ELE538 Quiz/Answers (2004)

Name: _____ Student #: _____ Time: 30 minutes

Chun,Clowes,Guerkov

Table of Contents

Answer all questions. All questions have equal weight.

More difficult version (sorry)

This version of the Quiz was (unintentionally) more difficult than the other two versions. Even worse, 2 of the 3 questions were harder. I (kclowes) apologize. Marking of this version will be more generous

To make matters even worse, the unluckly students (one out of three) who got this version also had to deal with another error. (The source code used an instruction `ldx #stuff-1` but the first question referred to an instruction `ldx #stuff` which did not exist. These on-line versions of the Quiz have been fixed.) Once again, I apologize.

Reference Material

This material contains technical details that may be required to answer certain questions.

Instructions

Table 1. Instruction Details (Abridged)

Assembler	Mode	Encoding	Cycles
ldaa	IMM	86 ii	2
ldab	IMM	C6 ii	2
mul	INH	3D	10

A/D system

The bits in the Control/Status register (`ADCTL`, mapped to address 0x1030) are:

Figure 1. AD Control/Status Register

```
 7   6   5   4   3   2   1   0
CCF  -   M*  S   0   n   n   n
```

The interpretation of the bits is:

CCF 0: conversion NOT complete; 1: conversion complete.

M* 0: Convert 4 channels; 1: Convert single channel.

S 0: continuous conversion; 1: one-shot conversion.

nnn Channel number (0-7).

Questions

1. Given the following program:

```
        org $6000
main:
        ldx #stuff-1
        ldaa #0

loop:
        inx
        ldab 0,x
        beq done
        adda 0,x
        bra loop
done:
        swi

        org $7000
stuff   fcb 3, 1, 4, -1, 0, 2, 7
```

Assume that the CPU begins executing at address 0x6000.

a. What value (in hex) will be in index register X following the execution of the instruction `ldx #stuff-1`?

b. What will the values in index register X, AccA and AccB be just before the `swi` instruction is executed?

ANSWER

a.

$6FFF

b.

AccA: 7
AccB: 0
IX: $7004

2. The following program uses the A/D converter subsystem to read some voltages. The program does work. (i.e. there are no logical or syntactical errors.)

```
; A simple program using adc module.
; Author: Foo Bar
; Date: October 6, 2004

ADCTL equ $1030     ;address of ADC Control register
ADR1  equ ADCTL+1   ;address of first result register
ADR2  equ ADCTL+2   ;address of second result register
ADR3  equ ADCTL+3   ;address of third result register
ADR4  equ ADCTL+4   ;address of fourth result register

        org $6000
main:
        ldaa #%00010100
        staa ADCTL

        jsr foo

        ldaa ADR1
        ldab ADR4
        swi

foo:
         tst ADCTL
         bpl foo
         rts
```

a. The subroutine "foo" performs an essential task. The name of the subroutine, however, is poorly chosen since it does not hint at the task it performs.

What is a better name for the subroutine?

b. Suppose that all 8 analog channels are connected to DC voltages as follows (assume that "full scale analog voltage" is 5.0 V):

Channel 1: 2.5 V
Channel 2: 1.25 V
Channel 3: 3.75 V
Channel 4: 5.0 V
Channel 5: 2.5 V
Channel 6: 0.0 V
Channel 7: 5.0 V
Channel 8: 1.25 V

The program is run from address 0x6000. What values will be in Acc. A and Acc B. when the "swi" instruction is encountered?

ANSWER

a. A better name would be something like `WaitConvDone`.

b. Channels 5-8 are converted. Channel 5 is 2.5 volts (analog), converted to digital %10000000. Channel 8 is 1.25 volts (analog), converted to digital %01000000. Since AccA reads Channel 5, it is $80; since AccB reads Channel 8, it is $40.

3. Write a code fragment that performs the following:

a. Divides the signed 8-bit binary number in Acc by 2. (For example, if AccA were 19, it would be 9 after division.)

b. If Acc A is an odd number, convert it to the next bigger even number. (For example, a 9 would become a 10.)

c. Invert the bits 2 and 3 of the result. (For example, 10—00001010 in binary— would become 6—00000110 in binary.)

ANSWER

```
asra         ;Part a: arithmetic right shift divides by 2 (signed)
inca         ;Part b: increment odd or even number
anda #$FE    ;        ensure result is even
eora #%1100  ;Part c: XORing with 1 inverts bit at same position
```