Q1 Data Multiplexors

3 Points

You can find the lecture slides for todays lecture here!

You can access the YouTube playlist here!

Q1.1 Which of the following are true?

3 Points

The MUX is most similar to:

○ Variable assignment

○ The `while` loop

● The `if` statement

If I wish to select between n different inputs to a mux, how many selector bits will I need?

○ n

○ 2^n

● $\log_2(n)$

If the boolean expression for a 2-to-1 MUX is $\overline{s}a + sb$, then what should the boolean expression for a 4-to-1 MUX be? (Let a, b, c, d, s_0, s_1 be the 4 inputs and 2 selector bits, respectively)

○ $s_1 s_0 a + \overline{s_1} s_0 b + s_1 \overline{s_0} c + \overline{s_1 s_0} d$

○ $s_1 s_0 a + s_1 \overline{s_0} b + \overline{s_1} s_0 c + \overline{s_1 s_0} d$

● $\overline{s_1 s_0} a + \overline{s_1} s_0 b + s_1 \overline{s_0} c + s_1 s_0 d$

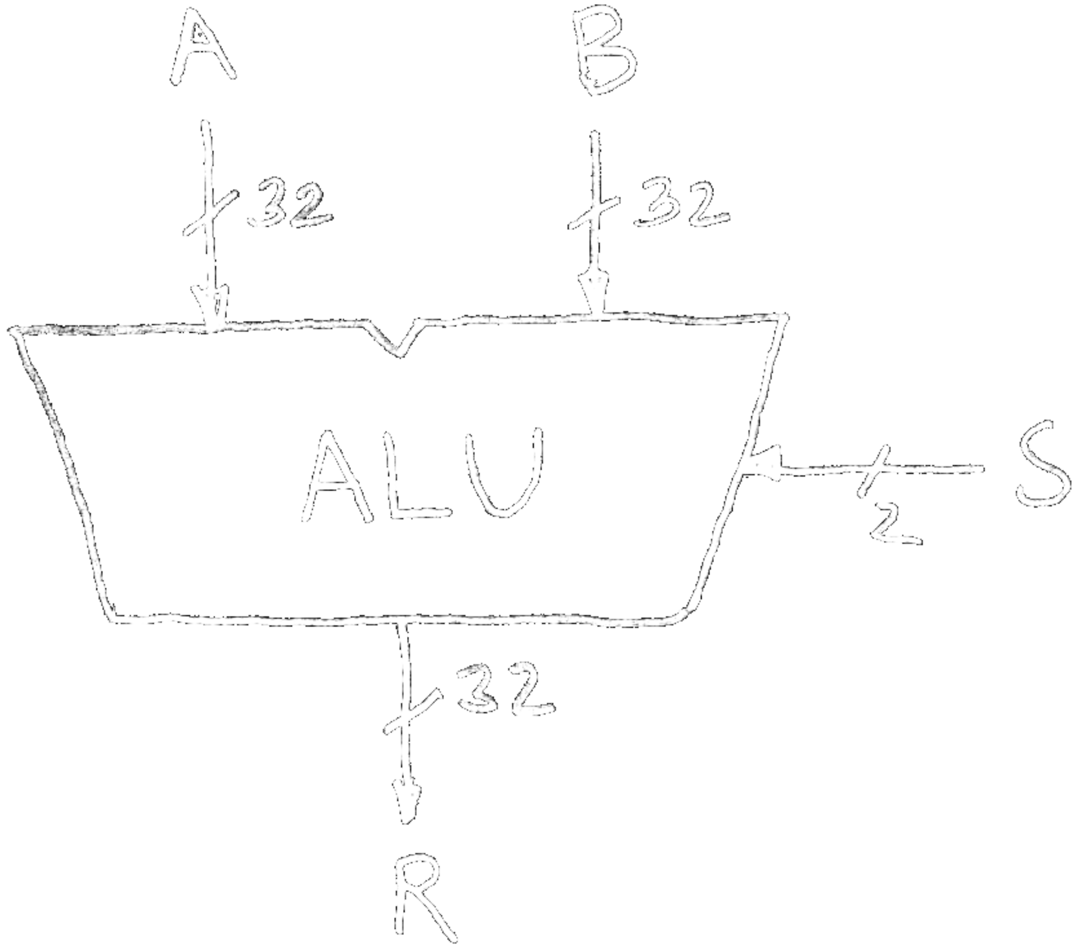
○ $\overline{s_1 s_0} a + s_1 \overline{s_0} b + \overline{s_1} s_0 c + s_1 s_0 d$

Q2 ALU

4 Points

Q2.1 Which of the following are true?

4 Points



Our ALU will only be able to compute 2 different things, since S is only 2 bits

○ True

● False

The signal bits S control which operation our ALU does. The machinery for our hardware for the operations not selected are not active.

○ True

● False

We're going to try to build an adder/subtractor box and reuse machinery (rather than build a separate adder and separate subtractor)

● True

○ False

We’re going to try to build an ANDer/ORer box and reuse machinery (rather than build a separate ANDer and separate ORer)

○ True

● False

Q3 Adder/Subtractor

2 Points

Q3.1 How do we calculate the overflow of two signed numbers?

2 Points

○ We OR together the c_n and c_{n-1} bits

○ We AND together the c_n and c_{n-1} bits

● We XOR together the c_n and c_{n-1} bits

○ We XNOR together the c_n and c_{n-1} bits

Q4 Subtractor Design

4 Points

Q4.1 Which of the following are true?

4 Points

How did we implement subtraction?

- □ We built a whole new subtraction unit
- ☑ We leveraged the 2's complement trick of negating a number by flipping the bits and adding 1
- □ We leveraged the 2's complement trick of negating a number by flipping the bits
- ☑ We leveraged the mathematical fact that $a - b == a + (-b)$
- □ We flipped one input's bits by ANDing them all with 1
- □ We flipped one input's bits by ANDing them all with 0
- □ We flipped one input's bits by ORing them all with 1
- ☑ We flipped one input's bits by XORing them all with 1
- □ We flipped one input's bits by XORing them all with 0
- □ We added 1 to one input by adding one more addition unit that will manually add a constant 1
- ☑ We added 1 to one input by inserting a carry-in bit c_0
- ☑ We connected the flipping bit and the +1 together

Lecture 17 - Combinational Logic Blocks

● UNGRADED

STUDENT

Shauna Hannani

TOTAL POINTS

- / 13 pts

QUESTION 1

Data Multiplexors		3 pts
1.1	Which of the following are true?	3 pts

QUESTION 2

ALU		4 pts
2.1	Which of the following are true?	4 pts

QUESTION 3

Adder/Subtractor		2 pts
3.1	How do we calculate the overflow of two signed numbers?	2 pts

QUESTION 4

Subtractor Design		4 pts
4.1	Which of the following are true?	4 pts