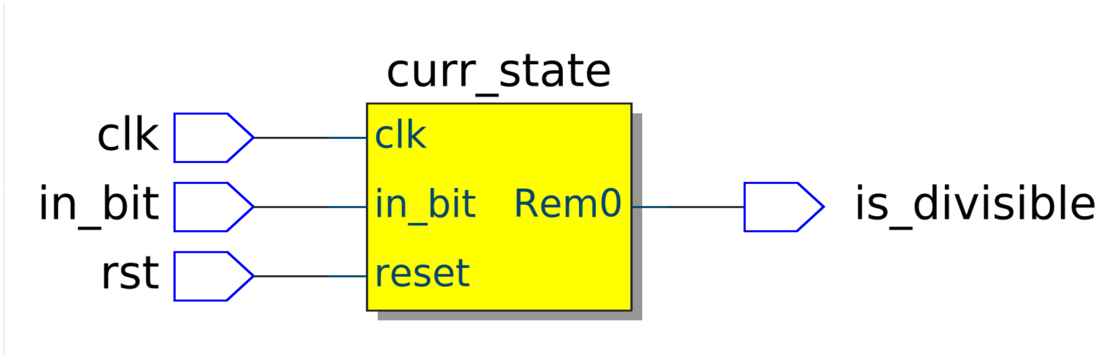
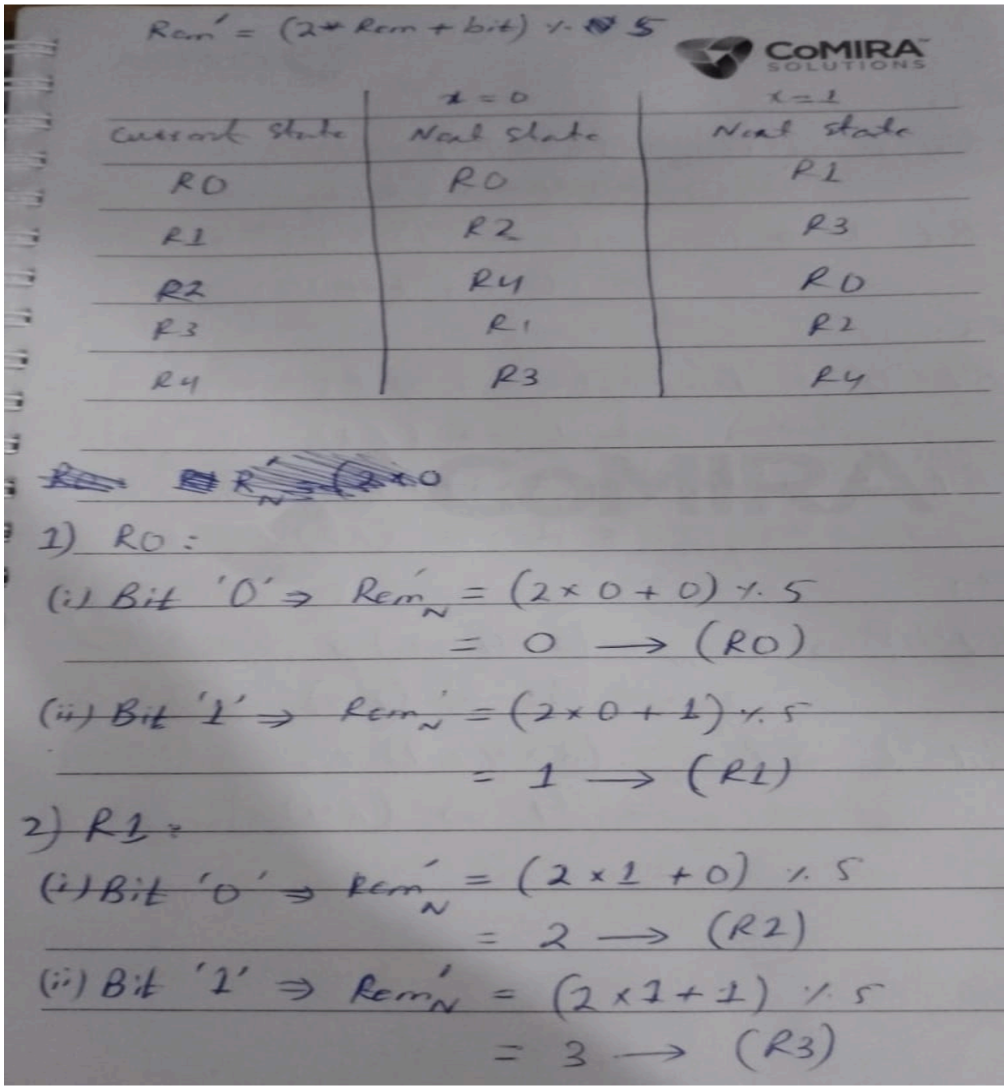
“Input Sequence Divisiblity by 5”

‘Moore Finite State Machine’

Design:





$Rem' = (2 * Rem + bit) \% 5$

CoMIRA SOLUTIONS

	$x = 0$	$x = 1$
Current State	Next State	Next State
R0	R0	R1
R1	R2	R3
R2	R4	R0
R3	R1	R2
R4	R3	R4

~~R0~~ ~~$R_N' = (2 \times 0$~~

1) R0 :

(i) Bit ‘0’ ⇒ $Rem_N' = (2 \times 0 + 0) \% 5$
$= 0 \longrightarrow (R0)$

(ii) Bit ‘1’ ⇒ $Rem_N' = (2 \times 0 + 1) \% 5$
$= 1 \longrightarrow (R1)$

2) R1 :

(i) Bit ‘0’ ⇒ $Rem_N' = (2 \times 1 + 0) \% 5$
$= 2 \longrightarrow (R2)$

(ii) Bit ‘1’ ⇒ $Rem_N' = (2 \times 1 + 1) \% 5$
$= 3 \longrightarrow (R3)$

3) R2 :

(i) Bit '0' ⇒ $Rem'_N = (2 \times 2 + 0) \% 5$
$= 4 \rightarrow (R4)$

(ii) Bit '1' ⇒ $Rem'_N = (2 \times 2 + 1) \% 5$
$= 0 \rightarrow (R0)$

4) R3 :

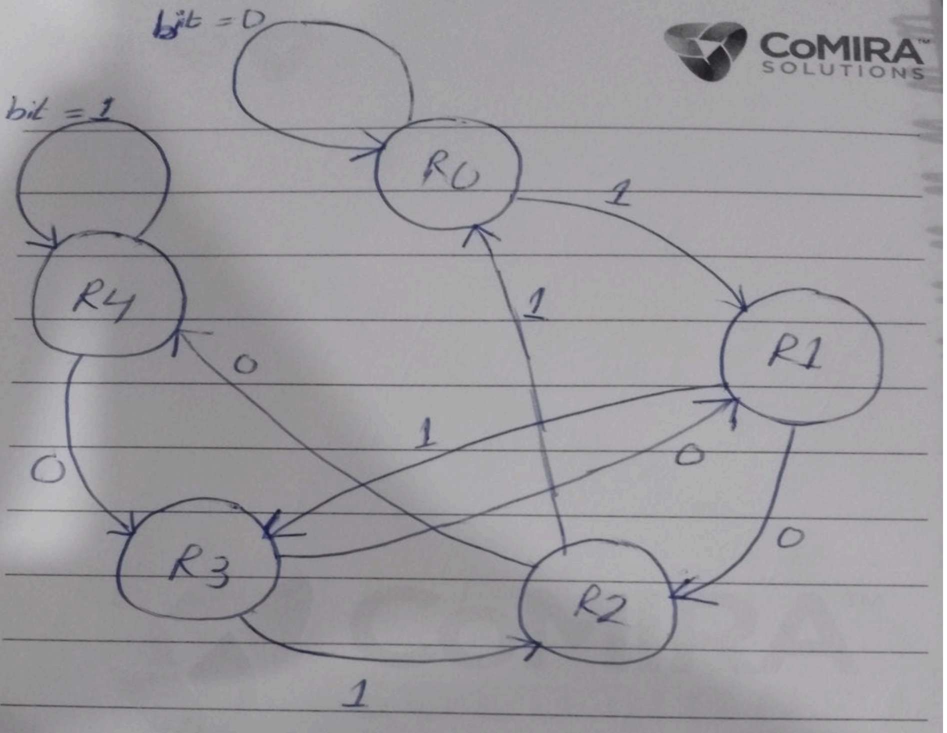
(i) Bit '0' ⇒ $Rem'_N = (2 \times 3 + 0) \% 5$
$= 1 \rightarrow (R1)$

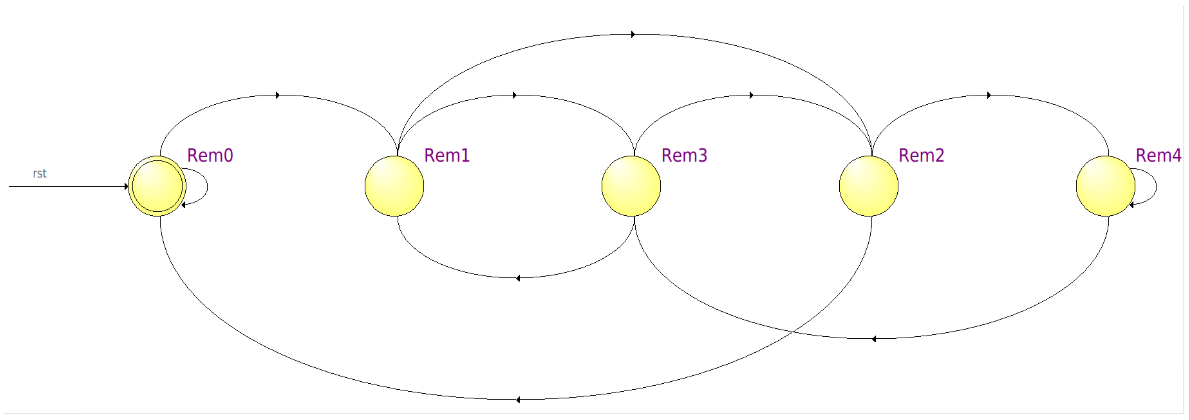
(ii) Bit '1' ⇒ $Rem'_N = (2 \times 3 + 1) \% 5$
$= 2 \rightarrow (R2)$

5) R4 :

(i) Bit '0' ⇒ $Rem'_N = (2 \times 4 + 0) \% 5$
$= 3 \rightarrow (R3)$

(ii) Bit '1' ⇒ $Rem'_N = (2 \times 4 + 1) \% 5$
$= 4 \rightarrow (R4)$



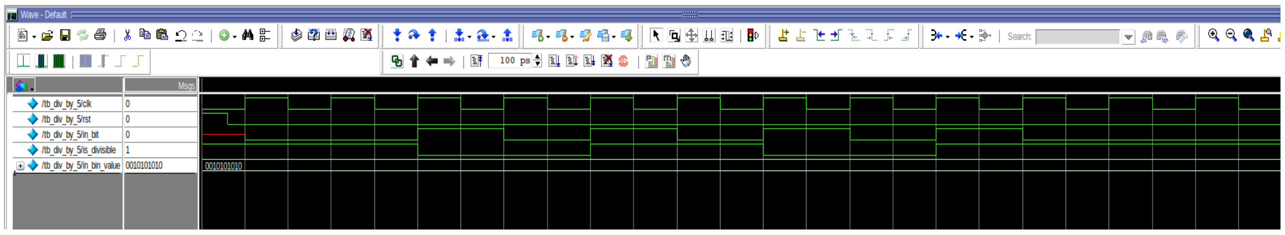


- The design models a moore finite state machine with five states (Rem0, Rem1, Rem2, Rem3, and Rem4), each representing the remainder when dividing input stream by 5.
- On every clock edge, the FSM transitions based on the incoming bit, updating the current remainder state through a reminder prediction formula.
- When the FSM reaches the 'Rem0' state, the output 'is_divisible' is asserted, indicating divisibility by 5.
- Reset initializes the FSM to 'Rem0' , ensuring proper startup.

Formula:

Reminder (new) = [(2 * Reminder (current)) + input_bit)] % 5

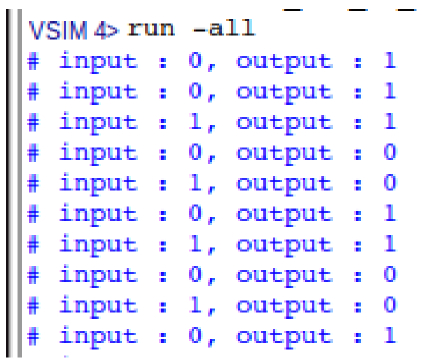
Verification:



```
/tb_div_by_5/clk          0
/tb_div_by_5/rst          0
/tb_div_by_5/in_bit       0
/tb_div_by_5/is_divisible 1
/tb_div_by_5/in_bin_value 0010101010
```

- The testbench instantiates the 'div_by_5' FSM and drives clock, reset, and input signals.
- A predefined 10-bit binary sequence (0010101010) was given in the question itself, so is applied bit-by-bit to the FSM as serial input stream.
- At every clock edge, the divisibility output is displayed alongside the current input bit, showing state behavior.
- The simulation runs through the entire sequence, and then stops and as result same expected output sequence is achieved on console.

Result:



```
VSIM 4> run -all
# input : 0, output : 1
# input : 0, output : 1
# input : 1, output : 1
# input : 0, output : 0
# input : 1, output : 0
# input : 0, output : 1
# input : 1, output : 1
# input : 0, output : 0
# input : 1, output : 0
# input : 0, output : 1
```