CE2810 Embedded Systems II

Dr. Walter Schilling

Spring, 2011-2012

For the exams, you are permitted one 8.5 x 11 sheet of paper with notes handwritten on it.

1. Week 1 - Introduction
   (a) Lecture 1: Introduction to C.
      i. Explain the history of the C programming language
      ii. Compare and Contrast C with Java
      iii. Describe the usage of the #include and #define preprocessor directives.
      iv. Explain why #define statements are useful
      v. Explain typical case conventions for #defines
      vi. Compare and contract definitions and declarations
      vii. Explain the concept function prototypes
      viii. Analyze a generated assembly file based on the constructed C code
      ix. Use the asm statement to place inline assembly code within a function
      x. Explain how to set the stack address in avr studio (Not an exam question)
   (b) Lecture 2: More information on C.
      i. Draw a flowchart showing the compilation phases for a C program
      ii. List the purposes for the preprocessor
      iii. List the purposes for the compiler
      iv. List the purposes for the linker
      v. Explain how to view C code with generated assembly code using avr studio
      vi. Explain the relationship between #includes and dependencies
      vii. Explain the relationship between .c and .h files
      viii. Explain how code can be partitioned into multiple files
      ix. List the fundamental data types for C and C99.
      x. Construct a constant table in C.
2. Week 2 - More C details
   (a) Lecture 1- Data Types, Operators, and Loops
      i. List the native C data types including their ranges and sizes
      ii. List the C99 data types including their ranges and sizes
      iii. Construct a program using a constant table to solve a problem in C.
      iv. Construct while, for, and do-while loops in C
      v. Construct expressions using bitwise or, and, and not operators
      vi. Construct expressions using left and right shift operations
      vii. Describe the usage of pointers to directly access I/O ports.
      viii. Define C macros to allow access to I/O ports.
      ix. Explain the meaning of the term volatile
      x. Explain why we must define pointers to I/O registers as volatile
   (b) C Operators
      i. Explain how variables are allocated in SRAM by the compiler and linker
      ii. Use the address of operator and the indirection operator to manipulate pointers
      iii. List the relationship between variable size and pointer size on the ATMEGA32.

iv. Explain the purpose for the NULL pointer
v. Explain the operation of pointer arithmetic
vi. Explain the danger of declaring multiple pointers on the same line
vii. Explain the relationship between arrays and pointers

3. Week 3

(a) Lecture 1- Arrays and Interrupts

i. Explain how one sets the size of an array.
ii. List two equivalent mechanisms for dereferencing arrays
iii. Determine the size of an array using the sizeof operator
iv. Determine the number of elements in an array using the sizeof operator
v. Compare and contrast interrupts and polling
vi. Explain the purpose for the ISR macro
vii. Construct an interrupt service routine in C for the external interrupts.
viii. Explain the need for the volatile keyword when referencing variables from interrupts.

(b) Lecture 2 - Text Manipulation

i. Explain the difference between passing by value and passing by reference.
ii. Explain the meaning of the const keyword in C.
iii. Explain the relationship between arrays and pointers
iv. Explain how parameters are passed on the ATMEGA32 environment
v. Compare and contrast pass by value and pass by reference
vi. List the capabilities of the c string handling library
vii. Construct code which uses the C string handling library

4. Week 4

(a) Lecture 1 - Text Manipulation

i. Explain the difference between the allocated length and the string length for a String.
ii. Explain how to compare two strings using the C library.
iii. Explain the relationship between char arrays and numbers.
iv. Explain the purpose for the C standard library.
v. Explain the problem with returning pointers from a function.
vi. Explain why functions that return a pointer often pass a pointer in as a parameter.
vii. Using the atoi function, convert an ascii string into an integer.
viii. Use the sprintf method to print to a buffer, converting a number into a string.
ix. Construct code which converts strings into integers.
x. Construct code which converts integers into strings.

i. Lecture 2 Interrupts in C

A. Explain the steps necessary for disabling interrupts.
B. Explain the risks of improperly disabling interrupts.
C. Explain the concept of scope.
D. Given a variable, define its scope.
E. Explain the purpose for the C static keyword.

5. Week 5

(a) Lecture 1 Midterm Exam

i. Successfully pass the exam.