

1. Floating point has one sign bit, 8 exponent bits, and 23 fraction bits. You can convert to a real number with $N = (-1)^S \times 1.\text{fraction} \times 2^{\text{exponent} - 127}$, for exponents in the range of 1 to 254. Show the bit pattern that would represent $-6 \frac{5}{8}$. Separate out the sections of bits, so that it's easy to read (as was done in the book).

1 10000001 1010100000000000000000

2. Write the bit pattern for the hexadecimal number 0xABCD

1010 1011 1100 1101

3. Convert the decimal number 42 to binary, octal, and hexadecimal.

101010 52 2A

4. Now we get tricky! Convert the decimal number 42 into base 3!

1120

5. Write truth tables for NOT, AND, OR, and XOR

6. Draw an inverter using N and P transistors.

Name:

CS210 Final, May 14, 2007

7. Draw an R-S latch.

8. Give a truth table for a full adder (inputs A, B, Cin, outputs S, Cout)

9. Sketch a gate-level 2-bit decoder circuit; this has two inputs, S_1 and S_2 , and four output lines; the different combinations of input values for S_1 and S_2 cause one of the four output lines to turn on.

10. Sketch a gate-level 2-to-1 mux circuit. It has inputs A, B, and S. If S is 1, the output is the same as A. If S is 0, the output of the circuit is B.

11. Circuit functions can be written with a plus symbol for logical “or”, a multiply for logical “and”, and an exclamation point for logical “not”. Give a truth table for $(A \oplus B * C) + (A * B * C)$. Please put the bits in a reasonable order!

| ABC | Value | ABC | Value |
|-----|-------|-----|-------|
| 0 | 0 | 100 | 0 |
| 1 | 0 | 101 | 1 |
| 10 | 0 | 110 | 0 |
| 11 | 0 | 111 | 1 |

12. Draw a gate-level circuit for the function from the previous question. Make the circuit as simple as possible.

And gate “A and C” -- B does not come into the circuit.

13. 8 points (4 each part). For the truth table given, draw a Karnaugh map, and give a simple circuit that implements the function. If you had web access, you could just use the one off the Wikipedia page; for the exam, you have to use your brain.

| ABCD | | ABCD | |
|------|---|------|---|
| 0 | 1 | 1000 | 1 |
| 1 | 0 | 1001 | 0 |
| 10 | 1 | 1010 | 1 |
| 11 | 0 | 1011 | 0 |
| 100 | 0 | 1100 | 0 |
| 101 | 0 | 1101 | 0 |
| 110 | 0 | 1110 | 0 |
| 111 | 0 | 1111 | 0 |

Karnaugh Map

| | | AB | | | |
|----|----|----|----|----|----|
| | | 00 | 01 | 11 | 10 |
| CD | 00 | 1 | 0 | 0 | 1 |
| | 01 | 0 | 0 | 0 | 0 |
| | 11 | 0 | 0 | 0 | 0 |
| | 10 | 1 | 0 | 0 | 1 |

$$F(ABCD) = \bar{B} \bar{D}$$

This information might be useful. For any instruction that might need it, assume the instruction is located in memory at 0x4000.

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----|----|----|----|----|----|---|-----------|-----------|---|---------|------|---|-----|---|---|
| ADD | 0 | 0 | 0 | 1 | DR | | | SR1 | | | 0 | 0 | 0 | SR2 | | |
| ADD | 0 | 0 | 0 | 1 | DR | | | SR1 | | | 1 | imm5 | | | | |
| AND | 0 | 1 | 0 | 1 | DR | | | SR1 | | | 0 | 0 | 0 | SR2 | | |
| AND | 0 | 1 | 0 | 1 | DR | | | SR1 | | | 1 | imm5 | | | | |
| BR | 0 | 0 | 0 | 0 | n | z | p | PCoffset9 | | | | | | | | |
| LD | 0 | 0 | 1 | 0 | DR | | | PCoffset9 | | | | | | | | |
| LDI | 1 | 0 | 1 | 0 | DR | | | PCoffset9 | | | | | | | | |
| LDR | 0 | 1 | 1 | 0 | DR | | | BaseR | | | offset6 | | | | | |
| LEA | 1 | 1 | 1 | 0 | DR | | | PCoffset9 | | | | | | | | |
| TRAP | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | trapvect8 | | | | | | | |

14. Write the bit pattern for the LC3 instruction to add R2 to R3, storing the result in R6.

0001 110 010 000 011 or 0001 110 011 000 010

15. What does the instruction 0 0 1 0 0 1 0 1 1 0 1 0 1 1 1 1 do in the LC3?

Loads the value at 0x4000 + FFAF into register 2
Location is 3FAF

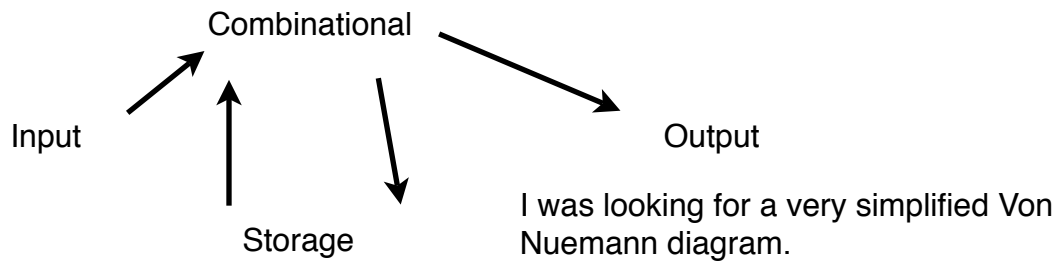
16. Write the bit pattern for an LC3 instruction which causes the program counter to jump forward 41 locations if the “negative” status register is set.

0000 100 000101001

17. Some microprocessors have a no-op instruction (a NO-OPERation instruction does nothing; it has no effect on any register). How would you make a no-op for the LC3? Hint: you can use one of the instructions in the table above for this.

0000 000 000000000 If the second group of numbers is 000, the third group does not matter

18. Draw a simple figure that represents how a sequential circuit operates. You'll have boxes for "input," "output," "combinational logic," "storage elements," and some arrows.



19. What do FSM, SEXT, MAR, and PC stand for? (1 point each).

Finite State Machine
 Sign Extension
 Memory Address Register
 Program Counter

20. Using a conditional branch instruction, how far away from the current program counter can you jump?

+255 to -256

21. What does the assembly instruction AND R2,R4,#0 do?

Clears R2

22. Write a few lines of assembly language that results in $R0 = R1 - R2$.

```

NOT R2, R2
ADD R2, R2, #1
ADD R0, R1, R2
  
```

23. The LC3 has an “indirect” addressing mode. Briefly describe what happens with an LDI or STI instruction.

The memory location indicated in the instruction is queried.

Then, the value in that location is used to access memory again

Reg = Mem[Mem[location specified in the instruction]]

24. What numbers are in registers R1, R2, and R3 when the following code finishes?

```

LD R1, A
LD R2, B
AND R3, R3, #0
X  ADD R3, R3, R2
   ADD R1, R1, #-1
   BRp X
   HALT
A  .FILL 5
B  .FILL 8    R3=40 R1 = 0 R2 = 8

```

25. The PostScript programming language uses a stack to do its math. To compute $(5+3)$, you would write a program that does:

Push 5

Push 3

ADD

In addition to add, there is also MUL (multiply).

Write a short PostScript program to compute $(3+5)*(2+3)$

Push 3

Push 5

ADD

Push 2

Push 3

ADD

MUL

26. Bonus question. What food is shown on the CS210 web site, above the word “Potential?”

French Fries