LAB 3

Deliverable 1: LUT Analysis

The truth table is the same as last week. It is equivalent to the logic expression: $O = I_0\ I_1\ I_2\ I_3$.

Deliverable 2: Counter LUT

The LUT $y[0]\_i\_1$ implements the logic $O = I_0'$, and the LUT $y[1]\_i\_1$ implements the logic $O = I_0 \oplus I_1$. These expressions represent the logic for the two least significant bits in a 4-bit counter which we found in homework.

Deliverable 3: Clock Divider Block Diagram

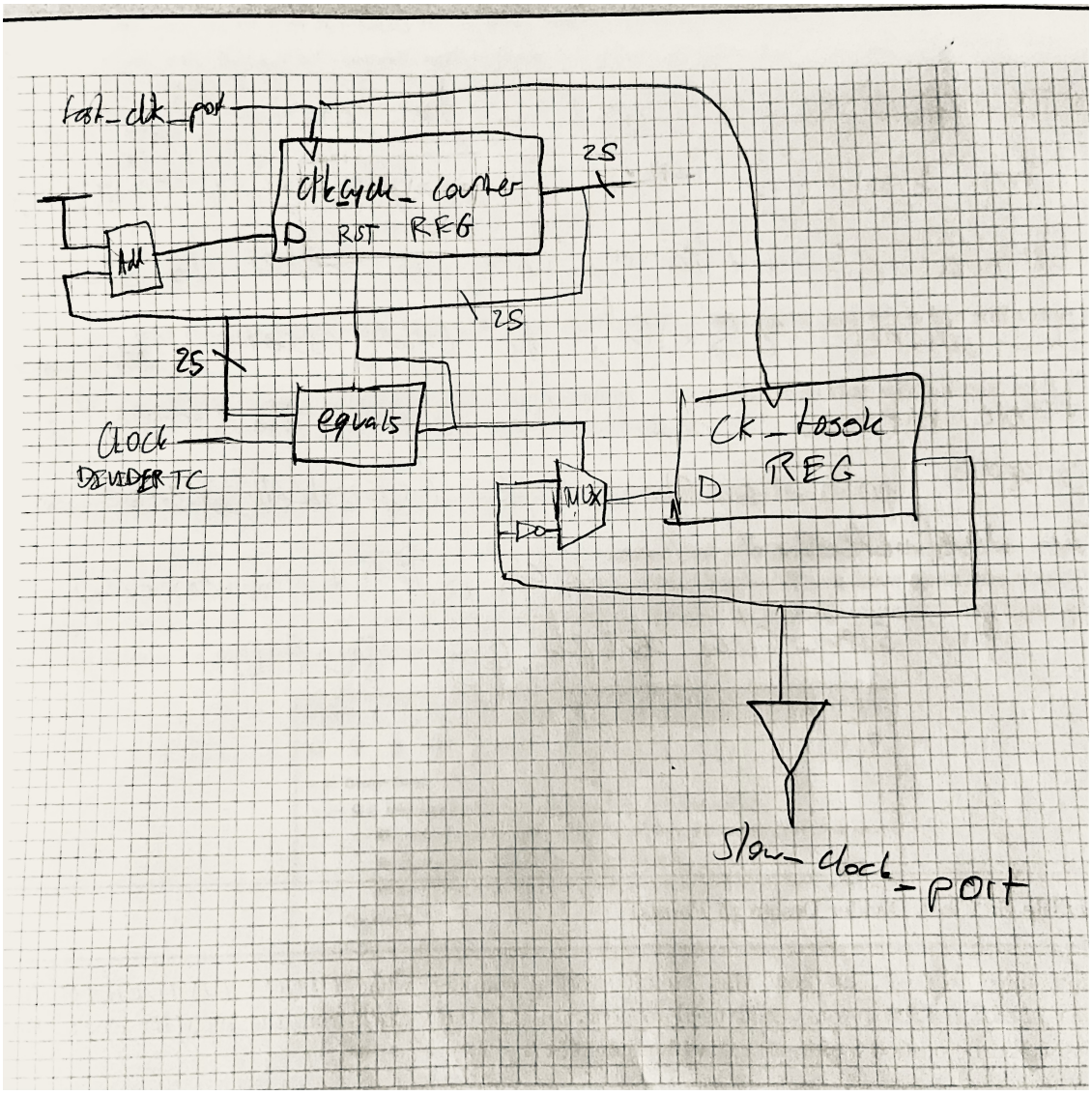


Figure 1: Circuit Divider Block Diagram

Deliverable 4: Clock Divider Discussion

The function of this circuit is to convert the fast clock signal (100 MHz) to a slower clock signal (2 Hz). Using a 25 bit counter, which is needed to store the divisor 25×10^6, the circuit resets at this divisor value, and asserts the toggle bit. The toggle bit corresponds with the slow clock (rising/falling/level) edges as it is connected to the slow_clock output using BUFD.

Deliverable 5: Clock Divider Design

$$CLOCK\_DIVIDER\_TC := 5$$

We would need 3 bits for the `clk_divider_counter`.

Deliverable 6: BCD Enumeration

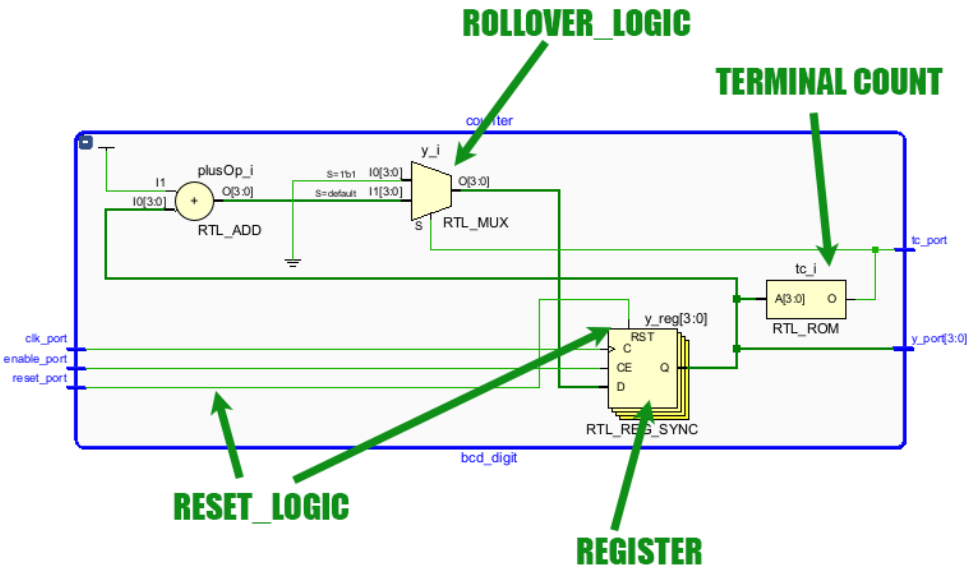


Figure 2: Annotated RTL Schematic for BCD Enumeration

Deliverable 7: Waveforms

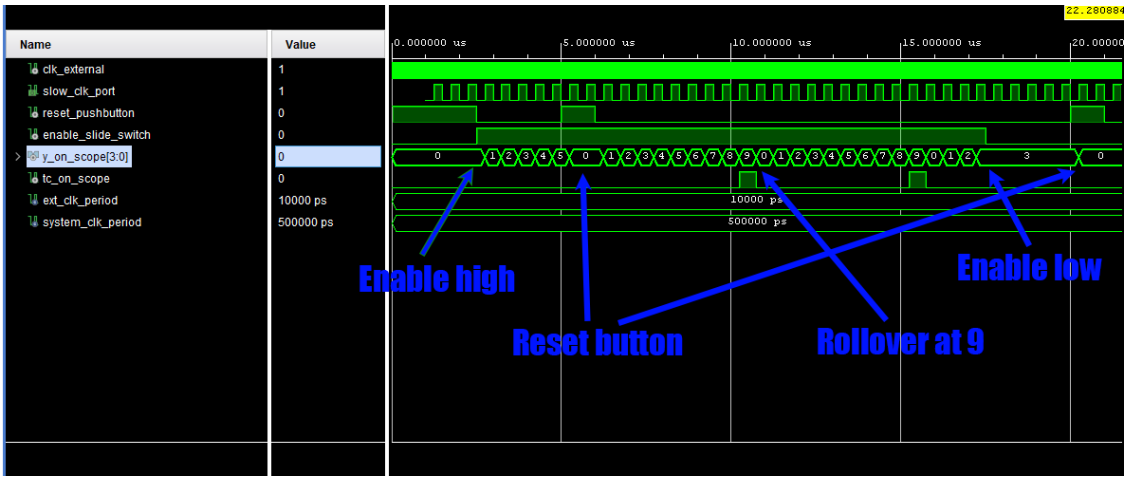


Figure 3: Entire Simulation Runtime, Annotated

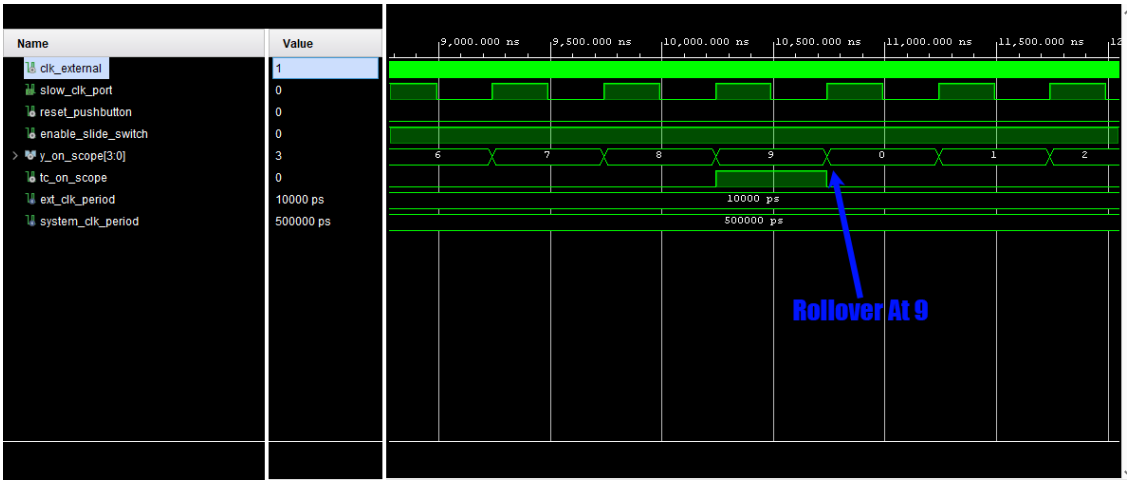
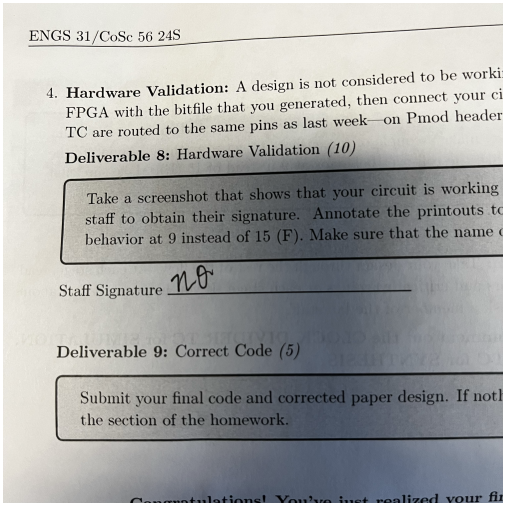


Figure 4: Simulation Zoomed-In on Rollover Behavior, Annotated

Deliverable 8: Hardware Validation


ENGS 31/CoSc 56 24S

4. **Hardware Validation:** A design is not considered to be worki
FPGA with the bitfile that you generated, then connect your ci
TC are routed to the same pins as last week—on Pmod header

Deliverable 8: Hardware Validation *(10)*

Take a screenshot that shows that your circuit is working
staff to obtain their signature. Annotate the printouts to
behavior at 9 instead of 15 (F). Make sure that the name

Staff Signature

Deliverable 9: Correct Code *(5)*

Submit your final code and corrected paper design. If notl
the section of the homework.

Congratulations! You've just realized your fi

Figure 5: Staff Signature

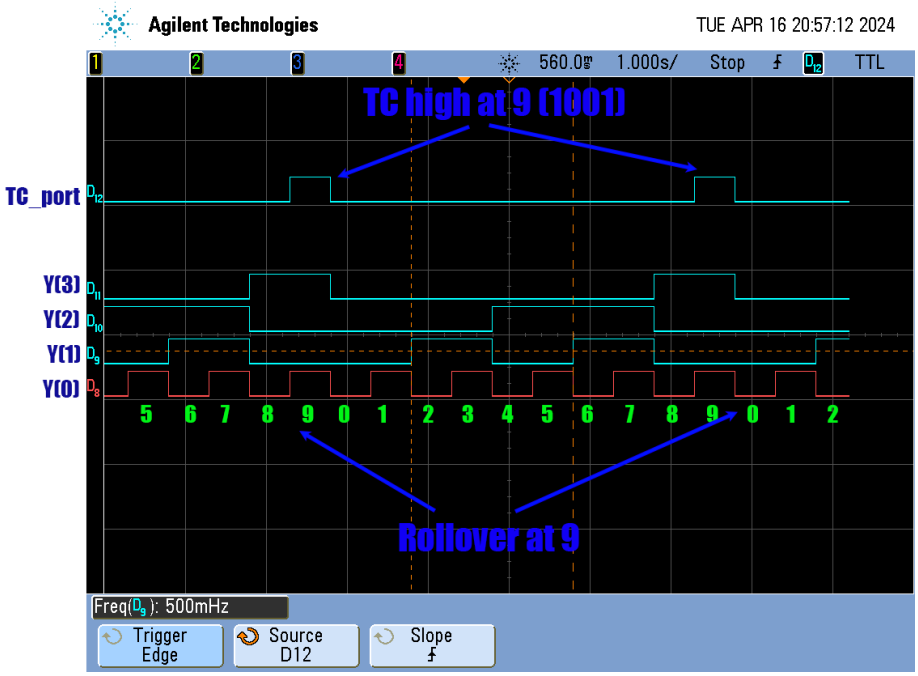


Figure 6: Normal Operation of BCD Circuit, Annotated

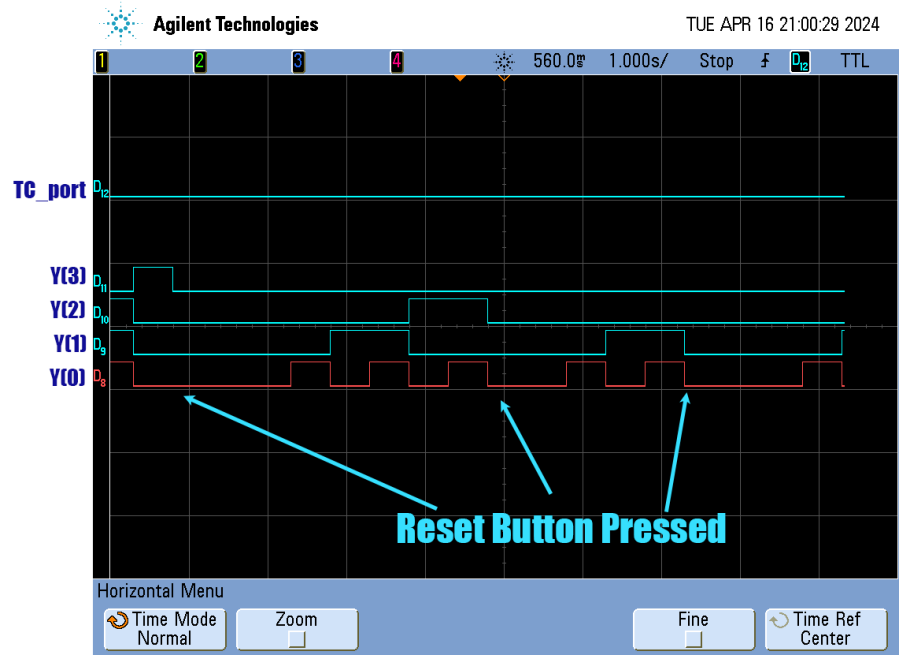


Figure 7: Pressing the Reset Button a Few Times, Annotated

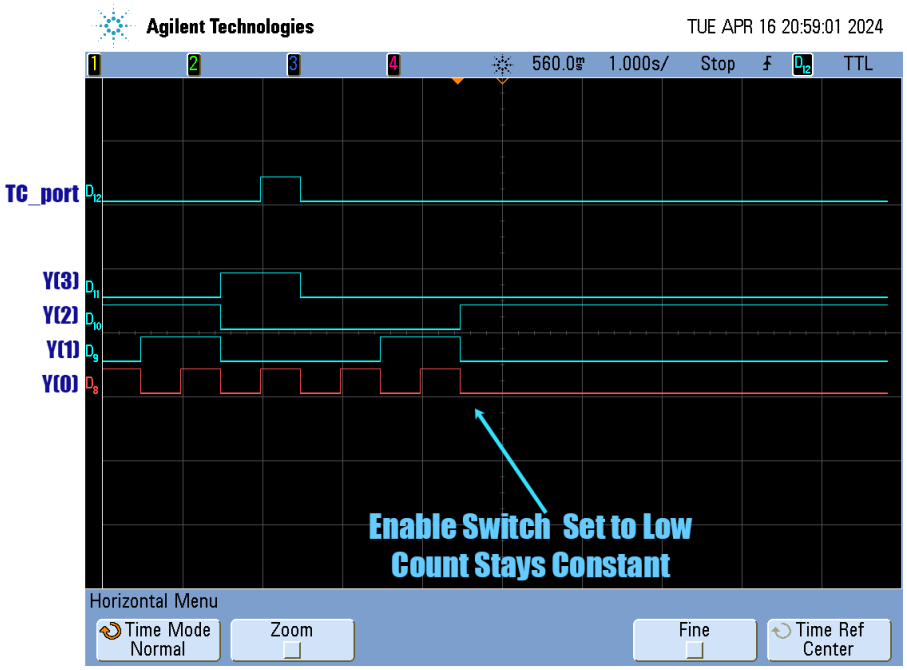


Figure 8: Disabling the Counting, Annotated

Deliverable 9: Correct Code

```
--================================================================================
--ENGS 31/ CoSc 56
--Basic4bitCounter
--Ben Dobbins
--Eric Hansen
--================================================================================

--================================================================================
--Library Declarations:
--================================================================================
library IEEE;
use IEEE.std_logic_1164.all;
use ieee.numeric_std.all;

--================================================================================
--Entity Declaration:
--================================================================================
entity bcd_digit is
  port(clk_port        : in  std_logic;
             reset_port        : in  std_logic;
       enable_port : in std_logic;
             y_port                  : out std_logic_vector(3 downto 0);
       tc_port        : out std_logic );
end entity;

--================================================================================
--Architecture Type:
--================================================================================
architecture behavior of bcd_digit is

--================================================================================
--Signal Declarations:
--================================================================================
-- internal unsigned signal for the counter's register
signal y: unsigned(3 downto 0) := "0000";
signal tc: std_logic := '0';

--================================================================================
--Processes:
--================================================================================
begin

--++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
--4-Bit Counter:
--++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
-- the counter itself
process(clk_port, enable_port)
begin
        if rising_edge(clk_port) then
                if reset_port = '1' then y <= "0000";
                elsif enable_port = '1' then
                  if tc = '1' then y <= "0000";
                  else y <= y+1;
                  end if;
                end if;
        end if;
end process;

--++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
--Terminal Count:
--++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
process(y)
begin
        -- tc is asynchronous, derived from y
        if y = "1001" then tc <= '1';
        else tc <= '0';
        end if;
end process;

--++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
--Output Mapping:
--++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
y_port <= std_logic_vector(y); -- typecast y to std_logic_vector for output
tc_port <= tc;

end behavior;


```

Figure 9: Full VHDL code for `bcd_digit.vhd`