# Assignment #14

User Defined Primitives

## User Defined Primitives

1. **Design UDP for 8:1 Multiplexer :**
   **Hint:** To model an 8:1 multiplexer using UDP, define 8 inputs (a7, a6, a5, a4, a3, a2, a1, a0), 3 selection inputs (s2, s1, s0), and 1

2. **4-Input Majority Voter using UDP:** Output is 1 when 3 or more inputs are 1.
   **Hint :** List all combinations of inputs a, b, c, d that yield 3 or more 1's. You'll need 16 rows in the truth table.

3. **3-Bit Even Parity Generator :** Output is 1 if number of 1's in a, b, c is even.
   **Hint:** Use `even = ~(a ^ b ^ c)` (or match all even-weighted inputs in the table).

4. **2-Bit Comparator (Equal Detector) : Output 1 when** $a[1:0] = b[1:0]$.
   **Hint:** Define 4 inputs (`a1`, `a0`, `b1`, `b0`). Create a truth table to check for equality cases.

5. **D Flip-Flop (Edge-Sensitive using UDP) :**
   **Hint:** Use sequential UDP (add `reg q`, and trigger state changes on `posedge clk`).

6. **SR Latch with Asynchronous Reset :** SR latch with an active-low reset.
   Hint: Priority logic: if reset == 0, force output 0; else do standard SR behavior

7. **Rising Edge Detector:** Output a pulse (1) when a rising edge is detected on input `signal`.
   **Hint:**Track previous state of `signal` to detect `0` $\rightarrow$ `1` transition.

## Optional Extra Assignment for Practice

1. Design UDP for positive edge triggered D-Flip Flop with asynchronous reset ( Clear) and asynchronous set ( Preset)

2. Design MOD 3 Counter using Verilog primitives and D-Flip Flop UDP

3. 8-Bit Adder/Subtractor Using UDP