Lab 8: Asynchronous Serial Transmitter

Shrey Patel 2019CS10400
Aayush Goyal 2019CS10452

May 2022

Contents

1 Implementation

1.1 Display

- The display of the received number is same as that in assignment 7, i.e. two digits on the seven-segment display. The number transmitted by the board is visible directly on the gtkterm console.

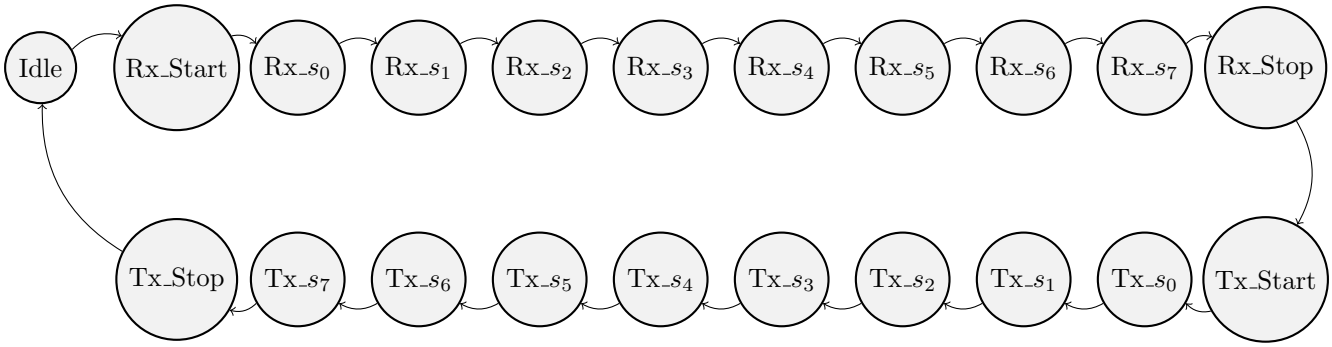
1.2 States

- The states of the receiver are the same as in assignment 7 i.e. idle, start, data and stop states. We will reuse the same states for the transmitter as well. But, to distinguish them both, we use an additional variable ***transmit***. The logic associated with the states remain the same.
- So, the states are as follows:
  1. Common idle state(9)
  2. Transmit = 0, Start(10), Stop(8) and Data(0-7) states -> Receiver States
  3. Transmit = 1, Start(10), Stop(8) and Data(0-7) states -> Transmitter States

1.3 State Transitions

- The transitions related to the receiver are same as in assignment 7. The only difference is that now, after successfully receiving '1' as the stop bit, the receiver doesn't move to the idle state but to the start state of the transmitter. Additionally, while our FSM is in the receiver mode, we need to keep the transmitter idle. For that, we keep sending the bit '1' while in receiver mode.
- Now, in transmitter mode, we generate a pattern by replicating the input received in receiver mode with only difference i.e. we send 16 '0' bits in the start state instead of 8. Similarly, in stop state of the transmitter mode, we send 16 '1' bits. After that, the FSM moves to the idle state and switches the transmit bit to '0' since FSM can become active only in the receive mode.

- In the data states, now, instead of waiting 15 cycles, we take the bit obtained in the same data state while in receiver mode earlier and send it in all the 16 cycles. So, all data bits will be sent 16 times.



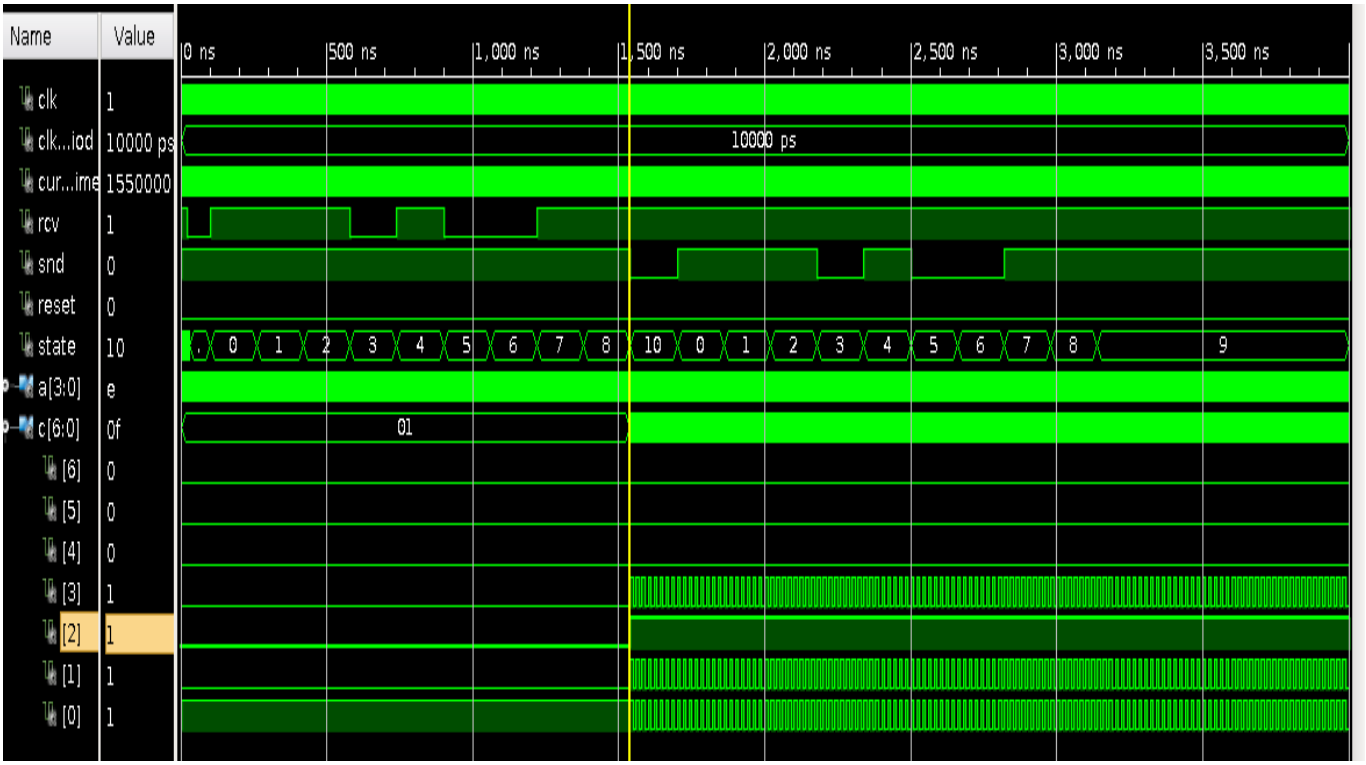
1.4 User Input

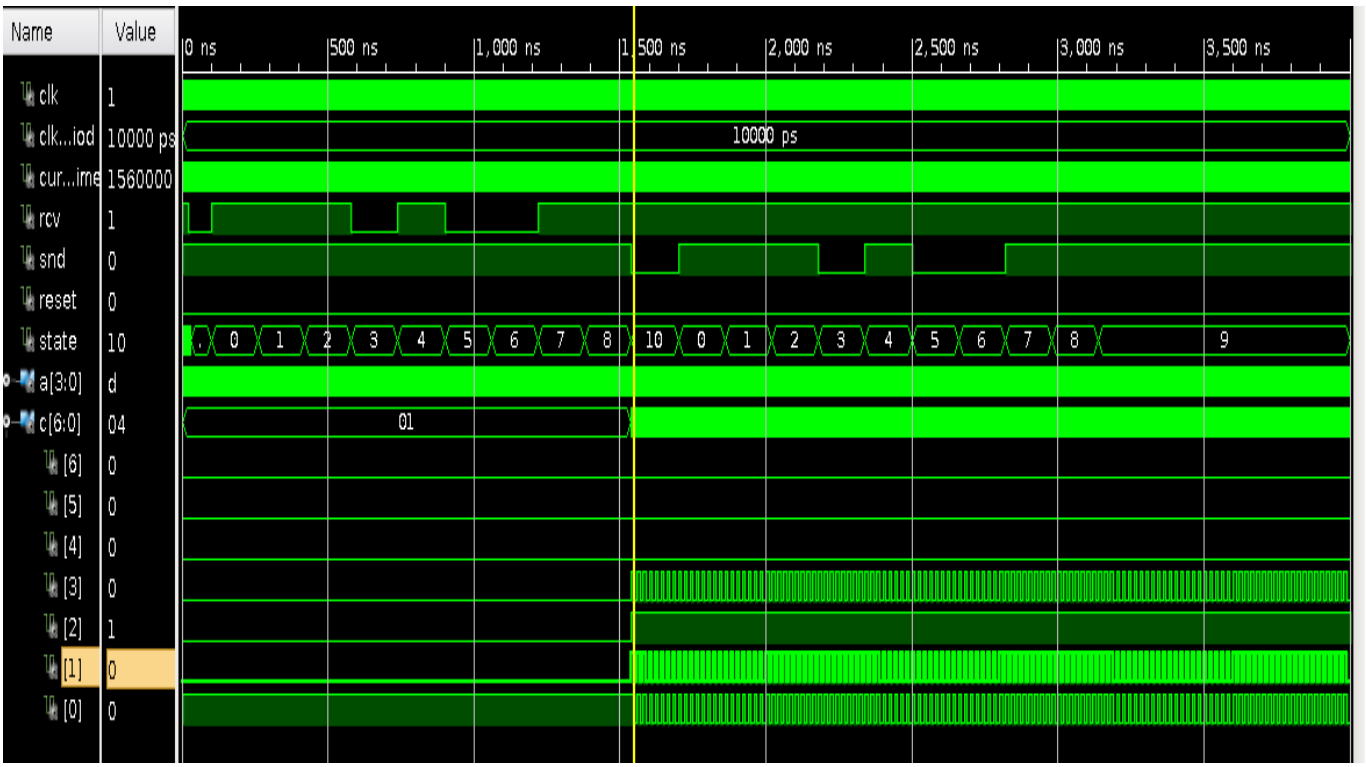
Similar to assignment 7, the reset button is mapped to the centre button of the board (BTNC, V18). On pressing it, the FSM moves to the idle state, zeroes state_counters and also resets the displayed number to "00".

2 Simulation

2.1 State Transitions

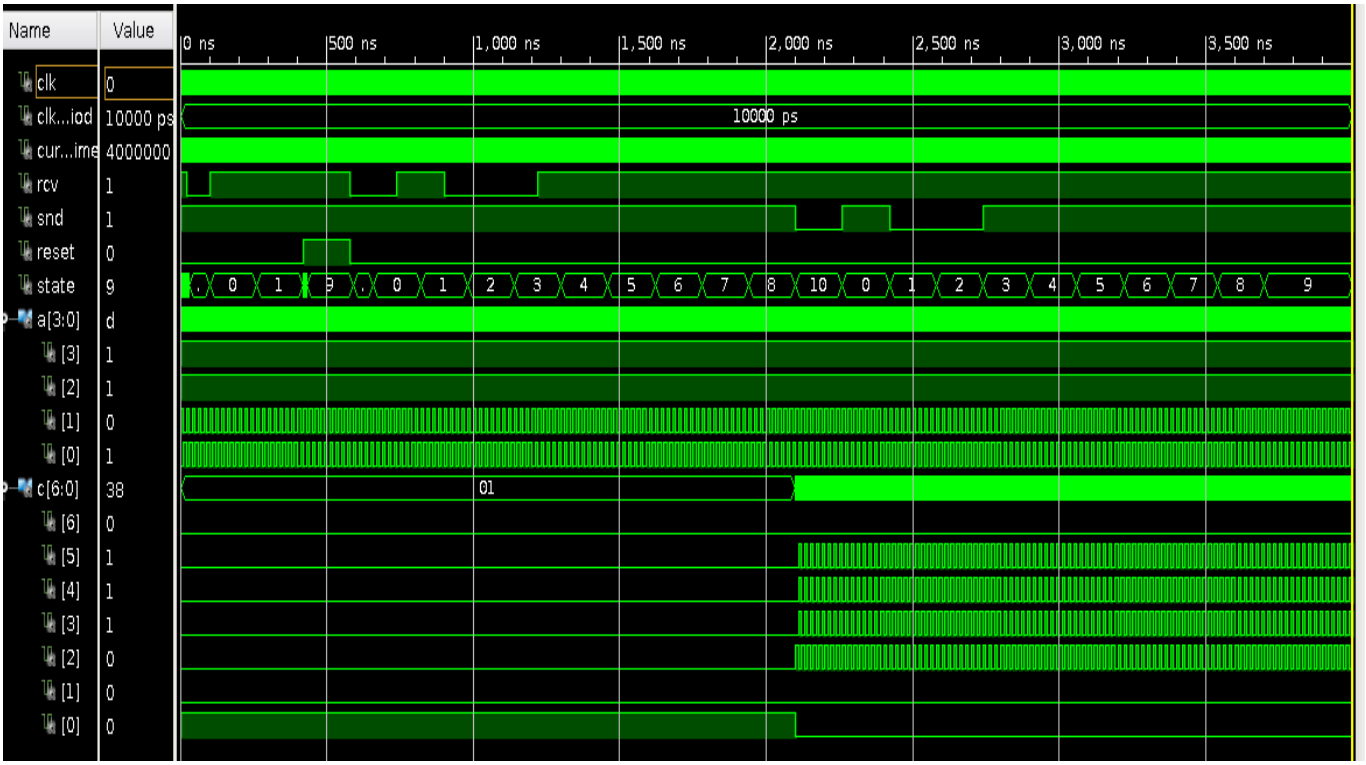
- The first simulation shows the state changes related to the FSM. The first 1540ns are same as in assignment 7. This is expected since the receiver mode has been left unchanged. Similarly, the anode and cathode configurations show that the number on display is "97" as last time.
- But this time, the FSM doesn't move to the idle state 9 but again to 10 which is the start state of the transmitter. Also the width of the start state is now double (i.e. 16 cycles) of the start state width of the receiver.
- Rest of the states are same as that of the receiver. The FSM finally moves to the idle state 9 from the stop state of the transmitter.
- Another thing to observe is that the sequence of received bits starting from 100ns till 1380ns exactly matches with the sequence of transmitted bits starting from 1860ns till 3140ns which proves that the received and transmitted numbers are the same.





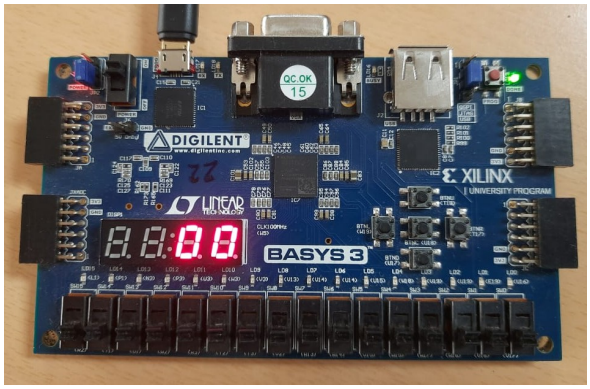
2.2 Reset

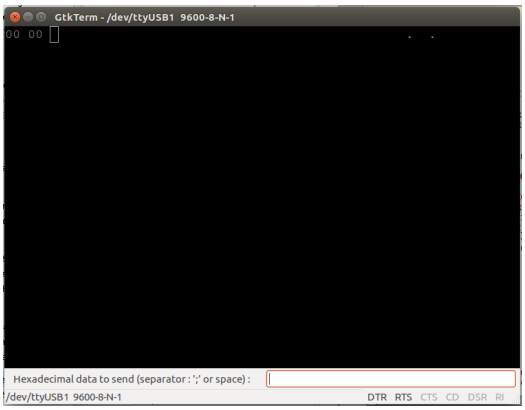
Similar to Assignment 7, the reset buttons resets the display and moves the FSM back to the idle state(9). This can be seen from the simulation below at 420ns.

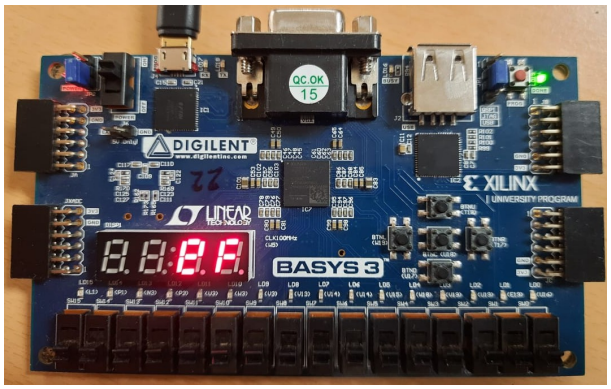


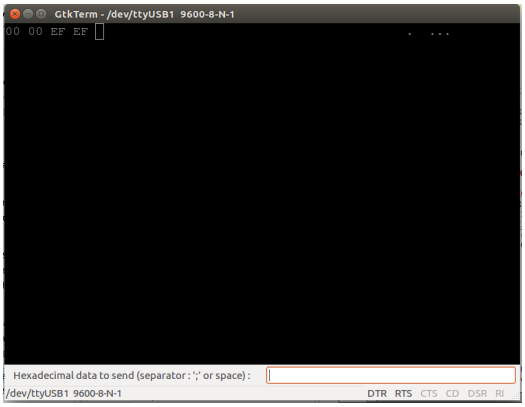
3 Output

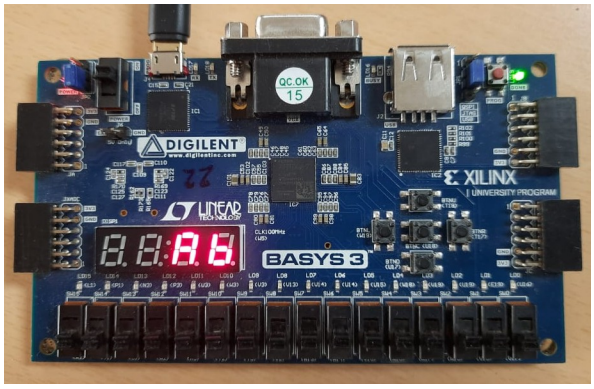
- Order of input: 00 => EF => AB => 56
- Note that the local echo is enabled in each of the below figures. So, after entering any input, we can see the copy of the input followed by the received(i.e. transmitted by the board) output.

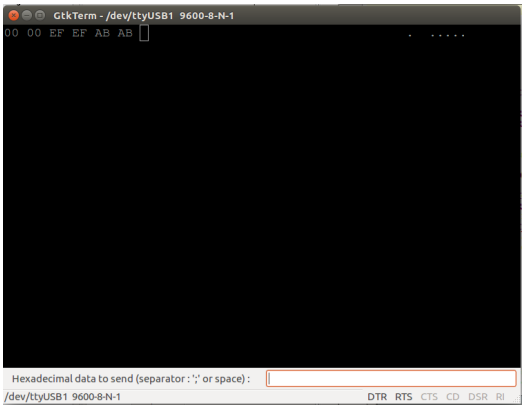


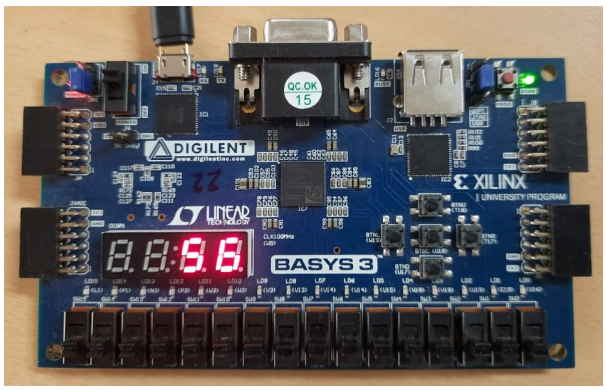


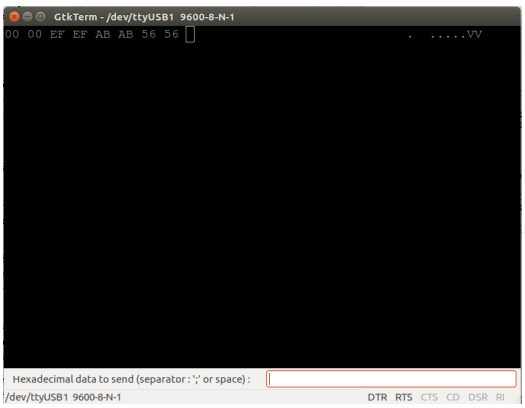












4 Resource Utilization

- **LUTs:** 162
- **Flip Flops:** 113
- **Block RAMs:** 0
- **DSPs:** 0