CS288 Documentation
Universal Asynchronous Receiver Transmitter (*Group 02*) Report

Anmol Garg 110050020
Mohit Gupta 110050085
S S Kausik 110050003
Vibhor Kanojia 110050036
Avinash Palaparthi 110050064

April 14, 2013

1 Specifications

- $send$: Send input to the Transmitter. (SW0)
- $input$: parallel input to the Transmitter.(7-bit : SW1-SW7).
- clk_in : FPGA Clock of 19.2 Mhz to the Frequency Modulator.
- $output$: Output from the receiver.(7-bit : LD1-LD7)
- $reset$: reset to the Transmitter.(F5)

2 Implementation of the UART

The UART consists of three components, a Transmitter , A Receiver and a Frequency Modulator to provide the modulated frequencies. The output of the Transmitter is connected to the input of the Receiver. The switches (SW1-SW7) are used to give the parallel input. When SW0 will be switched on(the $send$ is high), data is transmitted from the Transmitter. This data is received by the Receiver, interpreted accordingly, and the corresponding output is reflected on the LEDs(LD1-LD7). To transmit again, press the Reset button(F5)

3 Transmitter

3.1 Specifications

3.1.1 Input

- $send$: High input signal to start Transmission of data.
- $input$: 7-bit data received parallel to the Transmitter.
- clk_in : Clock Input to Transmitter. Baud rate at 19.2 Khz.
- $reset$: Resets the Transmitter to idle state.

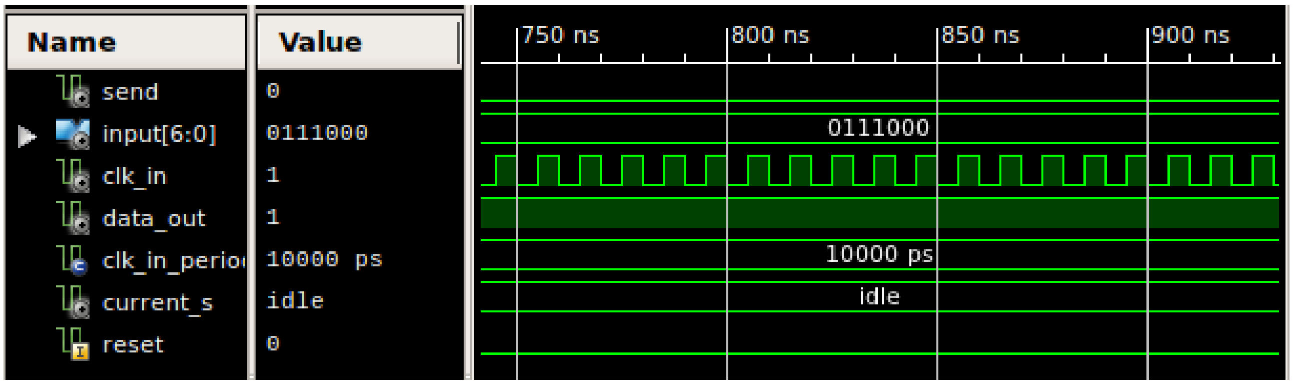
3.1.2 Output

- $data\_out$: Output line of the Transmitter. Connected to Receiver module.

3.2 Working of the Transmitter

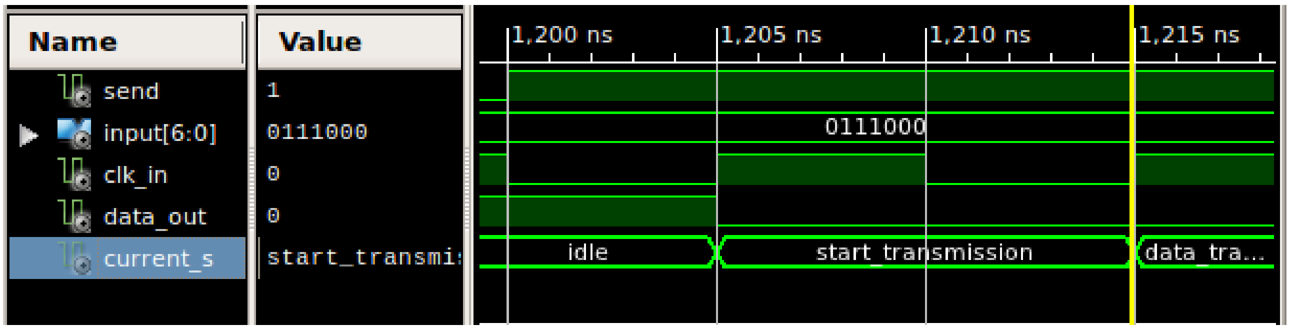
The data transmission form the transmitter happens in the following way:

1. Initially the transmitter is in the $IDLE$ state. In $IDLE$ state, the 7-bit parallel input is stored at the next clock edge and $data\_out$ is high in this state, indicating an inactive line. This data loaded into the transmitter will be transmitted when the $send$ input is set to high.



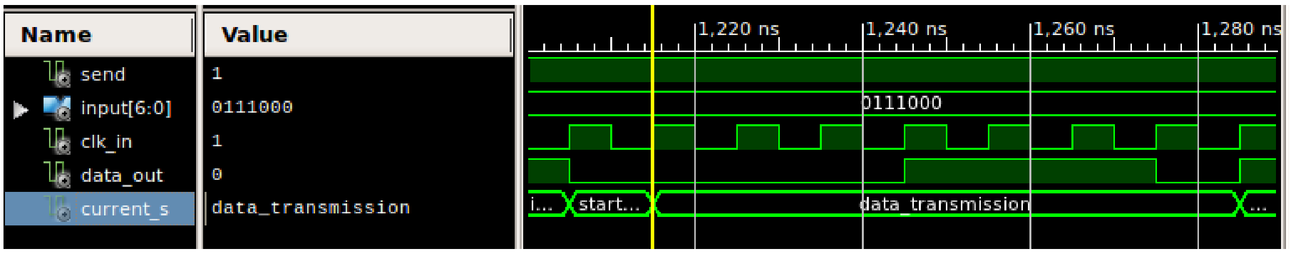
Name	Value
send	0
input[6:0]	0111000
clk_in	1
data_out	1
clk_in_perio	10000 ps
current_s	idle
reset	0

2. To start the Transmission, the $send$ input is set to high. The transmitter detects the $send$ input and makes a transition from $IDLE$ to the $START\_TRANSMISSION$ state. This state indicates the start of transmission and the $data\_out$ sends a start bit which is 0 for a clock cycle to start transmission.



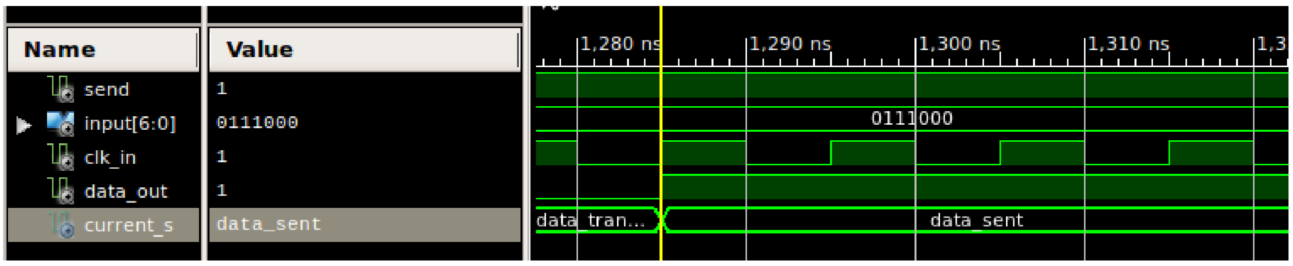
Name	Value
send	1
input[6:0]	0111000
clk_in	0
data_out	0
current_s	start_transmi

3. After the start bit has been sent on the $data\_out$, the transmitter changes state from $START\_TRANSMISSION$ to $DATA\_TRANSMISSION$. In this state, data stored in the transmitter is transmitted sequentially bit by bit on $data\_out$ from LSB to MSB. (each bit for one clock cycle). A counter keeps track of number of bits transmitted.

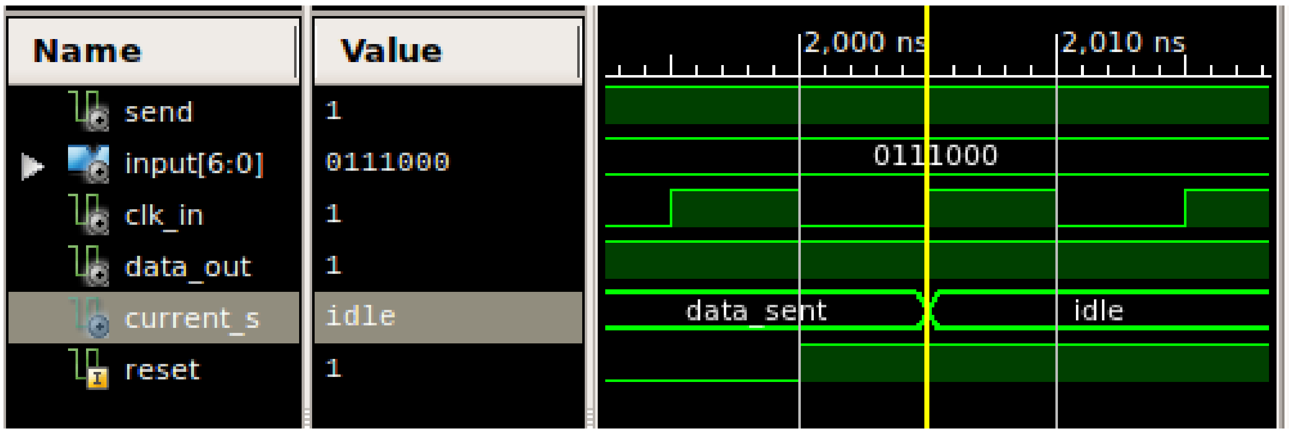


Name	Value
send	1
input[6:0]	0111000
clk_in	1
data_out	0
current_s	data_transmission

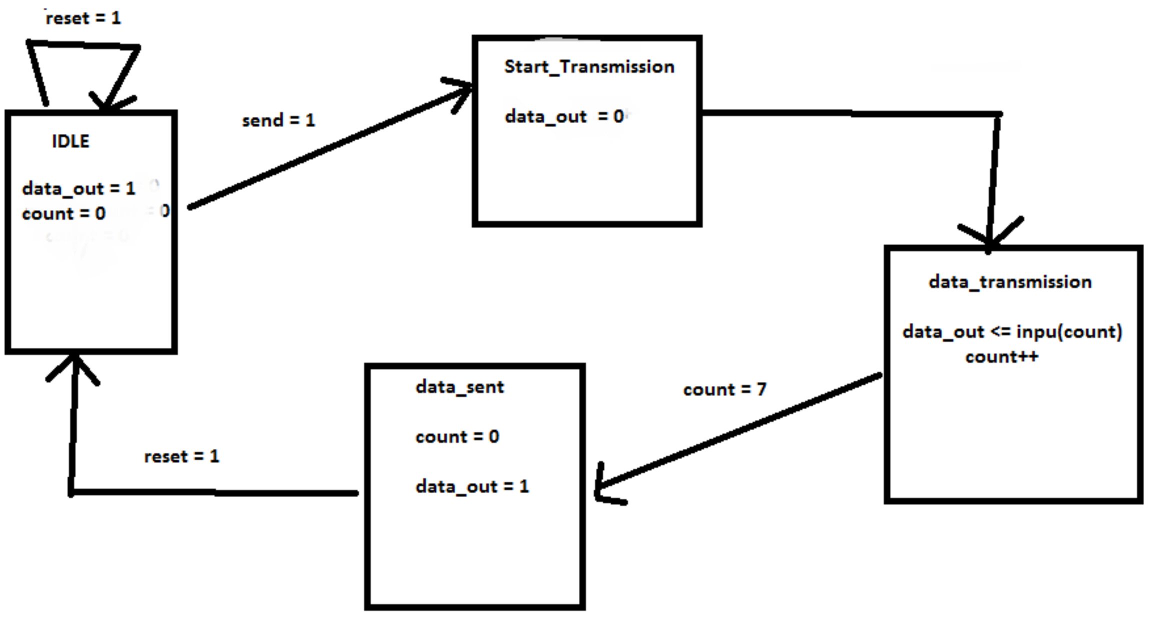
4. After all 7 bits have been transmitted on $data\_out$, the transmitter goes from $DATA\_TRANSMISSION$ to $DATA\_SENT$. This state acknowledges that data has been sent on the line. Now the transmitter sends the stop bit 1 to mark the end of transmission and the line is inactive again.



5. After the Transmitter is in $DATA\_SENT$ state, it does not transmit data anymore, even if you load the transmitter with new inputs. For the transmitter to transmit again, $reset$ input has to be set to 1. Then the transmitter goes from $DATA\_SENT$ to $IDLE$ and the loads the input again and is ready for transmission.



3.3 State Diagram



4 Receiver

4.1 Specifications

4.1.1 Input

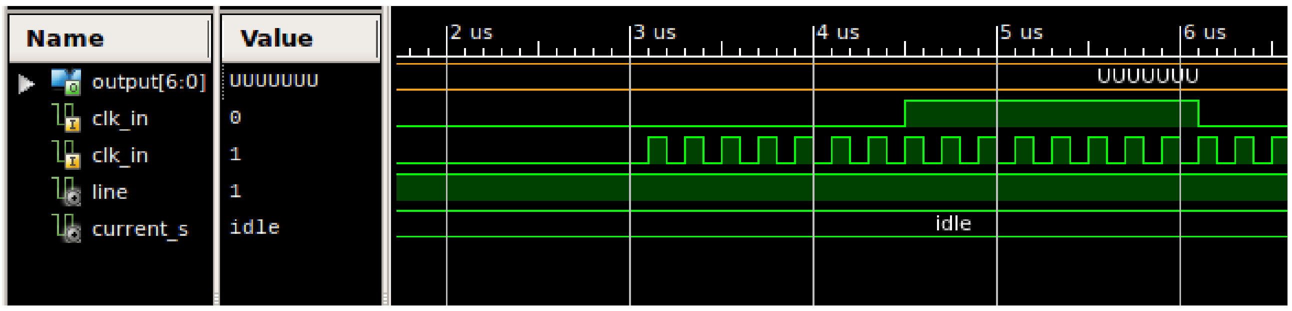
- $line$: the input data line of the receiver.
- clk_in : the clock working at 16F of the Transmitter clock Frequency.

4.1.2 Input

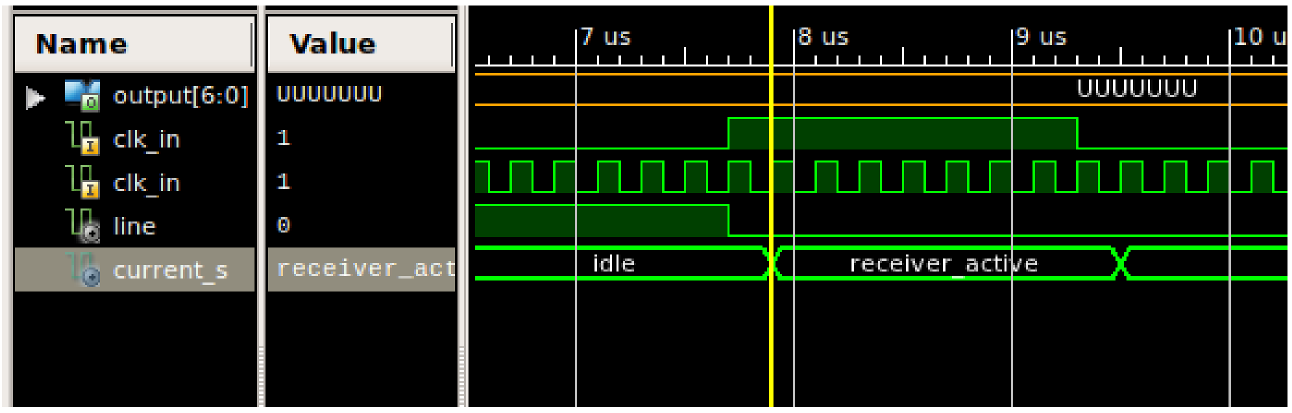
- $output$: the parallel output of the 7 bit signal (that is received sequentially);

4.2 Working of the Receiver

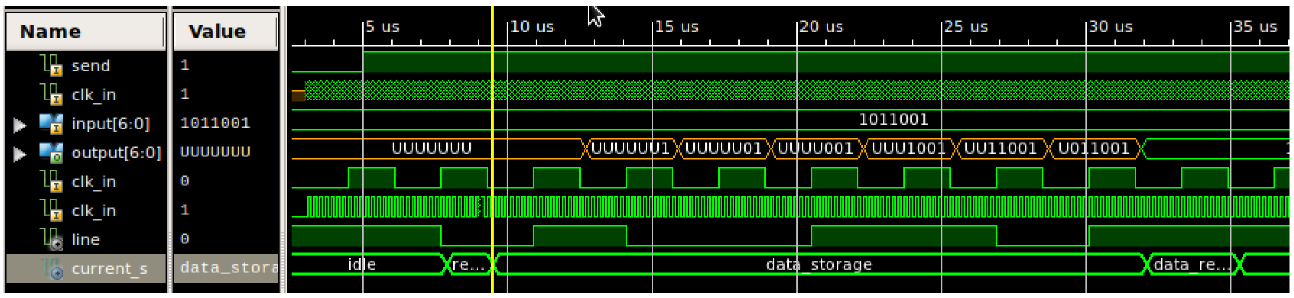
1. The receiver module operates a frequency 16 times that of the transmitter. It is always active(does not have an enable input). The receiver is in $IDLE$ state when the $line$ is 1(inactive). The output is the last received data to the receiver.



2. When the $line$ turns 0, the receiver changes state from $IDLE$ to $RECEIVER\_ACTIVE$ as a 0 on the line indicates the start bit and the line will transmit the bits one by one from LSB to MSB. The receiver counts 7 clock pulses from the start of the start bit (If valid start bit, receiver will reach the center of the start bit.)

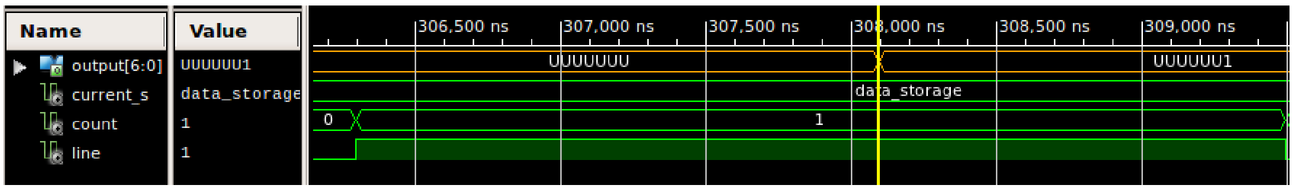


3. If the line is 0, which indicates that the line is transmitting a valid data packet(not noise on the line). The receiver changes state from $RECEIVER\_ACTIVE$ to $DATA\_STORAGE$. In the state, a count of number of bits received is stored and the output bits are assigned one by one.

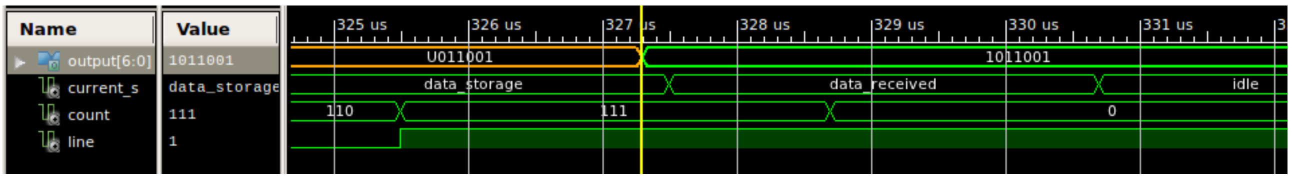


The bits on the line are read after 16 clock cycles. This is due to the fact that we are oversampling the data(16 times) to reduce error. We start from the center region of each bit and store its value. Then, after 16 clock pulses, we reach the center region of the next bit.

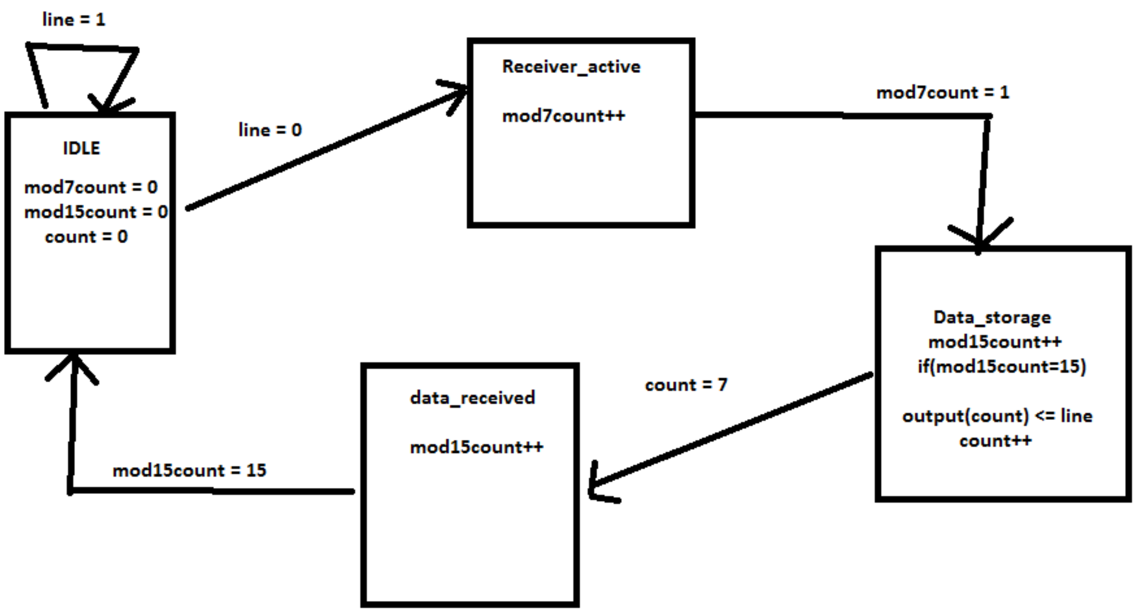
$$F_r = 16 * F_t$$



4. After all the 7 bits have been set to the output (count of bits is stored and incremented after 16 clock pulses), the receiver goes to $DATA\_RECEIVED$ state where all the bits of the output have been assigned. In this state, the receiver counts 16 more clock pulses to reach the stop bit on the line which is 1. Now the line is inactive and the receiver goes to $IDLE$ state, ready to receive data.



4.3 State Diagram



5 Frequency Modulator

Generic Modulator which takes ratio of frequency of the input and output clock and outputs a modulated clock.

$$F_{out} = F_{in}/Ratio$$