I2CSLAVE PAGE 1

1 ;======================================================================

2 ;

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

4 ;

5 ; Date : Oct 2000

6 ;

7 ; File : i2cslave.asm

8 ;

9 ; Hardware : ADuC812 (commented out = ADuC816/ADuC824)

10 ;

11 ; Description : Code for a slave in an I2C system. This code will

12 ; continuously receive and transmit a byte over the I2C

13 ; interface, then send the received byte out the UART,

14 ; then check if a character had been entered in the UART.

15 ; If so, it will send the ASCII value of the character

16 ; entered to the slave, the next time it transmits a byte.

17 ;

18 ; Reference : Tech Note, uC001: "MicroConverter I2C Compatible

19 ; Interface" find it at www.analog.com/microconverter

20 ;

21 ;======================================================================

22

23 $MOD812 ; use ADuC812 & 8052 predefined symbols

24 ;$MOD816

25 ;$MOD824

26

27

28 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

29 ; DEFINE VARIABLES IN INTERNAL RAM

30

0030 31 BYTECNT DATA 30h ; byte counter for I2C routines

0031 32 INPUT DATA 31h ; data recieved from master

0032 33 OUTPUT DATA 32h ; data to be transmitted to master

34

0000 35 GO BIT 00h ; flag for all the interrupts

0001 36 FIRST BIT 01h ; flag for recieve mode interrupt

0002 37 TR BIT 02h ; flag for transmit mode interrupt

38

00B4 39 LED EQU P3.4 ; P3.4 drives the LED on eval board

40

41 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

42 ; BEGINNING OF CODE

---- 43 CSEG

0000 44 ORG 0000h

0000 020060 45 JMP MAIN

46 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

47 ; INT0 ISR

0003 48 ORG 0003h

0003 0532 49 INC OUTPUT

0005 32 50 RETI

51 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

52 ; I2C ISR

003B 53 ORG 003Bh

54

003B 20E90D 55 JB I2CTX,TRANSMIT ; if slave-transmitter get data ready

56

003E 57 RECEIVE:

003E C202 58 CLR TR ; FLAG to indicate that this time

I2CSLAVE PAGE 2

59 ; we were receiving (not trans'ting)

0040 20010F 60 JB FIRST, ENDINT1 ; no need to store the address

0043 D200 61 SETB GO ; reception complete

0045 859A31 62 MOV INPUT, I2CDAT ; store data recieved in INPUT

0048 020052 63 JMP ENDINT1 ; ERROR => end interrupt

64

004B 65 TRANSMIT:

004B D202 66 SETB TR ; FLAG to indicate that this time

67 ; we were transmitting (not receiving)

004D D200 68 SETB GO ; transmission complete

004F 85329A 69 MOV I2CDAT,OUTPUT ; move data to be transmitted into I2CDAT

70 ; JMP ENDINT2 ; Note: On the ADuC824/816 the read or

71 ; write of I2CDAT register

72 ; automatically clears i2ci. If

73 ; I2CI is cleared twice then the

74 ; microconverter will hang.)

75

0052 76 ENDINT1:

0052 C2E8 77 CLR I2CI ; clear I2C interrupt bit (812 only)

0054 78 ENDINT2:

0054 C201 79 CLR FIRST ; address has already been recieved

80

0056 32 81 RETI

82

83

84 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

85 ; MAIN PROGRAM

0060 86 ORG 0060h

0060 87 MAIN:

88

89 ; configure the UART ADuC812

0060 759852 90 MOV SCON,#52h ; configure UART for 9600baud..

0063 758920 91 MOV TMOD,#20h ; ..assuming 11.0592MHz crystal

0066 758DFD 92 MOV TH1,#-3

0069 D28E 93 SETB TR1

94

95 ; configure the UART ADuC824/ADuC816

96 ; MOV RCAP2H,#0FFh ; config UART for 9830baud

97 ; MOV RCAP2L,#-5 ; (close enough to 9600baud)

98 ; MOV TH2,#0FFh

99 ; MOV TL2,#-5

100 ; MOV SCON,#52h

101 ; MOV T2CON,#34h

102

103 ;configure and enable interrupts

006B 75A901 104 MOV IE2,#01h ; enable I2C interrupt

105 ; MOV IEIP2,#01h ; enable I2C interrupt

006E D2A8 106 SETB EX0 ; enable INT0

0070 D288 107 SETB IT0 ; INT0 edge triggered

0072 D2AF 108 SETB EA ; allow all the interrupts

109

110

111 ;initialise settings

0074 759B44 112 MOV I2CADD,#044h ; slave address is 44h

0077 75E800 113 MOV I2CCON,#00h ; slave mode

007A C200 114 CLR GO ; clear flag used in the interrupt

115

007C 753200 116 MOV OUTPUT,#0 ; TX 0 as default

I2CSLAVE PAGE 3

007F D2B4 117 SETB LED

118

0081 119 RESET:

0081 D201 120 SETB FIRST ; first byte recieved will be the

121 ; address => no need to store it.

0083 3000FD 122 JNB GO,$ ; wait for the interrupt. If it is in

123 ; recieve mode, it will wait here for

124 ; a second interrupt, when it

125 ; recieves the data bit

0086 C200 126 CLR GO ; flag cleared for the next interrupt

0088 2002F6 127 JB TR,RESET ; if the slave has just transmitted,

128 ; wait for another interrupt. If it has

129 ; recieved a data byte send it out the UART

130

008B 131 SENDUART:

008B B2B4 132 CPL LED ; LED changes each time one byte has been

133 ; recieved and another transmitted

134

008D E531 135 MOV A,INPUT ; send value recieved out the UART

008F 1200BA 136 CALL SENDVAL

0092 740A 137 MOV A,#10

0094 1200A6 138 CALL SENDCHAR ; send LF + CR

0097 740D 139 MOV A,#13

0099 1200A6 140 CALL SENDCHAR

141

009C 3098E2 142 JNB RI, RESET ; repeat (unless UART data received)

143

144

145 ; WHEN UART DATA RECEIVED, MOVE DATA TO I2C OUTPUT...

146

009F 859932 147 MOV OUTPUT, SBUF ; update OUTPUT byte to new value

00A2 C298 148 CLR RI ; must clear RI

00A4 80DB 149 JMP RESET ; back to main loop

150

151

152

153 ;======================================================================

154 ; SUBROUTINES

155 ;======================================================================

156

157 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

158 ; SENDCHAR

159 ; sends ASCII value contained in A to UART

160

00A6 161 SENDCHAR:

162

00A6 3099FD 163 JNB TI,$ ; wait til present char gone

00A9 C299 164 CLR TI ; must clear TI

00AB F599 165 MOV SBUF,A

00AD 22 166 RET

167

168

169 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

170 ; HEX2ASCII

171 ; converts A into the hex character representing the value of A's

172 ; least significant nibble

173

00AE 174 HEX2ASCII:

I2CSLAVE PAGE 4

175

00AE 540F 176 ANL A,#00Fh

00B0 B40A00 177 CJNE A,#00Ah,$+3

00B3 4002 178 JC IO0030

00B5 2407 179 ADD A,#007h

00B7 2430 180 IO0030: ADD A,#'0'

00B9 22 181 RET

182

183

184 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

185 ; SENDVAL

186 ; converts the hex value of A into two ASCII chars, and then spits

187 ; these two characters up the UART. does not change the value of A.

188

00BA 189 SENDVAL:

00BA C0E0 190 PUSH ACC

00BC C4 191 SWAP A

00BD 11AE 192 CALL HEX2ASCII

00BF 11A6 193 CALL SENDCHAR ; send high nibble

00C1 D0E0 194 POP ACC

00C3 C0E0 195 PUSH ACC

00C5 11AE 196 CALL HEX2ASCII

00C7 11A6 197 CALL SENDCHAR ; send low nibble

00C9 D0E0 198 POP ACC

199

00CB 22 200 RET

201 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

202

203 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

I2CSLAVE PAGE 5

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

BYTECNT. . . . . . . . . . . . . D ADDR 0030H NOT USED

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

ENDINT1. . . . . . . . . . . . . C ADDR 0052H

ENDINT2. . . . . . . . . . . . . C ADDR 0054H NOT USED

EX0. . . . . . . . . . . . . . . B ADDR 00A8H PREDEFINED

FIRST. . . . . . . . . . . . . . B ADDR 0001H

GO . . . . . . . . . . . . . . . B ADDR 0000H

HEX2ASCII. . . . . . . . . . . . C ADDR 00AEH

I2CADD . . . . . . . . . . . . . D ADDR 009BH PREDEFINED

I2CCON . . . . . . . . . . . . . D ADDR 00E8H PREDEFINED

I2CDAT . . . . . . . . . . . . . D ADDR 009AH PREDEFINED

I2CI . . . . . . . . . . . . . . B ADDR 00E8H PREDEFINED

I2CTX. . . . . . . . . . . . . . B ADDR 00E9H PREDEFINED

IE2. . . . . . . . . . . . . . . D ADDR 00A9H PREDEFINED

INPUT. . . . . . . . . . . . . . D ADDR 0031H

IO0030 . . . . . . . . . . . . . C ADDR 00B7H

IT0. . . . . . . . . . . . . . . B ADDR 0088H PREDEFINED

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

MAIN . . . . . . . . . . . . . . C ADDR 0060H

OUTPUT . . . . . . . . . . . . . D ADDR 0032H

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

RECEIVE. . . . . . . . . . . . . C ADDR 003EH NOT USED

RESET. . . . . . . . . . . . . . C ADDR 0081H

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

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

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

SENDCHAR . . . . . . . . . . . . C ADDR 00A6H

SENDUART . . . . . . . . . . . . C ADDR 008BH NOT USED

SENDVAL. . . . . . . . . . . . . C ADDR 00BAH

TH1. . . . . . . . . . . . . . . D ADDR 008DH PREDEFINED

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

TMOD . . . . . . . . . . . . . . D ADDR 0089H PREDEFINED

TR . . . . . . . . . . . . . . . B ADDR 0002H

TR1. . . . . . . . . . . . . . . B ADDR 008EH PREDEFINED

TRANSMIT . . . . . . . . . . . . C ADDR 004BH