

***100 DAYS RTL PART-2***

# Day 91: SERIAL-IN SERIAL-OUT

# Day 92: SERIAL-IN SERIAL-OUT USING JK FF

# Day 93: SERIAL-IN PARALLEL-OUT

# Day 94: PARALLE-IN PARALLEL-OUT

# Day 95: ASYNCHRONOUS 5-BIT RIPPLE UP COUNTER UISNG T-FF

# Day 96: ASYNCHRONOUS 5-BIT RIPPLE DOWN COUNTER UISNG T-FF

# Day 97: 2-BIT SYNCHRONOUS COUNTER

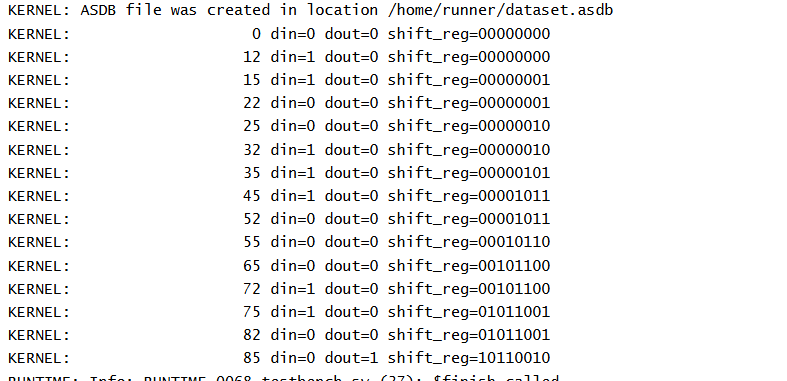
# Day 98: 3-BIT SYNCHRONOUS COUNTER

# Day 99: 4-BIT SYNCHRONOUS COUNTER

# Day 100: MODULE 12 COUNTER

***Day 91: SERIAL-IN SERIAL-OUT***

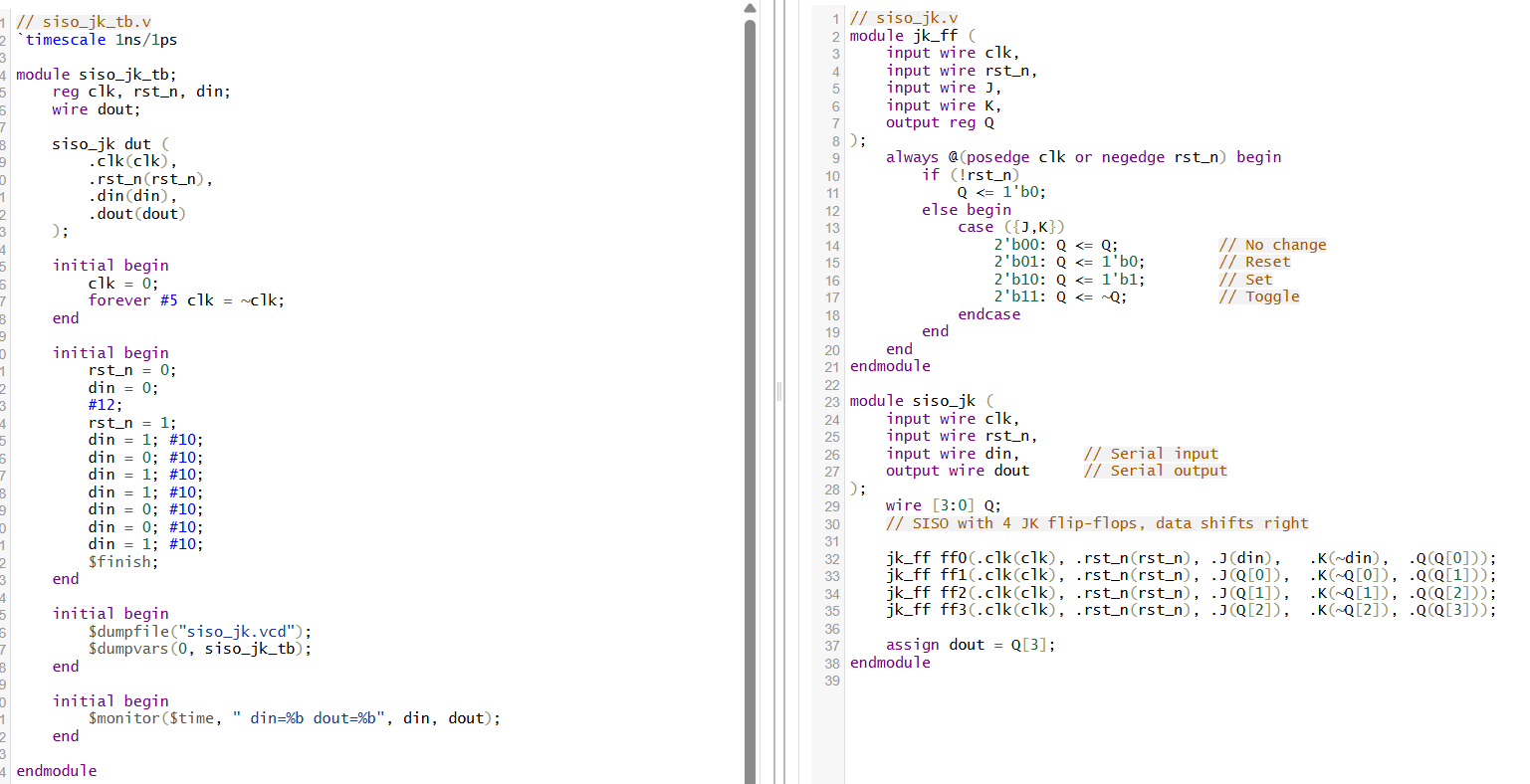
******

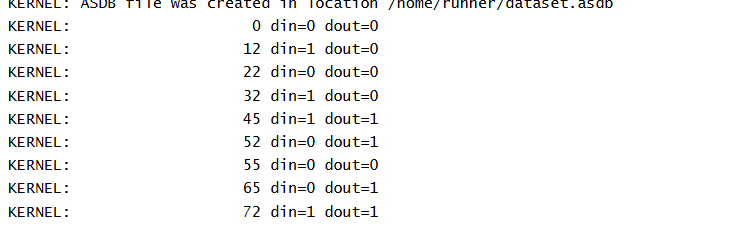
******

A Serial In Serial Out (SISO) shift register is a sequential digital circuit that moves and outputs data one bit at a time in a serial fashion. It consists of multiple flip-flops connected in series, with all flip-flops sharing the same clock signal.

* Working Principle:
  + Data enters the register via a single serial input line.
  + At every clock pulse, the input bit is stored in the first flip-flop; previous bits shift down the chain to the next flip-flop.
  + After N clock cycles (for an N-bit register), the first bit inputted appears at the serial output—the output of the last flip-flop.
  + This process provides a controlled *time delay* for each data bit as it propagates through the register.
* Key Features:
  + *In SISO registers, data comes in and leaves serially—one bit per clock cycle.*
  + Data is moved right (or left depending on design) at each clock edge.
  + Typical applications include temporary data storage, time delay generation, and serial-to-serial data transfer.
* Example (4-bit register, data input 1011):
  + After each clock pulse, the input bit shifts through the chain—initially all flip-flops are reset to zero.
  + With each clock edge, a new input bit is loaded, previous bits are shifted, and the oldest bit appears at the output.
* Applications:
  + *Delay elements* in digital circuits.
  + Serial data transfer for communication systems.
  + Simple finite state machines and basic counters.

***Day 92: SERIAL-IN SERIAL-OUT USING JK***

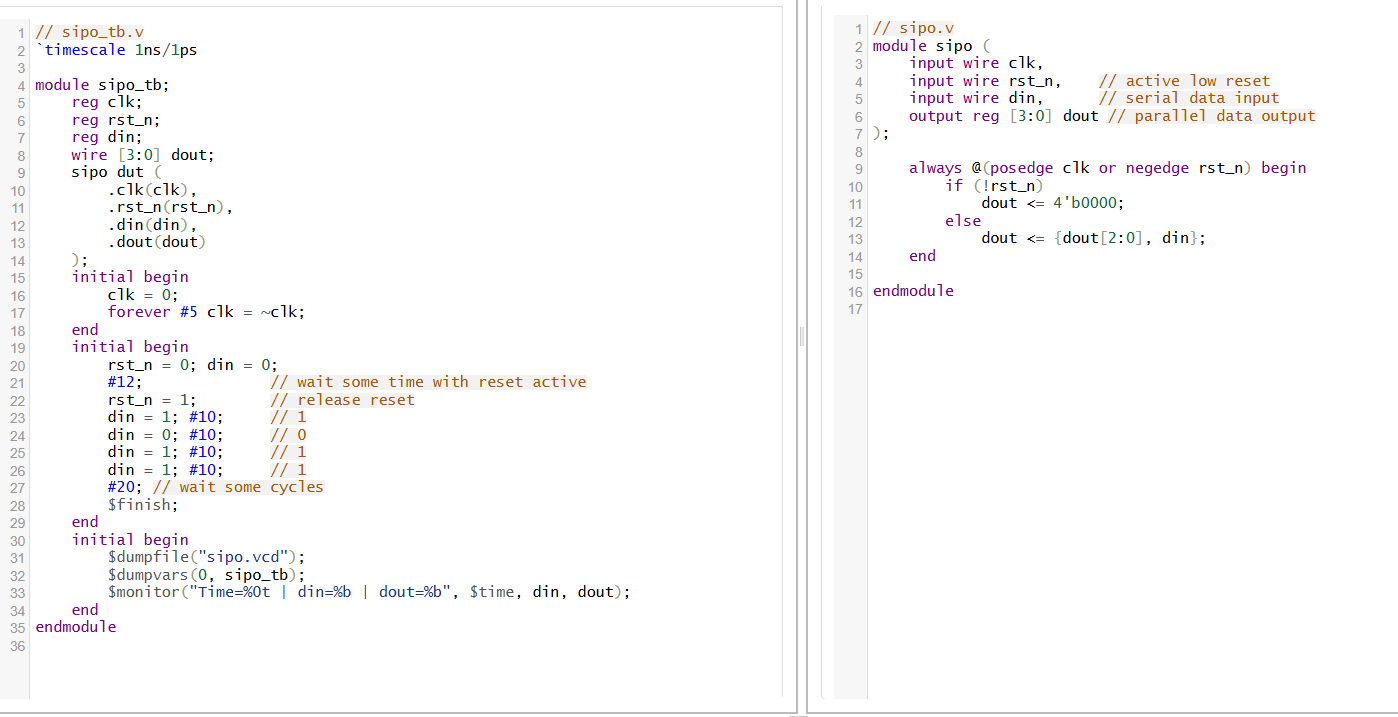
******

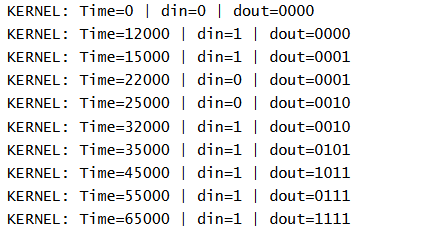
******

A Serial In Serial Out (SISO) shift register using JK flip-flops is a sequential digital circuit in which a sequence of JK flip-flops is connected in series to transfer data serially, one bit at a time, by the action of the clock pulse.

* Theory and Working Principle:
  + Each JK flip-flop can *store one bit* of data, changing its output state according to its J and K inputs and the clock pulse.
  + To shift a bit into a JK flip-flop:
    - For input bit '1', set J = 1, K = 0 (forces output Q to '1').
    - For input bit '0', set J = 0, K = 1 (forces output Q to '0').
  + In a SISO shift register, the serial input is provided to the first JK flip-flop's J and K inputs according to the above rule. The output Q of each flip-flop is fed into the J and K inputs of the next flip-flop in the chain, shifting the data with each clock edge.
  + After N clock cycles (for an N-bit register), the first input bit will appear at the output (the last flip-flop in the series).
* Key Features:
  + Shifts *one bit per clock pulse*, from input through the chain to output.
  + All flip-flops share a common clock, causing synchronous shifting.
  + The JK configuration ensures *no invalid state* occurs; if both J and K are '1', the flip-flop toggles.
* Applications:
  + Used for temporary storage and serial data transfer.
  + Timing and synchronizing digital signals.
  + Counters and simple state machines.
* Why JK Flip-Flops?
  + JK flip-flops offer flexibility with set, reset, and toggle actions depending on J and K values.

***Day 93: SERIAL-IN PARALLEL-OUT***

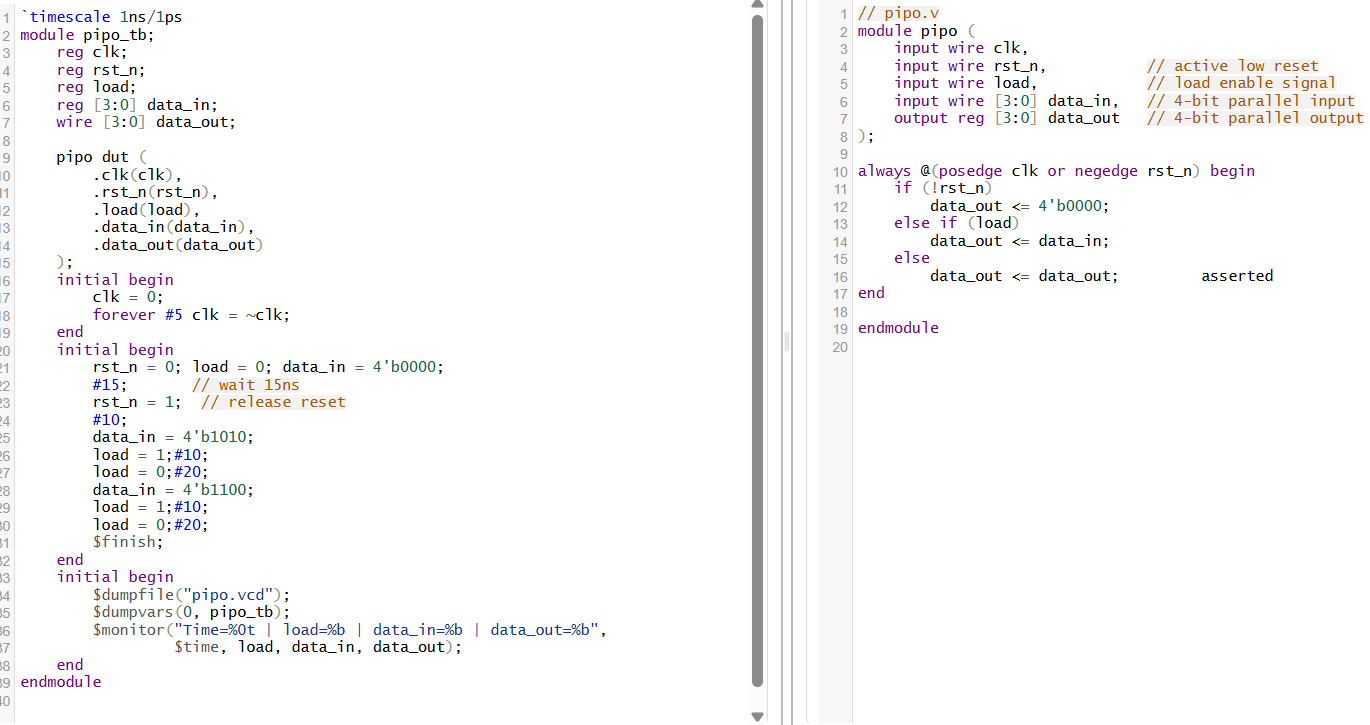
******

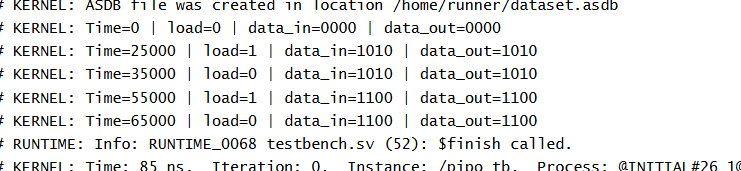
******

A Serial In Parallel Out (SIPO) shift register is a sequential logic device that converts data from serial format into parallel format. It consists of a chain of flip-flops connected in series, where data enters one bit at a time (serial input) and after shifting through the flip-flops, all stored bits become simultaneously available on parallel output lines.

* + Working Principle:
    - Data bits enter serially via a single input line.
    - With each clock pulse, the current input bit is loaded into the first flip-flop.
    - The previous bits are shifted one position down the line of flip-flops.
    - After N clock cycles (for an N-bit register), the input bits appear concurrently at all parallel outputs, representing the stored data in parallel.
    - This behavior effectively converts a serial data stream into a parallel data word.
  + Key Features:
    - The register stores the data sequentially, but outputs them all at once.
    - Each flip-flop output is connected to an output line, enabling simultaneous reading of all bits.
    - The clock synchronizes data shifting.
    - Typically used for buffering or converting serial communication data into parallel data for further processing.
  + Applications:
    - Converting serial data from communication lines to parallel signals usable by microprocessors or other parallel-input devices.
    - Temporary data storage and timing delay.

***Day 94: PARALLEL-IN PARALLEL-OUT***

******

******

A Parallel In Parallel Out (PIPO) shift register is a digital sequential circuit that simultaneously loads multiple bits of data in parallel into its flip-flops and also presents the stored data on its outputs simultaneously in parallel form.

**Theory and Working Principle:**

* The shift register consists of several flip-flops, each capable of storing one bit.
* Parallel loading: When the load signal is enabled, all flip-flops capture their respective input bits *at the same time* (in parallel) on the clock edge.
* Parallel output: The stored data bits are presented in parallel to the external outputs, allowing all bits to be read *simultaneously*.
* When load is not enabled, the register typically holds its current state.
* This capability allows the PIPO shift register to quickly load and provide an entire word of data in a single clock cycle.
* Unlike serial input/output registers, no shifting delay is involved in loading or accessing all bits.

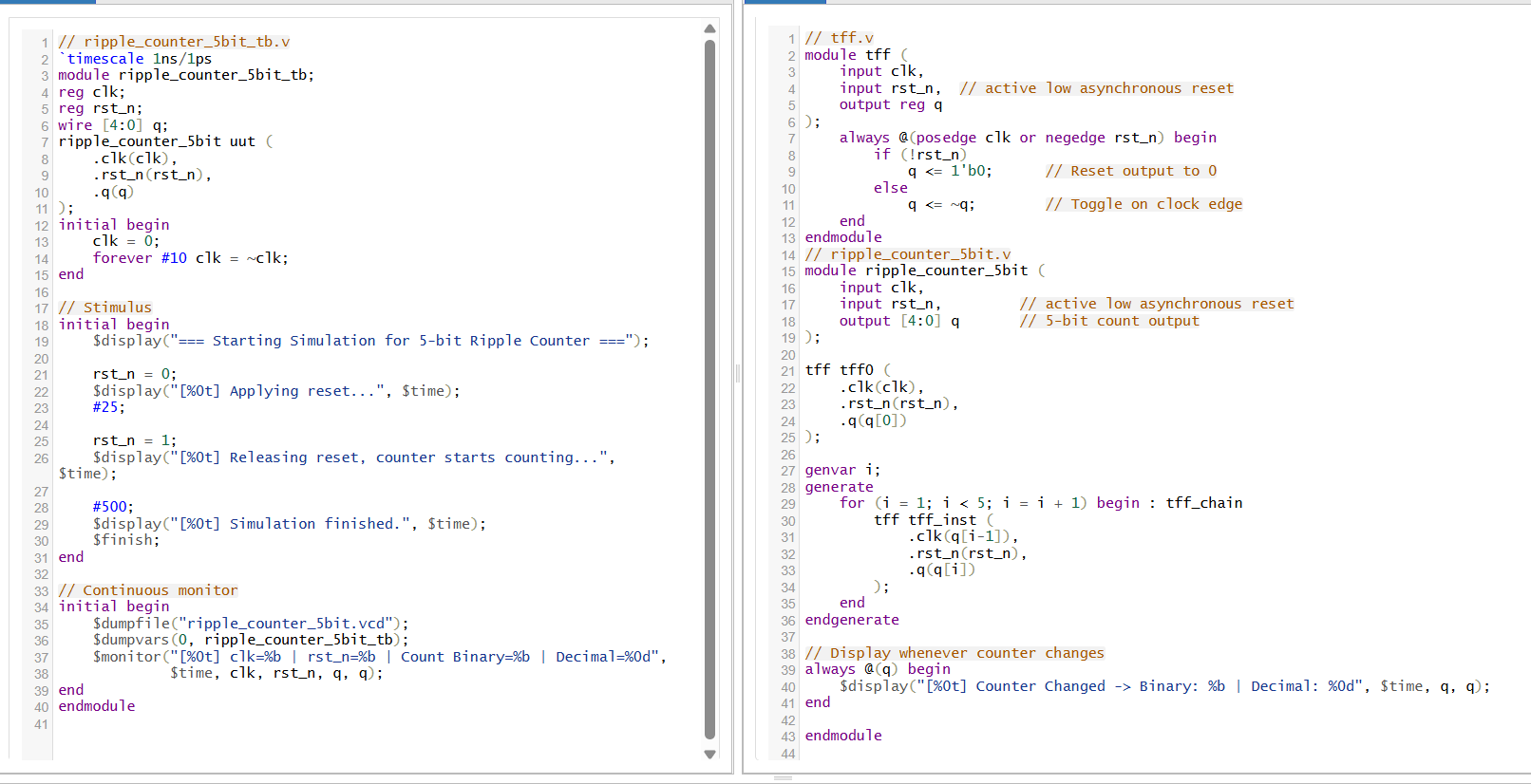
**Key Features:**

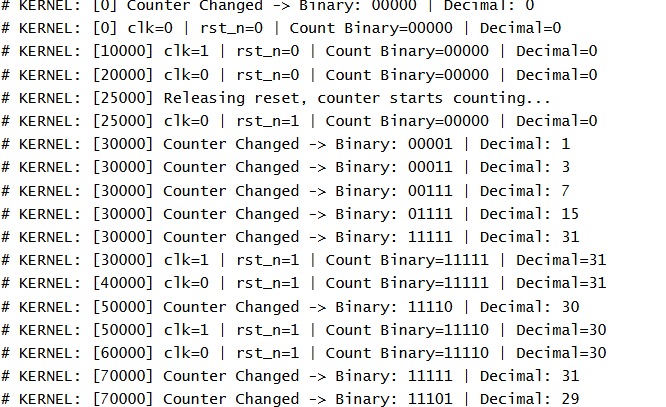
* Provides fast data loading and reading for multiple bits simultaneously.
* Generally used for temporary data storage, buffering, and data transfer where parallel data access is needed.
* Controlled by clock and load signals to synchronize data loading and holding.
* Often used in digital systems where wide data buses and parallel processing are required.

**Applications:**

* Buffering data between systems with different data widths or speeds.
* Holding data during computational operations in microprocessors or DSPs.
* Expanding I/O ports by storing multiple bits and outputting them in parallel.

***Day 95: ASYNCHRONOUS 5-BIT RIPPLE UP COUNTER USING T-FF***

******

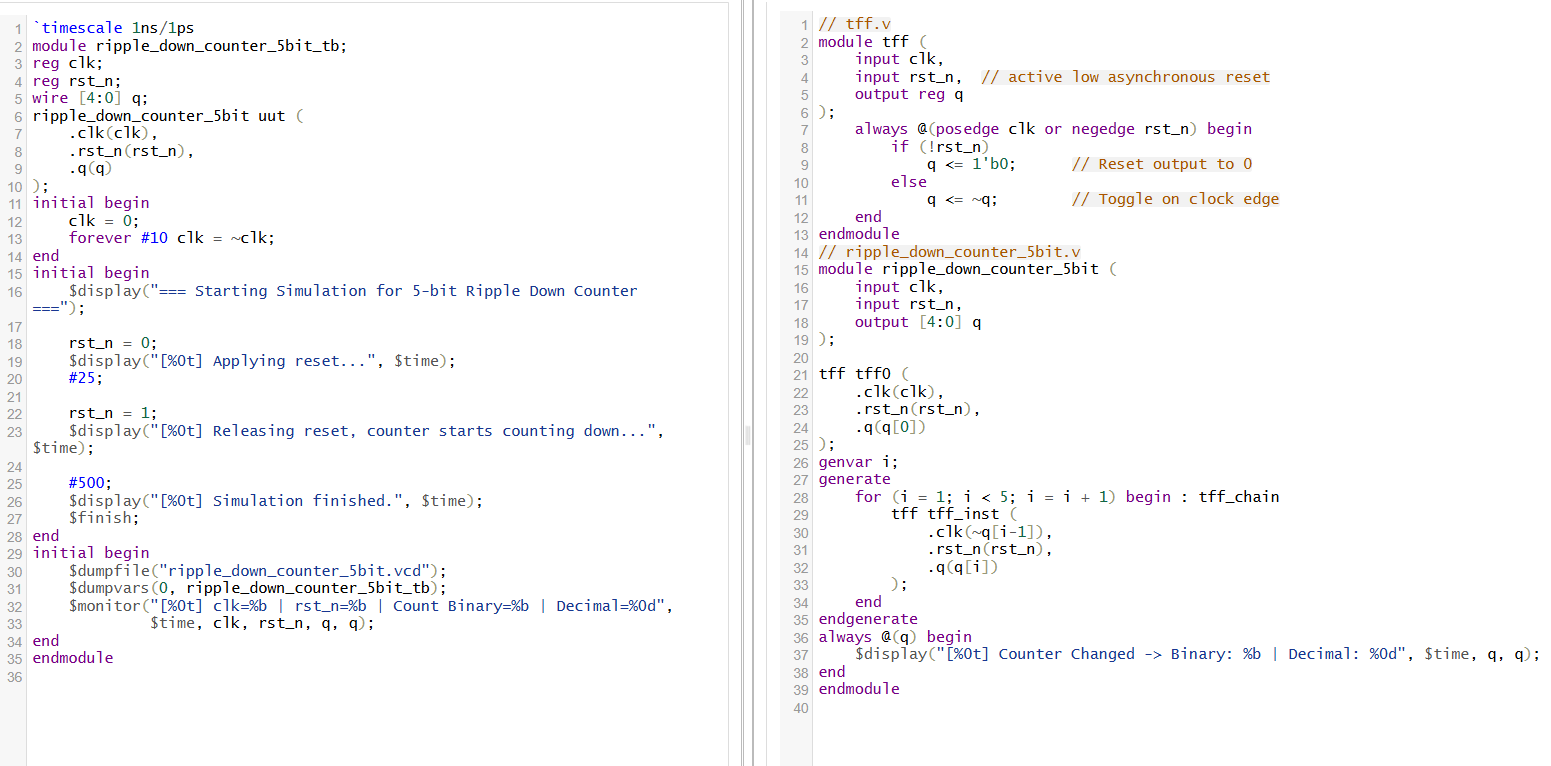
******

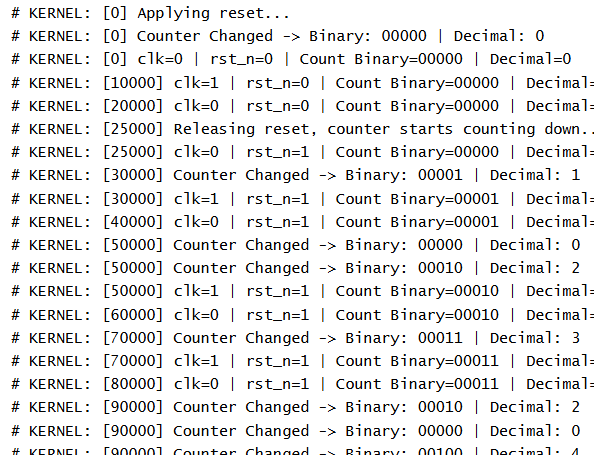
An Asynchronous 5-bit Ripple Up Counter is a type of digital counter built with a cascade of T flip-flops where each flip-flop toggles its state based on the output of the previous flip-flop rather than simultaneously on the main clock.

**Theory and Working Principle:**

* Asynchronous/Ripple Counter Basics:
  + Only the first T flip-flop is clocked by the external input clock signal.
  + Each subsequent flip-flop's clock input is triggered by the output (Q) of the preceding flip-flop.
  + Because flip-flops do not change state simultaneously but sequentially in a ripple fashion, it is called a ripple counter or asynchronous counter.
* Counting Behavior:
  + On every positive edge of the external clock, the first flip-flop toggles.
  + When the first flips (from 0 to 1 or 1 to 0), it triggers the second flip-flop to toggle on its next clock edge (which is the output of the first).
  + This cascading effect continues through all flip-flops, causing a binary count that increments by one each clock cycle.
  + The count progresses in binary from 00000 to 11111 (for 5 bits), then rolls over to 00000.
* Propagation Delay and Ripple Effect:
  + Because each flip-flop waits for the previous one to toggle before changing, there is a propagation delay through the chain.
  + This delay causes the output bits to change sequentially, creating a "ripple" effect. The total delay accumulates as the number of flip-flops increases

***Day 96: ASYNCHRONOUS 5-BIT RIPPLE DOWN COUNTER USING T-FF***

******

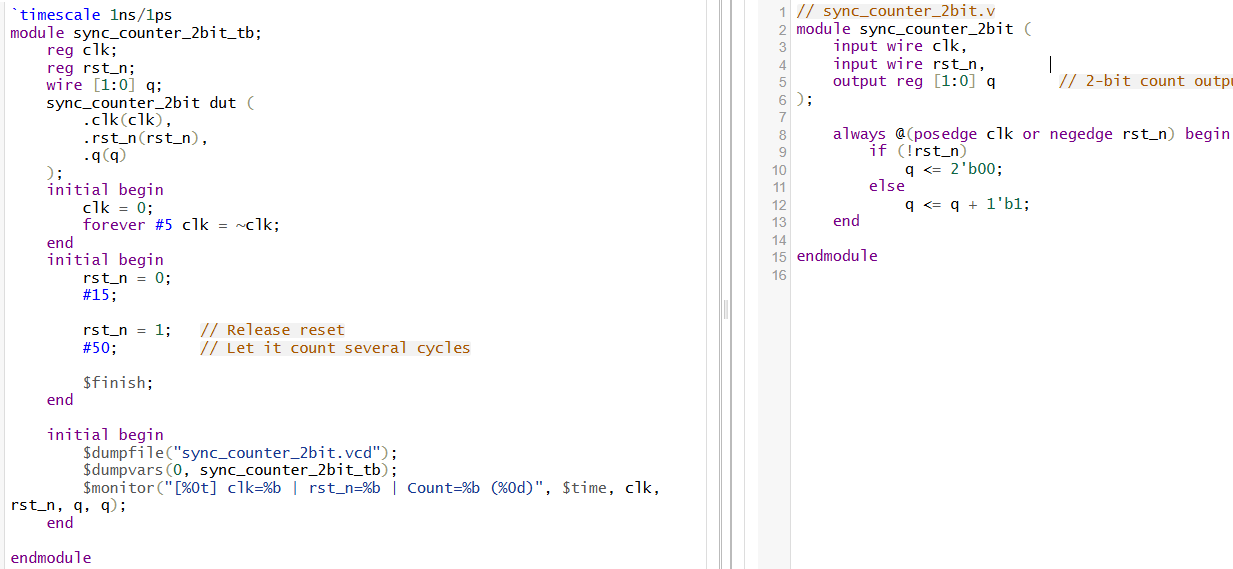
******

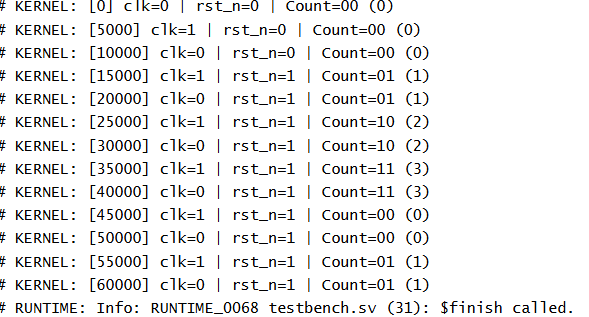
An Asynchronous 5-bit Ripple Down Counter is a type of asynchronous (ripple) counter that counts downwards in binary. It is constructed using T flip-flops connected in series, where each flip-flop toggles on the clock pulse derived from the previous stage but in an inverted manner to implement downward counting.

**Theory and Working Principle:**

* Asynchronous Counter Basics: Only the first flip-flop is driven directly by the external clock. Each subsequent flip-flop receives as its clock the output from the previous flip-flop (for up counters) or the *inverted* output (for down counters). This cascading forms a ripple effect, as the clock pulses "ripple" through each flip-flop asynchronously (not all at once).
* Counting Down Behavior: When counting down, the flip-flops change state so that the binary count decrements by one on each clock cycle:
  + The first flip-flop toggles on every clock pulse.
  + Each following flip-flop toggles when the inverted output of the previous flip-flop transitions, producing a count sequence like 11111, 11110, 11101,..., 00000, then wraps around.
* Ripple Effect and Propagation Delay: Because flip-flops toggle one after another triggered by preceding outputs, there is a cumulative delay (propagation delay), causing the output bits to change in sequence rather than simultaneously—this delay is called the ripple effect. The more flip-flops cascaded, the longer the delay, limiting the maximum operating frequency.
* Advantages:
  + Simple implementation with minimal control logic.
  + Suitable for low-speed counting where timing constraints are relaxed.
* Disadvantages:
  + Propagation delay accumulates with added flip-flops.
  + Not suitable for high-speed or precision synchronous applications due to asynchronous transitions.

***Day 97:2-BIT SYNCHRONOUS COUNTER***

******

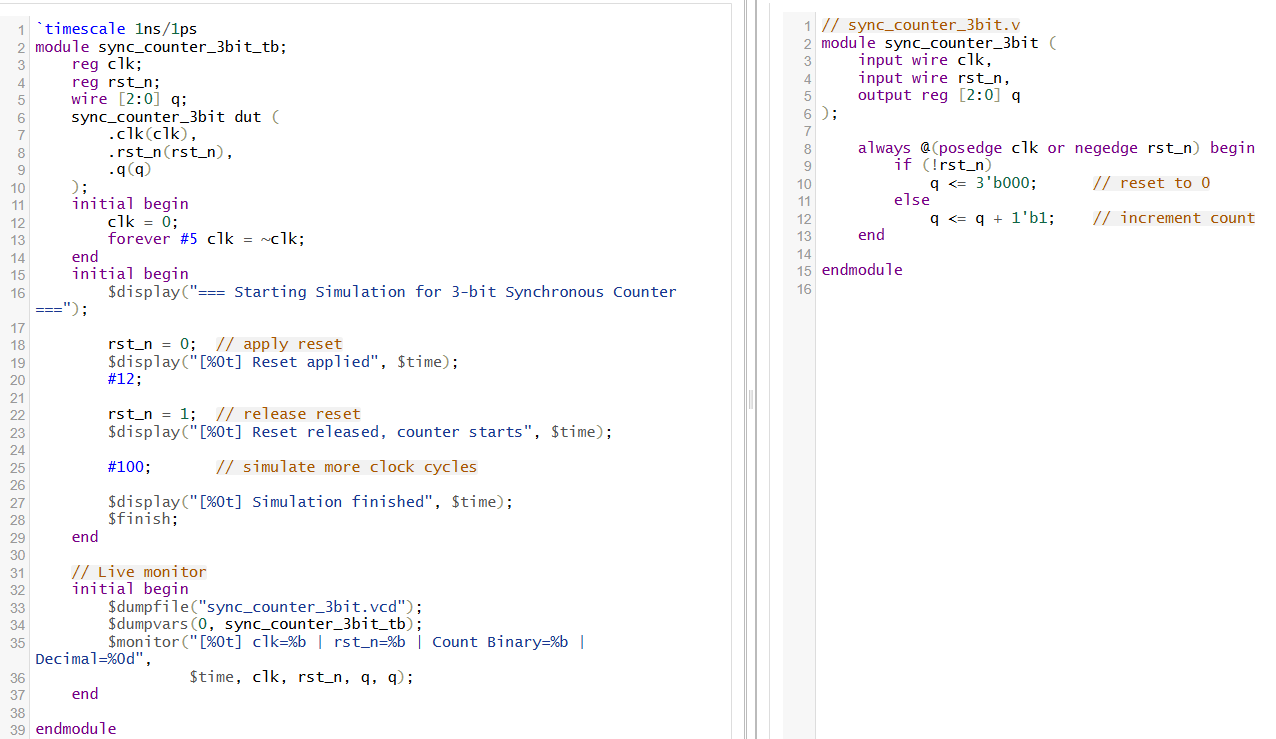
******

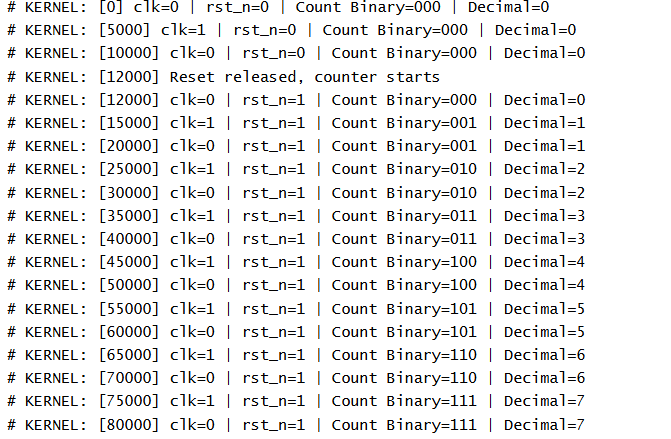
A 2-bit synchronous counter is a digital sequential circuit composed of two flip-flops that are all driven by a *common clock signal* simultaneously. Unlike asynchronous (ripple) counters where flip-flops toggle at different times, in synchronous counters every flip-flop receives the clock pulse at the same time, enabling all bits to update in parallel on each clock edge.

**Theory and Working Principle:**

* Synchronous Operation: All flip-flops in the 2-bit counter are triggered simultaneously by the same global clock signal. This eliminates ripple delays seen in asynchronous counters and allows the counter to operate correctly at higher frequencies.
* Counting Sequence: The counter counts in a binary sequence from 00 to 11 (decimal 0 to 3) incrementing by one on each clock pulse and then wraps back to 00.
* Flip-Flop Logic:
  + The least significant bit (LSB) flip-flop toggles on every clock pulse (by configuring its inputs to toggle mode).
  + The next flip-flop toggles only when the LSB output is high, effectively counting the carry from the first bit.
  + This is usually implemented by combinational logic feeding the J and K inputs of JK flip-flops or using T flip-flops to achieve toggling.
* Timing and Outputs:
  + Each clock rising edge causes the counter to increment by 1.
  + Because all flip-flops are clocked together, their outputs change simultaneously, and the counter output reflects the binary count at that instant.
* Advantages Over Asynchronous Counters:
  + No cumulative propagation delay since all flip-flops receive the clock at once.
  + Higher maximum clock frequency due to synchronous operation.
  + Predictable timing behavior, important for precise control in digital systems.
* Applications:
  + Digital clocks, timers, frequency counters.
  + State machines requiring synchronous state progression.
  + Digital systems requiring reliable multi-bit counting with reduced timing errors.

***Day 98:3-BIT SYNCHRONOUS COUNTER***

******

******

Theory of 3-bit Synchronous Counter

A 3-bit synchronous counter is a sequential digital circuit that counts in binary from 000 to 111 (0 to 7 in decimal) in a synchronized manner, meaning all flip-flops are triggered at the same time by the same clock signal.

Working Principle

* The counter consists of three flip-flops (FF0, FF1, FF2), each representing one bit of the count (Q0 = LSB, Q2 = MSB).
* All flip-flops receive the same clock pulse at their clock input — this is why it’s called *synchronous*.
* State changes happen *in parallel* at each clock’s rising (or falling) edge.
* The binary output increments by 1 on every clock pulse until it reaches its maximum value (111), then wraps around to 000.

Operation Steps

For an up-counter:

1. Q0 (LSB): Toggles on every clock pulse.
2. Q1: Toggles when Q0 changes from 1 to 0 (i.e., every 2 clock pulses).
3. Q2 (MSB): Toggles when Q1 changes from 1 to 0 (i.e., every 4 clock pulses).

Because all flip-flops share the same clock, no ripple delay occurs — the outputs change simultaneously.

Example counting sequence (binary → decimal):

text

000 → 0

001 → 1

010 → 2

011 → 3

100 → 4

105 → 5

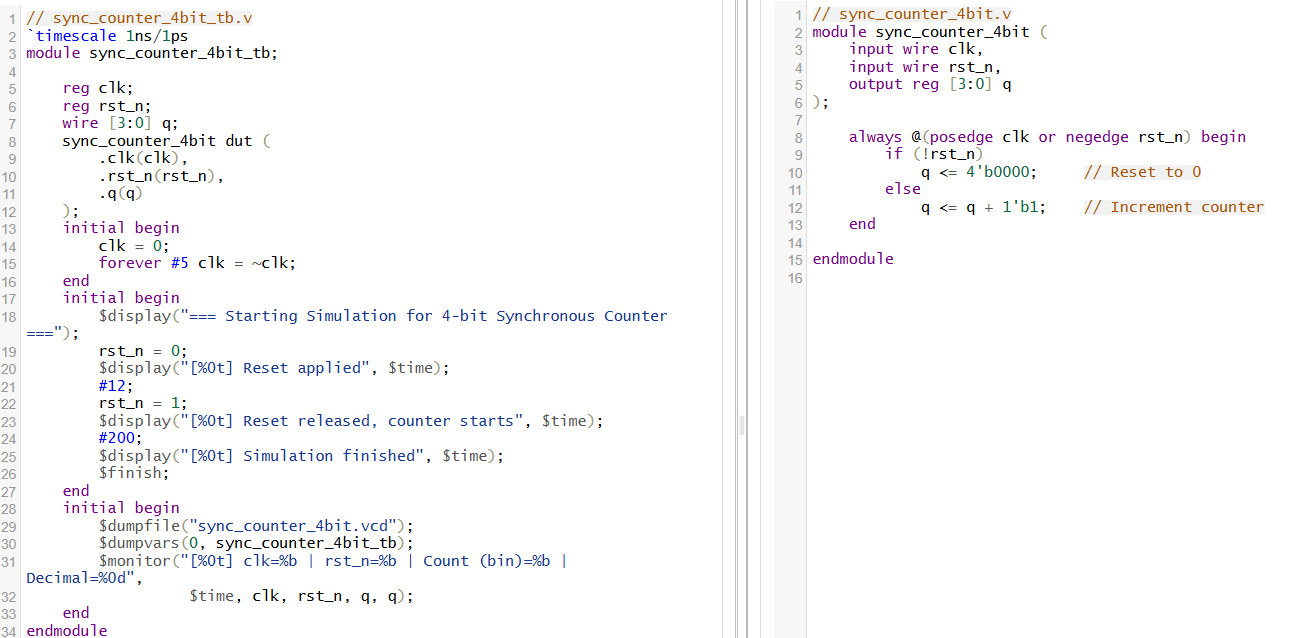
110 → 6

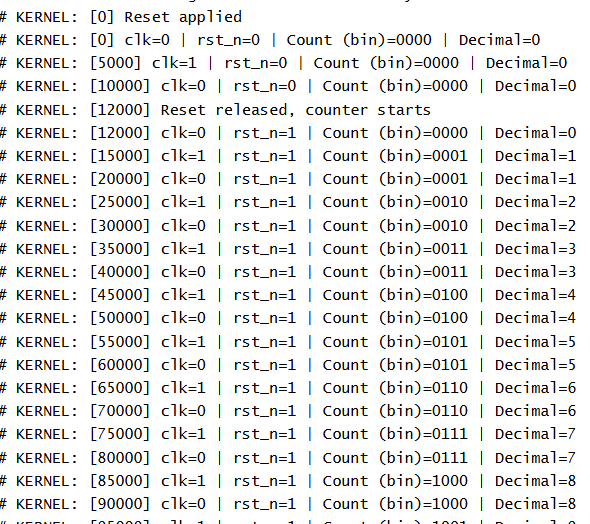
111 → 7 → wraps back to 000

Advantages over Asynchronous Counter

* No cumulative propagation delay, allowing higher operating speed.
* Predictable timing, making it suitable for precise digital systems.

***Day 99: 4-BIT SYNCHRONOUS COUNTER***

******

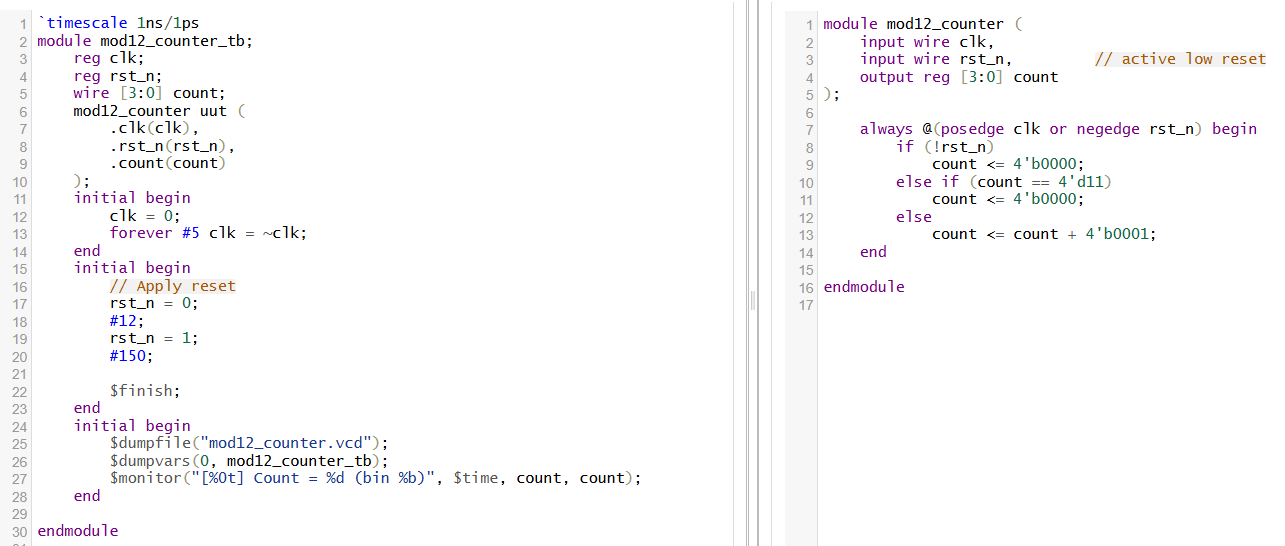
******

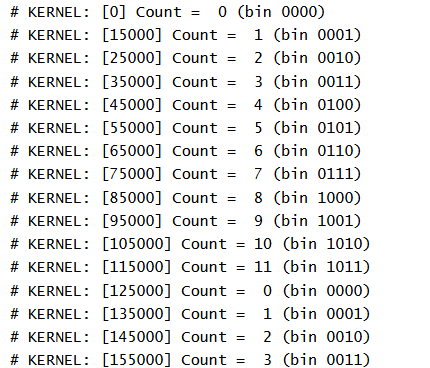
A 4-bit synchronous counter is a sequential digital circuit that goes through a fixed sequence of binary states in synchronization with a clock signal. In this counter, *all flip-flops are triggered by the same clock edge* (rising or falling), ensuring simultaneous state changes.

Working Principle

* Structure:
  + Contains 4 flip-flops (one for each bit: Q0 = LSB, Q3 = MSB).
  + All flip-flops share the same clock line → *synchronous operation*.
  + The flip-flop states change according to binary counting rules.
* Counting Sequence:
  + The counter increments its binary value on each clock pulse:  
    0000(0) → 0001(1) → 0010(2) → … → 1111(15) → back to 0000
* Flip-Flop Toggling:
  + LSB Flip-Flop (Q0): Toggles on every clock pulse.
  + Q1: Toggles when Q0 changes from 1→0 (every 2nd pulse).
  + Q2: Toggles on every 4th pulse.
  + Q3 (MSB): Toggles on every 8th pulse.

***Day 100: MODULE 12 COUNTER***





Definition

A Modulo-12 counter is a sequential counter that cycles through 12 distinct states (0 to 11 in decimal, i.e., 0000 to 1011 in binary) before repeating from 0 again.  
It is called *modulo-12* because:

Number of unique states=12Number of unique states=12

How It Works

* Since 24=1624=16 states are available in a 4-bit binary counter, but we only need 12, the counter is forced to reset when it reaches decimal 12 (1100 in binary).
* The counter increments on each clock pulse.
* When it detects the count value = 11 (decimal), the next clock cycle resets it back to 0.
* The reset can be implemented:
  + Synchronously: Reset signal to the storage elements on the same clock edge as the count is updated (as we did in the Verilog code).
  + Asynchronously: Decoding logic immediately clears flip-flops when count = 12 without waiting for a clock edge.

State Transition Table (Up Counting)

| **Clock Pulse** | **Count (Binary)** | **Count (Decimal)** |
| --- | --- | --- |
| 0 | 0000 | 0 |
| 1 | 0001 | 1 |
| 2 | 0010 | 2 |
| ... | ... | ... |
| 10 | 1010 | 10 |
| 11 | 1011 | 11 |
| 12 | 0000 | 0 (repeat) |

Key Features

* Synchronous Operation: All flip-flops are clocked together (if designed as a synchronous counter), allowing higher speed and predictable timing.
* Fixed Modulus: Modulus (number of states) is 12, less than 2n2*n*, so some binary states are unused.
* Reset Mechanism: The counter is designed to detect when the count reaches 12 and then reset.

Applications

* Timekeeping: Since 12 hours is a common time cycle (e.g., wall clocks).
* Frequency Division: A mod-12 counter can divide an input clock frequency by 12.
* Digital Systems Requiring Mod-N Counts: Common in control circuits, rotary encoders, etc.

