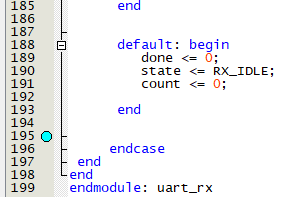
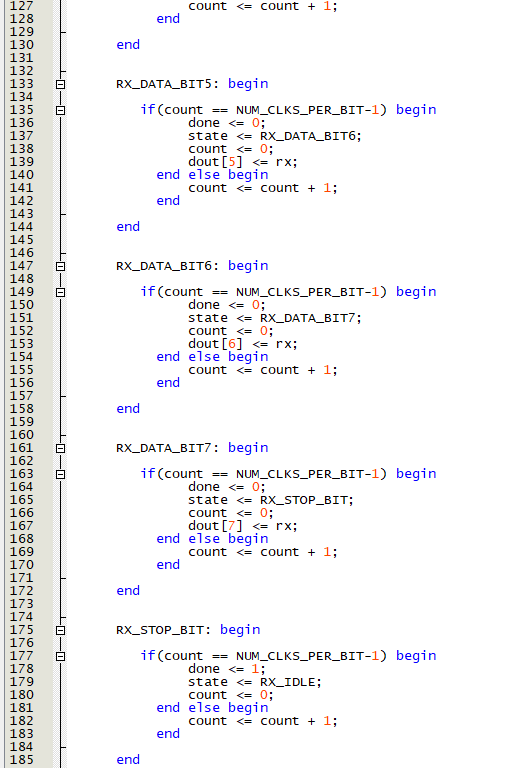
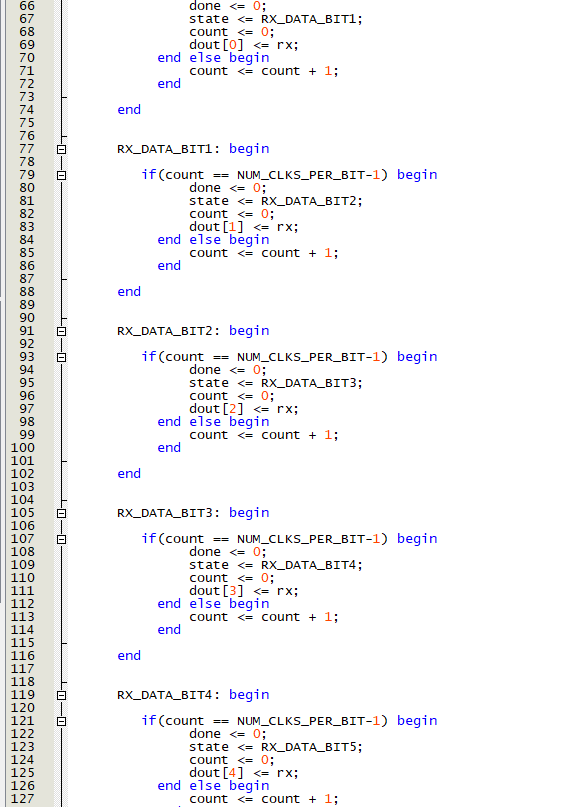
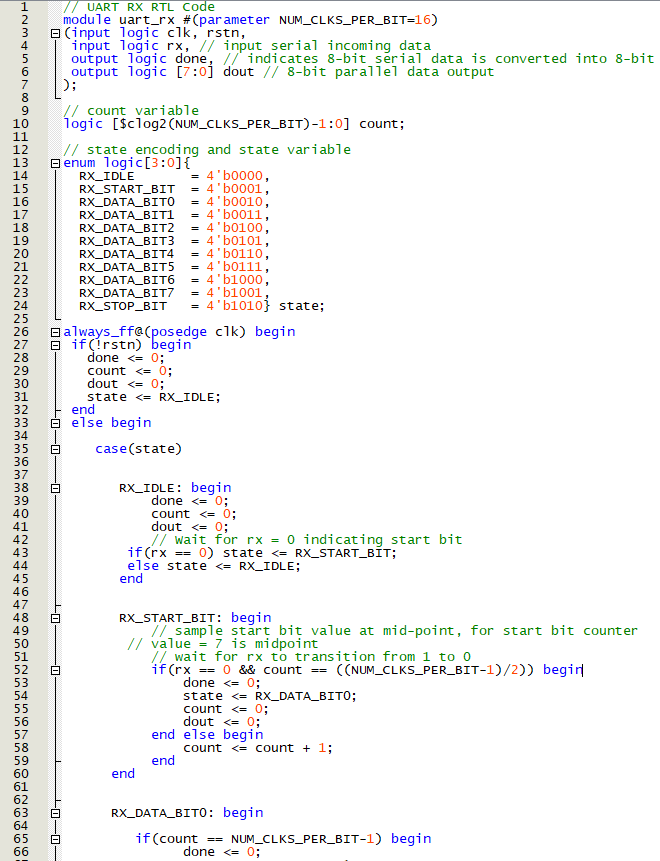
ECE 111 Winter 2022  
HW7  
Hao Le A15547504

**UART Receiver**

Code



RTL netlist

Diagram

Description automatically generated

Resource usage

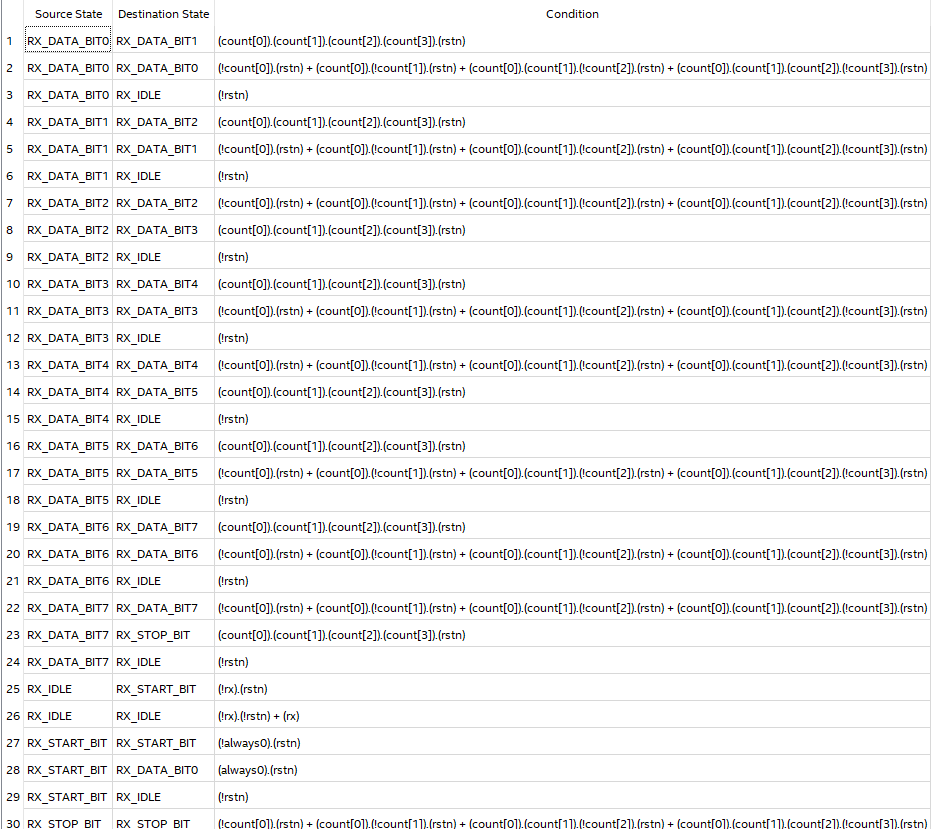
Graphical user interface

Description automatically generated with medium confidence

State diagram

A picture containing full, different, several

Description automatically generated



Testbench simulation waveform

Graphical user interface

Description automatically generated with medium confidence

Text

Description automatically generated

* When rx first goes low, this indicates a start bit, and we can observe the state variable going to that state
* Then, for 16 clock cycles from the halfway point of that start bit, we can observe the dout register getting values changed from LSB to MSB
* Moreover, the state also changes at the halfway point of every bit, indicating that the incoming bits are being sampled at the right times
* Once the eight bit has been sampled, the state changes to the stop state, where the stop bit is sampled as a high – subsequently the done bit is raised high as a pulse, dout represents the received byte, then the receiver goes back to idle state
* The receiver DUT passes all the tests
* Described above are signs that the receiver is behaving correctly

**UART TX-RX Communication System**

Code

Text

Description automatically generated

RTL netlist

Diagram

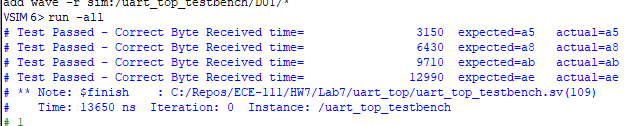
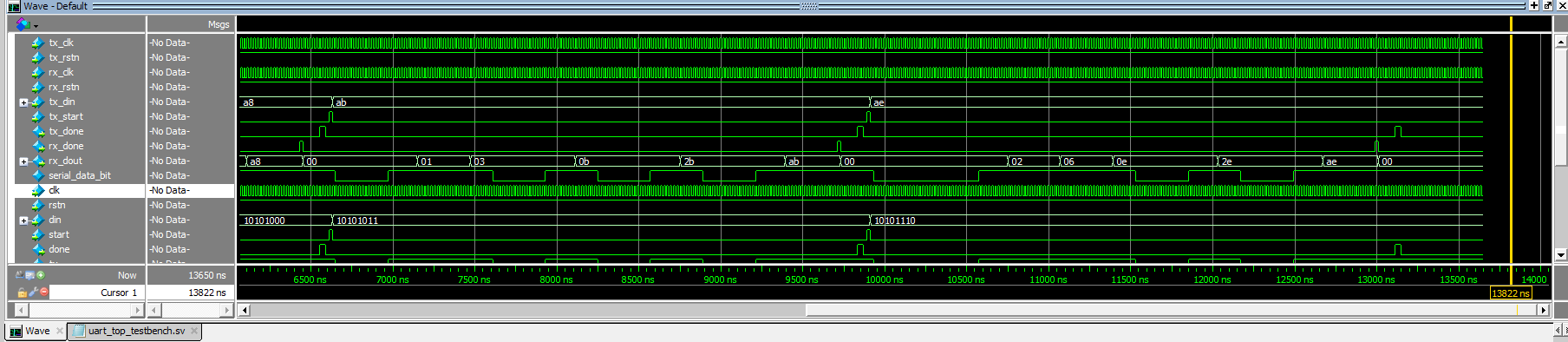
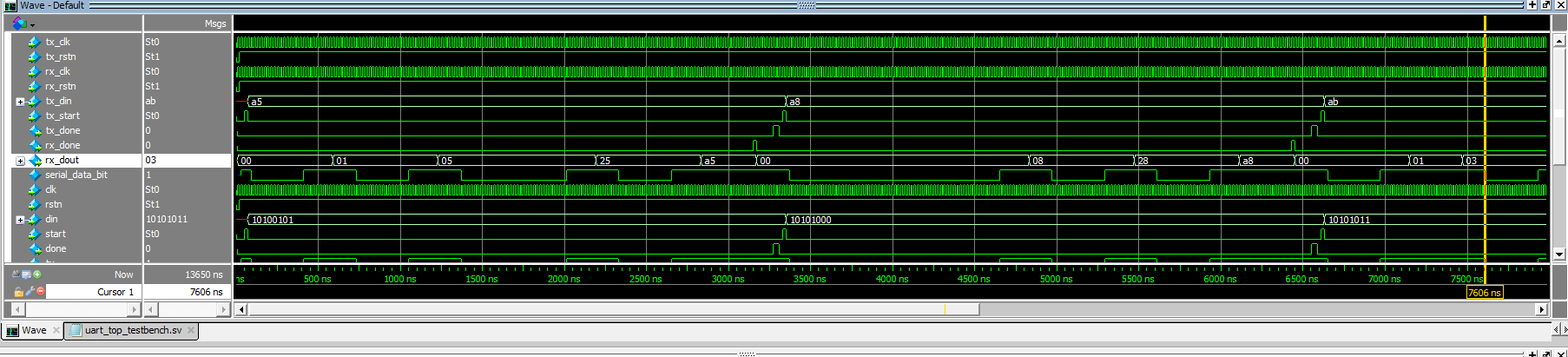
Description automatically generated

Resource usage

Graphical user interface

Description automatically generated with medium confidence

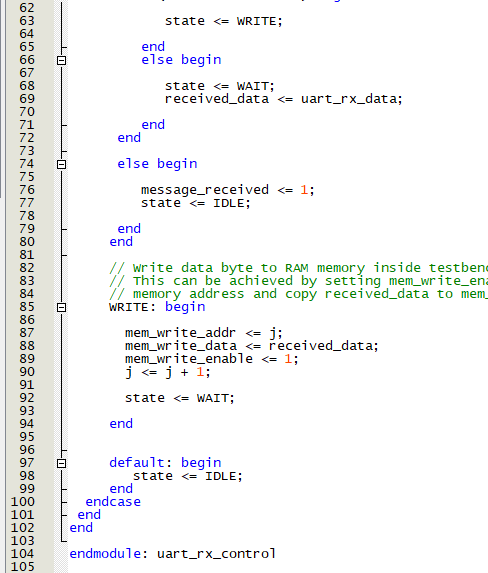
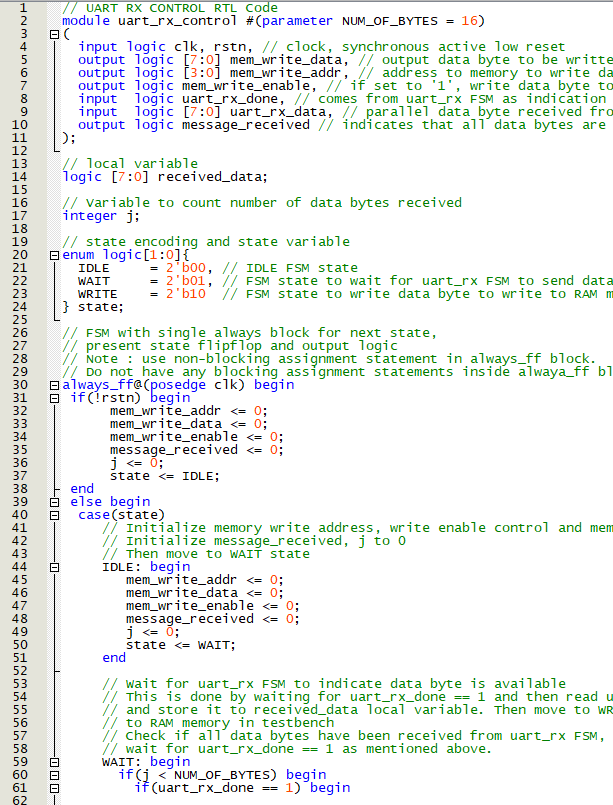
Testbench simulation waveform



* Shown above is the detailed waveform of both tx and rx UART modules working together, where the tx module transmits 4 different bytes loaded in through tx\_din, and the rx deserializes the byte and outputs that to rx\_dot
* For example, the first byte to be transmitted is A5 in hexadecimal. After tx\_start goes high, this signals tx to start serializing what is in tx\_din which is A5
* After the starting of tx, we also see rx\_dout start to change, indicating that rx has received that start bit, so it starts reading
* We can also observe the wire connecting the modules taking on the serialized data, on wire serial\_data\_bit
* Once rx has received all eight bits, it pulls rx\_done high to indicate so. Similarly, tx also pulls tx\_done high
* The end data is output to rx\_dout. It matches with what was initially fed into tx\_din
* Described above is the correct behavior between the two modules, for all four test cases

**UART Control System**

RX Control code



RX Control state diagram

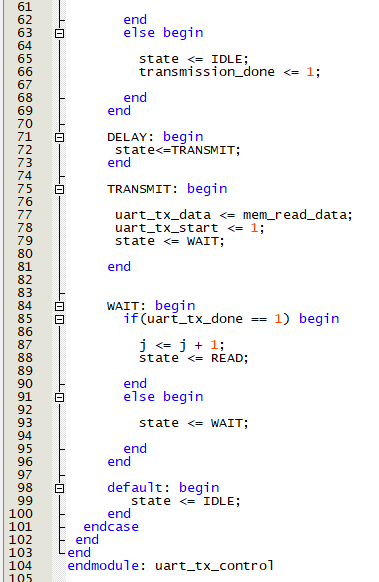
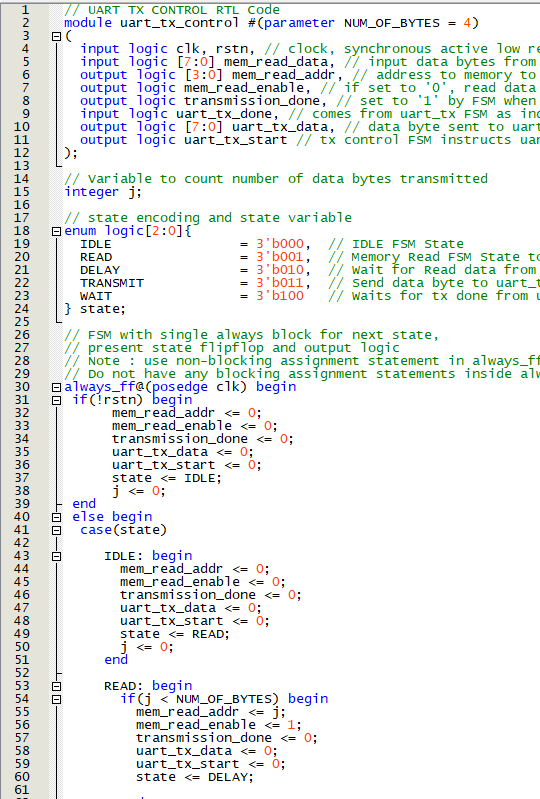
A picture containing text, sport

Description automatically generated

Table

Description automatically generated

TX Control code



TX Control state diagram

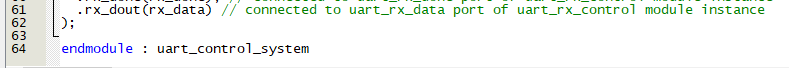
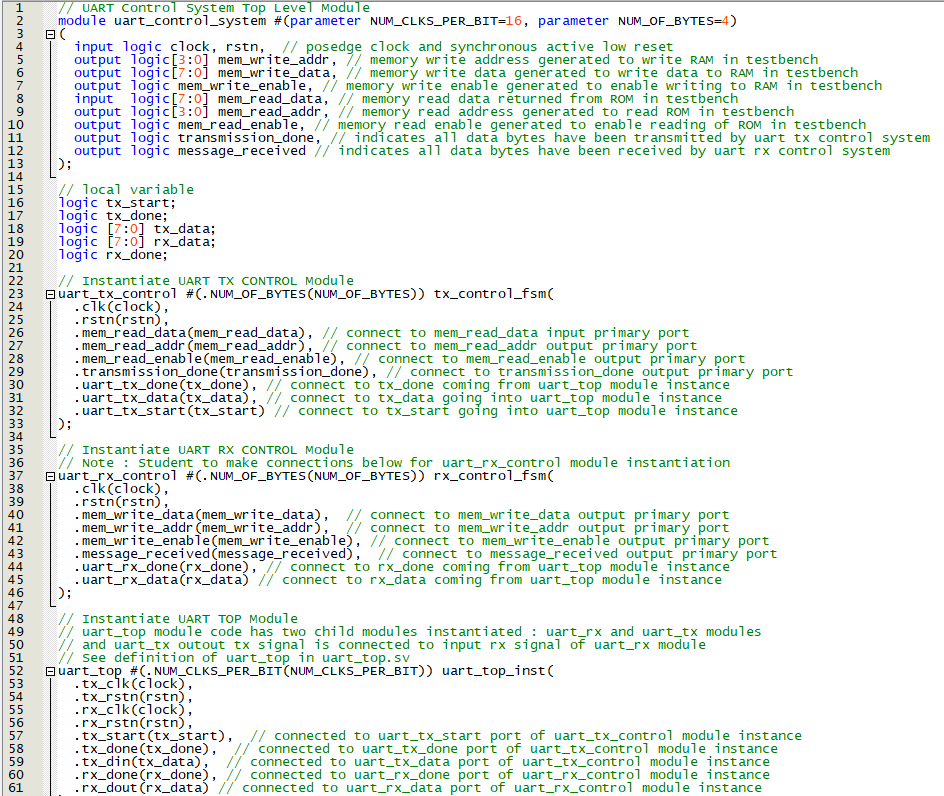
Diagram, schematic

Description automatically generated

Table

Description automatically generated

UART Control System code



RTL netlist

Chart

Description automatically generated with low confidence

Resource usage

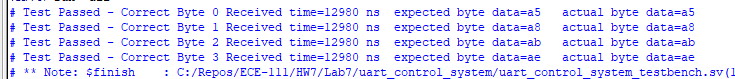
Graphical user interface

Description automatically generated

Testbench simulation waveform

Graphical user interface

Description automatically generated



* The data being read from memory by the tx controller is matched with the data written to memory by the rx controller – we can observe this by seeing mem\_read\_data being matched by mem\_write\_data after rx\_done signal goes high to indicate all 8 bits have been received by the UART top module
* When all four bytes have been transferred, message\_received goes high indicating that all 4 bytes have been written to memory
* Described above is the correct behavior for the UART control system