MASTUART PAGE 1

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

2 ;

3 ; Author : ADI - Apps

4 ;

5 ; Date : 11/19/99

6 ;

7 ; File : MASTuart.asm

8 ;

9 ; Hardware ; ADuC824

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 ADuC824

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 $MOD824 ;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 120095 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 75CBFF 70 MOV RCAP2H,#0FFh ; config UART for 9830baud

0063 75CAFB 71 MOV RCAP2L,#-5 ; (close enough to 9600baud)

0066 75CDFF 72 MOV TH2,#0FFh

0069 75CCFB 73 MOV TL2,#-5

006C 759852 74 MOV SCON,#52h

006F 75C834 75 MOV T2CON,#34h

76

0072 75F837 77 MOV SPICON,#037h ; Initialise SPICON to have

78 ; -bitrate=fosc/64

79 ; -CPHA=1

80 ; -CPOL=0, sclk idling low

81 ; -master mode select

82 ; -Enable SPI serial port

83

0075 75A901 84 MOV IEIP2, #01h ; Enable SPI interrrupt

0078 D2AF 85 SETB EA ; Enable interrupts

86

007A 7940 87 MOV R1, #40h ; initialise R1 to 40 to store the

88 ; input data from memory location 40

007C 7800 89 MOV R0, #00H ; initialise R0 to 0 to start

90 ; transmissions from 1

91

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

93 ; can be easily synchronised with the master program.

94

007E 7432 95 MOV A, #50

0080 120101 96 CALL DELAY

97

0083 98 TRNSMT:

0083 08 99 INC R0

0084 C2B5 100 CLR P3.5 ; clear the SS bit during transmission

0086 88F7 101 MOV SPIDAT, R0 ; transmit the current value on R0

0088 D200 102 SETB FLAG ; set flag so that we wait here until

103 ; the spi interrupt routine clears

104 ; the FLAG

105

008A 2000FD 106 JB FLAG, $ ; stay here until flag is cleared

107 ; by interrupt

108

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

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

111 ; from 1 again

112

008D E8 113 MOV A, R0

008E B40AF2 114 CJNE A, #0AH, TRNSMT ; if R0 is not 10, jump to TRNSMT

0091 7800 115 MOV R0, #00H ; if R0=10 make R0=0 & jump to TRNSMT

0093 80EE 116 JMP TRNSMT

MASTUART PAGE 3

117

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

119 ; 5 seconds and then transmit and receive values to/from the slave

120 ; again down the SPI port.

121

0095 122 SNDUART:

0095 B2B4 123 CPL LED ;CPL LED with each transmission

0097 90010D 124 MOV DPTR, #TITLE

009A 1200C1 125 CALL SENDSTRING ; write title block on screen

126

009D 7940 127 MOV R1, #40h ; move value at address 40 into R2

009F E7 128 MOV A, @R1

00A0 FA 129 MOV R2, A

130

00A1 131 NEXT: ; Put new value on a new line

00A1 740A 132 MOV A, #10 ; Transmit a linefeed (= ASCII 10)

00A3 1200D9 133 CALL SENDCHAR

00A6 740D 134 MOV A, #13 ;Transmit a carriage return (=ASCII 13)

00A8 1200D9 135 CALL SENDCHAR

136

00AB EA 137 MOV A, R2 ; Transmit R2 i.e. value @ address R1

00AC 1200E1 138 CALL SENDVAL

00AF 09 139 INC R1 ; Increment address

00B0 E7 140 MOV A, @R1

00B1 FA 141 MOV R2, A ; R2 holds the value @ addrR1

142

00B2 E9 143 MOV A, R1 ; Check if at address 50h

00B3 B450EB 144 CJNE A, #50h, NEXT ; if not jump to Next

00B6 0200B9 145 JMP WAIT5S ; if so wait 5s and repeat

146

00B9 7432 147 WAIT5S: MOV A, #50 ; wait 5s before sending down the

148 ; SPI port again for ease of viewing

149 ; on screen and to allow the slave

150 ; synchronise itself with the master

00BB 120101 151 CALL DELAY

00BE 7940 152 MOV R1, #40h ; store new inputs at address 40h again

00C0 32 153 RETI

154

155 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

156 ; SENDSTRING

157

00C1 158 SENDSTRING: ; sends ASCII string to UART starting at location

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

160

00C1 C0E0 161 PUSH ACC

00C3 C0F0 162 PUSH B

00C5 E4 163 CLR A

00C6 F5F0 164 MOV B,A

00C8 E5F0 165 IO0010: MOV A,B

00CA 05F0 166 INC B

00CC 93 167 MOVC A,@A+DPTR

00CD 6005 168 JZ IO0020

00CF 1200D9 169 CALL SENDCHAR

00D2 80F4 170 JMP IO0010

00D4 D0F0 171 IO0020: POP B

00D6 D0E0 172 POP ACC

173

00D8 22 174 RET

MASTUART PAGE 4

175

176 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

177 ; SENDCHAR

178

00D9 179 SENDCHAR: ; sends ASCII value contained in A to UART

180

00D9 3099FD 181 JNB TI,$ ; wait til present char gone

00DC C299 182 CLR TI ; must clear TI

00DE F599 183 MOV SBUF,A

184

00E0 22 185 RET

186

187 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

188 ; SENDVAL

189

00E1 190 SENDVAL: ; converts the hex value of A into two ASCII chars,

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

192 ; does not change the value of A.

193

00E1 C0E0 194 PUSH ACC

00E3 C4 195 SWAP A

00E4 1200F5 196 CALL HEX2ASCII

00E7 11D9 197 CALL SENDCHAR ; send high nibble

00E9 D0E0 198 POP ACC

00EB C0E0 199 PUSH ACC

00ED 1200F5 200 CALL HEX2ASCII

00F0 11D9 201 CALL SENDCHAR ; send low nibble

00F2 D0E0 202 POP ACC

203

00F4 22 204 RET

205

206

207 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

208 ; HEX2ASCII

209

00F5 210 HEX2ASCII: ; converts A into the hex character representing the

211 ; value of A's least significant nibble

212

00F5 540F 213 ANL A,#00Fh

00F7 B40A00 214 CJNE A,#00Ah,$+3

00FA 4002 215 JC IO0030

00FC 2407 216 ADD A,#007h

00FE 2430 217 IO0030: ADD A,#'0'

218

0100 22 219 RET

220

221 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

222 ; DELAY

223

0101 224 DELAY: ; Delays by 100ms \* A

225 ; 100mSec based on 1.573MHZ Core Clock

226

227

0101 FA 228 MOV R2,A ; Acc holds delay variable

0102 7B32 229 DLY0: MOV R3,#50 ; Set up delay loop0

0104 7C83 230 DLY1: MOV R4,#131 ; Set up delay loop1

0106 DCFE 231 DJNZ R4,$ ; Dec R4 & Jump here until R4 is 0

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

MASTUART PAGE 5

0108 DBFA 233 DJNZ R3,DLY1 ; Dec R3 & Jump DLY1 until R3 is 0

234 ; Wait for 50\*2ms

010A DAF6 235 DJNZ R2,DLY0 ; Dec R2 & Jump DLY0 until R2 is 0

236 ; wait for ACC\*100ms

010C 22 237 RET ; Return from subroutine

238

239

240 ;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

241

010D 0A0A0D5F 242 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 243 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 244 DB ' SPI MASTER Demo Routine',10,13

0161 20205350

0165 49204D41

0169 53544552

016D 2044656D

0171 6F20526F

0175 7574696E

0179 650A0D

017C 20204461 245 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

246

247

248

249

250 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