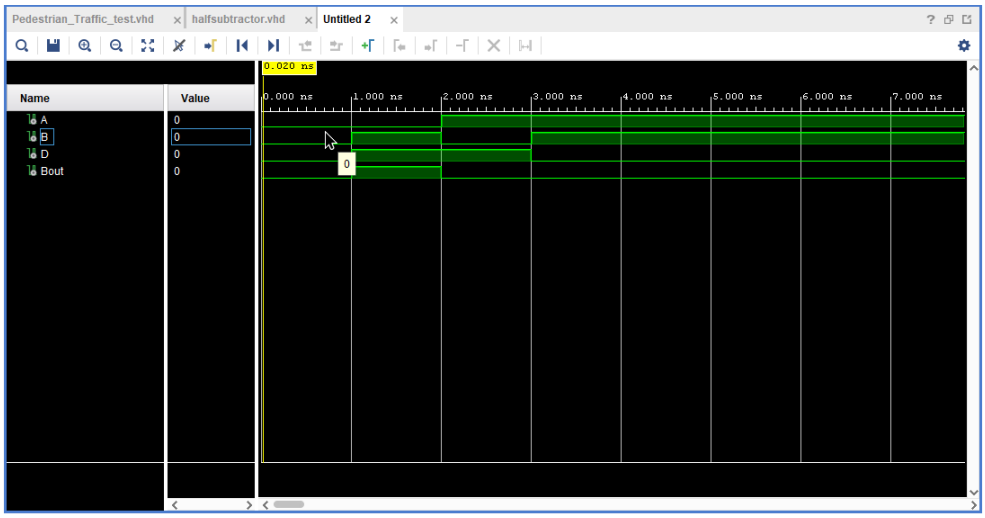
Lab 3

Madiba Hudson-Quansah



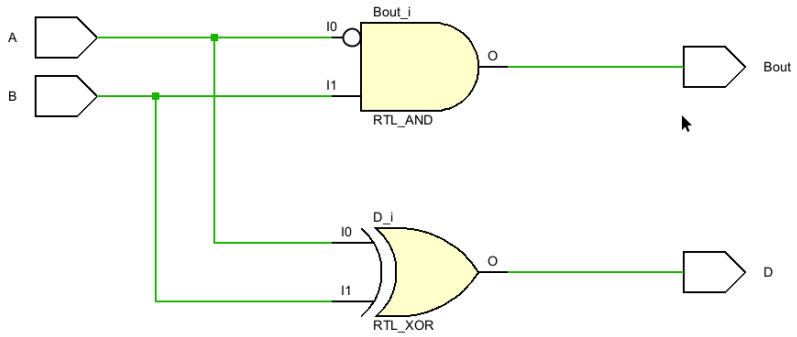
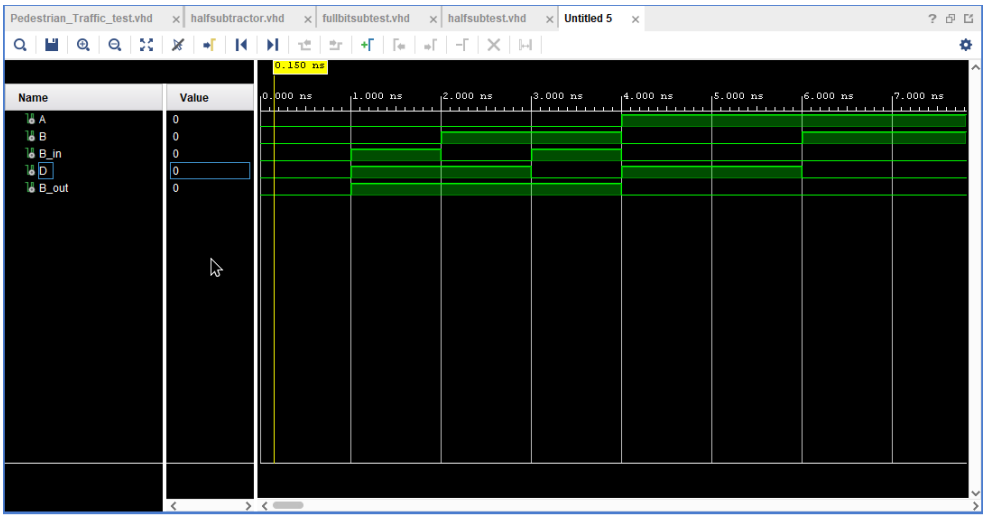


Figure 1: Half Subtracter



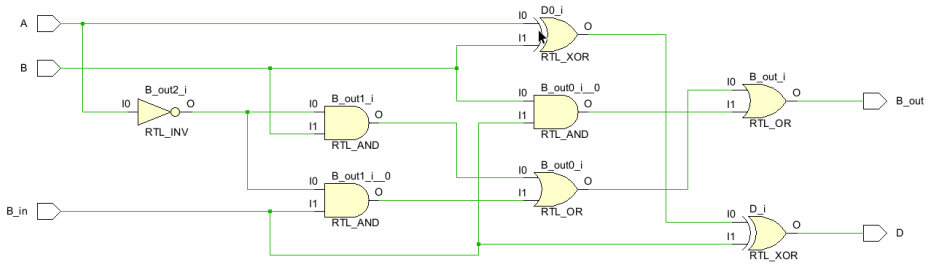


Figure 2: Full Subtracter

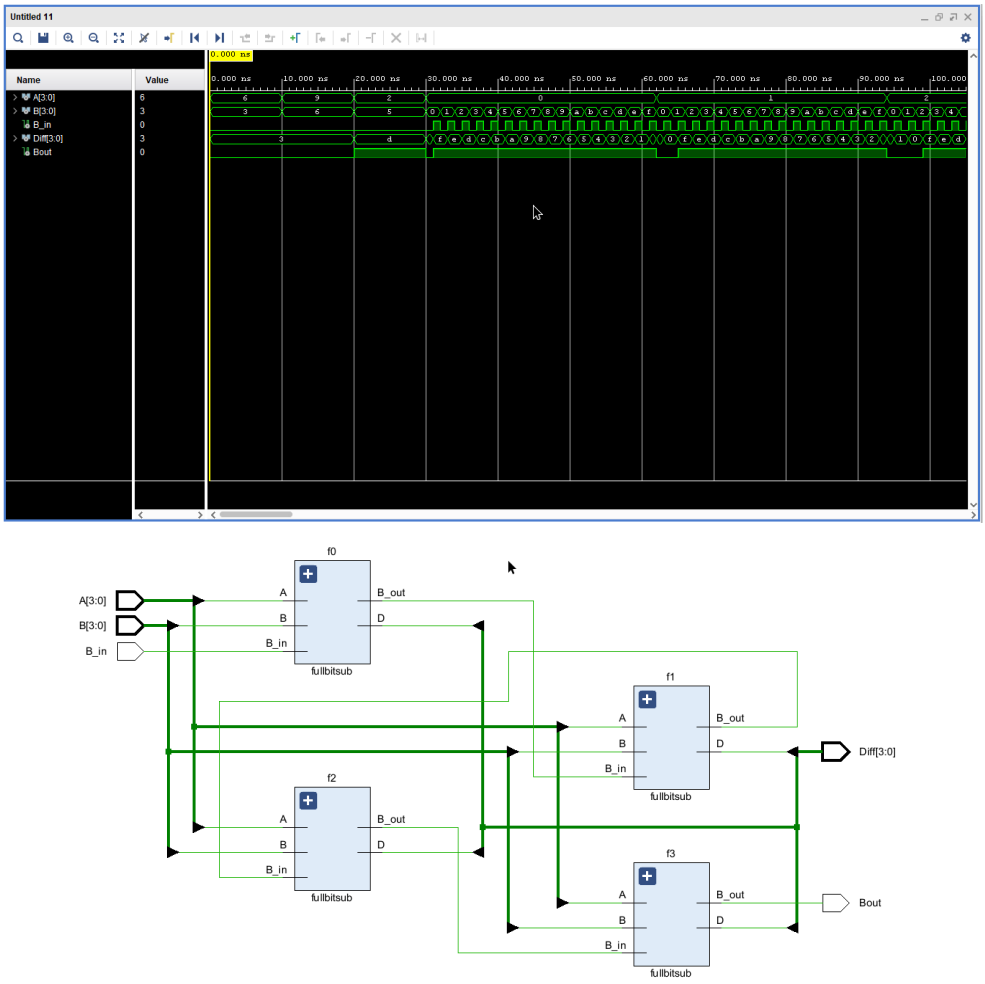


Figure 3: Four Bit Subtracter

A	B	Difference	Borrow Out
0110	0011	0011	0
1001	0110	0011	0
0010	0101	1101	1

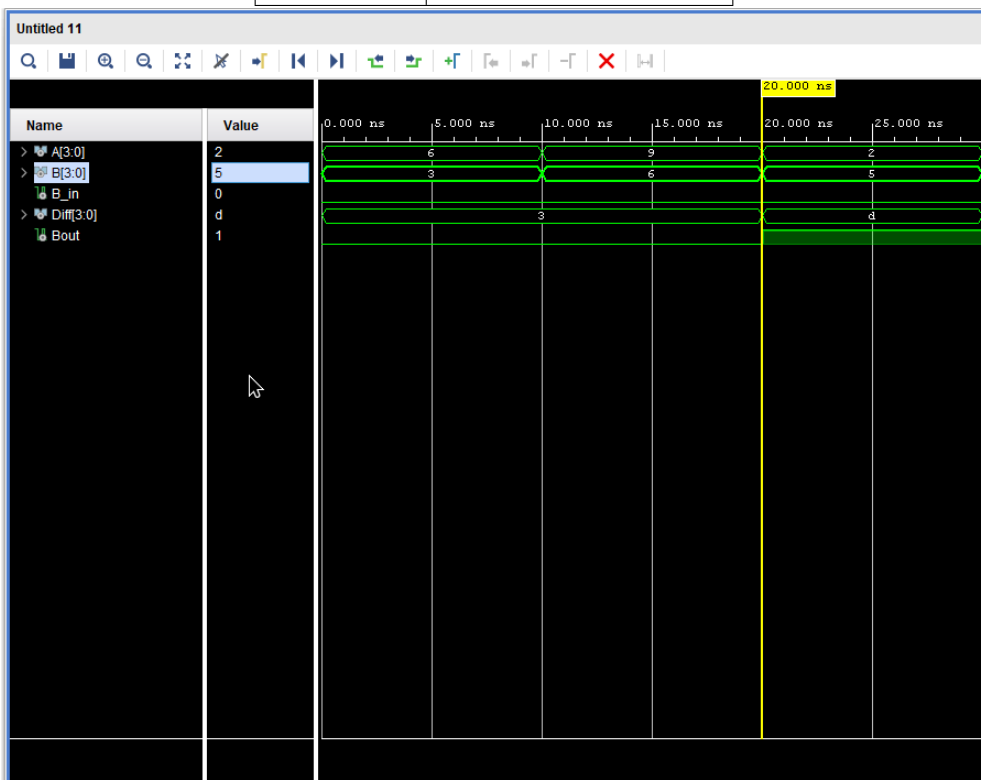


Figure 4: Four Bit Subtracter Table

Question 1

Pedestrian Crossing

1. Using Karnaugh maphs, provide the state equations
2. Implement your circuit design in Vivado providing a screenshot of your schematic design and simulation with signals in ns not ps. Name file as Pedestrian_Traffic.vhdl andtestbench as Pedestrian_Traffic.vht.
3. Is this a Moore or Mealy Machine? Justify.

Solution:

Current State	Next State	D_1	D_2
00	01	0	1
01	10	1	0
10	00	0	0

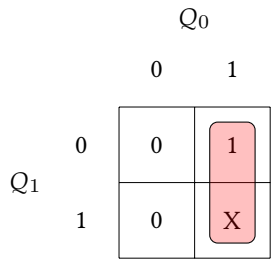


Figure 5: D_1

$$Q\,(t+1) = Q_0\overline{Q_1}$$

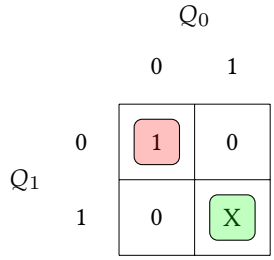
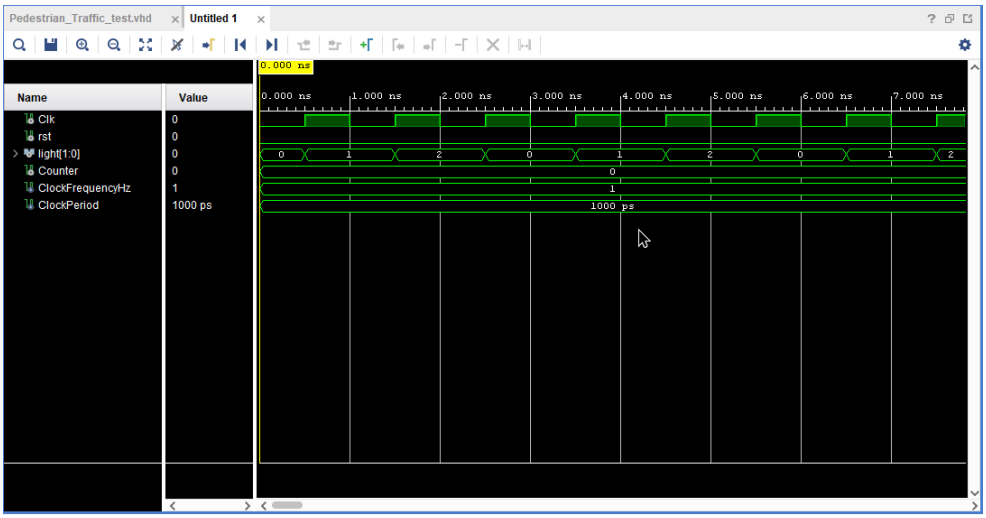
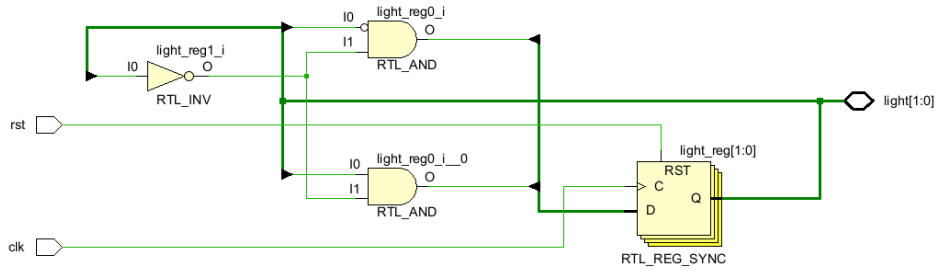


Figure 6: D_0

$$Q\,(t+1) = \overline{Q_1 Q_0}$$

1.





2.

3. This is a Moore machine, as the output, i.e. the lights, depends solely on the state.

Question 2

Conceptual Understanding

1. What is the fundamental difference between a half subtractor and a full subtractor?
2. Why does a full subtractor require a borrow-in, while a half subtractor does not?

Solution:

1. A half subtracter does not take in a borrow in bit while a full subtracter does.
2. A full subtracter needs a borrow in bit to account for previous borrows outs from possible previous subtracters. This allows multiple subtracters to be chained together allowing for subtraction of progressively larger bit widths.

Question 3

Logic and Implementation

1. What happens when you subtract a larger number from a smaller number in unsigned binary subtraction? How does the borrow bit behave?
2. If we cascade full subtractors to create an N-bit subtractor, how does the borrow propagate through the chain?

Solution:

1. When subtracting a larger number from a smaller number the operation underflows since unsigned numbers cannot be negative, i.e. the largest number that can be represented is subtracted from the smallest number that can be represented. The borrow bit is then set to 1 indicating an underflow occured
2. Each successive subtracter will take in the borrow out from the previous subtracter as its borrow in with the first subtracter in the chain taking in the original borrow in.

Question 4

Overflow & Sign in Signed vs. Unsigned Numbers

1. What is the difference between sign and overflow in signed and unsigned number systems?
2. Can an overflow occur in an unsigned subtraction? Explain why or why not.
3. How does two's complement affect the way we handle subtraction in signed binary numbers?

Solution:

1. Sign refers to whether a number is positive or negative and only signed numbers have a sign bit that represents this. Overflow refers to when the result of an operation exceeds the limits of the current number system and so wraps around to be contained in the number system again.
2. In the case of unsigned subtraction, overflow cannot occur as the result will always be positive and within the bounds of the number system, underflow can occur however if the result is negative.

Question 5

FSM Understanding

1. How does the system transition between states without requiring an explicit input?
2. Why do we include the clock (CLK) and reset (R) signals, even in an automatic FSM?
3. What would happen if we added a pedestrian push button as an input? How would this change the FSM design?
4. If we wanted to add a yellow (caution) phase before stopping, how would the state diagram and VHDL implementation change?

Solution:

1. The system transitions based on the rising edge of the clock signal this allows the system to transition between states without requiring an explicit input.
2. The clock signal is used to synchronize the system and ensure that the system transitions between states at the correct time. The reset signal is used to reset the system to a known initial state
3. The FSM would have to include a state for the pedestrian push button and the system would have to transition to the stop state when the button is pressed.
4. The state diagram would now have four states instead of three, which is possible to include in the current implementation as we disregard the 11 state. This new state would be in-between the BLINK and STOP states making the state assignment

   - S_0 / 00 - STOP
   - S_1 / 01 - GO
   - S_2 / 10 - BLINK
   - S_3 / 11 - YELLOW

   This would change the next state equations for D_1 and D_0 flip-flops to include the new state. Another possible implementation in VHDL would just to detect the current state enum and update the light outputs accordingly in a switch statement, i.e. if the current state is STOP, we set the State to GO and set the light output to 01.