MASTUART PAGE 1

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

2 ;

3 ; Author : ADI - Apps

4 ;

5 ; Date : November 2001

6 ;

7 ; File : MASTuart.asm

8 ;

9 ; Hardware ; ADuC836

10 ;

11 ; Description : This Program transmits the numbers 1-10 in binary

12 ; form continuously down the SPI serial port.

13 ; After the transmission of each byte the incoming

14 ; byte is saved in order between internal RAM

15 ; addresses 40h and 50h.

16 ;

17 ; After the 16 bytes have been writen into memory

18 ; the program outputs the received data up the UART

19 ; where it can be viewed using Hyperterminal.

20 ;

21 ; An SPI slave program can be run on a second ADuC836

22 ; to communicate with this master code.

23 ; The Slave program (SLAVuart.asm in the SPI\SLAVE

24 ; directory) should be started after the master

25 ; program (MASTuart.asm) but within the time delay

26 ; of 5s in order that the slave program is

27 ; synchronised by the first outputted clock of the

28 ; master.

29 ;

30 ; The clock is outputted at sclock (pin 26)

31 ; The data is outputted at sdata/MOSI (pin 27)

32 ; The data is inputted at MISO (pin 14)

33 ;====================================================================

34 ;

35 $MOD836 ;Use 8052 predefined Symbols

36

00B4 37 LED EQU P3.4

0000 38 FLAG BIT 00H

39

40 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

41 ; BEGINNING OF CODE

---- 42 CSEG

0000 43 ORG 0000H

44

0000 020060 45 JMP MAIN

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

47 ; SPI INTERRUPT ROUTINE

003B 48 ORG 003BH

003B D2B5 49 SETB P3.5 ; set the SS bit after transmission

003D C200 50 CLR FLAG ; Clear flag to leave loop

51

003F A7F7 52 MOV @R1, SPIDAT ; move input into memory

0041 09 53 INC R1 ; increment memory location so new

54 ; data is stored in new address

55

0042 B95003 56 CJNE R1, #50H, CONT ; reset memory location to 40h when

57 ; memory location reaches 50h saving

58 ; 16 bytes of data

MASTUART PAGE 2

59

0045 12008C 60 CALL SNDUART ; send the data up the UART

0048 32 61 CONT: RETI

62

63 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

64 ; MAIN PROGRAM

65

0060 66 ORG 0060H ; Start code at address above interrupts

67

0060 68 MAIN: ; Main program

69

0060 759E82 70 MOV T3CON,#82h

0063 759D12 71 MOV T3FD,#12h

0066 759852 72 MOV SCON,#52h

73

0069 75F837 74 MOV SPICON,#037h ; Initialise SPICON to have

75 ; -bitrate=fosc/64

76 ; -CPHA=1

77 ; -CPOL=0, sclk idling low

78 ; -master mode select

79 ; -Enable SPI serial port

80

006C 75A901 81 MOV IEIP2, #01h ; Enable SPI interrrupt

006F D2AF 82 SETB EA ; Enable interrupts

83

0071 7940 84 MOV R1, #40h ; initialise R1 to 40 to store the

85 ; input data from memory location 40

0073 7800 86 MOV R0, #00H ; initialise R0 to 0 to start

87 ; transmissions from 1

88

89 ; Delay the output of data by 5.0s in order that the slave program

90 ; can be easily synchronised with the master program.

91

0075 7432 92 MOV A, #50

0077 1200F8 93 CALL DELAY

94

007A 95 TRNSMT:

007A 08 96 INC R0

007B C2B5 97 CLR P3.5 ; clear the SS bit during transmission

007D 88F7 98 MOV SPIDAT, R0 ; transmit the current value on R0

007F D200 99 SETB FLAG ; set flag so that we wait here until

100 ; the spi interrupt routine clears

101 ; the FLAG

102

0081 2000FD 103 JB FLAG, $ ; stay here until flag is cleared

104 ; by interrupt

105

106 ; check if R0 is equal to 10. If so the number 10 has been

107 ; transmitted and we should reset R0 to 0 to start transmission

108 ; from 1 again

109

0084 E8 110 MOV A, R0

0085 B40AF2 111 CJNE A, #0AH, TRNSMT ; if R0 is not 10, jump to TRNSMT

0088 7800 112 MOV R0, #00H ; if R0=10 make R0=0 & jump to TRNSMT

008A 80EE 113 JMP TRNSMT

114

115 ; Transmit the values in locations 40h->50h up the UART wait for

116 ; 5 seconds and then transmit and receive values to/from the slave

MASTUART PAGE 3

117 ; again down the SPI port.

118

008C 119 SNDUART:

008C B2B4 120 CPL LED ;CPL LED with each transmission

008E 900104 121 MOV DPTR, #TITLE

0091 1200B8 122 CALL SENDSTRING ; write title block on screen

123

0094 7940 124 MOV R1, #40h ; move value at address 40 into R2

0096 E7 125 MOV A, @R1

0097 FA 126 MOV R2, A

127

0098 128 NEXT: ; Put new value on a new line

0098 740A 129 MOV A, #10 ; Transmit a linefeed (= ASCII 10)

009A 1200D0 130 CALL SENDCHAR

009D 740D 131 MOV A, #13 ;Transmit a carriage return (=ASCII 13)

009F 1200D0 132 CALL SENDCHAR

133

00A2 EA 134 MOV A, R2 ; Transmit R2 i.e. value @ address R1

00A3 1200D8 135 CALL SENDVAL

00A6 09 136 INC R1 ; Increment address

00A7 E7 137 MOV A, @R1

00A8 FA 138 MOV R2, A ; R2 holds the value @ addrR1

139

00A9 E9 140 MOV A, R1 ; Check if at address 50h

00AA B450EB 141 CJNE A, #50h, NEXT ; if not jump to Next

00AD 0200B0 142 JMP WAIT5S ; if so wait 5s and repeat

143

00B0 7432 144 WAIT5S: MOV A, #50 ; wait 5s before sending down the

145 ; SPI port again for ease of viewing

146 ; on screen and to allow the slave

147 ; synchronise itself with the master

00B2 1200F8 148 CALL DELAY

00B5 7940 149 MOV R1, #40h ; store new inputs at address 40h again

00B7 32 150 RETI

151

152 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

153 ; SENDSTRING

154

00B8 155 SENDSTRING: ; sends ASCII string to UART starting at location

156 ; DPTR and ending with a null (0) value

157

00B8 C0E0 158 PUSH ACC

00BA C0F0 159 PUSH B

00BC E4 160 CLR A

00BD F5F0 161 MOV B,A

00BF E5F0 162 IO0010: MOV A,B

00C1 05F0 163 INC B

00C3 93 164 MOVC A,@A+DPTR

00C4 6005 165 JZ IO0020

00C6 1200D0 166 CALL SENDCHAR

00C9 80F4 167 JMP IO0010

00CB D0F0 168 IO0020: POP B

00CD D0E0 169 POP ACC

170

00CF 22 171 RET

172

173 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

174 ; SENDCHAR

MASTUART PAGE 4

175

00D0 176 SENDCHAR: ; sends ASCII value contained in A to UART

177

00D0 3099FD 178 JNB TI,$ ; wait til present char gone

00D3 C299 179 CLR TI ; must clear TI

00D5 F599 180 MOV SBUF,A

181

00D7 22 182 RET

183

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

185 ; SENDVAL

186

00D8 187 SENDVAL: ; converts the hex value of A into two ASCII chars,

188 ; and then spits these two characters up the UART.

189 ; does not change the value of A.

190

00D8 C0E0 191 PUSH ACC

00DA C4 192 SWAP A

00DB 1200EC 193 CALL HEX2ASCII

00DE 11D0 194 CALL SENDCHAR ; send high nibble

00E0 D0E0 195 POP ACC

00E2 C0E0 196 PUSH ACC

00E4 1200EC 197 CALL HEX2ASCII

00E7 11D0 198 CALL SENDCHAR ; send low nibble

00E9 D0E0 199 POP ACC

200

00EB 22 201 RET

202

203

204 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

205 ; HEX2ASCII

206

00EC 207 HEX2ASCII: ; converts A into the hex character representing the

208 ; value of A's least significant nibble

209

00EC 540F 210 ANL A,#00Fh

00EE B40A00 211 CJNE A,#00Ah,$+3

00F1 4002 212 JC IO0030

00F3 2407 213 ADD A,#007h

00F5 2430 214 IO0030: ADD A,#'0'

215

00F7 22 216 RET

217

218 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

219 ; DELAY

220

00F8 221 DELAY: ; Delays by 100ms \* A

222 ; 100mSec based on 1.573MHZ Core Clock

223

224

00F8 FA 225 MOV R2,A ; Acc holds delay variable

00F9 7B32 226 DLY0: MOV R3,#50 ; Set up delay loop0

00FB 7C83 227 DLY1: MOV R4,#131 ; Set up delay loop1

00FD DCFE 228 DJNZ R4,$ ; Dec R4 & Jump here until R4 is 0

229 ; wait here for 131\*15.3us=2ms

00FF DBFA 230 DJNZ R3,DLY1 ; Dec R3 & Jump DLY1 until R3 is 0

231 ; Wait for 50\*2ms

0101 DAF6 232 DJNZ R2,DLY0 ; Dec R2 & Jump DLY0 until R2 is 0

MASTUART PAGE 5

233 ; wait for ACC\*100ms

0103 22 234 RET ; Return from subroutine

235

236

237 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

238

0104 0A0A0D5F 239 TITLE: DB 10,10,13,'\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_',10,13

0108 5F5F5F5F

010C 5F5F5F5F

0110 5F5F5F5F

0114 5F5F5F5F

0118 5F5F5F5F

011C 5F5F5F5F

0120 5F5F5F5F

0124 5F5F5F5F

0128 5F5F5F0A

012C 0D

012D 416E616C 240 DB 'Analog Devices MicroConverter ADuC836',10,13

0131 6F672044

0135 65766963

0139 6573204D

013D 6963726F

0141 436F6E76

0145 65727465

0149 72204144

014D 75433833

0151 360A0D

0154 20202020 241 DB ' SPI MASTER Demo Routine',10,13

0158 20205350

015C 49204D41

0160 53544552

0164 2044656D

0168 6F20526F

016C 7574696E

0170 650A0D

0173 20204461 242 DB ' Data Stored in Memory in Hex Form',10,13,0

0177 74612053

017B 746F7265

017F 6420696E

0183 204D656D

0187 6F727920

018B 696E2048

018F 65782046

0193 6F726D0A

0197 0D00

243

244

245

246

247 END

VERSION 1.2h ASSEMBLY COMPLETE, 0 ERRORS FOUND

MASTUART PAGE 6

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

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

CONT . . . . . . . . . . . . . . C ADDR 0048H

DELAY. . . . . . . . . . . . . . C ADDR 00F8H

DLY0 . . . . . . . . . . . . . . C ADDR 00F9H

DLY1 . . . . . . . . . . . . . . C ADDR 00FBH

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

FLAG . . . . . . . . . . . . . . B ADDR 0000H

HEX2ASCII. . . . . . . . . . . . C ADDR 00ECH

IEIP2. . . . . . . . . . . . . . D ADDR 00A9H PREDEFINED

IO0010 . . . . . . . . . . . . . C ADDR 00BFH

IO0020 . . . . . . . . . . . . . C ADDR 00CBH

IO0030 . . . . . . . . . . . . . C ADDR 00F5H

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

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

NEXT . . . . . . . . . . . . . . C ADDR 0098H

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

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

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

SENDCHAR . . . . . . . . . . . . C ADDR 00D0H

SENDSTRING . . . . . . . . . . . C ADDR 00B8H

SENDVAL. . . . . . . . . . . . . C ADDR 00D8H

SNDUART. . . . . . . . . . . . . C ADDR 008CH

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

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

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

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

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

TITLE. . . . . . . . . . . . . . C ADDR 0104H

TRNSMT . . . . . . . . . . . . . C ADDR 007AH

WAIT5S . . . . . . . . . . . . . C ADDR 00B0H