UART Transmitter Project Documentation

---

1. Study of the Existing Code

The uart_tx project from the VSDSquadron_FM repository implements a UART transmitter module on the FPGA. The Verilog code consists of several key components as outlined below:

Key Components:

1. **top.v (Top-Level Module):**
    - **UART Transmission:**
        - This module connects the FPGA's uarttx pin to a UART transmitter block and sends a byte upon trigger.
    - **Clock Generation:**
        - The internal clock is derived from a 12 MHz oscillator, and a 9600 Hz clock is generated by counting cycles of the main clock.
    - **RGB LED Feedback:**

        - RGB LEDs are driven using the SB_RGBA_DRV primitive, providing visual feedback based on internal counter values.
    - **Frequency Counter:**
        - A counter is used to generate the 9600 Hz clock signal required for UART transmission.

2. **uart_tx_8n1.v (UART Transmitter):**
    - **Transmission Protocol:**
        - Implements an 8N1 UART transmission protocol (8 data bits, no parity, 1 stop bit).
    - **Finite State Machine (FSM):**
        - The FSM manages states for idle, start bit, data bits transmission, and stop bit.
    - **Transmission Control:**
        - The module sends one byte at a time, waits for a trigger signal (senddata), and transmits the byte through the UART TX pin.

3. **VSDSquadronFM.pcf (Pin Constraint File):**

- Defines the pin assignments for the FPGA:
  - **UART TX:** Pin 14
  - **UART RX:** Not used in this transmitter-only design.
  - **RGB LED Pins:** Pins 39, 40, and 41.
  - **Clock Input:** Pin 20.

4. **Makefile (Build System):**
   - Automates the synthesis, place & route, timing analysis, and FPGA programming.
   - Uses **yosys**, **nextpnr-ice40**, **icepack**, and **iceprog** tools for building and uploading the bitstream.
   - Provides a UART terminal setup for communication via **picocom**.

Implementation Steps

Hardware Setup:

1. **Connect FPGA to PC via USB:**
   - Connect the FPGA board to the PC through a USB-to-UART adapter or directly to the device's UART port.
2. **Pin Configuration:**

- Ensure that the FPGA's uarttx pin is correctly connected to the TX pin of the receiving UART device.
- Connect the RGB LEDs to the designated pins (39, 40, 41).

3. **Serial Terminal Setup:**
   - Use **picocom** or another serial terminal tool to send and receive data through the UART interface.

Synthesis & Upload:

1. **Run Make Command:**
   - Compile the Verilog code and generate the bitstream using:

     make

2. **Flash FPGA:**
   - Upload the bitstream to the FPGA using:

     sudo make flash

---

4. Testing & Verification

Test Procedure:

1. **Open a Serial Terminal:**
   - Use the following command to open a serial terminal:

     sudo picocom -b 9600 /dev/ttyUSB0

2. **Send Test Data:**
   - Send some test characters from the terminal to the FPGA.
3. **Observe Transmitted Data:**
   - Ensure that the same data sent from the terminal is transmitted back to the terminal by the FPGA via UART.
4. **Verify RGB LEDs:**
   - Check if the RGB LEDs respond based on the transmitted data (using internal counter for feedback).

Expected Outcome:

- Sent characters should be echoed back to the serial terminal.

- RGB LEDs should show changes depending on the transmitted data, confirming correct operation.

5. Block diagram: