COL215 LAB Assignment 7 : Asynchronous Serial Receiver

Mohit Sharma(2019CS10372), Arka Mandal(2019CS50617)

1 Aim

Design asynchronous serial receiver with baud rate = 9600, 8 data bits, no parity bits and 1 stop bit. Connect this to the micro USB port of the BASYS 3 board. The receiver operates with a clock that is 16 x baud rate for proper detection of the start bit. Make a provision for resetting the FSM to idle state by a push button.

2 Design

The receiver takes input RsRx which are the bits received serially with a baud rate of 9600.
It also takes input of push button to synchronise gtkterm and the basys board, before this all data sent from pc is ignored.
The output of the receiver are the 8 bits sent from the pc which are shown in the seven segment display.
To receive bits a rxclk is generated which completes one cycle after every 650 cycles of clock from basys board. This represents 10432 clock cycles for 1 bit receive where the actual baud rate corresponds to 10416.67 clock cycles of clock from basys board. The above frequency of (650 clock cycles) ensures that data is received properly without any error.

There are 6 states in the receiver.

State : 5
This state waits for the user to press the center push button to synchronize gtkterm and basys board. Initially the receiver is in this state and never returns to this state after the user presses the button.

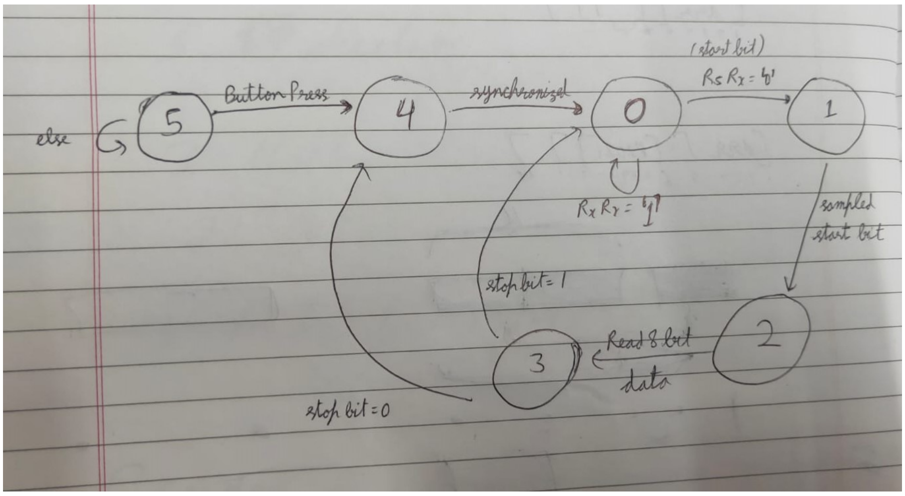
State : 4
This is synchronizing state. It does so by waiting for 10 continuous '1' bits input from the RsRx register. If 10 continuous bits are '1' then this implies that none of this bits corresponds to any data being sent from the pc as for a data to be sent, the first bit needs to be a start bit which is 0 and then 8 bits are data bits and last bit is stop bit. So this implies that pc is not sending any data and so it is in idle state. Hence receiver also transition to idle state.

State : 0
This is idle state. Receiver wait for start bit here. On receiving and '0' bit it transition to sampling that bit.

State : 1
This is a sampling state, it samples the start bit, means it checks that the start bit is '0' continuously for 8 clock cycles and this clock is run at 1/16th the rate at which 1 bit is received. So basically it divides the bit in 16 portions and checks is the first 8 portions are '0' or not.

State : 2
This states is used to read the 8 bits one by one at the middle approximately.

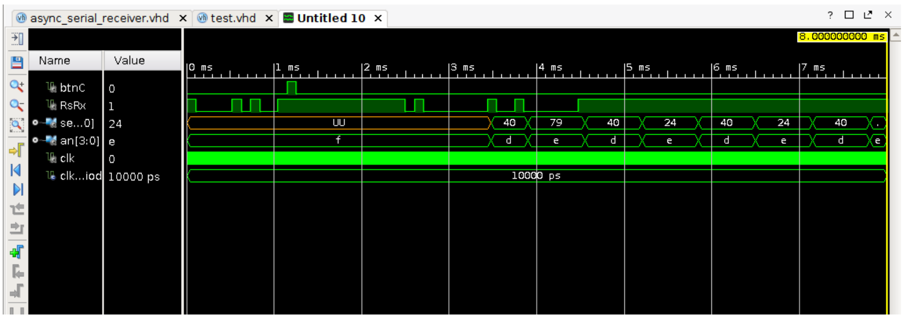
State : 3
This states is used for reading the stop bit. If stop bit is '1' then its good and transition is done to idle state, otherwise there is some problem with software or synchronisation and in this case the state transition is done to synchronizing state to auto correct the error and till synchronization all data is ignored.

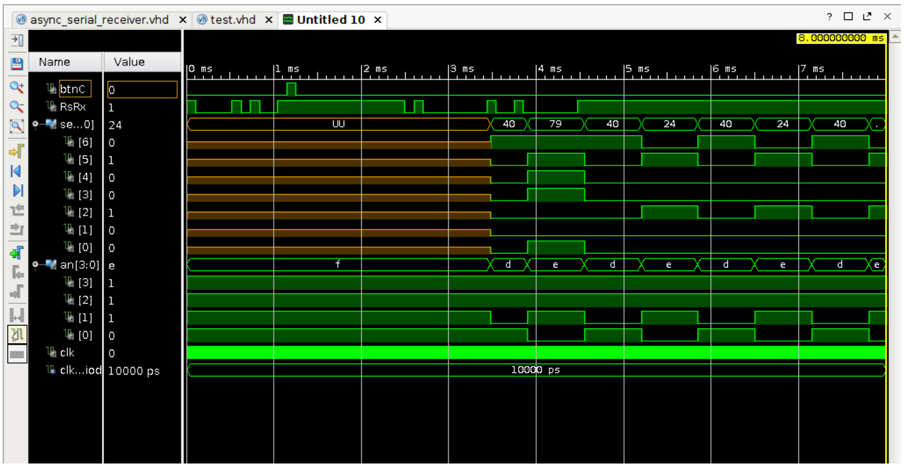


3 Simulation

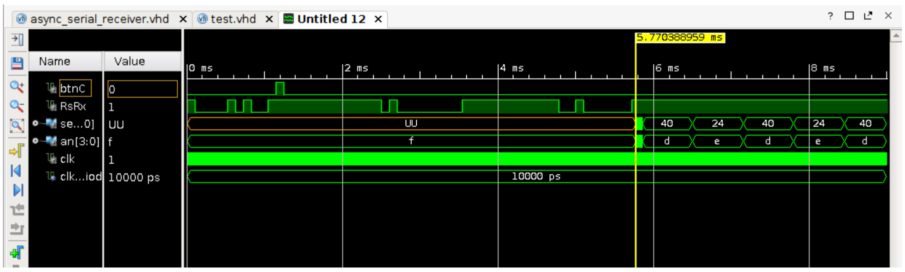
The code is simulated on test-bench created by the name test.vhd.

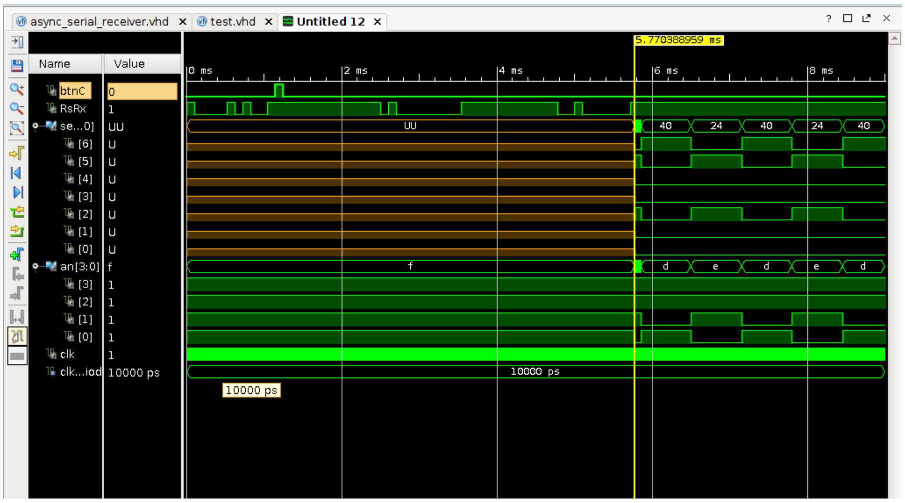
The below images show the normal case of data being transmitted from gtkterm and how it is ignored until push button is pressed for reset and then later it continuously receive data and display it in segment.





The below two images shows how the receiver goes in synchronization state to auto correct things itself in case there is some problem with the gtkterm software and it does not send the stop bit properly.





4 Utilisation Report

88 LUT's are used (as logic)
62 registers are used as flip-flops.
14 Input Output Blocks are used (1 RsRx input(data to be received) and 1 clock input(clk) and 7 segment outputs(seg) and 4 display outputs(an) and 1 push button inputs)
2 BUFGCTRL is used (to enable clock)

```
+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |   62 |        Flop & Latch |
| LUT5     |   27 |                 LUT |
| LUT6     |   21 |                 LUT |
| LUT4     |   16 |                 LUT |
| LUT1     |   16 |                 LUT |
| LUT3     |   15 |                 LUT |
| OBUF     |   11 |                  IO |
| LUT2     |    9 |                 LUT |
| CARRY4   |    4 |          CarryLogic |
| IBUF     |    3 |                  IO |
| MUXF7    |    2 |               MuxFx |
| BUFG     |    2 |               Clock |
+----------+------+---------------------+
```

(Utilisation report is also given in the submission.)