MASTUART PAGE 1

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

2 ;

3 ; Author : ADI - Apps

4 ;

5 ; Date : January 2001

6 ;

7 ; File : MASTuart.asm

8 ;

9 ; Hardware ; ADuC816

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 device

22 ; (ADuC816/ADuC824/ADuC812/other) to communicate

23 ; with this master code.

24 ;

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

26 ; directory) should be started after the master

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

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

29 ; synchronised by the first outputted clock of the

30 ; master.

31 ;

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

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

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

35 ;====================================================================

36 ;

37 $MOD816 ; Use 8052/ADuC816 predefined Symbols

38

00B4 39 LED EQU P3.4

0000 40 FLAG BIT 00H

41

42 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

43 ; BEGINNING OF CODE

---- 44 CSEG

0000 45 ORG 0000H

46

0000 020060 47 JMP MAIN

48 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

49 ; SPI INTERRUPT ROUTINE

003B 50 ORG 003BH

003B D2B5 51 SETB P3.5 ; set the SS bit after transmission

003D C200 52 CLR FLAG ; Clear flag to leave loop

53

003F A7F7 54 MOV @R1, SPIDAT ; move input into memory

0041 09 55 INC R1 ; increment memory location so new

56 ; data is stored in new address

57

0042 B95003 58 CJNE R1, #50H, CONT ; reset memory location to 40h when

MASTUART PAGE 2

59 ; memory location reaches 50h saving

60 ; 16 bytes of data

61

0045 120095 62 CALL SNDUART ; send the data up the UART

0048 32 63 CONT: RETI

64

65 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

66 ; MAIN PROGRAM

67

0060 68 ORG 0060H ; Start code at address above interrupts

69

0060 70 MAIN: ; Main program

71

0060 75CBFF 72 MOV RCAP2H,#0FFh ; config UART for 9830baud

0063 75CAFB 73 MOV RCAP2L,#-5 ; (close enough to 9600baud)

0066 75CDFF 74 MOV TH2,#0FFh

0069 75CCFB 75 MOV TL2,#-5

006C 759852 76 MOV SCON,#52h

006F 75C834 77 MOV T2CON,#34h

78

0072 75F837 79 MOV SPICON,#037h ; Initialise SPICON to have

80 ; -bitrate=fosc/64

81 ; -CPHA=1

82 ; -CPOL=0, sclk idling low

83 ; -master mode select

84 ; -Enable SPI serial port

85

0075 75A901 86 MOV IEIP2, #01h ; Enable SPI interrrupt

0078 D2AF 87 SETB EA ; Enable interrupts

88

007A 7940 89 MOV R1, #40h ; initialise R1 to 40 to store the

90 ; input data from memory location 40

007C 7800 91 MOV R0, #00H ; initialise R0 to 0 to start

92 ; transmissions from 1

93

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

95 ; can be easily synchronised with the master program.

96

007E 7432 97 MOV A, #50

0080 120101 98 CALL DELAY

99

0083 100 TRNSMT:

0083 08 101 INC R0

0084 C2B5 102 CLR P3.5 ; clear the SS bit during transmission

0086 88F7 103 MOV SPIDAT, R0 ; transmit the current value on R0

0088 D200 104 SETB FLAG ; set flag so that we wait here until

105 ; the spi interrupt routine clears

106 ; the FLAG

107

008A 2000FD 108 JB FLAG, $ ; stay here until flag is cleared

109 ; by interrupt

110

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

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

113 ; from 1 again

114

008D E8 115 MOV A, R0

008E B40AF2 116 CJNE A, #0AH, TRNSMT ; if R0 is not 10, jump to TRNSMT

MASTUART PAGE 3

0091 7800 117 MOV R0, #00H ; if R0=10 make R0=0 & jump to TRNSMT

0093 80EE 118 JMP TRNSMT

119

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

121 ; 5 seconds and then transmit and receive values to/from the slave

122 ; again down the SPI port.

123

0095 124 SNDUART:

0095 B2B4 125 CPL LED ;CPL LED with each transmission

0097 90010D 126 MOV DPTR, #TITLE

009A 1200C1 127 CALL SENDSTRING ; write title block on screen

128

009D 7940 129 MOV R1, #40h ; move value at address 40 into R2

009F E7 130 MOV A, @R1

00A0 FA 131 MOV R2, A

132

00A1 133 NEXT: ; Put new value on a new line

00A1 740A 134 MOV A, #10 ; Transmit a linefeed (= ASCII 10)

00A3 1200D9 135 CALL SENDCHAR

00A6 740D 136 MOV A, #13 ;Transmit a carriage return (=ASCII 13)

00A8 1200D9 137 CALL SENDCHAR

138

00AB EA 139 MOV A, R2 ; Transmit R2 i.e. value @ address R1

00AC 1200E1 140 CALL SENDVAL

00AF 09 141 INC R1 ; Increment address

00B0 E7 142 MOV A, @R1

00B1 FA 143 MOV R2, A ; R2 holds the value @ addrR1

144

00B2 E9 145 MOV A, R1 ; Check if at address 50h

00B3 B450EB 146 CJNE A, #50h, NEXT ; if not jump to Next

00B6 0200B9 147 JMP WAIT5S ; if so wait 5s and repeat

148

00B9 7432 149 WAIT5S: MOV A, #50 ; wait 5s before sending down the

150 ; SPI port again for ease of viewing

151 ; on screen and to allow the slave

152 ; synchronise itself with the master

00BB 120101 153 CALL DELAY

00BE 7940 154 MOV R1, #40h ; store new inputs at address 40h again

00C0 32 155 RETI

156

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

158 ; SENDSTRING

159

00C1 160 SENDSTRING: ; sends ASCII string to UART starting at location

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

162

00C1 C0E0 163 PUSH ACC

00C3 C0F0 164 PUSH B

00C5 E4 165 CLR A

00C6 F5F0 166 MOV B,A

00C8 E5F0 167 IO0010: MOV A,B

00CA 05F0 168 INC B

00CC 93 169 MOVC A,@A+DPTR

00CD 6005 170 JZ IO0020

00CF 1200D9 171 CALL SENDCHAR

00D2 80F4 172 JMP IO0010

00D4 D0F0 173 IO0020: POP B

00D6 D0E0 174 POP ACC

MASTUART PAGE 4

175

00D8 22 176 RET

177

178 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

179 ; SENDCHAR

180

00D9 181 SENDCHAR: ; sends ASCII value contained in A to UART

182

00D9 3099FD 183 JNB TI,$ ; wait til present char gone

00DC C299 184 CLR TI ; must clear TI

00DE F599 185 MOV SBUF,A

186

00E0 22 187 RET

188

189 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

190 ; SENDVAL

191

00E1 192 SENDVAL: ; converts the hex value of A into two ASCII chars,

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

194 ; does not change the value of A.

195

00E1 C0E0 196 PUSH ACC

00E3 C4 197 SWAP A

00E4 1200F5 198 CALL HEX2ASCII

00E7 11D9 199 CALL SENDCHAR ; send high nibble

00E9 D0E0 200 POP ACC

00EB C0E0 201 PUSH ACC

00ED 1200F5 202 CALL HEX2ASCII

00F0 11D9 203 CALL SENDCHAR ; send low nibble

00F2 D0E0 204 POP ACC

205

00F4 22 206 RET

207

208

209 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

210 ; HEX2ASCII

211

00F5 212 HEX2ASCII: ; converts A into the hex character representing the

213 ; value of A's least significant nibble

214

00F5 540F 215 ANL A,#00Fh

00F7 B40A00 216 CJNE A,#00Ah,$+3

00FA 4002 217 JC IO0030

00FC 2407 218 ADD A,#007h

00FE 2430 219 IO0030: ADD A,#'0'

220

0100 22 221 RET

222

223 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

224 ; DELAY

225

0101 226 DELAY: ; Delays by 100ms \* A

227 ; 100mSec based on 1.573MHZ Core Clock

228

229

0101 FA 230 MOV R2,A ; Acc holds delay variable

0102 7B32 231 DLY0: MOV R3,#50 ; Set up delay loop0

0104 7C83 232 DLY1: MOV R4,#131 ; Set up delay loop1

MASTUART PAGE 5

0106 DCFE 233 DJNZ R4,$ ; Dec R4 & Jump here until R4 is 0

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

0108 DBFA 235 DJNZ R3,DLY1 ; Dec R3 & Jump DLY1 until R3 is 0

236 ; Wait for 50\*2ms

010A DAF6 237 DJNZ R2,DLY0 ; Dec R2 & Jump DLY0 until R2 is 0

238 ; wait for ACC\*100ms

010C 22 239 RET ; Return from subroutine

240

241

242 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

243

010D 0A0A0D5F 244 TITLE: DB 10,10,13,'\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_',10,13

0111 5F5F5F5F

0115 5F5F5F5F

0119 5F5F5F5F

011D 5F5F5F5F

0121 5F5F5F5F

0125 5F5F5F5F

0129 5F5F5F5F

012D 5F5F5F5F

0131 5F5F5F0A

0135 0D

0136 416E616C 245 DB 'Analog Devices MicroConverter ADuC824',10,13

013A 6F672044

013E 65766963

0142 6573204D

0146 6963726F

014A 436F6E76

014E 65727465

0152 72204144

0156 75433832

015A 340A0D

015D 20202020 246 DB ' SPI MASTER Demo Routine',10,13

0161 20205350

0165 49204D41

0169 53544552

016D 2044656D

0171 6F20526F

0175 7574696E

0179 650A0D

017C 20204461 247 DB ' Data Stored in Memory in Hex Form',10,13,0

0180 74612053

0184 746F7265

0188 6420696E

018C 204D656D

0190 6F727920

0194 696E2048

0198 65782046

019C 6F726D0A

01A0 0D00

248

249

250

251

252 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 0101H

DLY0 . . . . . . . . . . . . . . C ADDR 0102H

DLY1 . . . . . . . . . . . . . . C ADDR 0104H

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

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

HEX2ASCII. . . . . . . . . . . . C ADDR 00F5H

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

IO0010 . . . . . . . . . . . . . C ADDR 00C8H

IO0020 . . . . . . . . . . . . . C ADDR 00D4H

IO0030 . . . . . . . . . . . . . C ADDR 00FEH

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

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

NEXT . . . . . . . . . . . . . . C ADDR 00A1H

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

RCAP2H . . . . . . . . . . . . . D ADDR 00CBH PREDEFINED

RCAP2L . . . . . . . . . . . . . D ADDR 00CAH PREDEFINED

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

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

SENDCHAR . . . . . . . . . . . . C ADDR 00D9H

SENDSTRING . . . . . . . . . . . C ADDR 00C1H

SENDVAL. . . . . . . . . . . . . C ADDR 00E1H

SNDUART. . . . . . . . . . . . . C ADDR 0095H

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

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

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

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

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

TITLE. . . . . . . . . . . . . . C ADDR 010DH

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

TRNSMT . . . . . . . . . . . . . C ADDR 0083H

WAIT5S . . . . . . . . . . . . . C ADDR 00B9H