ECE 322
Digital Design with VHDL

Counters and Simple Design Example

Lecture 12
Sequential Logic Review


Chapter 7 Flip-flop, Registers, Counters, and a Simple Processor

In this lecture, we learn how to implement basic sequential blocks using VHDL

- Counters

Example of digital system that make use of sequential logic blocks
Counters
Asynchronous Counters

Up-Counter with T Flip-Flops

(a) Circuit

(b) Timing diagram

A three-bit up-counter
Asynchronous Counters

Down-Counter with T Flip-Flops

(a) Circuit

(b) Timing diagram

A three-bit down-counter
## Synchronous Counters

### Synchronous Up-Counter with T Flip-Flops

<table>
<thead>
<tr>
<th>Clock cycle</th>
<th>$Q_2$</th>
<th>$Q_1$</th>
<th>$Q_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Derivation of the synchronous up-counter
Synchronous Counters

Synchronous Up-Counter with T Flip-Flops

(a) Circuit

(b) Timing diagram
A four-bit synchronous up-counter
Counters with Enable & Clear

Synchronous Up-Counter with Enable & Clear Inputs
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;
ENTITY upcount IS
    PORT (Clock, Resetn, Enable : IN STD_LOGIC;
          Q       : OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
END upcount;

ARCHITECTURE behavioral OF upcount IS
    SIGNAL Count : std_logic_vector(3 DOWNTO 0);
BEGIN
    PROCESS (Clock, Resetn)
    BEGIN
        IF Resetn = '0' THEN
            Count <= "0000";
        ELSIF rising_edge(Clock) THEN
            IF Enable = '1' THEN
                Count <= Count + 1;
            ELSE
                Count <= Count;
            END IF;
        END IF;
    END PROCESS;
    Q <= Count;
END behavioral;
Design Example
A digital system with \( k \) \( n \)-bit registers

Design Example: Bus Structure
Design Example: Bus Structure

Details for connecting registers to a bus
Design Example: Bus Structure

Using a shift register for control
Design Example: Bus Structure

VHDL Code for an $n$-bit register

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY regn IS
  GENERIC ( N : INTEGER := 8 ) ;
  PORT ( R : IN STD_LOGIC_VECTOR(N-1 DOWNTO 0) ;
         Rin, Clock : IN STD_LOGIC ;
         Q : OUT STD_LOGIC_VECTOR(N-1 DOWNTO 0) ) ;
END regn ;
ARCHITECTURE Behavior OF regn IS
BEGIN
  PROCESS
  BEGIN
    WAIT UNTIL Clock'EVENT AND Clock = '1' ;
    IF Rin = '1' THEN
      Q <= R ;
    END IF ;
  END PROCESS ;
END Behavior ;
VHDL Code for an $n$-bit tri-state buffer

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY trin IS
  GENERIC ( N : INTEGER := 8 ) ;
  PORT ( X : IN STD_LOGIC_VECTOR(N-1 DOWNTO 0) ;
        E : IN STD_LOGIC ;
        F : OUT STD_LOGIC_VECTOR(N-1 DOWNTO 0) ) ;
END trin ;

ARCHITECTURE Behavior OF trin IS
BEGIN
  F <= (OTHERS => 'Z') WHEN E = '0' ELSE X ;
END Behavior ;
Design Example: Bus Structure

VHDL Code for the shift register

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY shiftr IS -- left-to-right shift register with async reset
  GENERIC ( K : INTEGER := 4 );
  PORT ( Resetn, Clock, w : IN STD_LOGIC ;
         Q : BUFFER STD_LOGIC_VECTOR(1 TO K) ) ;
END shiftr ;
ARCHITECTURE Behavior OF shiftr IS
BEGIN
  PROCESS ( Resetn, Clock )
  BEGIN
    IF Resetn = '0' THEN
      Q <= (OTHERS => '0') ;
    ELSIF Clock'EVENT AND Clock = '1' THEN
      FOR i IN K DOWNTO 2 LOOP 
        Q(i) <= Q(i-1) ;
      END LOOP ;
      Q(1) <= w ;
    END IF ;
  END PROCESS ;
END Behavior ;
PACKAGE components IS
  COMPONENT regn -- register
    GENERIC ( N : INTEGER := 8 ) ;
    PORT ( R : IN STD_LOGIC_VECTOR(N-1 DOWNTO 0) ;
           Rin, Clock : IN STD_LOGIC ;
           Q : OUT STD_LOGIC_VECTOR(N-1 DOWNTO 0) ) ;
  END COMPONENT ;

  COMPONENT shiftr -- left-to-right shift register with async reset
    GENERIC ( K : INTEGER := 4 ) ;
    PORT ( Resetn, Clock, w : IN STD_LOGIC ;
           Q : BUFFER STD_LOGIC_VECTOR(1 TO K) ) ;
  END component ;

  COMPONENT trin -- tri-state buffers
    GENERIC ( N : INTEGER := 8 ) ;
    PORT ( X : IN STD_LOGIC_VECTOR(N-1 DOWNTO 0) ;
           E : IN STD_LOGIC ;
           F : OUT STD_LOGIC_VECTOR(N-1 DOWNTO 0) ) ;
  END COMPONENT ;

END components ;
Design Example: Bus Structure

VHDL code for a circuit that swaps the contents of two registers

LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
USE work.components.all ;

ENTITY swap IS
    PORT (Data : IN STD_LOGIC_VECTOR(7 DOWNTO 0) ;
          Resetn, w : IN STD_LOGIC ;
          Clock, Extern : IN STD_LOGIC ;
          RinExt : IN STD_LOGIC_VECTOR(1 TO 3) ;
          BusWires : INOUT STD_LOGIC_VECTOR(7 DOWNTO 0) ) ;
END swap ;
Design Example: Bus Structure

VHDL code for a circuit that swaps the contents of two registers

ARCHITECTURE Behavior OF swap IS
  SIGNAL Rin, Rout, Q : STD_LOGIC_VECTOR(1 TO 3);
  SIGNAL R1, R2, R3 : STD_LOGIC_VECTOR(7 DOWNTO 0);
BEGIN
  control: shiftr GENERIC MAP ( K => 3 )
    PORT MAP ( Resetn, Clock, w, Q ) ;
  Rin(1) <= RinExt(1) OR Q(3) ;
  Rin(2) <= RinExt(2) OR Q(2) ;
  Rin(3) <= RinExt(3) OR Q(1) ;
  Rout(1) <= Q(2) ; Rout(2) <= Q(1) ; Rout(3) <= Q(3) ;
  tri_ext: trin PORT MAP ( Data, Extern, BusWires ) ;
  reg1: regn PORT MAP ( BusWires, Rin(1), Clock, R1 ) ;
  reg2: regn PORT MAP ( BusWires, Rin(2), Clock, R2 ) ;
  reg3: regn PORT MAP ( BusWires, Rin(3), Clock, R3 ) ;
  tri1: trin PORT MAP ( R1, Rout(1), BusWires ) ;
  tri2: trin PORT MAP ( R2, Rout(2), BusWires ) ;
  tri3: trin PORT MAP ( R3, Rout(3), BusWires ) ;
END Behavior ;