

ECE 327 Spring 2005
Assignment 4 - Due 03/18/05
UART Design and Simulation

Using Verilog design a circuit for a Universal Asynchronous Receiver/Transmitter (UART):

```
module uart (clk, rst, data, select, chip_select,
            read_strobe, write_strobe, ack,
            interrupt, serial_in, serial_out );
    input clk, rst, select, chip_select;
    input read_strobe, write_strobe serial_in;
    output interrupt, serial_out, ack;
    inout [7:0]data;
```

with the following specification:

The UART is rising edge triggered, and as an active low asynchronous reset. The UART has two interfaces - one is a typical memory interface with an 8 bit bi-directional data path (`data`) a single address line (`select`) and five control lines (`chip_select`, `read_strobe`, `write_strobe`, `ack`, and `interrupt`). All lines in this interface are active high. The UART is being addressed only when `chip_select` is high (1). Address zero (0) selects the receive data buffer (read) and transmit data buffer (write) while address one (1) selects the control register (on a write) and the status register (on a read). Data written to the transmit data buffer is transmitted out of `serial_out` while data received on `serial_in` is read from the receive buffer.

The serial interface connects the UART to another UART (`serial_in` to `serial_out` and vice-versa). The serial lines are held high (1) when there is no data present. A single word is sent in a frame that consists of a single zero (0) start bit, followed by eight (8) payload data bits and ended with two (2) stop high (1) stop bits. Data is sent least significant bit first. Each of the 11 bits of a frame is held for between 4 and 64 clock cycles as specified in the clock multiplier field of the control register. The receiver attempts to sample each value as close to the middle of the bit as possible. The transition from stop bit to start bit (1 to 0) is used to synchronize the receiver. Each frame is resynchronized on that transition. The sending and receiving UART need not run on the same clock.

The UART generates an interrupt under two conditions: first, if the `enable_transmit_interrupt` bit is set in the control register and transmit buffer does not contain data that has not already been transmitted, and second, if the `enable_receive_interrupt` bit is set in the control register and the receive buffer contains data that has not been read yet. Writing the transmit buffer and reading the receive buffer clear these interrupts (respectively). The status register includes the `data_available` bit, which is true if there is unread data in the receive buffer and the `ready_to_send` bit which is true if there is no unsent data in the transmit buffer.

<i>Bit</i>	<i>Function</i>
0	enable_receive_interrupt
1	enable_transmit_interrupt
2	clock divide bit 2
3	clock divide bit 3
4	clock divide bit 4
5	clock divide bit 5

Control register bit fields.

<i>Bit</i>	<i>Function</i>
0	data_available
1	ready_to_send
2	error

Status register bit fields.

Assume that once data has been written into the transmit buffer it cannot be over-written. Assume the receive buffer cannot be read unless there is data in it (an then only once). Assume that register reads and writes will never take more on than one clock cycle. A read or write that violates the above asumptions should complete normally (asserting `ack`) but have no effect on the UART other than to set the error bit in the status register until the next time it is read. An erroneous read returns 0. Record all other assumptions or resultions to incomplete specifications in your report.

What to turn in: Write a report with an introduction, including motivation, block diagram, state diagrams and/or truth tables, and circuit diagram, a listing of the Verilog code, and simulation outputs showing correct operation. The simulation outputs must be annotated to demonstrate that the output is correct and a section of the report must describe theory of operation and testing methodology used.