

**Exam 2 objectives (in assembly and C) (running uVision in simulation not on real board)**

- 1) During the exam we try to answer questions with “yes”, “no”, “won’t answer”
- 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, 16-bit data and 32-bit data
- 3) Understanding differences between signed and unsigned integers
- 4) Programming if-then if-then-else for-loops while-loops and do-while-loops in assembly
- 5) Processing a variable-length array or string, either size first or terminating code at end
- 6) Addition subtraction multiplication, division, shift, and, orr, eor (signed and unsigned)
- 7) Structures using **DCB DCW DCD EQU SPACE** syntax
- 8) Call by value, call by reference, return by value
- 9) AAPCS Program conventions
  - Save and restore R4-R11,LR if your subroutine wishes to modify
  - Parameter passing in registers R0,R1,R2,R3, all input parameters promoted to 32 bits
  - Return parameter in R0
- 10) ~~Implementation of FSM~~ **no FSM Fall 2018**
  - ~~Moore FSM (not Mealy)~~
  - ~~It will be your choice to implement table index or pointer linked data structure~~
  - ~~Not a real port, so no bit specific addressing, no timer waits~~
  - ~~Not a real port, so no port initialization,~~
- 11) Accessing arrays and strings using pointers and indices
  - Stepping through two or more arrays at a time
  - 8/16/32-bit data, signed/unsigned numbers
- 12) Structs and Arrays of structs
  - Access with dot-format and arrow-format

**List of potential programming problems**

- A) You may be given one or more variable-length arrays of data,
  - The size may be the first entry, there may be a termination code, or the array may have two fields: size and data
  - The data may be 8-bit ASCII characters or integers
  - The integers may be 8- or 16-bit or 32-bit, signed or unsigned
  - A pointer to this array may be passed to your subroutine in registers
  - You may be asked to deal with special cases: size=0, size too big, overflow
- B) Your subroutine(s) may be asked to perform operations including, but not limited to these
  - Calculating mathematical functions (e.g.,  $y = a*x^2 + b*x + c$ ) with ceiling (on overflow, set to max), with floor (on underflow, set to min)
  - 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  
 Sort the array (we will give the steps)  
 Searching a data base made with an array of structs

**C) ~~Because 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 Moore FSM controller, using variable-based I/O (without timer wait)~~  
~~It also may involve accessing a linked structure like Lab 5~~ No FSM Fall 2018  
 However, you can expect to access structs or arrays of structs

**D) We may give you a function with mistakes and ask you to find and correct the bugs**

**Homework involves old Exam2 problems (not all exams were the same length).**

**CExam2\_StringCompare** Easy practice Exam 2 involving ASCII strings  
**CExam2C\_CalculusSpring2013** Medium difficulty practice Exam 2 involving Math  
**CExam2\_Merge** Medium difficulty practice Exam 2 involving ASCII strings  
**CExam2\_Unicode** Practice Exam 2 involving 8 and 16-bit arrays  
**CExam2C\_PermuteCombine** Medium difficulty practice Exam 2 involving Math  
~~**CExam2\_Moore** Practice Exam 2 involving Moore FSM, some C some assembly~~  
**CExam2\_Mode** Practice Exam 2 involving arrays and structures  
~~**CExam2\_Mealy** Practice Exam 2 involving a Mealy FSM~~  
**CExam2\_DataBase** Practice Exam 2 involving arrays and structures  
**CExam2\_ManhattanDistance** Exam 2 involving math and structures

**Assembly exams**

**Exam2\_Sum.zip** Easy practice Exam2 involving strings and addition  
**Exam2\_Quad.zip** Easy practice Exam2 involving arrays and multiplication  
**Exam2\_Mode8** Hard practice Exam 2 involving strings and pointers  
~~**Exam2\_Moore.zip** Hard practice Exam 2 involving Port initialization and a Moore FSM~~  
**StringCompare.zip** Easy practice Exam 2 involving ASCII strings  
**Exam2\_Merge.zip** Hard practice Exam 2 involving ASCII strings  
**Exam2V.zip** Easy practice Exam 2 involving BCD numbers  
**Exam2\_Sum32.zip** Easy practice Exam 2 involving 32-bit numbers and overflow (35min)  
**Exam2C\_CalculusSpring2013.zip** Practice Exam with math  
**Exam2\_ArrayOfStruct.zip** Exam 2 with structures and arrays of structures

There will be no SysTick, no I/O initialization, no floating point, no circuits, and no interrupts.

**Material**

- 1) Book sections 1.3, 1.4, 1.5, 1.6, 1.12, 1.13, 2.5, 2.7, 2.8, 4.4, 4.5, 4.6, 4.7, 5.1
- 2) Embedded Software in C for an ARM Cortex M
  - <http://users.ece.utexas.edu/~valvano/embed/toc1.htm>
  - Chapter 1 Introduction (no I/O, no files, no preprocessor commands)
  - Chapter 2 Tokens (no colon, no switch statement)
  - Chapter 3 Numbers (no octal)
  - Chapter 4 Variables (no statics, no volatile, no externals)
  - Chapter 5 Expressions (no selection operator)
  - Chapter 6 Flow of control (no switch statements, no goto)
  - Chapter 7 Pointers (no FIFO, no I/O)
  - Chapter 8 Arrays and strings (no negative index, no string.h functions, no FIFO)
  - Chapter 9 Structures (no FSMs, no structs inside structs, no linked lists, no binary trees)
  - Chapter 10 Functions (no private functions, no function pointers, no FSM, no linked lists)
- 3) Labs 1,2,3,4,5 (in C and/or assembly, no I/O, no SysTick, no PLL, no FSM)
- 4) Worksheet questions: 2.10, 4.1, 4.2, 4.3, 4.6, 4.7, 5.1, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 7.3, 7.4, 8.17

**Grading based mostly on numerical results and some part of the grading will be based on programming style** (style to be determined by professor after the exam is given). We consider it necessary to actually solve the problem. We will substantially lower grades to solutions that trick the grader into giving points (hard coding so it returns correct answers without actually calculating the output from the inputs).

Your laptop needs to be running Keil uVision in simulation and have 75 minutes of power. You will use the internet to download the exam at the beginning, and then use the internet to upload the solution to canvas at the end. The instructions for your exam will be very similar to the practice exams.