

\*\*\*\*\*  
EE/CS 119A Homework 4  
Edward Speer  
October 30, 2024  
\*\*\*\*\*

```
-----  
--  
-- BCD2Binary8 Combinatorial Logic Design  
--  
-- This is an entity and architecture declaration for a combinatorial  
-- circuit which converts an 8-bit BCD value into an 8-bit binary value.  
-- This design assumes the input will be a valid 8-bit BCD value.  
--  
-- Inputs:  
--     BCD[7..0] - 8-bit BCD value to convert to binary  
--  
-- Outputs:  
--     B[7..0]   - 8-bit binary value converted from BCD  
--  
-- Revision History:  
--     10/27/24 Edward Speer   Initial Revision  
--  
-----  
  
-- IMPORTS  
library ieee;  
use ieee.std_logic_1164.all;  
use ieee.numeric_std.all;  
  
--  
-- BCD2Binary8 entity declaration  
--  
entity BCD2Binary8 is  
    port (  
        BCD : in  std_logic_vector(7 downto 0);    -- 8-bit BCD in  
        B    : out std_logic_vector(7 downto 0)     -- 8-bit Binary out  
    );  
end BCD2Binary8;  
  
--  
-- BCD2Binary8 dataFlow architecture  
--  
architecture dataFlow of BCD2Binary8 is  
begin  
    B <= std_logic_vector(  
  
        -- Multiply the high nibble by 10, using left shifts:  $10 = 2^3 + 2^1$   
        -- Pad operations on high nibble out to 8-bit to avoid overflow  
        (unsigned("0000" & BCD(7 downto 4)) sll 3) +  
        (unsigned("0000" & BCD(7 downto 4)) sll 1) +
```

```
        -- Pad the low nibble out to 8-bit to avoid overflow
        unsigned("0000" & BCD(3 downto 0))
    );
end dataFlow;
```

```

-----
--
-- ic74194 Sequential Logic Design
--
-- This is an entity and architecture declaration for a 4-bit bi-directional
-- shift register. This shift register implements the integrated circuit
-- 74LS194A. The ic74194 includes an asynchronous global clear.
--
-- Inputs:
--     CLK      - Clock signal
--     CLR      - Asynchronous clear
--     LSI      - Left shift serial input
--     RSI      - Right shift serial input
--     S[1..0]  - Select signals for shift register mode mux
--     DI[3..0] - Parallel data inputs
--
-- Outputs:
--     DO[3..0] - Data buffer/output signals
--
-- Mode Selection:
--     00 => HOLD
--     01 => SHIFT RIGHT
--     10 => SHIFT LEFT
--     11 => LOAD PARALLEL
--
-- Revision History:
--     10/27/24  Edward Speer      Initial Revision
--
-----

-- IMPORTS
library ieee;
use ieee.std_logic_1164.all;

--
-- ic74194 entity declaration
--

entity ic74194 is
    port(
        CLK : in std_logic;           -- Clock signal
        CLR : in std_logic;           -- Active low asynch clear
        LSI : in std_logic;           -- Left shift serial input
        RSI : in std_logic;           -- Right shift serial input
        S   : in std_logic_vector(1 downto 0); -- Mode mux select signals
        DI  : in std_logic_vector(3 downto 0); -- Parallel data input
        DO  : buffer std_logic_vector(3 downto 0) -- Data outputs
    );
end ic74194;

--
-- ic74194 dataFlow architecture
--

```

```

architecture dataFlow of ic74194 is
begin
  process(CLK, CLR)          -- CLR is asynch, so is in sensitivity list
  begin
    if CLR = '0' then        -- If CLR low, override mux mode behavior
      DO <= "0000";
    elsif rising_edge(CLK) then
      case S is
        when "01"    => DO <= RSI & DO(3 downto 1); -- 01 => SHIFT RIGHT
        when "10"    => DO <= DO(2 downto 0) & LSI; -- 10 => SHIFT LEFT
        when "11"    => DO <= DI;                  -- 11 => LOAD
        when others => DO <= DO;                    -- 00 => HOLD
      end case;
    end if;
  end process;
end dataFlow;

```

```

-----
--
-- UniversalSR8 sequential logic design
--
-- This is an entity and architecture declaration for an 8-bit universal
-- shift register. This register is implemented using a pair of 74LS194A
-- 4-bit shift registers as components. UniversalSR8 includes an
-- asynchronous global clear.
--
-- Inputs:
--     CLK      - Clock Signal
--     CLR      - Active low asynchronous clear
--     LSI      - Left shift serial input
--     RSI      - Right shift serial input
--     S[1..0]  - Select signals for shift register mode mux
--     D[7..0]  - Parallel data Input
--
-- Outputs:
--     Q[7..0]  - Data buffer/output signals
--
-- Mode Selection:
--     00 => HOLD
--     01 => SHIFT RIGHT
--     10 => SHIFT LEFT
--     11 => LOAD PARALLEL
--
-- Revision History:
--     10/27/24  Edward Speer      Initial Revision
--
-----

-- IMPORTS
library ieee;
use ieee.std_logic_1164.all;

--
-- UniversalSR8 entity declaration
--

entity UniversalSR8 is
    port(
        CLK      : in std_logic;           -- Clock signal
        CLR      : in std_logic;           -- Active low asynch clear
        LSer     : in std_logic;           -- Left shift serial input
        RSer     : in std_logic;           -- Right shift serial input
        Mode     : in std_logic_vector(1 downto 0); -- Mode mux select signals
        D        : in std_logic_vector(7 downto 0); -- Parallel data input
        Q        : buffer std_logic_vector(7 downto 0) -- Data outputs
    );
end UniversalSR8;

architecture Structural of UniversalSR8 is

    --

```

```

-- Component declaration of ic74174
--

component ic74194 is
    port(
        CLK : in std_logic;           -- Clock signal
        CLR : in std_logic;           -- Active low asynch clear
        LSI : in std_logic;           -- Left shift serial in
        RSI : in std_logic;           -- Right shift serial in
        S   : in std_logic_vector(1 downto 0); -- Mode mux select signals
        DI  : in std_logic_vector(3 downto 0); -- Parallel data input
        DO  : buffer std_logic_vector(3 downto 0) -- Data outputs
    );
end component;

begin

    -- The lower nibble is one instance of an ic74194
    U1: ic74194 port map (
        CLK => CLK,
        CLR => CLR,
        LSI => LSer,           -- Lower nibble shifts in system LSer on left
        RSI => Q(4),           -- Shift in low bit of upper nibble from right
        S   => Mode,
        DI  => D(3 downto 0), -- Lower nibble is bits 3..0 for I/O
        DO  => Q(3 downto 0)
    );

    -- The upper nibble is a second instance of an ic74194
    U2: ic74194 port map (
        CLK => CLK,
        CLR => CLR,
        LSI => Q(3),           -- Shift in high bit of lower nibble on left
        RSI => RSer,           -- Shift in system RSer on the right
        S   => Mode,
        DI  => D(7 downto 4), -- Upper nibble is bits 7..4 for I/O
        DO  => Q(7 downto 4)
    );

end Structural;

```