// FUNCTION: Analog to Digital Converter

// VERSION: $Revision: 2.12 $

// AUTHOR: Cadence Design Systems, Inc.

//

// GENERATED BY: Cadence Modelwriter 2.31

// ON: Fri Mar 07 09:37:38 EST 2008

//

// Description: Ideal Analog to Digital Converter

// Generates an N bit ADC.

// - selectable logic output levels

// - model valid for negative values of vmin

// - adjustable conversion time, and rise/fall time

// This model is an example, provided "as is" without express or

// implied warranty and with no claim as to its suitability for

// any purpose.

//

// PARAMETERS:

// slack = The smallest time interval considered negligible for

// cross event on clock [S]

// tconv = Delay from threshold crossing to output change [S]

// trise = Rise time for digital output signals [S]

// trise = Rise time for digital output signals [S]

// vmax = ADC Full scale output voltage [V]

// vmin = ADC Zero scale output voltage [V]

// vone = The voltage of a logical 1 on digital outputs [V]

// vth = Threshold value of clock signal [V]

// vzero = The voltage of a logical 0 on digital outputs [V]

//

`include "discipline.h"

`include "constants.h"

`define NUM\_ADC\_BITS 10

module adc\_8bit (vin, clk, data);

input vin, clk;

electrical vin, clk;

output [`NUM\_ADC\_BITS-1:0] data;

electrical [`NUM\_ADC\_BITS-1:0] data;

parameter real vmax = 0.750;

parameter real vmin = 0.250;

parameter real one = 1.8;

parameter real zero =0;

parameter real vth = 0;

parameter real slack = 0.5p from (0:inf);

parameter real trise = 1.0p from (0:inf);

parameter real tfall = 1.0p from (0:inf);

parameter real tconv = 0.5p from [0:inf);

parameter integer traceflag = 1;

real sample, vref, lsb, voffset;

real vd[0:`NUM\_ADC\_BITS-1];

integer ii, binvalue;

analog begin

@(initial\_step or initial\_step("dc", "ac", "tran", "xf")) begin

vref = (vmax - vmin) / 2.0;

lsb = (vmax - vmin) / (1 << `NUM\_ADC\_BITS) ;

voffset = vmin;

if (traceflag)

$display("%M ADC range ( %g v ) / %d bits = lsb %g volts.\n",

vmax - vmin, `NUM\_ADC\_BITS, lsb );

generate i ( `NUM\_ADC\_BITS-1, 0) begin

vd[i] = 0 ;

end

end

@(cross ( V(clk)-vth, 1, slack, clk.potential.abstol)) begin

binvalue = 0;

sample = V(vin) - voffset;

for ( ii = `NUM\_ADC\_BITS -1 ; ii>=0 ; ii = ii -1 ) begin

vd[ii] = 0;

if (sample > vref ) begin

vd[ii] = one;

sample = sample - vref;

binvalue = binvalue + ( 1 << ii );

end

else begin

vd[ii] = zero;

end

sample = sample \* 2.0;

end

if (traceflag)

$strobe("%M at %g sec. digital out: %d vin: %g (d2a: %g)\n",

$abstime, binvalue, V(vin), (binvalue\*lsb)+voffset);

end

generate i ( `NUM\_ADC\_BITS-1, 0) begin

V(data[i]) <+ transition ( vd[i] , tconv, trise, tfall );

end

end

endmodule

`undef NUM\_ADC\_BITS