ECE 2534 - Homework 1 – Fall 2015

(Patterson section)

Due at Scholar before 5:00 p.m. on September 1

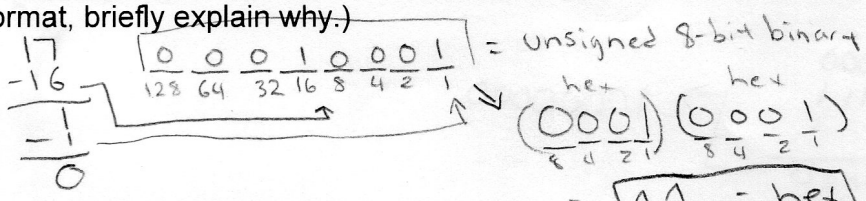
Name: Bowei Zhao. ID Number (last 4 digits): 9193

Instructions: Work all of the problems, and submit your solutions to Scholar as a single document. Do not hand in paper copies. It is okay to do the work by hand, and then scan those pages. The preferred format for submission is PDF, although MS Word is acceptable. Show your work to a reasonable extent, and clearly indicate your answers. In order to give you faster feedback, only a subset of these problems will be graded in detail.

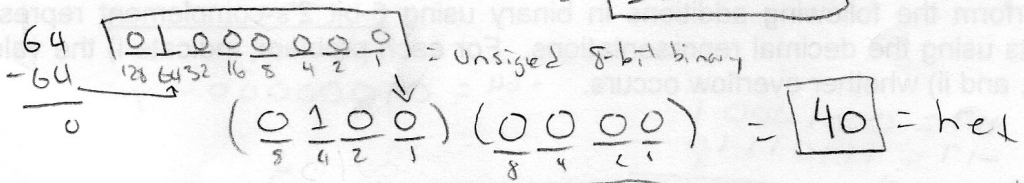
You may use any appropriate reference materials, including your ECE 2504 textbook and on-line materials. You may share references with your classmates, but you may not share your solutions. The work that you submit must be your own.

Problem 1. Convert each of the following decimal numbers to an unsigned 8-bit binary value. Also give the hexadecimal equivalent for each answer. (If one of the numbers cannot be represented in these format, briefly explain why.)

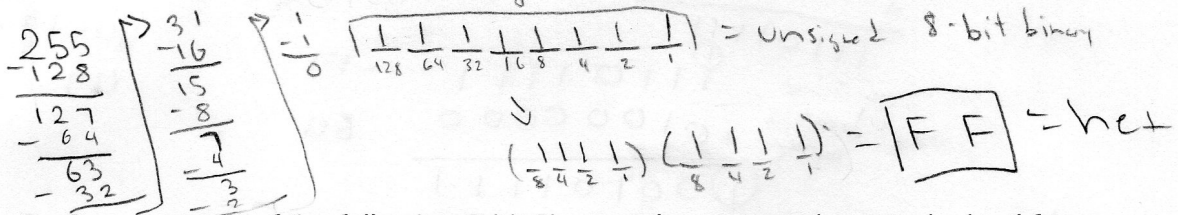
a. 17



b. 64

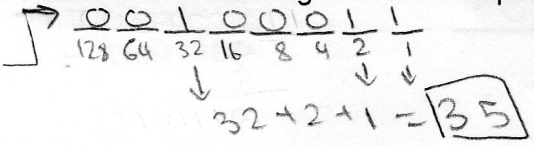


c. 255

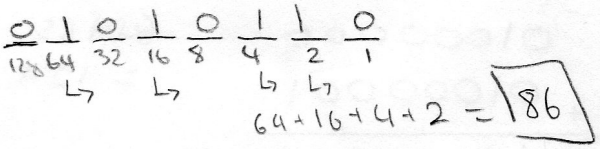


Problem 2. Convert each of the following 8-bit 2's-complement numbers to decimal form.

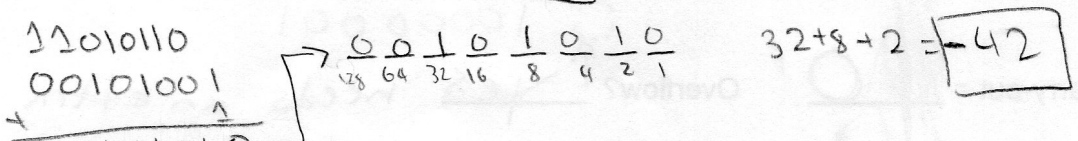
a. $(00100011)_{2cm}$



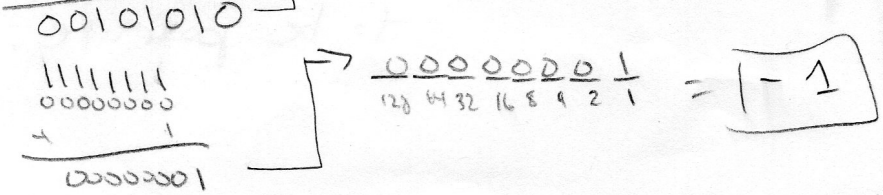
b. $(01010110)_{2cm}$



c. $(11010110)_{2cm}$



d. $(11111111)_{2cm}$



Problem 3. Convert the following decimal numbers to 8-bit 2's-complement form. (If one of the numbers cannot be represented in this format, briefly explain why.)

a. +9 $9 - 8 = 1$, $1 - 1 = 0$ → 00001001

b. -9 +9 = 00001001 → 11110110 + 1 = 11110111

c. +128 128 − 128 = 0 → 10000000 ERROR can not be represented. Decimal positive so leading bit must be 0. To represent +128 in 8-bit 2's comp will require 9-bits or it will be another number and be negative.

d. -128 +128 = 10000000 → 01111111 + 1 = 10000000

Problem 4. Perform the following additions in binary using 8-bit 2's-complement representation. Verify your results using the decimal representations. For each problem, indicate i) the value of the final carry-out bit, and ii) whether overflow occurs. +64 = 01000000

a. (-17) + (64) +17 = 00010001 → −17 = 11101110 + 1 = 11101111

11101111 −17

+ 01000000 64

(1) 00101111 (1 doesn't matter)

64 − 17 = 47

47 = 00101111

Carry-out = 1 Overflow? No

b. (64) + (65)

01000000 = 64

01000000

+ 01000001 = 65

(1) 10000001

64 + 65 = 129 = 10000001

Carry-out = 0 Overflow? yes needs an extra 0 for it to be positive.