

Exam 2 objectives

- 0) being able to quickly design, implement, and debug software
- 1) understanding differences between data and address
- 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**
- 8) 8- and 16-bit shift left and right with **asla asra asld asrd lsla lsra lslld lsrd**
- 9) no local variables, but simple use of **psha pula pshx pulx pshy puly**
- 10) simple subroutines with parameters passed in registers using **jsr bsr rts**.
- 11) call by value and call by reference parameter passing in registers

List of potential programming problems

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, integers or fixed-point numbers

The integers and/or fixed-point may be 8- or 16-bit, signed or unsigned

A pointer to this array will 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 one or more of these operations

Determine the size of the array

Return the first element of the array

Find the minimum element

Find the maximum element

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 slope (**buf [i+1]-buf [i]**)

Find the minimum slope (**buf [i+1]-buf [i]**)

Find the maximum absolute value

Find the minimum absolute value

Find the maximum binary fixed-point square (**buf [i]*buf [i]**) / 256

Find the maximum decimal fixed-point square (**buf [i]*buf [i]**) / 100

Count the number of times a particular value occurs (**buf [i]==1000**)

Count the number of times two adjacent elements are equal (**buf [i+1]==buf [i]**)

Count the number of times the slope is positive (**buf [i+1]>buf [i]**)

Count the number of times the slope is negative (**buf [i+1]<buf [i]**)

Two practice versions are posted at

<http://users.ece.utexas.edu/~valvano/EE319K/Exam2C.zip>

<http://users.ece.utexas.edu/~valvano/EE319K/Exam2y.zip>

A long list of potential Exam2 problems can be found in

[http:// users.ece.utexas.edu/~valvano/EE319K/Exam2study.rtf](http://users.ece.utexas.edu/~valvano/EE319K/Exam2study.rtf)

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