

```

// *****
// **
// ** AD7847.v - AD7847 12-BIT PARALLEL MDAC (A/B/S GRADES)
// **
// *****
// **
// **          COPYRIGHT (c) 2000 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 : 02/28/2000
// ** Revision History:
// **
// ** 02/28/2000: Initial design
// **
// *****
// **          TABLE OF CONTENTS
// **          -----
// **  DECLARATIONS
// **          -----
// **  INITIALIZATION
// **          -----
// **  CORE LOGIC
// **          -----
// **  TIMING CHECKS
// **          -----
// **
// *****

```

```
`timescale 1ns/10ps
```

```

module AD7847      ( VOUTA,
                   VOUTB,

                   DB00,
                   DB01,
                   DB02,
                   DB03,
                   DB04,
                   DB05,
                   DB06,
                   DB07,
                   DB08,
                   DB09,
                   DB10,
                   DB11,

                   WR_N,
                   CSA_N,
                   CSB_N,

                   RESET
                 );

input  DB00;          // parallel data bus in
input  DB01;          // parallel data bus in
input  DB02;          // parallel data bus in
input  DB03;          // parallel data bus in
input  DB04;          // parallel data bus in
input  DB05;          // parallel data bus in
input  DB06;          // parallel data bus in
input  DB07;          // parallel data bus in
input  DB08;          // parallel data bus in
input  DB09;          // parallel data bus in
input  DB10;          // parallel data bus in
input  DB11;          // parallel data bus in

input  WR_N;          // data write strobe

```

```

input          CSA_N;          // DAC A chip select
input          CSB_N;          // DAC B chip select

input          RESET;         // system reset

output [11:00] VOUTA;         // DAC A voltage output
output [11:00] VOUTB;         // DAC B voltage output

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

reg [11:00]    DACA_InputLatch; // DAC A input data latch
reg [11:00]    DACB_InputLatch; // DAC B input data latch

reg [11:00]    VOUTA;           // DAC A output voltage
reg [11:00]    VOUTB;           // DAC B output voltage

wire [11:00]   DataDI;         // parallel data bus input

wire           WrStrobe_DACA;   // DAC A decoded write strobe
wire           WrStrobe_DACB;   // DAC B decoded write strobe

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

initial begin
    VOUTA = 12'hXXX;
    VOUTB = 12'hXXX;
end

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

assign DataDI[11:00] = {DB11,DB10,DB09,DB08,DB07,DB06,DB05,DB04,DB03,DB02,DB01,DB00};

assign WrStrobe_DACA = WR_N | CSA_N;
assign WrStrobe_DACB = WR_N | CSB_N;

always @(posedge WrStrobe_DACA) begin
    VOUTA = DataDI[11:00];
end

always @(posedge WrStrobe_DACB) begin
    VOUTB = DataDI[11:00];
end

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

specify
    specparam
        t3 = 30, // WR_N pulse width - low
        t4 = 80; // DB to WR_N setup time

    $width (negedge WR_N, t3);

    $setup (DB00, posedge WR_N &&& (RESET==0), t4);
    $setup (DB01, posedge WR_N &&& (RESET==0), t4);
    $setup (DB02, posedge WR_N &&& (RESET==0), t4);
    $setup (DB03, posedge WR_N &&& (RESET==0), t4);
    $setup (DB04, posedge WR_N &&& (RESET==0), t4);
    $setup (DB05, posedge WR_N &&& (RESET==0), t4);
    $setup (DB06, posedge WR_N &&& (RESET==0), t4);
    $setup (DB07, posedge WR_N &&& (RESET==0), t4);
    $setup (DB08, posedge WR_N &&& (RESET==0), t4);
    $setup (DB09, posedge WR_N &&& (RESET==0), t4);
    $setup (DB10, posedge WR_N &&& (RESET==0), t4);
    $setup (DB11, posedge WR_N &&& (RESET==0), t4);
endspecify
endmodule

```

