1

Assignment: Arithmetic and Logic

Instructor: Elaheh Sadredini

elaheh@cs.ucr.edu

University of California, Riverside

2

Grading (total 4.2 points)

- Question 1: 0.6 point
- Question 2: 0.2 point
- Question 3: 1.2 point
- Question 4: 0.6 point
- Question 5: 0.2 point
- Question 6: 1.4 point

Question 1: Two's Complement

Convert the following decimal numbers into 4-bit two's complement:

1. 4
2. -4
3. -8
4. 8

Answer

1. 4 --> 0100 --> 1011 --> 1011 + 1 = 1100
2. -4 --> 1100 --> 0011 --> 0011 + 1 = 0100
3. -8 --> 1 1000 --> 0 0111 --> 0 0111 + 1 = 0 1000
   (overflow since range is from -8 to +7)
4. 8 --> 0 1000 --> 1 0111 --> 1 0111 + 1 = 1 1000

Question 2: Two's Complement Subtraction

Compute the following subtraction in 7-bit two's complement using the A-B = A+(!B+1) formula: **8 – 12**

Write the results in binary.

Answer

- Write the final result in binary in 7-bit format

A - B = A + (!B + 1)
8 - 12 = 8 + (!12 + 1)
000 1000 - 000 1100 = 000 1000 + (!(000 1100) + 1)
= 000 1000 + 111 0100
= 111 1100

Question 3: Overflow in Two's Complement

Compute if the following operations overflow in 5-bit two's complement. For the answer, provide the carry bits, whether there is an overflow, and the decimal equivalent operation

1. 01000 – 11000
   - Result and carry-out bit:
   - Does it overflow?
   - Decimal version:

2. 10000 + 01111
   - Result and carry-out bit:
   - Does it overflow?
   - Decimal version:

Answer

- Write the final result. For "overflow" question, shortly explain your answer in one sentence.

1. 01000 - 11000
   = 01000 + 01000
   = 10000
   Result and carry out-bit: 10000, none
   Does it overflow: no overflow since there is no carry out bit.
   Decimal: -16
2. 10000 + 01111
   = 10000 + 10001
   = 1 00001
   Result and carry out-bit: 00001, 1
   Does it overflow: yes because there is a carry out bit and 100001 is not the same as 00001.
   Decimal: +1

Question 4: Floating Point

Below is a simplified **8 bit floating point format**, with 1 bit for sign, 4 bits for the fraction, and 3 bits for the exponent. (Format: Sffffeee = $(-1)^S$x0.ffffx$2^{(eee-4)}$) For each example, write the decimal equivalent.

1. 1 0101 110
   - Decimal:
2. 0 1111 111
   - Decimal:
3. 0 0001 000
   - Decimal:

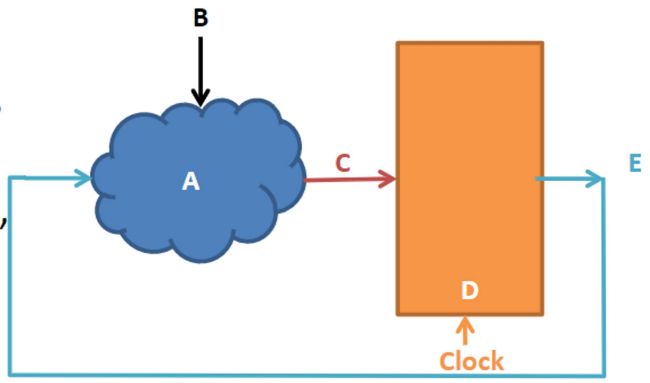
Answer

1. 1 0101 110
   -1 x 0.0101 x 2^(110-4)
   = -1 x 0.5 x 2^(6-4)
   = -2
2. 0 1111 111
   = 1 x 0.1111 x 2^(111-4)
   = 1 x 0.15 x 2^(7-4)
   = 1.2
3. 0 0001 000
   = 1 x 0.0001 x 2^(000-4)
   = 0.00625

Question 5: State

What do parts A-E need to do in this circuit to make a counter?

1. A: adder, B: input 1, C: next_count, D: state latch, E: current_count
2. A: adder, B: input 1, C: current_count, D: state latch, E: next_count
3. A: adder, B: next_count, C: current_count, D: state latch, E: current_count
4. A: state latch, B: input 1, C: current_count, D: adder, E: next_count



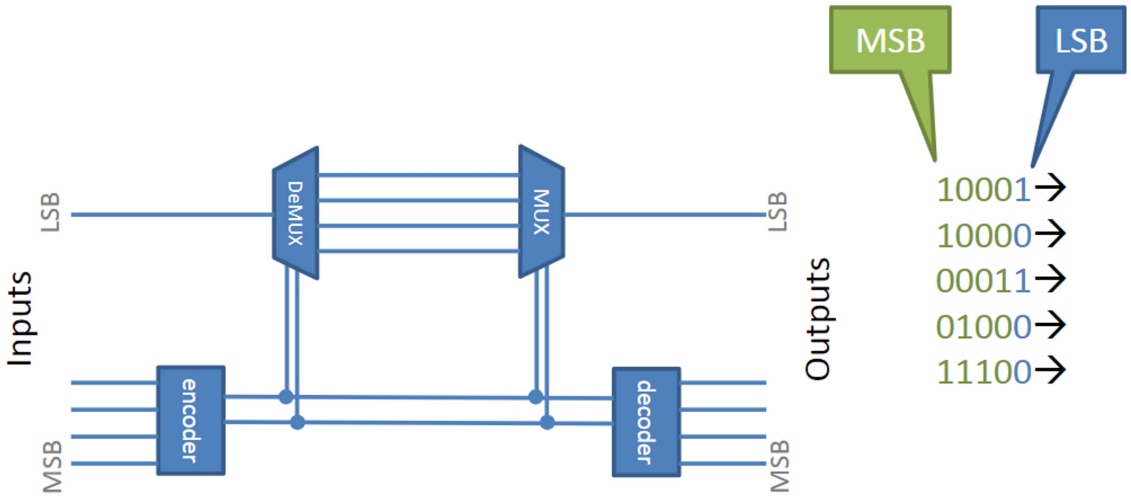
Answer

- Choose the correct answer.

1. A: adder, B: input 1, C: next_count, D: state latch, E: current_count

Question 6: Logic blocks

A: What is the output for each of the input shown below.
B. What does the block do?



Answer

- For part A, write the output for each input. For part B, briefly explain your observation.

A. 10001 --> 10001
10000 --> 10000
00011 --> 00011
01000 --> 01000
11100 --> unknown

B. The block encodes and decodes the MSB, and using the encoded MSB chooses which output to output to, then using the same encoded MSB, chooses which input signal to output as the new LSB. Overall the system block seems to not do much other than change the input and then change it back to what it was originally and then output that.

1000 --> 11 --> 1000
0001 --> 00 --> 0001
0100 --> 10 --> 0100
1110 --> unknown