Exam 2 objectives (in assembly not C) (running TExaS in simulation not on real board)

0) Being able to quickly design, implement, and debug assembly software
1) Understanding differences between data and address, being able to use pointers and indices
2) Understanding differences between 8-bit and 16-bit data
3) Understanding differences between signed and unsigned integers
4) Programming loops and conditionals
5) Processing a variable-length array or string, either size first or terminating code at end
6) 8- and 16-bit addition subtraction with **`adda suba addd subd`**
7) 16-bit multiply and divide with **`mul idiv idivs`** E.g.,
   `mul` is unsigned multiply of A times B, putting the product in D
   `idiv` is unsigned division of D divided by X, putting the quotient in X, remainder in D
   `idivs` is signed division of D divided by X, putting the quotient in X, remainder in D
8) 8- and 16-bit shift left and right with **`asla asra lsla lsra lsld lsrd`**
9) Numbers and pointers on the stack
   Push/pull, including but not limited to **`psha pula pshx pulx pshy puly`**
   Push values on the stack, e.g.,
   ```
   ldaa #5
   psha
   ```
   or
   ```
   movb #5,1,-sp
   ```
   Push addresses on the stack. For example, assume **`Data`** is a variable in RAM
   ```
   Data rmb 1
   ```
   these instructions push a pointer to **`Data`** on the stack
   ```
   ldx  #Data
   pshx
   ```
   or
   ```
   movw #Data,2,-sp
   ```
10) Subroutines with parameter passing
   Called using **`jsr bsr`**, returned using **`rts`**
   Call by value and call by reference
   Parameter passing in registers
11) Implementation of FSM using a linked data structure or using a table structure with an index
12) Accessing arrays and strings using pointers and indices

List of potential programming problems

A) You may be given one or more variable length arrays of data, `buf[i]`

The size may be the first entry or there may be a termination code
The data may be 8-bit ASCII characters or integers
The integers may be 8- or 16-bit, signed or unsigned
A pointer to this array may be passed to your subroutine in RegX or RegY
You may be asked to deal with special cases: size=0, size too big, overflow

Your subroutine(s) may be asked to perform operations including, but not limited to these
Determine the size of the array
Return the first element of the array
Find the maximum or minimum element in an array
Find the sum of all the elements
Find the average of all the elements

Find the mode of all the elements
Find the range = maximum - minimum
Find the maximum or minimum slope (**`buf[i+1]-buf[i]`**)
Find the maximum or minimum absolute value
Count the number of times a particular value occurs (**`buf[i]==1000`**)
Search for the occurrence of one string in another
Concatenate two strings together
Delete characters from a string
Insert one string into another
Move data from one place to another within an array or string

B) Since this exam covers Lab 5, you may be asked to implement a FSM

Convert a FSM graph to a linked data structure or table with an index
Write a Mealy FSM controller (with or without timer wait)
Write a Moore FSM controller (with or without timer wait)

Homework 6 is an old Exam2 problem. For full credit on HW6, work one old exam problem until you get a full score of 100. Print one screen shot for each exam showing your code in the source window (RTF file) and theCRT.rtf with your score. We strongly suggest doing all three.

Four FSM problems can be found in the textbook

Homework 6.25, 6.26, 6.27, and 6.28

A long list of potential Exam2 problems can be found in

http:// users.ece.utexas.edu/~valvano/EE319K/Exam2study.rtf

Exam 2 is later in time as compared to some previous semesters, therefore you can expect this semester's Exam2 to be more complex than what previous EE319K students might tell you.

Grading based both on numerical results and programming style (weighting to be determined by professor after the exam is given)