**Lab 2 Exercise 4:**

**Design and realize a three-digit decimal counter**

**姓名：王宇 学号：12112725**

**原理图：**

分频器采用的设计如图所示，因100MHZ的时钟频率对应的是10ns，与1s相差，转换为二进制共有27位，故c\_next和c\_reg均为27位。当c\_reg的值不等于-1 时，c\_next+1，当c\_reg的值等于-1 时，即过去1s时，c\_next置为0，重新开始计时。

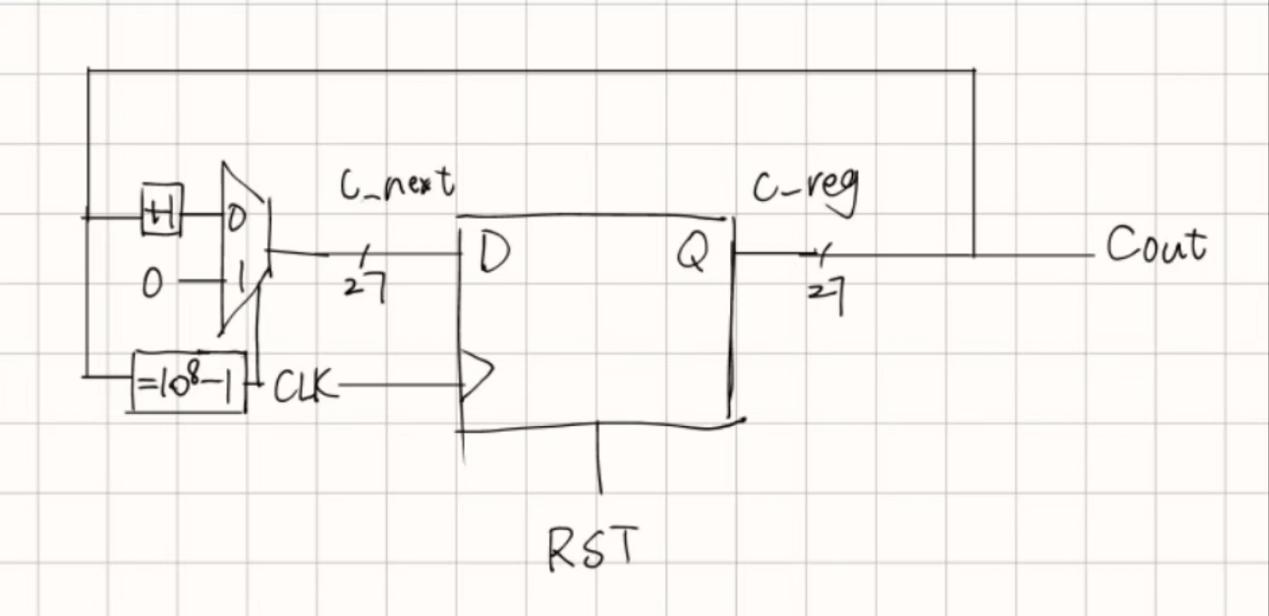
****

图 1 frequency divider概念图

三位计数器采用的设计如图所示，当d1\_reg不等于9时，则+1，等于9时置0，并且传递到下一位的判断，若d2\_reg不等于9且d1\_reg等于9时+1，二者均等于9时，d2\_reg置0，其余逻辑均类似。

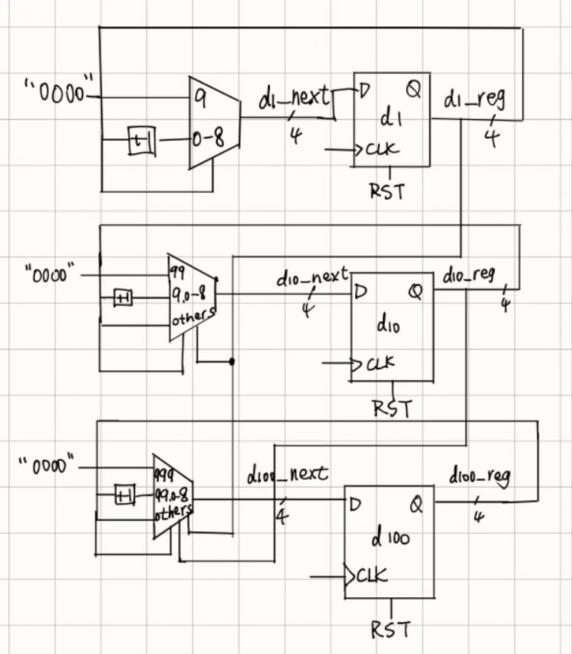


图 2

3-digit decimal counter概念图

总体设计如下图所示，在分频器的输出端加了一个判断条件，并在d1上加了一个使能端EN。若分频器输出为-1，则使能端置为1，计数器开始工作，否则停止工作。图中所有时钟信号均为同一时钟信号。

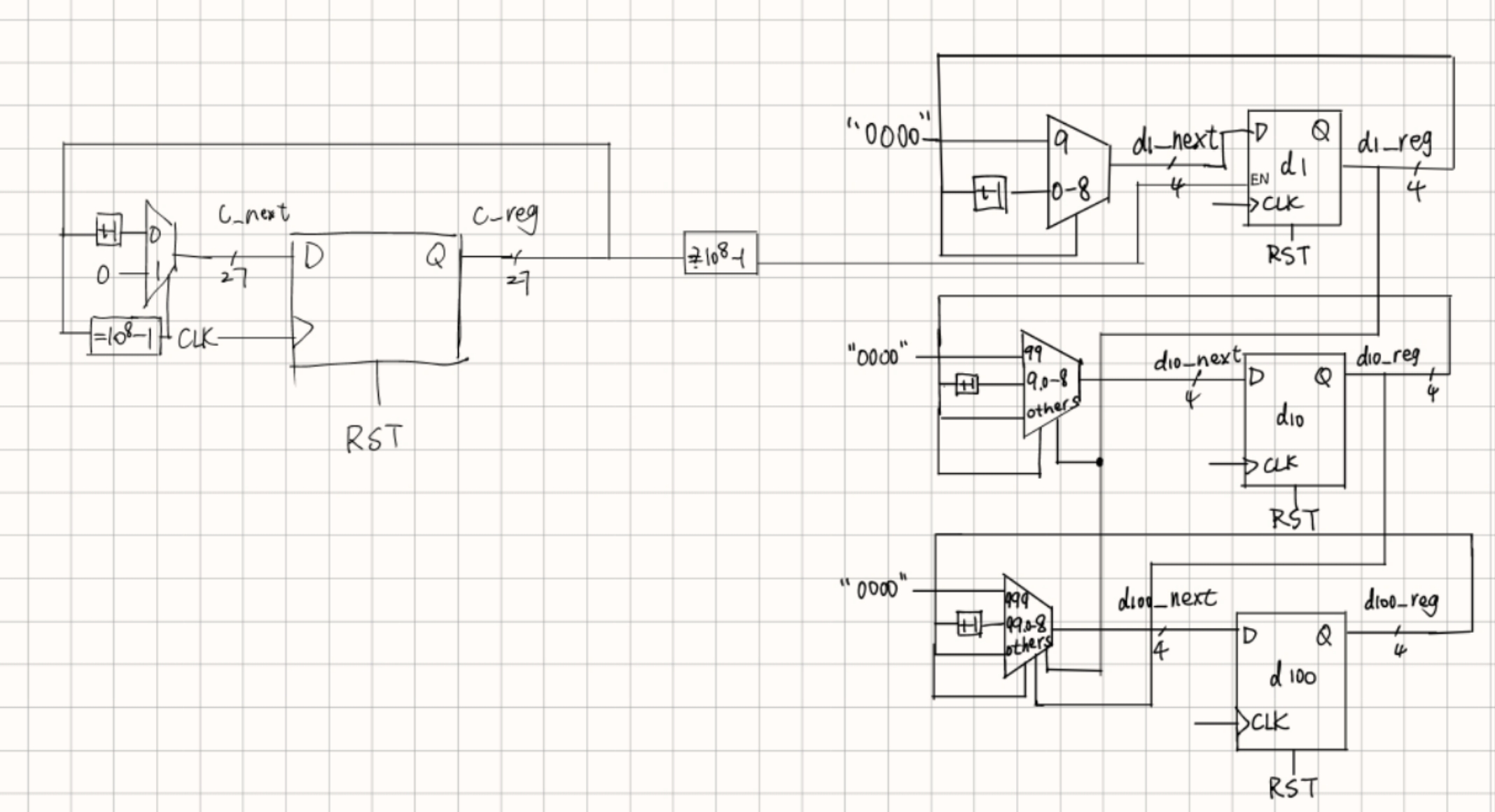


图 3 top level结构图

**代码：**

*VHDL部分：*

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use ieee.numeric\_std.all;

entity decimal\_counter is

Port ( CLOCK : in STD\_LOGIC;

RESET : in STD\_LOGIC;

SW : in STD\_LOGIC\_VECTOR(11 downto 0);

d1, d10, d100: out STD\_LOGIC\_VECTOR(3 downto 0));

end decimal\_counter;

architecture Behavioral of decimal\_counter is

signal d1\_reg, d10\_reg, d100\_reg: STD\_LOGIC\_VECTOR(3 downto 0);

signal c\_reg: STD\_LOGIC\_VECTOR(26 downto 0);

signal bin\_value : STD\_LOGIC\_VECTOR(11 downto 0);

signal EN: STD\_LOGIC;

begin

-- 时钟部分

process(CLOCK, RESET)

begin

if RESET = '0' then

c\_reg <= (others => '0');

EN <= '0';

elsif rising\_edge(CLOCK) then

if c\_reg = "101111101011110000100000000" then -- Replace with your condition

c\_reg <= (others => '0');

EN <= '1';

else

c\_reg <= std\_logic\_vector(unsigned(c\_reg) + 1);

EN <= '0';

end if;

end if;

end process;

-- 计数器部分

process(CLOCK, RESET)

begin

if RESET = '0' then

d1\_reg <= SW(3 downto 0);

d10\_reg <= SW(7 downto 4);

d100\_reg <= SW(11 downto 8);

elsif rising\_edge(CLOCK) then

if EN = '1' then

if unsigned(d1\_reg) = 9 then

d1\_reg <= "0000";

if unsigned(d10\_reg) = 9 then

d10\_reg <= "0000";

if unsigned(d100\_reg) = 9 then

d100\_reg <= "0000";

else

d100\_reg <= std\_logic\_vector(unsigned(d100\_reg) + 1);

end if;

else

d10\_reg <= std\_logic\_vector(unsigned(d10\_reg) + 1);

end if;

else

d1\_reg <= std\_logic\_vector(unsigned(d1\_reg) + 1);

end if;

end if;

end if;

end process;

-- 输出部分

d1 <= d1\_reg;

d10 <= d10\_reg;

d100 <= d100\_reg;

end Behavioral;

*Testbench 部分：*

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use ieee.numeric\_std.all;

ENTITY tb\_decimal\_counter IS

END tb\_decimal\_counter;

ARCHITECTURE behavior OF tb\_decimal\_counter IS

COMPONENT decimal\_counter

PORT(

CLOCK : IN std\_logic;

RESET : IN std\_logic;

SW : IN STD\_LOGIC\_VECTOR(11 downto 0);

d1 : OUT std\_logic\_vector(3 downto 0);

d10 : OUT std\_logic\_vector(3 downto 0);

d100 : OUT std\_logic\_vector(3 downto 0)

);

END COMPONENT;

signal CLOCK : std\_logic := '0';

signal RESET : std\_logic := '0';

signal SW : std\_logic\_vector(11 downto 0) := (others => '0');

signal d1 : std\_logic\_vector(3 downto 0);

signal d10 : std\_logic\_vector(3 downto 0);

signal d100 : std\_logic\_vector(3 downto 0);

constant CLOCK\_PERIOD : time := 10 ns; -- 100MHz时钟

BEGIN

uut: decimal\_counter PORT MAP (

CLOCK => CLOCK,

RESET => RESET,

SW => SW,

d1 => d1,

d10 => d10,

d100 => d100

);

CLOCK\_process :process

begin

CLOCK <= '0';

wait for CLOCK\_PERIOD/2;

CLOCK <= '1';

wait for CLOCK\_PERIOD/2;

end process;

stim\_proc: process

begin

RESET <= '1';

wait for 1000 ns;

RESET <= '0';

wait for 1000 ns;

SW <= "000000000001";

wait for 5000 ms;

end process;

END;

*约束条件部分：*

# Reset button (suppose the button you wish to use for reset is BTN0)

set\_property PACKAGE\_PIN C12 [get\_ports {RESET}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {RESET}]

# Clock signal

set\_property PACKAGE\_PIN E3 [get\_ports {CLOCK}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {CLOCK}]

create\_clock -add -name sys\_clk\_pin -period 10.000 -waveform {0 5} [get\_ports {CLOCK}]

# 12 switches for setting the start number (suppose SW1 - SW12 are used)

set\_property PACKAGE\_PIN V10 [get\_ports {SW[0]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[0]}]

set\_property PACKAGE\_PIN U11 [get\_ports {SW[1]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[1]}]

set\_property PACKAGE\_PIN U12 [get\_ports {SW[2]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[2]}]

set\_property PACKAGE\_PIN H6 [get\_ports {SW[3]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[3]}]

set\_property PACKAGE\_PIN T13 [get\_ports {SW[4]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[4]}]

set\_property PACKAGE\_PIN R16 [get\_ports {SW[5]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[5]}]

set\_property PACKAGE\_PIN U8 [get\_ports {SW[6]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[6]}]

set\_property PACKAGE\_PIN T8 [get\_ports {SW[7]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[7]}]

set\_property PACKAGE\_PIN R13 [get\_ports {SW[8]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[8]}]

set\_property PACKAGE\_PIN U18 [get\_ports {SW[9]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[9]}]

set\_property PACKAGE\_PIN T18 [get\_ports {SW[10]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[10]}]

set\_property PACKAGE\_PIN R17 [get\_ports {SW[11]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {SW[11]}]

# 12 LEDs for showing

set\_property PACKAGE\_PIN R18 [get\_ports { d1[0] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d1[0]}]

set\_property PACKAGE\_PIN V17 [get\_ports { d1[1] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d1[1]}]

set\_property PACKAGE\_PIN U17 [get\_ports { d1[2] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d1[2]}]

set\_property PACKAGE\_PIN U16 [get\_ports { d1[3] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d1[3]}]

set\_property PACKAGE\_PIN V16 [get\_ports { d10[0] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d10[0]}]

set\_property PACKAGE\_PIN T15 [get\_ports { d10[1] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d10[1]}]

set\_property PACKAGE\_PIN U14 [get\_ports { d10[2] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d10[2]}]

set\_property PACKAGE\_PIN T16 [get\_ports { d10[3] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d10[3]}]

set\_property PACKAGE\_PIN V15 [get\_ports { d100[0] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d100[0]}]

set\_property PACKAGE\_PIN V14 [get\_ports { d100[1] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d100[1]}]

set\_property PACKAGE\_PIN V12 [get\_ports { d100[2] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d100[2]}]

set\_property PACKAGE\_PIN V11 [get\_ports { d100[3] }]

set\_property IOSTANDARD LVCMOS33 [get\_ports {d100[3]}]