EE 3724 Test #3 Solutions - Fall '01– Reese

You may skip 1 problem out of problems 1 – 17. You must cross out the problem that you do not want me to grade. You must answer problems 18 and 19.

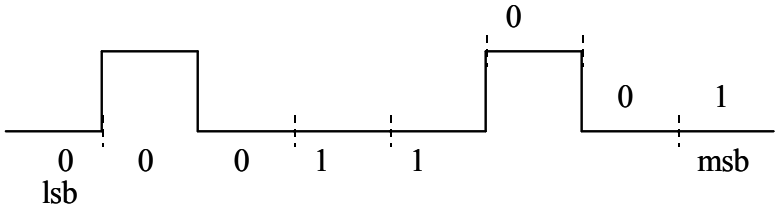
1. (5 pts) Convert the following number in single precision floating point format to its **decimal value** (no exponents allowed in the final decimal value).

Sign bit: 1
Exponent field: 10000010
Significand Field: 1010100….0000

exponent = 82h – 7Fh = 130 – 127 = 3

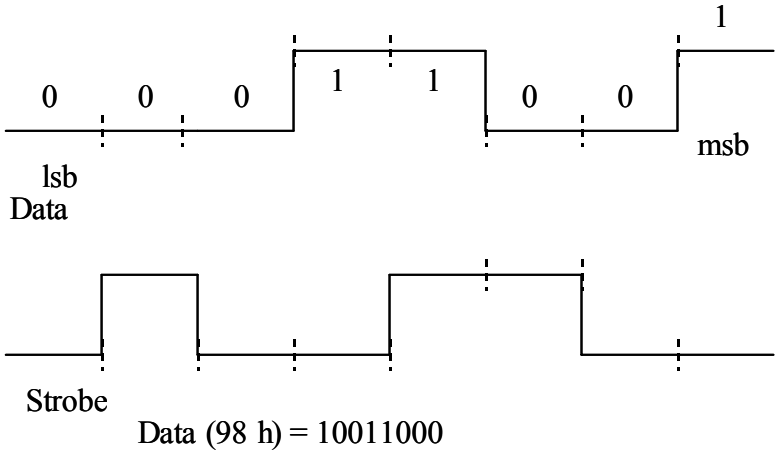
-1. *10101* x 2^3 = -1101.01 = -13.25

2. (5 pts) Draw the waveform for sending an 8 bit value of 98h using **NRZI** encoding (used by USB). Assume the initial value of the waveform signal is a low voltage. Send the value LSB first! (LSB to MSB)



Data (98 h) = 10011000

3. (5 pts) Draw the waveform for sending an 8 bit value of 98h using **DATA STROBE** encoding. Assume the initial values of both data and strobe are both low voltage. Show the signaling for the 8-data bits only (no framing bits such as start/stop)..Send the value LSB first! (LSB to MSB)



Data (98 h) = 10011000

4. (5 pts) What is the key feature of the USB physical signaling method that gives it good protection against injected noise?

   Differential signaling – this rejects common mode noise injected into the cable.

5. (5 pts) What does arbitration mean on a bus? Does the USB support bus arbitration?

   Bus arbitration is the logic that decides which device has control of the bus (bus master) in the case of simultaneous requests for the bus. USD does not support bus arbitration because the host is always bus master and controls all transfers.

6. (5 pts) How does the X86 determine the interrupt number for an INTR interrupt?

   It uses an interrupt acknowledge bus cycle to read the bus number from the interrupting device via the lower 8 data lines.

7. (5 pts) Why must an IRET instruction be used for an interrupt service routine instead of a normal RET instruction?

   The IRET pops off the CS, IP, and flag register which is pushed during as part of the interrupt service by the X86. The RET only pops of the CS, and IP registers.

8. (5 pts) Give me an example of an 8-bit addition that will produce different results in 8-bit signed saturated mode and 8-bit signed unsaturated mode.

   signed saturated 7Fh + 01h = 7Fh (saturates to positive limit)
   signed unsaturated 7Fh = 01h = 80h (normal addition, this is overflow).

9. (5 pts) Convert the number -0.375 to single precision floating point format:

-0.375 = 0.25 + 0.125 = -0.011 = -1.1 x 2^{-2}
exponent field = -2 + 127 = 125 = 7Dh = 01111101
significand is simply 1000000..000.
Sign bit: 1
Exponent field: 01111101 (8 bits)
Significand field: 100000….0 (23 bits)

10. (5 pts) The floating point instructions on the x86 can be written without any arguments, such as 'FMULT' and 'FADD', etc. If no arguments are specified for these instructions, where do they get their operands?

    From the registers at the top of the stack – ST0 and ST1

11. (5 pts) In the IEEE floating point format, what field do I add bits to if I want to extend the range of a floating point number?

The exponent field.

12. (5 pts) Why is the IEEE Firewire Data/Strobe signaling method considered to be a synchronous signaling method?

    Because a clock signal can be extracted by XOR'ing the data and strobe signals – this clock signal can be used to clock in the data bits and keep the receiver synchronized to the bit stream.

13. (5 pts) We looked a data structure called a *circular buffer* for storing data from an interrupt service routine. How did we know if there was data in the buffer?

    tail pointer not equal to head pointer.

14. (5 pts) If I allow a device to hold the bus for a longer period of time, what does this do to the bus latency? Increase bus latency or decrease bus latency?

    This increases the bus latency because other devices have to wait longer for the bus.

15. (5 pts) Give two reasons why the PCI bus cannot achieve its theoretical peak bandwidth.

    multiplexed address/data lines, turnaround bus cycles,

16. (5 pts) Give three methods for increasing bus bandwidth.

    increase data width, increase clock speed, clock data on both clock edges

17. (5 pts) What advantage does a synchronous signaling method like used by USB and Firewire have over an asynchronous signaling method such as used by RS232 (and 'higher speed' is not the answer).

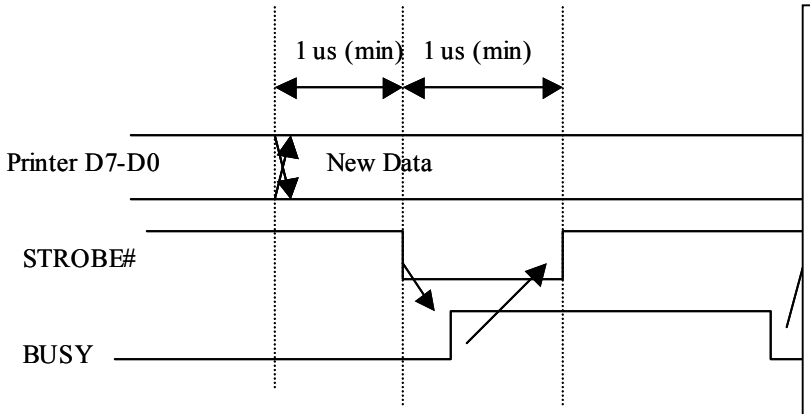
    With synchronous transmission, long streams of bits can be sent without framing bits (start/stop bits). With asynchronous transfers, the number of bits sent at one time must be kept low so that the clocks can be resynchronized via the start/stop bits (framing bits). This means the ratio of framing bits to data bits is much lower in synchronous transfers than asynchronous transfers, to the data transfer is more efficient.

18. (10 pts) For the 16550D, assume the following port definitions
Write a subroutine that will wait for an input byte to become ready from the 16650 and return the byte in register AL. Also, the carry flag should be set upon return if either a framing, parity, or overrun error occurs. The carry flag should be cleared upon return if these errors did not occur. For the 16550D, assume the port definitions shown below and use these port names in your code.

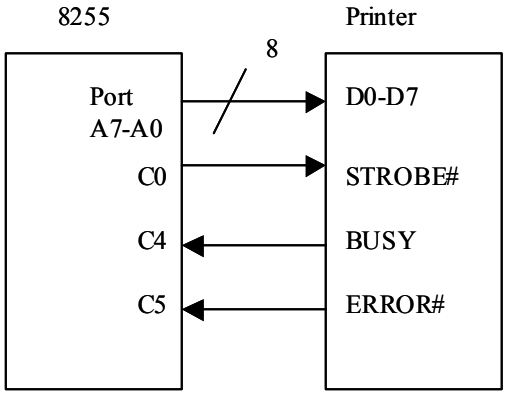
```
; all port definitions for COM1
rbr        equ   03f8h
thr        equ   03f8h
ier        equ   03f9h
iir        equ   03fAh
fcr        equ   03fAh
lcr        equ   03fBh
mcr        equ   03fCh
lsr        equ   03fDh
msr        equ   03feh
scr        equ   03ffh

dl_lsb     equ   03f8h
dl_msb     equ   03f9h


    indata     proc    near
               mov     dx,lsr
    lp1:       in      al,dx           ;read line status
               test    al, 01h         ;check data rdy bit
               jz      lp1             ;wait for data
               mv      bl,al           ;save flags in bl
               mov     dx,rbr
               in      al,dx           ;get byte from receive buffer
               test    bl,000001110b   ;check for errors
               jnz     err             ;jump if any error bit set
               clc                     ;no error
               ret
    err:       stc                     ;set carry
               ret
    indata     endp
```

19. (10 pts) The diagram below shows a 8255 connected to a parallel printer. The 8255 responds to the same ports (0400h=PortA, 0404h=PortB, 0408h=PortC, 040Ch=Control) using BE0# in the chip select as specified in the notes. The timing diagram illustrates how to send one byte of data to the printer:
    a. Place data on data lines
    b. Wait for 1 us (1 microsecond)
    c. Bring the strobe line low.
    d. Wait for 1 us.
    e. Bring the strobe line high
    f. Wait for BUSY line to go low
    g. Go back to 'a' and start again.

    Write a subroutine called PRINT and will accept two parameters: a pointer to some bytes to be printed (starting address is passed in BX), and the number of bytes to print (passed in CX, will never be zero, and never greater than FFFFh). The AL register should return a '0' if all bytes were printed successfully. If the ERROR# line ever goes low during printing, then an error has occurred and the subroutine should a non-zero value in AL to indicate the error. The CX register should return the number of bytes left to be printed in the case of error. Assume you have access to a subroutine called '*delay1us*' that will delay for 1 us.



If ERROR# goes low, then printer error has occurred.



8255 responds to ports 400h, 404h, 408h, 40Ch with BE0# as in example in notes.

```
Print   proc    near
Lp1:    mov     dx,0408h        ;port C
        Mov     al,1
        Out     al,dx           ;Strobe = high
Lp3:    in      al,dx
        Test    al,00100000b    ;check error
        Jz      err             ; err if = 0
        Test    al,00010000b    ;check busy
        Jnz     lp3             ;loop while busy=1
        mov     dx,0400h        ;port A
        Mov     al,[bx]         ;get char
        Out     al,dx           ;snd to Port A
        Call    delay1us        ;wait
        Mov     dx,0408h        ;port C
        Mov     al,0
        Out     al,dx           ;Strobe = low
        Call    delay1us
Lp2:    in      al,dx           ;read port C
        Test    al,00100000b    ;check error
        Jz      err             ; err if = 0
        Test    al,00010000b    ;check busy
        Jz      lp2             ;loop while busy=0
        Inc     bx              ;next char
        Loop    lp1             ;CX has count
        Xor     al,al           ; no error while printing
        Ret
Err:    mov     al,1            ;error while printing
        ret
print   endp
```