QUIZ 3 STUDY GUIDE

While I always hope to cover all the material and examples in the lecture material, I typically run out of time before material. Here are some of the lectures where you can find problems and examples you can work on at home.

1. Lecture 01 to 09 Programming and Lab Basics (Load-Store, SREG, Branching and Looping, Subroutines, GPIO)
2. Lectures 10 to 12 Interrupts , for example lecture 11 page 15 “Practice Problems”
3. Lecture 13 AVR Indirect Addressing Modes – pages 11 and 16 Program Examples
4. Lecture 14 AVR Logic and Shift – Page 5 Knight Rider, examples in slides, and Questions page 14
5. Lecture 15 AVR Stack Operations – Review example on page 6
6. Lecture 17 – C++ Introduction –pages 12,14, and 15. You can find the answers on page 19.

You can find **review question** in the AVR Final Review document under the corresponding lecture headings.

Lecture 01 to 09 Programming and Lab Basics

1. Given two numbers, calculate the difference, signed and unsigned relationship.
2. For a given arithmetic operation (add or subtract) define the state of SREG bits H, S, V, N, Z, C.
3. Know how to simulate a call to `ReadSwitches`.
4. Know how to save and restore the Status Register (SREG)
5. Know the methods for sending information to and from a subroutine.
6. Be prepared to write a program to send data to and receive data from a subroutine. Specifically, in a register or one of the SREG bits. Your program will not be required to implement a stack frame.
7. Be able to identify code which violates one or more of the rules for working with a subroutine or an interrupt service routine. For example the code jumps out of a subroutine, a push is not matched to a pop instruction, or a `ret` instruction is used to end an ISR.

Lecture 10 to 12 - Interrupts

1. Understand how an ISR is different from and similar to a subroutine.
2. Be able to locate interrupts within the Interrupt Vector Table (IVT) and the priority of each.
3. Know how to configure an interrupt to be triggered based on the nature of the input signal (low logic level, logic change, falling or rising edge.
4. Know how to enable a given external interrupt(s). This external interrupt(s) might be one of our two dedicated external interrupt lines or one or more of our pin-change interrupt lines.
5. Know what happens when an interrupt is triggered and what if any registers are placed on the stack.
6. Given our ground bounce (low-pass filter) circuit be able to generate the output for a given clock a button input condition.

Lecture 13 - Indirect Addressing Mode

1. What does defining a table actually do? Does it give addresses for constants that already exist in program memory or does it do something else?
2. What is an index and why is it used?
3. If the least significant bit selects whether ZL or ZH is used, is one of them (ZL or ZH) a 7 bit register?
4. Do the mnemonics "low" and "high" automatically correspond to the low and high bytes of the Z-register?
5. Specifically for lab 10, why is the index equal to 20*row+col?
6. What register pair is found in the source operand address field of an `lpm` instruction? Z
7. What register numbers correspond to pre-defined mnemonics ZH:ZL?

Lecture 14 – Logic and Shift Instructions

1. Be able to clear, set, and toggle bits, including setting a bit pattern (see page 12)
2. Know how to test if one or more bits are set or cleared.
3. Understand multiplication and division by 2 using shift instructions.

Lecture 15 – AVR Stack Operations

1. Understand the difference between the difference between a LIFO and a FIFO stack
2. Understand the difference between an implicit and explicit stack operation.
3. Given the address of an rcall instruction, the address where the called subroutine begins, and the value of the stack pointer before the call; be prepared to calculate what the stack pointer will be equal to after the call instruction; at the beginning of the subroutine and the contents of the stack.
4. Be able to identify the code within a subroutine that will result in stack incoherence or a poorly formed series of pushes and pops.

Lecture 16 – Instruction Encoding

1. Encoding of those instructions included in previous lectures (i.e., not all instructions)

Lecture 17 – C++ Introduction

1. Explicit Data Types
2. Variable Scope and Qualifiers
3. Setting and Clearing Bits
4. Setting a Bit Pattern