOV B,A ; store results in B

010F E500 =1 247 MOV A,0 ; echo received char

0111 1197 =1 248 CALL SENDCHAR

0113 E5F0 =1 249 MOV A,B ; final result

0115 D000 =1 250 POP 0

0117 D0F0 =1 251 POP B

=1 252

0119 22 =1 253 RET

=1 254

255 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 006AH

ASCII2HEX. . . . . . . . . . . . C ADDR 00BFH

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

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

DELAY. . . . . . . . . . . . . . C ADDR 0073H

DLY0 . . . . . . . . . . . . . . C ADDR 0074H

DLY1 . . . . . . . . . . . . . . C ADDR 0076H

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

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

GETCHAR. . . . . . . . . . . . . C ADDR 00E4H

GETVAL . . . . . . . . . . . . . C ADDR 00ECH NOT USED

HEX2ASCII. . . . . . . . . . . . C ADDR 00B3H

IO0010 . . . . . . . . . . . . . C ADDR 0086H

IO0020 . . . . . . . . . . . . . C ADDR 0092H

IO0030 . . . . . . . . . . . . . C ADDR 00BCH

IO0040 . . . . . . . . . . . . . C ADDR 00DFH

IO0050 . . . . . . . . . . . . . C ADDR 00E2H

IO0060 . . . . . . . . . . . . . C ADDR 00F0H

IO0070 . . . . . . . . . . . . . C ADDR 0101H

LED. . . . . . . . . . . . . . . NUMB 00B3H

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

SENDCHAR . . . . . . . . . . . . C ADDR 0097H

SENDSTRING . . . . . . . . . . . C ADDR 007FH NOT USED

SENDVAL. . . . . . . . . . . . . C ADDR 009FH

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

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

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

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