```
// ********************************************************************************************************
// **                                                                                                    **
// **   DAC8420.v - DAC8420 12-BIT SERIAL DAC (VDD = +5.0V)                                              **
// **                                                                                                    **
// ********************************************************************************************************
// **                                                                                                    **
// **                              COPYRIGHT (c) 2002 YOUNG ENGINEERING                                  **
// **                                      ALL RIGHTS RESERVED                                           **
// **                                                                                                    **
// **   THIS PROGRAM IS CONFIDENTIAL AND  A  TRADE SECRET  OF  YOUNG  ENGINEERING.  THE RECEIPT OR       **
// **   POSSESSION  OF THIS PROGRAM  DOES NOT  CONVEY  ANY  RIGHTS TO  REPRODUCE  OR  DISCLOSE ITS      **
// **   CONTENTS,  OR TO MANUFACTURE, USE, OR SELL  ANYTHING  THAT IT MAY DESCRIBE, IN WHOLE OR IN      **
// **   PART, WITHOUT THE SPECIFIC WRITTEN CONSENT OF YOUNG ENGINEERING.                                 **
// **                                                                                                    **
// ********************************************************************************************************
// **   Revision       : 1.0                                                                             **
// **   Modified Date  : 01/15/2002                                                                      **
// **   Revision History:                                                                                **
// **                                                                                                    **
// **   01/15/2002:  Initial design                                                                      **
// **                                                                                                    **
// ********************************************************************************************************
// **                                       TABLE OF CONTENTS                                            **
// ********************************************************************************************************
// **----------------------------------------------------------------------------------------------------**
// **   DECLARATIONS                                                                                     **
// **----------------------------------------------------------------------------------------------------**
// **----------------------------------------------------------------------------------------------------**
// **   INITIALIZATION                                                                                   **
// **----------------------------------------------------------------------------------------------------**
// **----------------------------------------------------------------------------------------------------**
// **   CORE LOGIC                                                                                       **
// **----------------------------------------------------------------------------------------------------**
// **----------------------------------------------------------------------------------------------------**
// **   TIMING CHECKS                                                                                    **
// **----------------------------------------------------------------------------------------------------**
// **                                                                                                    **
// ********************************************************************************************************


`timescale 1ns/10ps

module DAC8420 (VOUTA, VOUTB, VOUTC, VOUTD, SDI, CLK, CS_N, LD_N, CLR_N, CLSEL, RESET);

   input               SDI;                        // serial data in
   input               CLK;                        // serial data clock

   input               CS_N;                       // chip select
   input               LD_N;                       // DAC load strobe

   input               CLR_N;                      // DAC clear strobe
   input               CLSEL;                      // clear command select

   input               RESET;                      // system reset

   output [11:00]      VOUTA;                      // DAC A output voltage
   output [11:00]      VOUTB;                      // DAC B output voltage
   output [11:00]      VOUTC;                      // DAC C output voltage
   output [11:00]      VOUTD;                      // DAC D output voltage


// ********************************************************************************************************
// **   DECLARATIONS                                                                                     **
// ********************************************************************************************************

   reg  [15:00]        ShiftRegister;              // serial shift register
   wire                ShiftClock;                 // shift register clock

   reg  [11:00]        VOUTA;                      // DAC A output voltage
   reg  [11:00]        VOUTB;                      // DAC B output voltage
   reg  [11:00]        VOUTC;                      // DAC C output voltage
   reg  [11:00]        VOUTD;                      // DAC D output voltage


// ********************************************************************************************************
// **   INITIALIZATION                                                                                   **
// ********************************************************************************************************

   initial begin
```

```
      VOUTA = 12'hXXX;
      VOUTA = 12'hXXX;
      VOUTA = 12'hXXX;
      VOUTA = 12'hXXX;
   end


// ***************************************************************************************************
// **   CORE LOGIC                                                                                  **
// ***************************************************************************************************

   assign ShiftClock = CLK | CS_N;

   always @(posedge ShiftClock) begin
      ShiftRegister[00] <= SDI;
      ShiftRegister[01] <= ShiftRegister[00];
      ShiftRegister[02] <= ShiftRegister[01];
      ShiftRegister[03] <= ShiftRegister[02];
      ShiftRegister[04] <= ShiftRegister[03];
      ShiftRegister[05] <= ShiftRegister[04];
      ShiftRegister[06] <= ShiftRegister[05];
      ShiftRegister[07] <= ShiftRegister[06];
      ShiftRegister[08] <= ShiftRegister[07];
      ShiftRegister[09] <= ShiftRegister[08];
      ShiftRegister[10] <= ShiftRegister[09];
      ShiftRegister[11] <= ShiftRegister[10];
      ShiftRegister[12] <= ShiftRegister[11];
      ShiftRegister[13] <= ShiftRegister[12];
      ShiftRegister[14] <= ShiftRegister[13];
      ShiftRegister[15] <= ShiftRegister[14];
   end

   always @(LD_N or CLR_N or CLSEL or ShiftRegister) begin
      if (CLR_N == 0) begin
         if (CLSEL == 0) begin
            VOUTA = 12'h000;
            VOUTB = 12'h000;
            VOUTC = 12'h000;
            VOUTD = 12'h000;
         end
         else begin
            VOUTA = 12'h800;
            VOUTB = 12'h800;
            VOUTC = 12'h800;
            VOUTD = 12'h800;
         end
      end
      else if (LD_N == 0) begin
         case (ShiftRegister[15:14])
            2'b00 : VOUTA = ShiftRegister[11:00];
            2'b01 : VOUTB = ShiftRegister[11:00];
            2'b10 : VOUTC = ShiftRegister[11:00];
            2'b11 : VOUTD = ShiftRegister[11:00];
         endcase
      end
   end


// ***************************************************************************************************
// **   TIMING CHECKS                                                                               **
// ***************************************************************************************************

   specify
      specparam
         tCL  = 120,                                  // CLK pulse width - low
         tCH  = 90,                                   // CLK pulse width - high
         tDS  = 25,                                   // SDI to CLK setup time
         tDH  = 55,                                   // SDI to CLK hold time
         tCSS = 90,                                   // CS_N to CLK setup time
         tCSH = 5,                                    // CS_N to CLK hold time
         tLD1 = 130,                                  // LD_N to CLK setup time
         tLD2 = 35,                                   // LD_N to CLK hold time
         tLDW = 80,                                   // LD_N pulse width - low
         tCLRW = 150;                                 // CLR_N pulse width - low

      $width (posedge CLK, tCH);
      $width (negedge CLK, tCL);
      $width (negedge LD_N,  tLDW);
      $width (negedge CLR_N, tCLRW);
```

```
      $setup (SDI,  posedge CLK &&& (RESET==0), tDS);
      $setup (CS_N, posedge CLK &&& (RESET==0), tCSS);
      $setup (LD_N, posedge CLK &&& (RESET==0), tLD1);

      $hold  (posedge CLK &&& (RESET==0), SDI,  tDH);
      $hold  (posedge CLK &&& (RESET==0), CS_N, tCSH);
      $hold  (posedge CLK &&& (RESET==0), LD_N, tLD2);
   endspecify

endmodule
```