SPIFRAM PAGE 1

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

2 ;

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

4 ;

5 ; Date : April '01

6 ;

7 ; File : SPIfram.asm

8 ;

9 ; Hardware : ADuC814

10 ;

11 ; Description : sends sample values via SPI to FRAM

12 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

13

14 $MOD814

00B5 15 SS EQU P3.5 ; P3.5 drives slave device's SS pin

16

---- 17 CSEG

18

0050 19 ORG 0050H ; locate code above interrupts

20

0050 75CBFF 21 MOV RCAP2H,#0FFh ; config UART for 9600 baud

0053 75CAF9 22 MOV RCAP2L,#-7 ;

0056 75CDFF 23 MOV TH2,#0FFh

0059 75CCF9 24 MOV TL2,#-7

005C 759852 25 MOV SCON,#52h

005F 75C834 26 MOV T2CON,#34h

27

28

0062 D2B5 29 setb ss

0064 759C01 30 mov cfg814,#01h

0067 75F831 31 mov SPICON,#031H

006A C2B5 32 CLR SS ; Set slave select low

006C 7406 33 MOV A,#00000110B ; write enable

006E F5F7 34 MOV SPIDAT,A ; trigger data transfer

0070 30FFFD 35 JNB ISPI,$

0073 C2FF 36 CLR ISPI

0075 D2B5 37 setb ss ; Complete transmission of op-code

0077 00 38 NOP

0078 00 39 NOP

0079 00 40 NOP

41

007A C2B5 42 clr ss ;

007C 7402 43 MOV A,#00000010b ; write data

007E F5F7 44 MOV SPIDAT,A ; trigger data transfer

0080 30FFFD 45 JNB ISPI,$

0083 C2FF 46 CLR ISPI

47

0085 7400 48 MOV A,#00h

0087 F5F7 49 MOV SPIDAT,A ; trigger data transfer

0089 30FFFD 50 JNB ISPI,$

008C C2FF 51 CLR ISPI

52

008E 7400 53 MOV A,#00h ; start at address 0000h

0090 F5F7 54 MOV SPIDAT,A ; trigger data transfer

0092 30FFFD 55 JNB ISPI,$

0095 C2FF 56 CLR ISPI

57

0097 7800 58 MOV R0,#00H ; clear R0

SPIFRAM PAGE 2

59

0099 E8 60 loop: MOV A,R0 ; send test value

009A F5F7 61 MOV SPIDAT,A ; trigger data transfer

009C 30FFFD 62 JNB ISPI,$

009F C2FF 63 CLR ISPI

00A1 08 64 INC R0

00A2 B814F4 65 CJNE R0,#20,loop ; Loop till 20 values sent

66

67

00A5 D2B5 68 SETB SS ; finish write sequence

69

00A7 00 70 NOP

00A8 00 71 NOP

00A9 00 72 NOP

73

00AA C2B5 74 CLR SS

00AC 7403 75 MOV A,#00000011b ; READ data

00AE F5F7 76 MOV SPIDAT,A ; trigger data transfer

00B0 30FFFD 77 JNB ISPI,$

00B3 C2FF 78 CLR ISPI

79

00B5 7400 80 MOV A,#00h

00B7 F5F7 81 MOV SPIDAT,A ; trigger data transfer

00B9 30FFFD 82 JNB ISPI,$

00BC C2FF 83 CLR ISPI

84

00BE 7400 85 MOV A,#00h ; start at address 0000h

00C0 F5F7 86 MOV SPIDAT,A ; trigger data transfer

00C2 30FFFD 87 JNB ISPI,$

00C5 C2FF 88 CLR ISPI

89

00C7 7400 90 loop2: MOV A,#00h ; generate clocks for reception

00C9 F5F7 91 MOV SPIDAT,A

00CB 30FFFD 92 JNB ISPI,$

00CE C2FF 93 CLR ISPI

94

00D0 E5F7 95 mov A,SPIDAT

00D2 1200FA 96 CALL SENDVAL

00D5 B413EF 97 CJNE A,#19,loop2

00D8 80FE 98 exit: jmp $

99

=1 100 $INCLUDE(UARTIO.ASM)

=1 101 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 102 ;

=1 103 ; Author : ADI - Apps www.analog.com/MicroConverter

=1 104 ;

=1 105 ; Date : 12 October 1999

=1 106 ;

=1 107 ; File : UARTIO.hex

=1 108 ;

=1 109 ; Hardware : any 8051 based microcontroller or MicroConverter

=1 110 ;

=1 111 ; Description : standard UART I/O subroutines. total size of this

=1 112 ; code when assembled is 155 bytes. routines for use

=1 113 ; external to this file are:

=1 114 ;

=1 115 ; SENDSTRING - sends a string of characters

=1 116 ; SENDCHAR - sends a single character

SPIFRAM PAGE 3

=1 117 ; SENDVAL - sends a byte as 2 ASCII characters

=1 118 ; HEX2ASCII - converts from HEX to ASCII

=1 119 ; ASCII2HEX - converts from ASCII to HEX

=1 120 ; GETCHAR - gets a single character

=1 121 ; GETVAL - gets a byte as 2 ASCII characters

=1 122 ;

=1 123 ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=1 124

=1 125 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 126 ; SENDSTRING

=1 127

00DA =1 128 SENDSTRING: ; sends ASCII string to UART starting at location

=1 129 ; DPTR and ending with a null (0) value

=1 130

00DA C0E0 =1 131 PUSH ACC

00DC C0F0 =1 132 PUSH B

00DE E4 =1 133 CLR A

00DF F5F0 =1 134 MOV B,A

00E1 E5F0 =1 135 IO0010: MOV A,B

00E3 05F0 =1 136 INC B

00E5 93 =1 137 MOVC A,@A+DPTR

00E6 6005 =1 138 JZ IO0020

00E8 1200F2 =1 139 CALL SENDCHAR

00EB 80F4 =1 140 JMP IO0010

00ED D0F0 =1 141 IO0020: POP B

00EF D0E0 =1 142 POP ACC

=1 143

00F1 22 =1 144 RET

=1 145

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

=1 147 ; SENDCHAR

=1 148

00F2 =1 149 SENDCHAR: ; sends ASCII value contained in A to UART

=1 150

00F2 3099FD =1 151 JNB TI,$ ; wait til present char gone

00F5 C299 =1 152 CLR TI ; must clear TI

00F7 F599 =1 153 MOV SBUF,A

=1 154

00F9 22 =1 155 RET

=1 156

=1 157 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 158 ; SENDVAL

=1 159

00FA =1 160 SENDVAL: ; converts the hex value of A into two ASCII chars,

=1 161 ; and then spits these two characters up the UART.

=1 162 ; does not change the value of A.

=1 163

00FA C0E0 =1 164 PUSH ACC

00FC C4 =1 165 SWAP A

00FD 12010E =1 166 CALL HEX2ASCII

0100 11F2 =1 167 CALL SENDCHAR ; send high nibble

0102 D0E0 =1 168 POP ACC

0104 C0E0 =1 169 PUSH ACC

0106 12010E =1 170 CALL HEX2ASCII

0109 11F2 =1 171 CALL SENDCHAR ; send low nibble

010B D0E0 =1 172 POP ACC

=1 173

010D 22 =1 174 RET

SPIFRAM PAGE 4

=1 175

=1 176 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 177 ; HEX2ASCII

=1 178

010E =1 179 HEX2ASCII: ; converts A into the hex character representing the

=1 180 ; value of A's least significant nibble

=1 181

010E 540F =1 182 ANL A,#00Fh

0110 B40A00 =1 183 CJNE A,#00Ah,$+3

0113 4002 =1 184 JC IO0030

0115 2407 =1 185 ADD A,#007h

0117 2430 =1 186 IO0030: ADD A,#'0'

=1 187

0119 22 =1 188 RET

=1 189

=1 190 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 191 ; ASCII2HEX

=1 192

011A =1 193 ASCII2HEX: ; converts A from an ASCII digit ('0'-'9' or 'A'-'F')

=1 194 ; into the corresponding number (0-15). returns C=1

=1 195 ; when input is other than an ASCII digit,

=1 196 ; indicating invalid output (returned as 255).

=1 197

011A C3 =1 198 CLR C

011B 9430 =1 199 SUBB A,#'0'

011D B40A00 =1 200 CJNE A,#10,$+3

0120 401B =1 201 JC IO0050 ; if '0'<=char<='9', return OK

0122 B41100 =1 202 CJNE A,#17,$+3

0125 4013 =1 203 JC IO0040 ; if '9'<char<'A', return FAIL

0127 9407 =1 204 SUBB A,#7

0129 B41000 =1 205 CJNE A,#10h,$+3

012C 400F =1 206 JC IO0050 ; if 'A'<=char<='F', return OK

012E B42A00 =1 207 CJNE A,#42,$+3

0131 4007 =1 208 JC IO0040 ; if 'F'<char<'a', return FAIL

0133 9420 =1 209 SUBB A,#20h

0135 B41000 =1 210 CJNE A,#10h,$+3

0138 4003 =1 211 JC IO0050 ; if 'a'<=char<='f', return OK..

=1 212

013A C3 =1 213 IO0040: CLR C ; ..else return FAIL

013B 74FF =1 214 MOV A,#0FFh

=1 215

013D B3 =1 216 IO0050: CPL C

013E 22 =1 217 RET

=1 218

=1 219 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 220 ; GETCHAR

=1 221

013F =1 222 GETCHAR: ; waits for a single ASCII character to be received

=1 223 ; by the UART. places this character into A.

=1 224

013F 3098FD =1 225 JNB RI,$

0142 E599 =1 226 MOV A,SBUF

0144 C298 =1 227 CLR RI

=1 228

0146 22 =1 229 RET

=1 230

=1 231 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

=1 232 ; GETVAL

SPIFRAM PAGE 5

=1 233

0147 =1 234 GETVAL: ; waits for two ASCII hex digits to be received by

=1 235 ; the UART. returns the hex value in A.

=1 236

0147 C0F0 =1 237 PUSH B

0149 C000 =1 238 PUSH 0

014B C298 =1 239 IO0060: CLR RI

014D 313F =1 240 CALL GETCHAR ; first nibble

014F F500 =1 241 MOV 0,A ; store received char

0151 311A =1 242 CALL ASCII2HEX

0153 40F6 =1 243 JC IO0060 ; if not '0' thru 'F', don't accept

0155 C4 =1 244 SWAP A ; swap nibbles

0156 F5F0 =1 245 MOV B,A ; store nibble in B

0158 E500 =1 246 MOV A,0 ; echo received char

015A 11F2 =1 247 CALL SENDCHAR

015C C298 =1 248 IO0070: CLR RI

015E 313F =1 249 CALL GETCHAR ; second nibble

0160 F500 =1 250 MOV 0,A ; store received char

0162 311A =1 251 CALL ASCII2HEX

0164 40F6 =1 252 JC IO0070 ; if not '0' thru 'F', don't accept

0166 45F0 =1 253 ORL A,B ; combine nibbles

0168 F5F0 =1 254 MOV B,A ; store results in B

016A E500 =1 255 MOV A,0 ; echo received char

016C 11F2 =1 256 CALL SENDCHAR

016E E5F0 =1 257 MOV A,B ; final result

0170 D000 =1 258 POP 0

0172 D0F0 =1 259 POP B

=1 260

0174 22 =1 261 RET

=1 262

263 end

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

SPIFRAM PAGE 6

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

ASCII2HEX. . . . . . . . . . . . C ADDR 011AH

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

CFG814 . . . . . . . . . . . . . D ADDR 009CH PREDEFINED

EXIT . . . . . . . . . . . . . . C ADDR 00D8H NOT USED

GETCHAR. . . . . . . . . . . . . C ADDR 013FH

GETVAL . . . . . . . . . . . . . C ADDR 0147H NOT USED

HEX2ASCII. . . . . . . . . . . . C ADDR 010EH

IO0010 . . . . . . . . . . . . . C ADDR 00E1H

IO0020 . . . . . . . . . . . . . C ADDR 00EDH

IO0030 . . . . . . . . . . . . . C ADDR 0117H

IO0040 . . . . . . . . . . . . . C ADDR 013AH

IO0050 . . . . . . . . . . . . . C ADDR 013DH

IO0060 . . . . . . . . . . . . . C ADDR 014BH

IO0070 . . . . . . . . . . . . . C ADDR 015CH

ISPI . . . . . . . . . . . . . . B ADDR 00FFH PREDEFINED

LOOP . . . . . . . . . . . . . . C ADDR 0099H

LOOP2. . . . . . . . . . . . . . C ADDR 00C7H

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 00F2H

SENDSTRING . . . . . . . . . . . C ADDR 00DAH NOT USED

SENDVAL. . . . . . . . . . . . . C ADDR 00FAH

SPICON . . . . . . . . . . . . . D ADDR 00F8H PREDEFINED

SPIDAT . . . . . . . . . . . . . D ADDR 00F7H PREDEFINED

SS . . . . . . . . . . . . . . . NUMB 00B5H

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

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

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

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