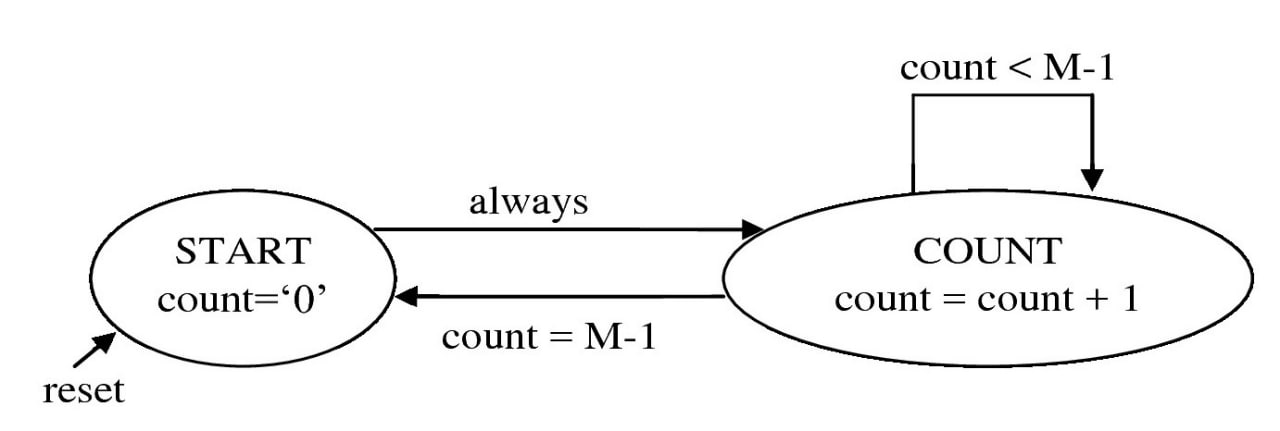
**پروژه state machine درس طراحی سیستم های دیجیتال  
ارایه دهندگان:**

**محمدحسین حسنی  
علی سلطانی  
عارف علیجان نژاد**

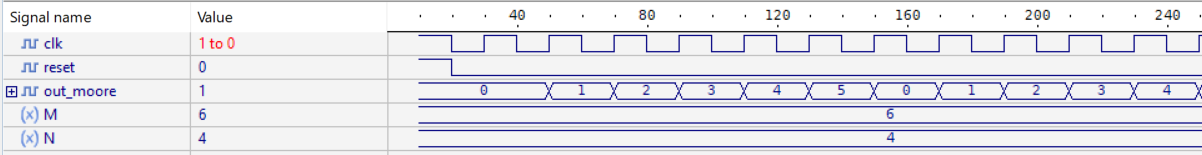
**مهدی حسینی زاده**

**موضوع پروژه:**

**Recursive Machine : Mod-m counter**

**عکس state machine:**

**عکس Waveform:**

****

**توضیحات مربوط به کد:**

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.numeric\_std.all;

entity counterEx is

  generic (

    M : natural := 6;

    N : natural := 4 -- N bits are required for M

  );

  port (

    clk, reset : in std\_logic;

    out\_moore : out std\_logic\_vector(N-1 downto 0)

  );

end entity;

architecture arch of counterEx is

  type stateType\_moore is (start\_moore, count\_moore); -- Define an enumeration type for Moore state

  signal state\_moore\_reg, state\_moore\_next : stateType\_moore; -- Registers to hold current and next Moore state

  signal count\_moore\_reg, count\_moore\_next : unsigned(N-1 downto 0); -- Registers to hold current and next count value

begin

  process (clk, reset)

  begin

    if reset = '1' then -- Asynchronous reset condition

      state\_moore\_reg <= start\_moore; -- Reset the state to start\_moore

      count\_moore\_reg <= (others => '0'); -- Reset the count to all zeros

    elsif rising\_edge(clk) then -- Synchronous behavior on rising edge of the clock

      state\_moore\_reg <= state\_moore\_next; -- Update the current state with the next state

      count\_moore\_reg <= count\_moore\_next; -- Update the current count with the next count

    end if;

  end process;

  process (count\_moore\_reg, state\_moore\_reg)

  begin

    case state\_moore\_reg is

      when start\_moore =>

        count\_moore\_next <= (others => '0'); -- Reset the count to all zeros

        state\_moore\_next <= count\_moore; -- Transition to count\_moore state

      when count\_moore =>

        count\_moore\_next <= count\_moore\_reg + 1; -- Increment the count by 1

        if (count\_moore\_reg + 1) = M - 1 then -- Check if the count reached (M - 1)

          state\_moore\_next <= start\_moore; -- Transition back to start\_moore state

        else

          state\_moore\_next <= count\_moore; -- Stay in the count\_moore state

        end if;

    end case;

  end process;

  out\_moore <= std\_logic\_vector(count\_moore\_reg); -- Convert the count to std\_logic\_vector and assign it to the output

end arch;

**این کد، یک ماشین شمارنده Recursive با نام "Mod-m counter" را پیاده‌سازی می‌کند. این شمارنده، به طور پیوسته از صفر شروع می‌کند و به تعدادی (M) دفعات به بالا شمارش می‌کند، سپس به صفر باز می‌گردد و این فرایند را تکرار می‌کند.**

**کتابخانه‌های ieee.std\_logic\_1164 و ieee.numeric\_std در این کد استفاده شده‌اند. اولی برای استفاده از نوع داده std\_logic و دومی برای استفاده از نوع داده unsigned و توابع مرتبط با آن.**

**پارامترهای generic که در تعریف entity وجود دارند، شامل M و N هستند. M بیانگر تعداد دفعاتی است که شمارنده به بالا می‌شمارد تا به مقدار (M - 1) برسد، و N تعداد بیت‌های مورد نیاز برای نمایش این مقدار است.**

**در بخش architecture، یک enumeration type به نام stateType\_moore تعریف شده است که دو حالت "start\_moore" و "count\_moore" را دارا می‌باشد. سیگنال‌های state\_moore\_reg و state\_moore\_next برای نگهداری حالت فعلی و حالت بعدی ماشین شمارنده استفاده می‌شوند. همچنین، سیگنال‌های count\_moore\_reg و count\_moore\_next برای نگهداری مقدار فعلی و مقدار بعدی شمارنده به کار می‌روند.**

**در دو فرآیند (process) اصلی که در کد وجود دارند، رفتار سیگنال‌ها به صورت همزمان و ناهمزمان تعیین می‌شود.**

**در فرآیند اول، سیگنال reset و clk بررسی می‌شوند. اگر سیگنال reset برابر با '1' باشد، یعنی شرط ریست ناهمزمان برقرار است و در این صورت حالت شمارنده به "start\_moore" تنظیم می‌شود و شمارنده به صفر بازمی‌گردد. اگر سیگنال reset برابر با '0' باشد و در لبه بالارونده (rising edge) سیگنال clk باشیم، حالت و مقدار فعلی شمارنده با حالت و مقدار بعدی جایگزین می‌شوند.**

**در فرآیند دوم، بر اساس حالت فعلی ماشین شمارنده، عملکرد مشخصی انجام می‌شود. در حالت "start\_moore"، مقدار شمارنده به صفر تنظیم می‌شود و حالت بعدی به "count\_moore" تغییر می‌کند. در حالت "count\_moore"، مقدار شمارنده یک واحد افزایش می‌یابد و سپس بررسی می‌شود که آیا مقدار شمارنده برابر با (M - 1) است یا خیر. اگر برابر باشد، حالت بعدی به "start\_moore" تغییر می‌کند و در غیر این صورت، در حالت "count\_moore" باقی می‌ماند.**

**در نهایت، مقدار فعلی شمارنده در قالب std\_logic\_vector تبدیل شده و به سیگنال out\_moore اختصاص داده می‌شود. این سیگنال مقدار شمارنده را به صورت بیت‌ها نمایش می‌دهد.**