Chapter 3

3.0 A Modeling Template

3.1 Introduction

3.1.1 Overview

This chapter was established to provide a template approach to the development of a DID compliant VHDL model. It is hoped that this chapter will give the design engineer a "quick start" entry into model development. In this chapter, the design engineer will be walked through an example design to show the modeling process, changing only those elements necessary to support the model.

3.1.2 Scope

This chapter will not delve into the specifics of the code being used. Rather, it will just provide the design engineer with the information necessary to implement the modeling process. If additional information is desired on the modeling process, the engineer is encouraged to read the remaining chapters of this document.

3.2 A Design Example

3.2.1 The CD4029B Counter

The example selected for this chapter is the CD4029B counter. This is a rather versatile presettable 4-bit counter featuring binary/decade and up/down capabilities. Table 3.2.1-1 contains the static electrical characteristics for this device while Table 3.2.1-2 contains the dynamic electrical characteristics which the model will capture. This data was pulled directly from the data sheet for the CD4029B counter.

	Conditions		*Limits at Indicated Tempeatures*					
Charaxteristic	*Vo*	*Vdd*	*-55*	*25min*	*25typ*	*25max*	*125*	*Units*
idd	*-*	*5*	*5*	*-*	*1*	*5*	*150*	*uA*
	-	*10*	*10*	*-*	*1*	*10*	*300*	*uA*
iol	*0.4*	*5*	*0.64*	*0.51*	*1.00*	*-*	*0.36*	*mA*
	0.5	*10*	*1.30*	*1.60*	*2.60*	*-*	*0.90*	*mA*
ioh	*4.6*	*5*	*-0.64*	*-0.51*	*-1.00*	*-*	*-0.36*	*mA*
	9.5	*10*	*-1.30*	*-1.60*	*-2.60*	*-*	*-0.90*	*mA*
vil	*-*	*5*	*1.5*	*-*	*-*	*1.5*	*1.5*	*V*
	-	*10*	*3.0*	*-*	*-*	*3.0*	*3.0*	*V*
vih	*-*	*5*	*3.5*	*-*	*-*	*3.5*	*3.5*	*V*
	-	*10*	*7.0*	*-*	*-*	*7.0*	*7.0*	*V*
iin	*-*	*-*	*1*	*-*	*1*	*1*	*1*	*V*
ci	*-*	*-*	*-*	*-*	*5*	*-*	*-*	*pF*

Notes

Only 5 and 10 volt characteristics are charted. 15 and 20 volt characteristics could be included in a similar manner.

All currents less than 1 ua have been rounded up to 1 ua. This is based upon the limitation of the current physical type defined in the EIA standard package.

The unloaded output voltage specifications are not captured in the table below. In this design, we will concern ourselves with loaded conditions.

Table 3.2.1-1. Static Electrical Characteristics.

3.2.2 Fixed Files

The files of Table 3.2.2-1 are used without modification in this model. For additional information on these files and their purpose, refer to the indicated chapter.

Delay Constraints

characteristic	*vdd*	*typical value*	*units*
propagation delays (tlh,thl)			
clock to q	*5*	*165*	*ns*
	10	*75*	*ns*
clock to co	*5*	*280*	*ns*
	10	*130*	*ns*
transititon times (ttlh,tthl)			
all outputs	*5*	*100*	*ns*
	10	*50*	*ns*

Asynchronous Constraints

characteristic	*vdd*	*typical value*	*units*
pulse width (tw)			
min clk width	*5*	*60*	*ns*
	10	*30*	*ns*
min pe width	*5*	*50*	*ns*
	10	*35*	*ns*
clock frequency (fcl)			
max clk freq	*5*	*4.6*	*MHz*
	10	*10*	*MHz*

Synchronous Constraints

characteristic	*vdd*	*typical value*	*units*
setup time (ts)			
min setup ci	*5*	*30*	*ns*
	10	*10*	*ns*
min setup bd	*5*	*150*	*ns*
or ud	*10*	*70*	*ns*
removal time (trem)			
min pe removal	*5*	*120*	*ns*
	10	*55*	*ns*

Notes

Only 5 and 10 volt characteristics are charted. 15 and 20 volt characteristics could be included in a similar manner.

The propagation delays specified for this model are based upon the clock input only. To include the propagation delays from the preset enable and carry in signals moves the model beyond the scope of this example.

Table 3.2.1-2. Dynamic Electrical Characteristics.

3.3 Files with Context Clause Modifications

The files identified in this section require only minor changes in the way of name changes and library reference changes. Each of the changes will be identified in detail, however the source code will not be replicated here. Table 3.3-1 contains a listing of the files which fall into this category. For additional information, refer to the indicated package in the referenced chapter.

3.3.1 The EIA Design Specification Package

File	Design Unit	Chapter
eia_567p.vhd	*EIA_567*	*5*
i_drive.vhd	*i_driver*	*5*
o_drive.vhd	*o_driver*	*5*
b_drive.vhd	*b_driver*	*5*
async_c.vhd	*async_checker*	*5*
sync_c.vhd	*sync_checker*	*5*
wav_logp.wav	*wav_logic*	*7*
wav_int.wav	*WAVES_INTERFACE*	*6*

Table 3.2.2-1. Design Files Requiring No Modification.

File	New Design Reference	Old Design Reference	Chapter
4029eiap.vhd	*eia_cd4029_ds*	*eia_dqd_ds*	*5*
4029obj.wav	*cd4029_wav*	*dqd_wav*	*6*

Table 3.3-1. Files Requiring Context Clause Changes.

The EIA Design Specification Package (4029eiap.vhd for this model) was modified as follows. First, the filename was changed to identify this file as being specific to the CD4029 model being developed. Secondly, the package name was changed from "eia_dqd_ds" to "eia_cd4029_ds" (4 places). Finally, the USE clause referencing the "dqd_ds" package was changed to reference the "cd4029_ds" package.

3.3.2 The WAVES Objects File

The WAVES objects file requires visibility to the TEST_PINS declaration which is made in the <design>_DUTP package. For this example, the clause "USE work.cd4029_dut.ALL;" must appear in this package. This is the only change required to this package.

3.4 Design Files

3.4.1 The Core Component

Before a complete VHDL model can be developed, it is first necessary to generate a behavioral model of the design. For the CD4029 example, the CD4029_core model is provided in Figure 3.4.1-1.

3.4.2 The Design Specification (DS) File

The purpose of the DS file is to build the electronic data sheet for the model. In the paragraphs which follow, the design engineer will be instructed in how to take information from the data sheet and place it in the DS package. The DS package for this example is provided as Figure 3.4.2-1.

3.4.2.1 The first design related element to be entered is the range of the voltage and current vectors. For the example design, there is a single supply, so these vectors are established for one element (element 0).

```
TYPE VOLTAGE_VECTOR IS ARRAY (0 DOWNTO 0) OF VOLTAGE;
TYPE CURRENT_VECTOR IS ARRAY (0 DOWNTO 0) OF CURRENT;
```

3.4.2.2 The next element to be altered is the PIN_INDEX declaration. For this model, there will be an index for each pin on the device including power and ground. In order to obtain the PIN_INDEX name, simply attach the "ndx" suffix to each pin name. Following is the declaration for this data element.

```
TYPE PIN_INDEX IS (j1ndx,j2ndx,j3ndx,j4ndx,pendx,cindx,bdndx,udndx,
```

```
--***************************************************************
--  (c) Copyright 1994 by the
--      Naval Air Warfare Center, Aircraft Division, Indianapolis
--  Source
--      Author(s):          Charles K. Rogers
--      Organization:         NAWC-ADI
--                          Code 306, MS-42
--                          6000 E 21st St
--                          Indianapolis,  IN  46219-2189
--                          Phone:   317-353-3579
--                          EMail:   ROGERSC1@po2.nawc-ad-indy.navy.mil
--      Reference:          VHDL  MODELING  GUIDE
--      Project:            SHARP  TIREP
--  DESC Certification
--      Status:             TBD
--***************************************************************
--  Revision Histor
--      Version:            1.0
--          Date:           25 May 1994
--          Comments:       Original Release
--***************************************************************
--  Module Description
--      File:               4029cor.vhd
--      Module  Name(s):    cd4029_coreentity/arch_cd4029_core_beh
--          architecture
--      Constraints:        none
--      Limitations:        none
--      I/O  Format(s):     std_logic
--      Purpose and Use:    This file contains a behavioral model of the
--          CD4029 counter.
--      Notes:              none
--***************************************************************
--  Standard Libraries/Packages
--      std_logic_1164      standard multi-value logic package
--  Associated Packages (order of analysis implied)
--      none
--  Component  Models
--      none
--  Platform:               486/33MHz PC
--  Software/Version:       V-System for Windows, Version 3.0
--***************************************************************
--  Design Specification Elements
--      entire file
--***************************************************************
LIBRARY ieee;
USE ieee.STD_LOGIC_1164.ALL;
ENTITY cd4029_core IS
    PORT(bd,pe,ci,ud,clk,j1,j2,j3,j4: IN STD_LOGIC;
        co,q1,q2,q3,q4: OUT STD_LOGIC);
```

Figure 3.4.1-1. The CD4029B Counter Core Model.

```
        END cd4029_core;
ARCHITECTURE arch_cd4029_core_beh OF cd4029_core IS
--***************************************************************
-- Functions for conversions between integers and std_logic_vectors.
--***************************************************************
    FUNCTION to_integer(INPUT:STD_LOGIC_VECTOR) RETURN INTEGER IS
        VARIABLE result:INTEGER;
        VARIABLE weight:INTEGER;
        BEGIN
        weight:=1;result:=0;
        FOR i IN INPUT'LOW TO INPUT'HIGH LOOP
            IF INPUT(i)='1'  THEN
                result:=result+weight;
                ELSIF INPUT(i)/='0'  THEN RETURN -1;
                END IF;
            weight := weight * 2;
            END LOOP;
        RETURN result;
        END to_integer;
    FUNCTION to_vector(INPUT,num_bits:INTEGER) RETURN STD_LOGIC_VECTOR IS
        VARIABLE result:STD_LOGIC_VECTOR(num_bits-1 DOWNTO 0);
        VARIABLE weight:INTEGER;
        VARIABLE temp:INTEGER;
        BEGIN
        IF INPUT<0  THEN
            FOR i IN 1 TO num_bits LOOP
                result(i):='X';
                END LOOP;
            RETURN result;
            END IF;
        weight := 2**(num_bits-1);
        temp := INPUT;
        FOR i IN result'HIGH DOWNTO result'LOW LOOP
            IF temp >= weight THEN
                result(i) := '1';
                temp := temp - weight;
                ELSE
                result(i) := '0';
                END IF;
            weight := weight/2;
            END LOOP;
        RETURN result;
        END to_vector;
    SIGNAL buf:INTEGER RANGE 0 TO 15;
    SIGNAL iq1:STD_LOGIC;
    SIGNAL din,dout:STD_LOGIC_VECTOR(4 DOWNTO 1);
    BEGIN
--***************************************************************
-- Count routine
```

Figure 3.4.1-1 continued. The CD4029B Counter Model.

```
--*************************************************************
    PROCESS(pe,clk,din)
        BEGIN
        IF pe='1' THEN    buf<=to_integer(din);
            ELSIF clk'EVENT AND clk='1' THEN
            IF ci='0' THEN
                IF ud='1' THEN
                    IF bd='1' AND buf=15 THEN   buf<=0;
                        ELSIF bd='0' AND buf=9 THEN   buf<=0;
                        ELSE    buf<=buf+1;
                        END IF;
                    ELSE
                    IF bd='1' AND buf=0 THEN   buf<=15;
                        ELSIF bd='0' AND buf=0 THEN   buf<=9;
                        ELSE   buf<=buf-1;
                        END IF;
                    END IF;
                ELSE NULL;
                END IF;
            ELSE NULL;
            END IF;
        END PROCESS;
--*************************************************************
-- Carry out generation
--*************************************************************
    PROCESS(iq1,ud,bd,ci,pe,buf)
        BEGIN
        IF ci='0' THEN
            IF ud='1' AND buf=15 THEN   co<='0';
                ELSIF ud='1' AND bd='0' AND buf>=9 THEN   co<='0';
                ELSIF ud='0' AND buf=0 THEN   co<='0';
                ELSE   co<='1';
                END IF;
            ELSE   co<='1';
            END IF;
        END PROCESS;
--*************************************************************
-- Internal bus assignments to external signals
--*************************************************************
    din(1)<=j1;din(2)<=j2;din(3)<=j3;din(4)<=j4;
    q1<=dout(1);q2<=dout(2);q3<=dout(3);q4<=dout(4);
    dout<=to_vector(buf,4);iq1<=dout(1);
    END  arch_cd4029_core_beh;
```

Figure 3.4.1-1 continued. The CD4029B Counter Model.

```
    clkndx,q1ndx,q2ndx,q3ndx,q4ndx,condx,vddndx,vssndx);
```

3.4.2.3 Before leaving the package declaration, we will enter appropriate information in the following constants.

```
CONSTANT PART_NAME STRING:="cd4029b";
CONSTANT ESD_PROTECTION ESD_CLASS:= UNKNOWN;
```

```
--**************************************************************
--   (c) Copyright 1994 by the
--        Naval Air Warfare Center, Aircraft Division, Indianapolis
--   Source
--        Author(s):          Charles K. Rogers
--        Organization:         NAWC-ADI
--                            Code 306, MS-42
--                            6000 E 21st St
--                            Indianapolis,  IN  46219-2189
--                            Phone:   317-353-3579
--                            EMail:   ROGERSC1@po2.nawc-ad-indy.navy.mil
--        Reference:
--        Project:            SHARP   TIREP
--   DESC Certification
--        Status:             TBD
--**************************************************************
--   Revision History
--        Version:            1.1
--             Date:          23 September 1994
--             Comments:      Added cycle time checks to package
--        Version:            1.0
--             Date:          25 May 1994
--             Comments:      Original Release
--**************************************************************
--   Module Description
--        File:               4029ds_p.vhd
--        Module  Name(s):    cd4029_ds package
--        Constraints:        none
--        Limitations:        none
--        I/O  Format(s):     none
--        Purpose and Use:    This VHDL package defines the electrical pin
--             specifications, operating point limits, timing specifications
--             and the electronic data sheet for the CD4029 counter.
--        Notes:              none
--**************************************************************
--   Standard Libraries/Packages
--        none
--   Associated Packages (order of analysis implied)
--        eia_567             standard eia package
--   Component  Models
--        none
--   Platform:                486/33MHz PC
--   Software/Version:        V-System for Windows, Version 3.0
--**************************************************************
--   Design Specification Elements
--        entire file
--**************************************************************
LIBRARY eia;
USE eia.EIA_567.ALL;
```

Figure 3.4.2-1. The CD4029 Design Specification File.

```
PACKAGE cd4029_ds IS
--*****************************************************************
-- Declaration of number of voltage supplies included in the design
--*****************************************************************
   TYPE VOLTAGE_VECTOR IS ARRAY(0 DOWNTO 0) OF VOLTAGE;
   TYPE CURRENT_VECTOR IS ARRAY(0 DOWNTO 0) OF CURRENT;
---------------------------------------------------------------------
-- Declaration of constants which will be detailed in the package body
---------------------------------------------------------------------
   CONSTANT ELECTRICAL_PIN_SPEC: EV_SIGNAL_LIMITS;
   CONSTANT VMAX:VOLTAGE_VECTOR;
   CONSTANT VMIN:VOLTAGE_VECTOR;
   CONSTANT VNOM:VOLTAGE_VECTOR;
   CONSTANT IMAX:CURRENT_VECTOR;
   CONSTANT LOWER_TEMPERATURE_LIMIT:TEMPERATURE;
   CONSTANT UPPER_TEMPERATURE_LIMIT:TEMPERATURE;
---------------------------------------------------------------------
-- Declaration of input constraints which are appropriate for this model
---------------------------------------------------------------------
   CONSTANT ASYNCS:ASYNC_CONSTRAINTS;
   CONSTANT SYNCS:SYNC_CONSTRAINTS;
   CONSTANT DELAYS:OUTPUT_DELAYS;
--*****************************************************************
-- The following constants define the maximum number of constraints
-- which will be associated to any given pin.  An example is a data
-- which is sampled by three different clocks (triple sampling).  The
-- data will have three setups and three holds.      So if this were the
-- largest number of constraints associated with a pin the
-- max_async_constraints_per_pin should be assigned 3.        These  constants
-- are used in the eia_567_ds (design specification) package and must be
-- defined beforehand.
--*****************************************************************
   CONSTANT MAX_ASYNC_CONSTRAINTS_PER_PIN:NATURAL:=1;
   CONSTANT MAX_SYNC_CONSTRAINTS_PER_PIN:NATURAL:=1;
   CONSTANT MAX_DELAY_CONSTRAINTS_PER_PIN:NATURAL:=1;
--*****************************************************************
-- Definition of an enumerated type which is used as an index into most
-- of the data stored in the electronic data sheet, the enumeration uses
-- signal/supply/nc names.
--*****************************************************************
   TYPE PIN_INDEX IS  (j1ndx,j2ndx,j3ndx,j4ndx,pendx,cindx,bdndx,udndx,
      clkndx,q1ndx,q2ndx,q3ndx,q4ndx,condx,vddndx,vssndx);
---------------------------------------------------------------------
-- Definition of a type and the signal to pin corrolation
---------------------------------------------------------------------
   TYPE PIN_AND_SIGNAL_CORROLATIONS IS ARRAY(PIN_INDEX) OF PIN_RECORD;
---------------------------------------------------------------------
-- timing related declarations
--    declaration of an operating point type based on the number of
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
--   voltage supplies defined by the user
---------------------------------------------------------------------
     TYPE POINT IS RECORD
          selection: OPERATING_SELECTION;
          temp: TEMPERATURE;
          SUPPLY: VOLTAGE_VECTOR;
          END RECORD;
---------------------------------------------------------------------
--   single operating point definition
---------------------------------------------------------------------
     TYPE ASYNC_ARRAY IS ARRAY(MAX_ASYNC_CONSTRAINTS_PER_PIN DOWNTO 1)
          OF INTEGER;
     TYPE SYNC_ARRAY IS ARRAY(MAX_SYNC_CONSTRAINTS_PER_PIN DOWNTO 1)
          OF INTEGER;
     TYPE DELAY_ARRAY IS ARRAY(MAX_DELAY_CONSTRAINTS_PER_PIN DOWNTO 1)
          OF INTEGER;
     TYPE PIN_POINTERS IS RECORD
          async_spec: ASYNC_ARRAY;
          sync_spec: SYNC_ARRAY;
          delay_spec: DELAY_ARRAY;
          elec_spec: NATURAL;
          END RECORD;
---------------------------------------------------------------------
-- Definition of an array of records.  Each record holds the specific
-- indexes for the signal/pin into the stored information in the views
-- of the electronic data sheet.   A single array element of
-- point_specification contains all the information needed to describe
-- the timing characteristics at an operating point.
---------------------------------------------------------------------
     TYPE POINT_SPECIFICATION IS ARRAY(PIN_INDEX) OF PIN_POINTERS;
     TYPE OPNT IS RECORD
          selpoint: POINT;
          edsnfo: POINT_SPECIFICATION;
          END RECORD;
--***************************************************************
-- The array range for the ELECTRONIC_DATA_SHEET type may require
-- updating.
--***************************************************************
     TYPE ELECTRONIC_DATA_SHEET IS ARRAY(0 TO 6) OF OPNT;
-- The set of all operating points is an electronic data sheet.
--***************************************************************
-- Definition of name, esd protection, outline drawing, and power for
-- this particular model as well as a few other constant values that
-- need to be visible.
--***************************************************************
     CONSTANT PART_NAME: STRING:="cd4029b";
     CONSTANT ESD_PROTECTION: ESD_CLASS:= UNKNOWN;
     CONSTANT OUTLINE_DRAWING: STRING:="gdip-t16 (mil-m-1835)";
     CONSTANT MAXIMUM_POWER_DISSIPATION: POWER:=500 mw;
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
    CONSTANT EDS : ELECTRONIC_DATA_SHEET;
    END cd4029_ds;

PACKAGE BODY cd4029_ds IS
--*****************************************************************
-- For the design, identify the number of pin classes required. A
-- pin class is defined as an input or an output whose interface
-- characteristics are different from other inputs and outputs. As
-- a minimum, there should be three pin classes: default (class 0),
-- inputs and outputs. Pin classes for nominal (or typical), minimum
-- and/or maximum characteristics should be included when possible.
--*****************************************************************
-- Load circuits
--    The standard load circuit for this model consists of two resistors
--    (r1 and r2), a capacitor (cl), a diode (cr1) and a power supply (p01).
--    The power supply is connected to one side of r2. The remaining side
--    of r2 is connected to the annode of cr1. The cathode of cr1 is
--    connected to the output of the CD4029. Also from the CD4029 output,
--    the resistor r1 and capacitor cl are connected in parallel, to ground.
--    Following are the values which apply to this load circuit for each
--    class    specified. For class 0 and all input classes, load components
--    are not used (N/U).
------------------------------------------------------------------------
-- pin class   load class   p01 (1%)   cr1      r1 (1%)      r2 (1%)   cl
-- 3           1            5.0V       1N916    4.60kohm     4.00kohm   N/U
-- 4           2            10.0V      1N916    3.65kohm     3.42kohm   N/U
-- 5           3            5.0V       1N916    7.19kohm     6.25kohm   N/U
-- 6           4            10.0V      1N916    5.94kohm     5.56kohm   N/U
-- 7           5            5.0V       1N916    12.78kohm    11.11kohm  N/U
-- 8           6            10.0V      1N916    10.56kohm    9.89kohm   N/U
------------------------------------------------------------------------
    CONSTANT ELECTRICAL_PIN_SPEC : EV_SIGNAL_LIMITS := (
-- pin class #0 - no load
        (voh=>0.0  v,  ioh=>0.0  ua,  vol=>0.0  v,  iol=>0.0  ua,test_load=>0,
        vih=>0.0  v,  iih=>0.0  ua,  vil=>0.0  v,  iil=>0.0  ua,  pin_load=>  0  pf),
-- pin class #1 - all inputs operating at 5V
        (voh=>0.0  v,  ioh=>0.0  ma,  vol=>0.0  v,  iol=>0.0  ma,test_load=>0,
        vih=>3.5  v,  iih=>1.0  ua,  vil=>1.5  v,  iil=>-1.0  ua,  pin_load=>  5  pf),
-- pin class #2 - all inputs operating at 10V
        (voh=>0.0  v,  ioh=>0.0  ma,  vol=>0.0  v,  iol=>0.0  ma,test_load=>0,
        vih=>7.0  v,  iih=>1.0  ua,  vil=>3.0  v,  iil=>-1.0  ua,  pin_load=>  5  pf),
-- pin class #3 - all outputs at 5V, room temperature, typical values
        (voh=>4.6  v,  ioh=>-1.00  ma,  vol=>0.4  v,  iol=>1.00  ma,test_load=>1,
        vih=>0.0  v,  iih=>0.0  ua,  vil=>0.0  v,  iil=>0.0  ua,  pin_load=>  0  pf),
-- pin class #4 - all outputs at 10V, room temperature, typical values
        (voh=>9.5  v,  ioh=>-2.6  ma,  vol=>0.5  v,  iol=>2.6  ma,test_load=>2,
        vih=>0.0  v,  iih=>0.0  ua,  vil=>0.0  v,  iil=>0.0  ua,  pin_load=>  0  pf),
-- pin class #5 - all outputs operating at 5V and cold temperature
        (voh=>4.6  v,  ioh=>-0.64  ma,  vol=>0.4  v,  iol=>0.64  ma,test_load=>3,
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
        vih=>0.0 v, iih=>0.0 ua, vil=>0.0 v, iil=>0.0 ua, pin_load=> 0 pf),
-- pin class #6 - all outputs operating at 10V and cold temperature
        (voh=>9.5 v, ioh=>-1.6 ma, vol=>0.5 v, iol=>1.6 ma,test_load=>4,
        vih=>0.0 v, iih=>0.0 ua, vil=>0.0 v, iil=>0.0 ua, pin_load=> 0 pf),
-- pin class #7 - all outputs operating at 5V and hot temperature
        (voh=>4.6 v, ioh=>-0.36 ma, vol=>0.4 v, iol=>0.36 ma,test_load=>5,
        vih=>0.0 v, iih=>0.0 ua, vil=>0.0 v, iil=>0.0 ua, pin_load=> 0 pf),
-- pin class #8 - all outputs operating at 10V and hot temperature
        (voh=>9.5 v, ioh=>-0.9 ma, vol=>0.5 v, iol=>0.9 ma,test_load=>6,
        vih=>0.0 v, iih=>0.0 ua, vil=>0.0 v, iil=>0.0 ua, pin_load=> 0 pf));
--***********************************************************
-- Power limits
--***********************************************************
    CONSTANT VMAX: VOLTAGE_VECTOR := (0=>(18.0 v));
    CONSTANT VMIN: VOLTAGE_VECTOR := (0=>(3.0 v));
    CONSTANT VNOM: VOLTAGE_VECTOR := (0=>(5.0 v));
    CONSTANT IMAX: CURRENT_VECTOR := (0=>(300 ma));
--***********************************************************
-- Temperature limits
--***********************************************************
    CONSTANT LOWER_TEMPERATURE_LIMIT: TEMPERATURE := -55 degrees_c;
    CONSTANT UPPER_TEMPERATURE_LIMIT: TEMPERATURE := 125 degrees_c;
--***********************************************************
-- Definition of asynchronous constraint classes
--***********************************************************
    CONSTANT ASYNCS: ASYNC_CONSTRAINTS := (
-- Async constraint class #0 for no constraints
        (t1min=>0 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns),
-- Async constraint class #1 for clk at 5 volts, typical
        (t1min=>60 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>217 ns,cycmax=>0 ns),
-- Async constraint class #2 for clk at 10 volts, typical
        (t1min=>30 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>100 ns,cycmax=>0 ns),
-- Async constraint class #3 for pe at 5 volts, typical
        (t1min=>50 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns),
-- Async constraint class #4 for pe at 10 volts, typical
        (t1min=>35 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns),
-- Async constraint class #5 for clk at 5 volts, minimum
        (t1min=>21 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>76 ns,cycmax=>0 ns),
-- Async constraint class #6 for clk at 10 volts, minimum
        (t1min=>11 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>36 ns,cycmax=>0 ns),
-- Async constraint class #7 for pe at 5 volts, minimum
        (t1min=>18 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
            cycmin=>0    ns,cycmax=>0    ns),
 -- Async constraint class #8 for pe at 10 volts, minimum
            (t1min=>12    ns,t0min=>0    ns,t1max=>0    ns,t0max=>0    ns,
            cycmin=>0    ns,cycmax=>0    ns),
 -- Async constraint class #9 for clk at 5 volts, maximum
            (t1min=>120    ns,t0min=>0    ns,t1max=>0    ns,t0max=>0    ns,
            cycmin=>434    ns,cycmax=>0    ns),
 -- Async constraint class #10 for clk at 10 volts, maximum
            (t1min=>60    ns,t0min=>0    ns,t1max=>0    ns,t0max=>0    ns,
            cycmin=>200    ns,cycmax=>0    ns),
 -- Async constraint class #11 for pe at 5 volts, maximum
            (t1min=>100    ns,t0min=>0    ns,t1max=>0    ns,t0max=>0    ns,
            cycmin=>0    ns,cycmax=>0    ns),
 -- Async constraint class #12 for pe at 10 volts, maximum
            (t1min=>70    ns,t0min=>0    ns,t1max=>0    ns,t0max=>0    ns,
            cycmin=>0    ns,cycmax=>0    ns));
--*************************************************************
 -- Definition of synchronous constraint classes
--*************************************************************
      CONSTANT SYNCS:SYNC_CONSTRAINTS=(
 -- Sync constraint class #0 for no constraints
            (setup=>0      ns,hold=>0      ns,edge=>SIGNAL_EDGELEFT),
 -- Sync constraint class #1 for ci at 5 volts, typical
            (setup=>30      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #2 for ci at 10 volts, typical
            (setup=>10      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #3 for ud or bd at 5 volts, typical
            (setup=>150      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #4 for ud or bd at 10 volts, typical
            (setup=>70      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #5 for pe at 5 volts, typical
            (setup=>120      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #6 for pe at 10 volts, typical
            (setup=>55      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #7 for ci at 5 volts, minimum
            (setup=>13      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #8 for ci at 10 volts, minimum
            (setup=>4      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #9 for ud or bd at 5 volts, minimum
            (setup=>53      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #10 for ud or bd at 10 volts, minimum
            (setup=>25      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #11 for pe at 5 volts, minimum
            (setup=>42      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #12 for pe at 10 volts, minimum
            (setup=>19      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #13 for ci at 5 volts, maximum
            (setup=>60      ns,hold=>0      ns,edge=>e01),
 -- Sync constraint class #14 for ci at 10 volts, maximum
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
        (setup=>20    ns,hold=>0    ns,edge=>e01),
 -- Sync constraint class #15 for ud or bd at 5 volts, maximum
        (setup=>300    ns,hold=>0    ns,edge=>e01),
 -- Sync constraint class #16 for ud or bd at 10 volts, maximum
        (setup=>140    ns,hold=>0    ns,edge=>e01),
 -- Sync constraint class #17 for pe at 5 volts, maximum
        (setup=>240    ns,hold=>0    ns,edge=>e01),
 -- Sync constraint class #18 for pe at 10 volts, maximum
        (setup=>110    ns,hold=>0    ns,edge=>e01));
--***********************************************************
 -- Definition of output delay classes
--***********************************************************
    CONSTANT DELAYS : OUTPUT_DELAYS := (
 -- Ouput delay class #0, no delay association
        (tlh=>0   ns,thl=>0   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>0   ns,tthl=>0   ns),
 -- Output delay class #1 clk to q at 5 volts, typical
        (tlh=>165   ns,thl=>165   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>100   ns,tthl=>100   ns),
 -- Output delay class #2 clk to q at 10 volts, typical
        (tlh=>75   ns,thl=>75   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>50   ns,tthl=>50   ns),
 -- Output delay class #3 clk to co at 5 volts, typical
        (tlh=>280   ns,thl=>280   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>100   ns,tthl=>100   ns),
 -- Output delay class #4 clk to co at 10 volts, typical
        (tlh=>130   ns,thl=>130   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>50   ns,tthl=>50   ns),
 -- Output delay class #5 clk to q at 5 volts, minimum
        (tlh=>48   ns,thl=>48   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>35   ns,tthl=>35   ns),
 -- Output delay class #6 clk to q at 10 volts, minimum
        (tlh=>27   ns,thl=>27   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>18   ns,tthl=>18   ns),
 -- Output delay class #7 clk to co at 5 volts, minimum
        (tlh=>98   ns,thl=>98   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>35   ns,tthl=>35   ns),
 -- Output delay class #8 clk to co at 10 volts, minimum
        (tlh=>45   ns,thl=>45   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>18   ns,tthl=>18   ns),
 -- Output delay class #9 clk to q at 5 volts, maximum
        (tlh=>330   ns,thl=>330   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>200   ns,tthl=>200   ns),
 -- Output delay class #10 clk to q at 10 volts, maximum
        (tlh=>150   ns,thl=>150   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>100   ns,tthl=>100   ns),
 -- Output delay class #11 clk to co at 5 volts, maximum
        (tlh=>560   ns,thl=>560   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
        ttlh=>200   ns,tthl=>200   ns),
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
-- Output delay class #12 clk to co at 10 volts, maximum
        (tlh=>260 ns,thl=>260 ns,tlz=>0 ns,thz=>0 ns,tzl=>0 ns,tzh=>0 ns,
        ttlh=>100 ns,tthl=>100 ns));
--************************************************************
-- Definition of an array of the indexes.  Each array holds the specific
-- index for all of the singal/pins for a specific operating point.
--************************************************************
    CONSTANT EDS:ELECTRONIC_DATA_SHEET:=(
-- The nominal or typical operating point at 5 volts
        0=>(
            selpoint=>(selection=> TNOM,temp=>27 degrees_c,SUPPLY=>(0=>5.0 v)),
        edsnfo=>(
            j1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            j2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            j3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            j4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            pendx=>(sync_spec=>(1=>(5)),async_spec=>(1=>(3)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            cindx=>(sync_spec=>(1=>(1)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            udndx=>(sync_spec=>(1=>(3)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            bdndx=>(sync_spec=>(1=>(3)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            clkndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(1)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            q1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(1)),elec_spec=>3),
            q2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(1)),elec_spec=>3),
            q3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(1)),elec_spec=>3),
            q4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(1)),elec_spec=>3),
            condx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(3)),elec_spec=>3),
            vddndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0),
            vssndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0))),
-- The nominal or typical operating point at 10 volts
        1=>(
            selpoint=>(selection=> TNOM,temp=>27 degrees_c,
                SUPPLY=>(0=>10.0 v)),
        edsnfo=>(
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
            j1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            j2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            j3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            j4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            pendx=>(sync_spec=>(1=>(6)),async_spec=>(1=>(4)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            cindx=>(sync_spec=>(1=>(2)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            udndx=>(sync_spec=>(1=>(4)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            bdndx=>(sync_spec=>(1=>(4)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            clkndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(2)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            q1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(2)),elec_spec=>4),
            q2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(2)),elec_spec=>4),
            q3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(2)),elec_spec=>4),
            q4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(2)),elec_spec=>4),
            condx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(4)),elec_spec=>4),
            vddndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0),
            vssndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0))),
-- The best case operating point at 5 volts
        2=>(
            selpoint=>(selection=>  TMAX,temp=>-55   degrees_c,
                SUPPLY=>(0=>5.0   v)),
        edsnfo=>(
            j1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            j2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            j3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            j4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            pendx=>(sync_spec=>(1=>(11)),async_spec=>(1=>(7)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            cindx=>(sync_spec=>(1=>(7)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
            udndx=>(sync_spec=>(1=>(9)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            bdndx=>(sync_spec=>(1=>(9)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            clkndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(5)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            q1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(5)),elec_spec=>5),
            q2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(5)),elec_spec=>5),
            q3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(5)),elec_spec=>5),
            q4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(5)),elec_spec=>5),
            condx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(7)),elec_spec=>5),
            vddndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0),
            vssndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0))),
-- The best case operating point at 10 volts
        3=>(
            selpoint=>(selection=>    TMAX,temp=>-55    degrees_c,
                SUPPLY=>(0=>10.0    v)),
        edsnfo=>(
            j1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            j2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            j3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            j4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            pendx=>(sync_spec=>(1=>(11)),async_spec=>(1=>(8)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            cindx=>(sync_spec=>(1=>(8)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            udndx=>(sync_spec=>(1=>(10)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            bdndx=>(sync_spec=>(1=>(10)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            clkndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(6)),
                delay_spec=>(1=>(0)),elec_spec=>2),
            q1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(6)),elec_spec=>6),
            q2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(6)),elec_spec=>6),
            q3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(6)),elec_spec=>6),
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
            q4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(6)),elec_spec=>6),
            condx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(8)),elec_spec=>6),
            vddndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0),
            vssndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0))),
-- The worst case operating point at 5 volts
        4=>(
            selpoint=>(selection=>   TMAX,temp=>125   degrees_c,
                SUPPLY=>(0=>5.0    v)),
        edsnfo=>(
            j1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            j2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            j3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            j4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            pendx=>(sync_spec=>(1=>(17)),async_spec=>(1=>(11)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            cindx=>(sync_spec=>(1=>(13)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            udndx=>(sync_spec=>(1=>(15)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            bdndx=>(sync_spec=>(1=>(15)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            clkndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(9)),
                delay_spec=>(1=>(0)),elec_spec=>1),
            q1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(9)),elec_spec=>7),
            q2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(9)),elec_spec=>7),
            q3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(9)),elec_spec=>7),
            q4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(9)),elec_spec=>7),
            condx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(11)),elec_spec=>7),
            vddndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0),
            vssndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                delay_spec=>(1=>(0)),elec_spec=>0))),
-- The worst case operating point at 10 volts
        5=>(
            selpoint=>(selection=>   TMAX,temp=>125   degrees_c,
                SUPPLY=>(0=>10.0    v)),
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
            edsnfo=>(
                j1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>2),
                j2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>2),
                j3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>2),
                j4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>2),
                pendx=>(sync_spec=>(1=>(18)),async_spec=>(1=>(12)),
                    delay_spec=>(1=>(0)),elec_spec=>2),
                cindx=>(sync_spec=>(1=>(14)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>2),
                udndx=>(sync_spec=>(1=>(16)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>2),
                bdndx=>(sync_spec=>(1=>(16)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>2),
                clkndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(10)),
                    delay_spec=>(1=>(0)),elec_spec=>2),
                q1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(10)),elec_spec=>8),
                q2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(10)),elec_spec=>8),
                q3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(10)),elec_spec=>8),
                q4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(10)),elec_spec=>8),
                condx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(12)),elec_spec=>8),
                vddndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>0),
                vssndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>0))),
    -- The zero operating point
            6=>(
                selpoint=>(selection=>tzero,temp=>27        degrees_c,
                    SUPPLY=>(0=>5.0     v)),
            edsnfo=>(
                j1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>0),
                j2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>0),
                j3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>0),
                j4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>0),
                pendx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                    delay_spec=>(1=>(0)),elec_spec=>0),
                cindx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
                  delay_spec=>(1=>(0)),elec_spec=>0),
              udndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0),
              bdndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0),
              clkndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0),
              q1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0),
              q2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0),
              q3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0),
              q4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0),
              condx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0),
              vddndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0),
              vssndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
                  delay_spec=>(1=>(0)),elec_spec=>0))));
      END cd4029_ds;
```

Figure 3.4.2-1 continued. The CD4029 DS File.

```
   CONSTANT OUTLINE_DRAWING:STRING:="gdip-t16 (mil-m-1835)";
   CONSTANT MAXIMUM_POWER_DISSIPATION:POWER:=500 mw;
```

3.4.2.4 At this point, the ELECTRICAL_PIN_SPEC constant is assigned. This constant is made up of classes which provide the input and output pin characteristics (i.e. voltage, current, load).

```
-----------------------------------------------------------------------
-- pin class   load class   p01 (1%)   cr1     r1 (1%)      r2 (1%)      cl
-- 3           1            5.0V       1N916   4.60kohm     4.00kohm     N/U
-- 4           2            10.0V      1N916   3.65kohm     3.42kohm     N/U
-- 5           3            5.0V       1N916   7.19kohm     6.25kohm     N/U
-- 6           4            10.0V      1N916   5.94kohm     5.56kohm     N/U
-- 7           5            5.0V       1N916   12.78kohm    11.11kohm    N/U
-- 8           6            10.0V      1N916   10.56kohm    9.89kohm     N/U
-----------------------------------------------------------------------
   CONSTANT ELECTRICAL_PIN_SPEC:EV_SIGNAL_LIMITS:=(
-- pin class #0 - no load
      (voh=>0.0 v, ioh=>0.0 ua, vol=>0.0 v, iol=>0.0 ua,test_load=>0,
      vih=>0.0 v, iih=>0.0 ua, vil=>0.0 v, iil=>0.0 ua, pin_load=> 0 pf),
-- pin class #1 - all inputs operating at 5V
      (voh=>0.0 v, ioh=>0.0 ma, vol=>0.0 v, iol=>0.0 ma,test_load=>0,
      vih=>3.5 v, iih=>1.0 ua, vil=>1.5 v, iil=>-1.0 ua, pin_load=> 5 pf),
-- pin class #2 - all inputs operating at 10V
      (voh=>0.0 v, ioh=>0.0 ma, vol=>0.0 v, iol=>0.0 ma,test_load=>0,
      vih=>7.0 v, iih=>1.0 ua, vil=>3.0 v, iil=>-1.0 ua, pin_load=> 5 pf),
-- pin class #3 - all outputs at 5V, room temperature, typical values
```

```
        (voh=>4.6  v,  ioh=>-1.00  ma,  vol=>0.4  v,  iol=>1.00  ma,test_load=>1,
        vih=>0.0  v,  iih=>0.0  ua,  vil=>0.0  v,  iil=>0.0  ua,  pin_load=>  0  pf),
  -- pin class #4 - all outputs at 10V, room temperature, typical values
        (voh=>9.5  v,  ioh=>-2.6  ma,  vol=>0.5  v,  iol=>2.6  ma,test_load=>2,
        vih=>0.0  v,  iih=>0.0  ua,  vil=>0.0  v,  iil=>0.0  ua,  pin_load=>  0  pf),
  -- pin class #5 - all outputs operating at 5V and cold temperature
        (voh=>4.6  v,  ioh=>-0.64  ma,  vol=>0.4  v,  iol=>0.64  ma,test_load=>3,
        vih=>0.0  v,  iih=>0.0  ua,  vil=>0.0  v,  iil=>0.0  ua,  pin_load=>  0  pf),
  -- pin class #6 - all outputs operating at 10V and cold temperature
        (voh=>9.5  v,  ioh=>-1.6  ma,  vol=>0.5  v,  iol=>1.6  ma,test_load=>4,
        vih=>0.0  v,  iih=>0.0  ua,  vil=>0.0  v,  iil=>0.0  ua,  pin_load=>  0  pf),
  -- pin class #7 - all outputs operating at 5V and hot temperature
        (voh=>4.6  v,  ioh=>-0.36  ma,  vol=>0.4  v,  iol=>0.36  ma,test_load=>5,
        vih=>0.0  v,  iih=>0.0  ua,  vil=>0.0  v,  iil=>0.0  ua,  pin_load=>  0  pf),
  -- pin class #8 - all outputs operating at 10V and hot temperature
        (voh=>9.5  v,  ioh=>-0.9  ma,  vol=>0.5  v,  iol=>0.9  ma,test_load=>6,
        vih=>0.0  v,  iih=>0.0  ua,  vil=>0.0  v,  iil=>0.0  ua,  pin_load=>  0  pf));
```

3.4.2.4.1 The values entered for each element are pulled directly from the table above. Only the maximum/minimum conditions are captured for 5 and 10 volt operation. As noted above, only 5 and 10 volt operating points are captured. This process could readily be extended to capture 15 volt operation also.

3.4.2.4.2 A set of test load classes are defined which apply the appropriate loads to the outputs of the circuit. These load classes are determined as a result of the "ioh" and "iol" specifications from the data sheet. For this example, a load circuit is defined as shown in Figure 3.4.2.4.2-1. For each load class, the values of R1 and R2 are calculated. Based upon this load circuit, a resistance of 4.60kohm will apply a load of 1.0ma with a voltage of 4.6 v. The pull-up resistors are calculated in a similar fashion except that the forward voltage drop of CR1 (approximately 0.6 volts) is taken into account.

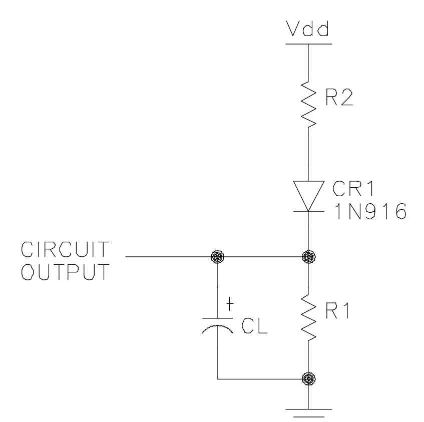

Figure 3.4.2.4.2-1. A Test Load Circuit.

3.4.2.5 At this point, the following constants are declared. Since we are only providing data sheet elements at 5 and 10 volts, the nominal operating voltage (vnom) is selected to be 5 volts. The minimum and maximum limits are set by the technology which allows the supply to range from 3 to 18 volts.

```
CONSTANT VMAX : VOLTAGE_VECTOR := (0=>(18.0  v));
CONSTANT VMIN : VOLTAGE_VECTOR := (0=>(3.0  v));
CONSTANT VNOM : VOLTAGE_VECTOR := (0=>(5.0  v));
CONSTANT IMAX : CURRENT_VECTOR := (0=>(300  ma));
CONSTANT LOWER_TEMPERATURE_LIMIT : TEMPERATURE := -55  degrees_c;
```

```
    CONSTANT UPPER_TEMPERATURE_LIMIT : TEMPERATURE := 125 degrees_c;
```

3.4.2.6 The asynchronous constraints dictate the minimum and maximum pulse widths. These constraints are captured in a fashion similar to the pin classes above. For each signal which has a minimum or maximum pulse width requirement, a asynchronous constraint class is declared.

```
    CONSTANT ASYNCS : ASYNC_CONSTRAINTS := (
  -- Async constraint class #0 for no constraints
        (t1min=>0 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns),
  -- Async constraint class #1 for clk at 5 volts, typical
        (t1min=>60 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>217 ns,cycmax=>0 ns),
  -- Async constraint class #2 for clk at 10 volts, typical
        (t1min=>30 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>100 ns,cycmax=>0 ns),
  -- Async constraint class #3 for pe at 5 volts, typical
        (t1min=>50 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns),
  -- Async constraint class #4 for pe at 10 volts, typical
        (t1min=>35 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns),
  -- Async constraint class #5 for clk at 5 volts, minimum
        (t1min=>21 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>76 ns,cycmax=>0 ns),
  -- Async constraint class #6 for clk at 10 volts, minimum
        (t1min=>11 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>36 ns,cycmax=>0 ns),
  -- Async constraint class #7 for pe at 5 volts, minimum
        (t1min=>18 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns),
  -- Async constraint class #8 for pe at 10 volts, minimum
        (t1min=>12 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns),
  -- Async constraint class #9 for clk at 5 volts, maximum
        (t1min=>120 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>434 ns,cycmax=>0 ns),
  -- Async constraint class #10 for clk at 10 volts, maximum
        (t1min=>60 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>200 ns,cycmax=>0 ns),
  -- Async constraint class #11 for pe at 5 volts, maximum
        (t1min=>100 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns),
  -- Async constraint class #12 for pe at 10 volts, maximum
        (t1min=>70 ns,t0min=>0 ns,t1max=>0 ns,t0max=>0 ns,
        cycmin=>0 ns,cycmax=>0 ns));
```

3.4.2.6.1 It should be noted that this model is not set up to readily check the maximum operating frequency of the device. A check has been approximated by subtracting the minimum pulse width (t1min) from the period of the maximum operating frequency and using the result as the minimum low pulse width (t0min). The minimum low pulse width is not, however specified in the data sheet. This approach imposes a corresponding limitation on this model in that it is possible to simulate this model in a manner which will meet the component requirements, yet generate a timing violation. This occurs when the minimum high pulse width (t1min) requirement is exceeded and the model is operated at (or near) the maximum frequency. Under this condition, the t0min pulse width could be violated, however the device would be operating within it's specifications.

3.4.2.6.2 In order to meet the requirements of the DID, it is necessary to include nominal, minimum and maximum operating points for the model. In the absence of specifications on these parameters, the guidelines of EIA-567A have been used. With the typical or nominal values provided, the minimum timing will be a factor of 0.35 times the nominal, while the maximum timing is obtained by multiplying the nominal by a factor of 2.00. These factors will be applied to the following synchronous, asynchronous and delay constraints in order to obtain the minimum and maximum timing constraints for this design.

3.4.2.7 Synchronous constraints are times established with respect to a signal transition. Setup times are measured prior to a transition while hold times are measured following the transition.

```
      CONSTANT SYNCS:SYNC_CONSTRAINTS=(
  -- Sync constraint class #0 for no constraints
          (setup=>0    ns,hold=>0    ns,edge=>SIGNAL_EDGELEFT),
  -- Sync constraint class #1 for ci at 5 volts, typical
          (setup=>30    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #2 for ci at 10 volts, typical
          (setup=>10    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #3 for ud or bd at 5 volts, typical
          (setup=>150    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #4 for ud or bd at 10 volts, typical
          (setup=>70    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #5 for pe at 5 volts, typical
          (setup=>120    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #6 for pe at 10 volts, typical
          (setup=>55    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #7 for ci at 5 volts, minimum
          (setup=>13    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #8 for ci at 10 volts, minimum
          (setup=>4    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #9 for ud or bd at 5 volts, minimum
          (setup=>53    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #10 for ud or bd at 10 volts, minimum
          (setup=>25    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #11 for pe at 5 volts, minimum
          (setup=>42    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #12 for pe at 10 volts, minimum
          (setup=>19    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #13 for ci at 5 volts, maximum
          (setup=>60    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #14 for ci at 10 volts, maximum
          (setup=>20    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #15 for ud or bd at 5 volts, maximum
          (setup=>300    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #16 for ud or bd at 10 volts, maximum
          (setup=>140    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #17 for pe at 5 volts, maximum
          (setup=>240    ns,hold=>0    ns,edge=>e01),
  -- Sync constraint class #18 for pe at 10 volts, maximum
          (setup=>110    ns,hold=>0    ns,edge=>e01));
```

3.4.2.8 Next, the delay constraints are specified. Because of the manner in which we have limited the propagation delays which will be modelled, this constant declaration is quite straightforward. This implementation does employ transition time modeling for the design.

```
      CONSTANT DELAYS:OUTPUT_DELAYS=(
  -- Output delay class #0, no delay association
          (tlh=>0   ns,thl=>0   ns,tlz=>0   ns,thz=>0   ns,tzl=>0   ns,tzh=>0   ns,
```

```
            ttlh=>0    ns,tthl=>0    ns),
    -- Output delay class #1 clk to q at 5 volts, typical
            (tlh=>165    ns,thl=>165    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>100    ns,tthl=>100    ns),
    -- Output delay class #2 clk to q at 10 volts, typical
            (tlh=>75    ns,thl=>75    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>50    ns,tthl=>50    ns),
    -- Output delay class #3 clk to co at 5 volts, typical
            (tlh=>280    ns,thl=>280    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>100    ns,tthl=>100    ns),
    -- Output delay class #4 clk to co at 10 volts, typical
            (tlh=>130    ns,thl=>130    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>50    ns,tthl=>50    ns),
    -- Output delay class #5 clk to q at 5 volts, minimum
            (tlh=>165    ns,thl=>165    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>35    ns,tthl=>35    ns),
    -- Output delay class #6 clk to q at 10 volts, minimum
            (tlh=>75    ns,thl=>75    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>18    ns,tthl=>18    ns),
    -- Output delay class #7 clk to co at 5 volts, minimum
            (tlh=>280    ns,thl=>280    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>35    ns,tthl=>35    ns),
    -- Output delay class #8 clk to co at 10 volts, minimum
            (tlh=>130    ns,thl=>130    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>18    ns,tthl=>18    ns),
    -- Output delay class #9 clk to q at 5 volts, maximum
            (tlh=>165    ns,thl=>165    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>200    ns,tthl=>200    ns),
    -- Output delay class #10 clk to q at 10 volts, maximum
            (tlh=>75    ns,thl=>75    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>100    ns,tthl=>100    ns),
    -- Output delay class #11 clk to co at 5 volts, maximum
            (tlh=>280    ns,thl=>280    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>200    ns,tthl=>200    ns),
    -- Output delay class #12 clk to co at 10 volts, maximum
            (tlh=>130    ns,thl=>130    ns,tlz=>0    ns,thz=>0    ns,tzl=>0    ns,tzh=>0    ns,
            ttlh=>100    ns,tthl=>100    ns));
```

3.4.2.9 Finally, it is time to assemble the electronic data sheet. In this section of the DS package, the design engineer establishes each of the operating points which will be declared in the EDS. To accomplish this, select the appropriate class for each pin at each operating point for the elec_spec, the sync_spec, the async_spec and the delay_spec.

3.4.2.9.1 The operating points supported by this model include the following. The "tzero" operating point allows this model to be used in other assemblies which are supported by an assembly specification.

Point	*Selection*	*Temp*	*Supply*
0	*tnom*	*27*	*5*
1	*tnom*	*27*	*10*
2	*tmin*	*-55*	*5*
3	*tmin*	*-55*	*10*
4	*tmax*	*125*	*5*
5	*tmax*	*125*	*10*
6	*tzero*	*27*	*5*

3.4.2.9.2 Due to the extensive size of this assignment, only the first (point 0) is shown below. It should be noted that all inputs and outputs have an elec_spec class other than 0. "delay_spec" applies only to outputs and "sync_spec" and "async_spec" values are used only when needed.

```
CONSTANT EDS:ELECTRONIC_DATA_SHEET:=(
-- The nominal or typical operating point at 5 volts
    0=>(
      selpoint=>(selection=>      TNOM,temp=>27     degrees_c,
          SUPPLY=>(0=>5.0      v)),
    edsnfo=>(
      j1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(0)),elec_spec=>1),
      j2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(0)),elec_spec=>1),
      j3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(0)),elec_spec=>1),
      j4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(0)),elec_spec=>1),
      pendx=>(sync_spec=>(1=>(5)),async_spec=>(1=>(3)),
          delay_spec=>(1=>(0)),elec_spec=>1),
      cindx=>(sync_spec=>(1=>(1)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(0)),elec_spec=>1),
      udndx=>(sync_spec=>(1=>(3)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(0)),elec_spec=>1),
      bdndx=>(sync_spec=>(1=>(3)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(0)),elec_spec=>1),
      clkndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(1)),
          delay_spec=>(1=>(0)),elec_spec=>1),
      q1ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(1)),elec_spec=>3),
      q2ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(1)),elec_spec=>3),
      q3ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(1)),elec_spec=>3),
      q4ndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(1)),elec_spec=>3),
      condx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(3)),elec_spec=>3),
      vddndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(0)),elec_spec=>0),
      vssndx=>(sync_spec=>(1=>(0)),async_spec=>(1=>(0)),
          delay_spec=>(1=>(0)),elec_spec=>0))),
```

3.4.2.10 Once the number of operating points has been established for the EDS, it is necessary to go back to the package declaration and insert the appropriate array limits into the ELECTRONIC_DATA_SHEET type declaration.

```
TYPE ELECTRONIC_DATA_SHEET IS ARRAY (0 TO 6) OF OPNT;
```

3.4.2.10.1 Additionally, the maximum number of constraints per pin values need to be checked. If more than one constraint per pin is allowed, these numbers need to be changed appropriately.

```
CONSTANT MAX_ASYNC_CONSTRAINTS_PER_PIN: NATURAL:=1;
CONSTANT MAX_SYNC_CONSTRAINTS_PER_PIN: NATURAL:=1;
CONSTANT MAX_DELAY_CONSTRAINTS_PER_PIN: NATURAL:=1;
```

3.4.2.10.2 In the example provided, we would have the option of using multiple delay constraints per pin in order to support the other propagation delay values provided in the data sheet. Approaches for accomplishing this will be discussed in Chapter 5.

3.4.3 The EDS Model

With the EDS model, the generic drivers and checkers are structurally assembled around the core component in order to implement the electronic data sheet. Figure 3.4.3-1 contains the EDS model for this example.

```
--*****************************************************************
--   (c) Copyright 1994 by the
--         Naval Air Warfare Center, Aircraft Division, Indianapolis
--   Source
--         Author(s):            Charles K. Rogers
--         Organization:           NAWC-ADI
--                               Code 306, MS-42
--                               6000 E 21st St
--                               Indianapolis,  IN  46219-2189
--                               Phone:   317-353-3579
--                               EMail:   ROGERSC1@po2.nawc-ad-indy.navy.mil
--         Reference:            VHDL   MODELING   GUIDE
--         Project:              SHARP   TIREP
--   DESC Certification
--         Status:               TBD
--*****************************************************************
--   Revision History
--         Version:              1.0
--               Date:           25 May 1994
--               Comments:       Original Release
--*****************************************************************
--   Module Description
--         File:                 4029eds.vhd
--         Module  Name(s):      cd4029_edsentity/arch_cd4029_eds_str
--               architecture
--         Constraints:          none
--         Limitations:          none
--         I/O  Format(s):       std_logic
--         Purpose and Use:      This VHDL model is the electronic data sheet
--               model of the CD4029 counter.
--         Notes:                none
--*****************************************************************
--   Standard Libraries/Packages
--         std_logic_1164        standard multi-value logic package
--   Associated Packages (order of analysis implied)
--         eia_567               standard eia package
--         cd4029_ds             cd4029 design specification package
--         eia_567_ds            eia design specification package
--   Component  Models
--         cd4029_core           behavorial VHDL module core
--         i_driver              generic input driver
```

Figure 3.4.3-1. The CD4029 EDS Model.

```
--         o_driver                generic output driver
--    Platform:                    486/33MHz PC
--    Software/Version:            V-System for Windows, Version 3.0
--****************************************************************
--    Design Specification Elements
--         entire file
--****************************************************************
LIBRARY ieee;
LIBRARY eia;
USE ieee.STD_LOGIC_1164.ALL;
USE eia.EIA_567.ALL;
USE work.cd4029_ds.ALL;
USE work.eia_cd4029_ds.ALL;
ENTITY cd4029_eds IS
    GENERIC(wi_j1,wi_j2,wi_j3,wi_j4,wi_pe,wi_ud,wi_bd,wi_ci,wi_clk,
        wi_q1,wi_q2,wi_q3,wi_q4,wi_co:TIME:=0 ns;
--*******************************************************************
-- Valid operating points are:
--     selection=tnom, temp=27 degrees_c, supply=(5 v)
--     selection=tnom, temp=27 degrees_c, supply=(10 v)
--     selection=tmin, temp=-55 degrees_c, supply=(5 v)
--     selection=tmin, temp=-55 degrees_c, supply=(10 v)
--     selection=tmax, temp=125 degrees_c, supply=(5 v)
--     selection=tmax, temp=125 degrees_c, supply=(10 v)
--     selection=tzero, temp=27 degrees_c, supply=(5 v)
--*******************************************************************
        user_operating_point: POINT:=(selection=> TNOM,temp=>27 degrees_c,
            SUPPLY=>(0=>5.0 v));
--*******************************************************************
-- The x_generation global variable controls 'X' state generation by
-- the "async_checker" and "sync_checker" modules when an
-- assert condition is violated.  If true, then 'X' states are generated
-- on assertion violations.  If false, then 'X' states will not be
-- generated.
--*******************************************************************
            x_generation: BOOLEAN:= TRUE;
--*******************************************************************
-- The m_generation variable is used to control reports provided by
-- assertion statements in the "sync_checker" and "async_checker
-- modules.  If the severity_level of the m_generation variable is less
-- than or equal to the serverity_level of an assertion violation, then
-- the assertion report will be generated.  Otherwise, the report will
-- not be provided.     Valid severity_level's are note, warning, error
-- or failure.
--*******************************************************************
            m_generation: SEVERITY_LEVEL:= WARNING);
    PORT(bd,pe,ci,ud,clk,j1,j2,j3,j4:IN STD_LOGIC;
        co,q1,q2,q3,q4:OUT STD_LOGIC;
        vdd,vss:IN STD_LOGIC);
```

Figure 3.4.3-1 continued. The CD4029 EDS Model.

```
    END cd4029_eds;
ARCHITECTURE arch_cd4029_eds_str OF cd4029_eds IS
    COMPONENT cd4029_core
        PORT(bd,pe,ci,ud,clk,j1,j2,j3,j4: IN STD_LOGIC;
            co,q1,q2,q3,q4: OUT STD_LOGIC);
        END COMPONENT;
    COMPONENT i_driver
        GENERIC(win_d: TIME;
            pin_data: EV_SIGNAL_LIMIT);
        PORT(a: IN STD_LOGIC;
            y:OUT STD_LOGIC);
        END COMPONENT;
    COMPONENT o_driver
        GENERIC(delay_y: DELAY;
            wir_o: TIME:=0 ns;
            pin_data: EV_SIGNAL_LIMIT;
            m_gen: SEVERITY_LEVEL;
            x_gen: BOOLEAN);
        PORT(a: IN STD_LOGIC;
            y:INOUT STD_LOGIC);
        END COMPONENT;
    COMPONENT async_checker
        GENERIC(x_gen: BOOLEAN:= TRUE;
            m_gen: SEVERITY_LEVEL:= WARNING;
            asyncconstraint: ASYNC);
        PORT(data_in: IN STD_LOGIC;
            data_out: OUT STD_LOGIC:='U');
        END COMPONENT;
    COMPONENT sync_checker
        GENERIC(x_gen: BOOLEAN:= TRUE;
            m_gen: SEVERITY_LEVEL:= WARNING;
            syncconstraint: SYNC);
        PORT(clk,data_in: IN STD_LOGIC;
            data_out: OUT STD_LOGIC:='U');
        END COMPONENT;
    SIGNAL ij1,ij2,ij3,ij4,ipe,ici,iud,ibd,iclk,obd,oud,oci,ppe: STD_LOGIC;
    SIGNAL iq1,iq2,iq3,iq4,ico,ope,oclk,oco,oq1,oq2,oq3,oq4: STD_LOGIC;
    FOR ALL: cd4029_core USE ENTITY work.cd4029_core;
    FOR ALL: i_driver USE ENTITY eia.i_driver;
    FOR ALL: o_driver USE ENTITY eia.o_driver;
    FOR ALL: async_checker USE ENTITY eia.async_checker;
    FOR ALL: sync_checker USE ENTITY eia.sync_checker;
    CONSTANT EDS: OPNT:= GET_TIMING(user_operating_point);
------------------------------------------------------------------------
-- The following constant definition is represented in one of its most
-- expanded forms.
------------------------------------------------------------------------
    CONSTANT INTERFACE_CONNECTIONS: PIN_AND_SIGNAL_CORROLATION:=(
-- Begins the array for each connection defined previously with the type
```

Figure 3.4.3-1 continued. The CD4029 EDS Model.

```
-- pin_index, there is an entry of information.Each entry contains
-- two elementsof information defined by the type.      Note that pin names
-- which are considered illegal can be represented in the text string
-- creating an explicit link between the port name used in VHDL and the
-- name actually used in hardware.
        j1ndx=>(pin_id=>"4                                          ",
            signal_name=>"jam    1                                  "),
        j2ndx=>(pin_id=>"12                                         ",
            signal_name=>"jam    2                                  "),
        j3ndx=>(pin_id=>"13                                         ",
            signal_name=>"jam    3                                  "),
        j4ndx=>(pin_id=>"3                                          ",
            signal_name=>"jam    4                                  "),
        pendx=>(pin_id=>"1                                          ",
            signal_name=>"preset      enable                        "),
        cindx=>(pin_id=>"5                                          ",
            signal_name=>"carry      in\                            "),
        bdndx=>(pin_id=>"9                                          ",
            signal_name=>"binary/decade                             "),
        udndx=>(pin_id=>"10                                         ",
            signal_name=>"up/down                                   "),
        clkndx=>(pin_id=>"15                                        ",
            signal_name=>"clock                                     "),
        q1ndx=>(pin_id=>"64                                         ",
            signal_name=>"q1                                        "),
        q2ndx=>(pin_id=>"11                                         ",
            signal_name=>"q2    .                                   "),
        q3ndx=>(pin_id=>"14                                         ",
            signal_name=>"q3                                        "),
        q4ndx=>(pin_id=>"2                                          ",
            signal_name=>"q4                                        "),
        condx=>(pin_id=>"7                                          ",
            signal_name=>"carry      out\                           "),
        vddndx=>(pin_id=>"16                                        ",
            signal_name=>"vdd                                       "),
        vssndx=>(pin_id=>"8                                         ",
            signal_name=>"vss                                       "));
    BEGIN
    ASSERT VALID_OPERATING_POINT(user_operating_point)
        REPORT "operating outside of the recommended range of this design"
        SEVERITY WARNING;
    s0:i_driver GENERIC MAP(win_d=>wi_j1,
            pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(j1ndx).elec_spec))
        PORT MAP(a=>j1,y=>ij1);
    s1:i_driver GENERIC MAP(win_d=>wi_j2,
            pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(j2ndx).elec_spec))
        PORT MAP(a=>j2,y=>ij2);
    s2:i_driver GENERIC MAP(win_d=>wi_j3,
            pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(j3ndx).elec_spec))
```

Figure 3.4.3-1 continued. The CD4029 EDS Model.

```
    PORT MAP(a=>j3,y=>ij3);
s3:i_driver GENERIC MAP(win_d=>wi_j4,
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(j4ndx).elec_spec))
    PORT MAP(a=>j4,y=>ij4);
s4:i_driver GENERIC MAP(win_d=>wi_pe,
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(pendx).elec_spec))
    PORT MAP(a=>pe,y=>ipe);
s5:i_driver GENERIC MAP(win_d=>wi_ci,
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(cindx).elec_spec))
    PORT MAP(a=>ci,y=>ici);
s6:i_driver GENERIC MAP(win_d=>wi_bd,
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(bdndx).elec_spec))
    PORT MAP(a=>bd,y=>ibd);
s7:i_driver GENERIC MAP(win_d=>wi_ud,
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(udndx).elec_spec))
    PORT MAP(a=>ud,y=>iud);
s8:i_driver GENERIC MAP(win_d=>wi_clk,
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(clkndx).elec_spec))
    PORT MAP(a=>clk,y=>iclk);
s9:o_driver
    GENERIC MAP(delay_y=>   DELAYS(EDS.edsnfo(q1ndx).delay_spec(1)),
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(q1ndx).elec_spec),
        m_gen=>m_generation,x_gen=>x_generation,wir_o=>wi_q1)
    PORT MAP(a=>iq1,y=>oq1);
s10:o_driver
    GENERIC MAP(delay_y=>   DELAYS(EDS.edsnfo(q2ndx).delay_spec(1)),
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(q2ndx).elec_spec),
        m_gen=>m_generation,x_gen=>x_generation,wir_o=>wi_q2)
    PORT MAP(a=>iq2,y=>oq2);
s11:o_driver
    GENERIC MAP(delay_y=>   DELAYS(EDS.edsnfo(q3ndx).delay_spec(1)),
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(q3ndx).elec_spec),
        m_gen=>m_generation,x_gen=>x_generation,wir_o=>wi_q3)
    PORT MAP(a=>iq3,y=>oq3);
s12:o_driver
    GENERIC MAP(delay_y=>   DELAYS(EDS.edsnfo(q4ndx).delay_spec(1)),
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(q4ndx).elec_spec),
        m_gen=>m_generation,x_gen=>x_generation,wir_o=>wi_q4)
    PORT MAP(a=>iq4,y=>oq4);
s13:o_driver
    GENERIC MAP(delay_y=>   DELAYS(EDS.edsnfo(condx).delay_spec(1)),
        pin_data=>   ELECTRICAL_PIN_SPEC(EDS.edsnfo(condx).elec_spec),
        m_gen=>m_generation,x_gen=>x_generation,wir_o=>wi_co)
    PORT MAP(a=>ico,y=>oco);
s14:async_checker
    GENERIC MAP(x_generation,m_generation,
        ASYNCS(EDS.edsnfo(clkndx).async_spec(1)))
    PORT MAP(data_in=>iclk,data_out=>oclk);
s15:async_checker
```

Figure 3.4.3-1 continued. The CD4029 EDS Model.

```
        GENERIC MAP(x_generation,m_generation,
            ASYNCS(EDS.edsnfo(pendx).async_spec(1)))
        PORT MAP(data_in=>ipe,data_out=>ppe);
    s16:sync_checker
        GENERIC MAP(x_generation,m_generation,
            SYNCS(EDS.edsnfo(cindx).sync_spec(1)))
        PORT MAP(iclk,ici,oci);
    s17:sync_checker
        GENERIC MAP(x_generation,m_generation,
            SYNCS(EDS.edsnfo(udndx).sync_spec(1)))
        PORT MAP(iclk,iud,oud);
    s18:sync_checker
        GENERIC MAP(x_generation,m_generation,
            SYNCS(EDS.edsnfo(bdndx).sync_spec(1)))
        PORT MAP(iclk,ibd,obd);
    s19:sync_checker
        GENERIC MAP(x_generation,m_generation,
            SYNCS(EDS.edsnfo(pendx).sync_spec(1)))
        PORT MAP(iclk,ppe,ope);
    s20:cd4029_core PORT MAP(obd,ope,oci,oud,iclk,ij1,ij2,ij3,ij4,      ico,iq1,iq2,iq3,iq4);
    co<=oco    WHEN vdd='1'  AND vss='0'  ELSE 'U';
    q1<=oq1    WHEN vdd='1'  AND vss='0'  ELSE 'U';
    q2<=oq2    WHEN vdd='1'  AND vss='0'  ELSE 'U';
    q3<=oq3    WHEN vdd='1'  AND vss='0'  ELSE 'U';
    q4<=oq4    WHEN vdd='1'  AND vss='0'  ELSE 'U';
    END  arch_cd4029_eds_str;
```

Figure 3.4.3-1 continued. The CD4029 EDS Model.

3.4.3.1 The GENERIC map for the EDS model contains a number of pieces of information. For the most part, it is not necessary that the design engineer make a lot of changes to this segment of code, however, there are a few items which must be addressed. First, the GENERIC clause should contain routing delays for all inputs and outputs (excluding power and ground). These elements are declared herein with a "wi_" prefix to the input and output signal names. Secondly, all of the valid operating points as declared in the EDS should be identified in comments at this point.

```
    GENERIC(wi_j1,wi_j2,wi_j3,wi_j4,wi_pe,wi_ud,wi_bd,wi_ci,wi_clk,
        wi_q1,wi_q2,wi_q3,wi_q4,wi_co:TIME:=0  ns;
--*****************************************************************
 -- Valid operating points are:
 --     selection=tnom, temp=27 degrees_c, supply=(5 v)
 --     selection=tnom, temp=27 degrees_c, supply=(10 v)
 --     selection=tmin, temp=-55 degrees_c, supply=(5 v)
 --     selection=tmin, temp=-55 degrees_c, supply=(10 v)
 --     selection=tmax, temp=125 degrees_c, supply=(5 v)
 --     selection=tmax, temp=125 degrees_c, supply=(10 v)
 --     selection=tzero, temp=27 degrees_c, supply=(5 v)
--*****************************************************************
        user_operating_point: POINT:=(selection=>   TNOM,temp=>27   degrees_c,
            SUPPLY=>(0=>5.0    v));
```

3.4.3.2 The PORT map must be modified for the design being developed. This map should reflect the inputs and outputs of the model including power and ground.

```
PORT(bd,pe,ci,ud,clk,j1,j2,j3,j4:IN STD_LOGIC;
  co,q1,q2,q3,q4:OUT STD_LOGIC;
  vdd,vss:IN STD_LOGIC);
END cd4029_eds;
```

3.4.3.3 In general, the components used in the EDS model will include the design core component (cd4029_core in this case) and the generic drivers and checkers (as needed). For this example, all of the generic components are used with the exception of the bidirectional driver.

```
COMPONENT cd4029_core
  PORT(bd,pe,ci,ud,clk,j1,j2,j3,j4:IN STD_LOGIC;
    co,q1,q2,q3,q4:OUT STD_LOGIC);
  END COMPONENT;
COMPONENT i_driver
  GENERIC(win_d:TIME;
    pin_data: EV_SIGNAL_LIMIT);
  PORT(a:IN STD_LOGIC;
    y:OUT STD_LOGIC);
  END COMPONENT;
COMPONENT o_driver
  GENERIC(delay_y:DELAY;
    wir_o:TIME:=0  ns;
    pin_data: EV_SIGNAL_LIMIT;
    m_gen:SEVERITY_LEVEL;
    x_gen:BOOLEAN);
  PORT(a:IN STD_LOGIC;
    y:INOUT STD_LOGIC);
  END COMPONENT;
COMPONENT async_checker
  GENERIC(x_gen:BOOLEAN:= TRUE;
    m_gen:SEVERITY_LEVEL:= WARNING;
    asyncconstraint: ASYNC);
  PORT(data_in: IN STD_LOGIC;
    data_out: OUT STD_LOGIC:='U');
  END COMPONENT;
COMPONENT sync_checker
  GENERIC(x_gen:BOOLEAN:= TRUE;
    m_gen:SEVERITY_LEVEL:= WARNING;
    syncconstraint: SYNC);
  PORT(clk,data_in: IN STD_LOGIC;
    data_out: OUT STD_LOGIC:='U');
  END COMPONENT;
```

3.4.3.4 Internal signals are declared to handle the interfaces between the core component and the generic drivers and checkers.

```
SIGNAL ij1,ij2,ij3,ij4,ipe,ici,iud,ibd,iclk,obd,oud,oci,ppe:STD_LOGIC;
SIGNAL iq1,iq2,iq3,iq4,ico,ope,oclk,oco,oq1,oq2,oq3,oq4:STD_LOGIC;
```

3.4.3.5 The component configuration clauses need to be updated for the design core component. Other clauses may or may not require alteration based upon the configuration of the design platform and the manner in which libraries are established.

```
FOR ALL:cd4029_core USE ENTITY work.cd4029_core;
```

```
FOR ALL:i_driver USE ENTITY eia.i_driver;
FOR ALL:o_driver USE ENTITY eia.o_driver;
FOR ALL:async_checker USE ENTITY eia.async_checker;
FOR ALL:sync_checker USE ENTITY eia.sync_checker;
```

3.4.3.6 Generic drivers and checkers are instantiated around the core component as required for the design. In this case, every input is equipped with an "i_driver" component. Similarly, outputs have "o_driver" components. The "sync_checker" and "async_checker" components are instantiated as needed. Following is a single instantiation for each of the components used in this model.

```
s0:i_driver GENERIC MAP(win_d=>wi_j1,
        pin_data=>  ELECTRICAL_PIN_SPEC(EDS.edsnfo(j1ndx).elec_spec))
    PORT MAP(a=>j1,y=>ij1);
s9:o_driver
    GENERIC MAP(delay_y=>  DELAYS(EDS.edsnfo(q1ndx).delay_spec(1)),
        pin_data=>  ELECTRICAL_PIN_SPEC(EDS.edsnfo(q1ndx).elec_spec),
        m_gen=>m_generation,x_gen=>x_generation,wir_o=>wi_q1)
    PORT MAP(a=>iq1,y=>oq1);
s14:async_checker
    GENERIC MAP(x_generation,m_generation,
        ASYNCS(EDS.edsnfo(clkndx).async_spec(1)))
    PORT MAP(data_in=>iclk,data_out=>oclk);
s16:sync_checker
    GENERIC MAP(x_generation,m_generation,
        SYNCS(EDS.edsnfo(cindx).sync_spec(1)))
    PORT MAP(iclk,ici,oci);
s20:cd4029_core PORT MAP(obd,ope,oci,oud,iclk,ij1,ij2,ij3,ij4,         ico,iq1,iq2,iq3,iq4);
```

3.4.3.7 Finally, the operation of power and ground is incorporated into the model. This is performed as follows. In this segment of code, the outputs are directly assigned from the output drivers unless "vdd" is not '1' or "vss" is not '0'.

```
co<=oco   WHEN vdd='1' AND vss='0' ELSE 'U';
q1<=oq1   WHEN vdd='1' AND vss='0' ELSE 'U';
q2<=oq2   WHEN vdd='1' AND vss='0' ELSE 'U';
q3<=oq3   WHEN vdd='1' AND vss='0' ELSE 'U';
q4<=oq4   WHEN vdd='1' AND vss='0' ELSE 'U';
```

3.5 WAVES Dataset Files and Testbench

As noted earlier, there are no changes required for the "wav_logic" package used by this model. Therefore, it will not be detained at this point. For more information on this package, refer to Chapter 7.

3.5.1 The WAVES Header File

The WAVES header file is provided for information only. It offers information about the WAVES dataset including how it is structured and analyzed. Figure 3.5.1-1 contains the WAVES header file for the CD4029 counter example.

3.5.2 The Wav_Frame Package

The "wav_frame" package is contained in Figure 3.5.2-1. This package requires very little change. The "simout" function of this package contains a timing parameter which is used by the testbench to determine when to check the outputs of the model. Therefore, this timing parameter needs to be something in excess of the maximum propagation delay through the model. A brief study of the delay constraints imposed on this model in the "4029ds_p" package reveals that the maximum propagation delay will be 560ns with an additional settling time of 200/2 or 100 ns. This means that the timing parameter should be at least 560+ 100 or 660ns. In this example, this time parameter has been set to 700ns.

```
FUNCTION simout RETURN FRAME_SET IS
```

```
--***************************************************************
--   (c) Copyright 1994 by the
--        Naval Air Warfare Center, Aircraft Division, Indianapolis
--   Source
--        Author(s):           Charles K. Rogers
--        Organization:          NAWC-ADI
--                             Code 306, MS-42
--                             6000 E 21st St
--                             Indianapolis,  IN  46219-2189
--                             Phone:   317-353-3579
--                             EMail:   ROGERSC1@po2.nawc-ad-indy.navy.mil
--        Reference:           VHDL  MODELING  GUIDE
--        Project:             SHARP   TIREP
--   DESC Certification
--        Status:              TBD
--***************************************************************
--   Revision History
--        Version:             1.0
--             Date:           26 May 1994
--             Comments:       Original Release
--***************************************************************
--   Module Description
--        File:                4029hdr.wav
--        Module  Name(s):      none
--        Constraints:         none
--        Limitations:         none
--        I/O  Format(s):      none
--        Purpose and Use:     This VHDL module describes the contents of the
--             WAVES dataset for the CD4029 counter.
--        Notes:               none
--***************************************************************
--   Standard Libraries/Packages
--        none
--   Associated Packages (order of analysis implied)
--        none
--   Component  Models
--        none
--   Platform:                 486/33MHz PC
--   Software/Version:         V-System for Windows, Version 3.0
--***************************************************************
--   Design Specification Elements
--        entire file
--***************************************************************
title                          cd4029 waves dataset
corporate_author               NAWC-ADI
individual_author              Charles  K.  Rogers
release_date_and_time          26 may 1994
origin
device_id                       cd4029b  counter
```

Figure 3.5.1-1. The CD4029 WAVES Header File.

```
-- Waves files
waves_filename               wav_logp.wav               work
waves_unit                   WAVES_INTERFACE            work
waves_filename               wav_frmp.wav               work
waves_filename               4029dutp.wav               work
waves_unit                   WAVES_OBJECTS              work
waves_filename               4029wavp.wav               work
-- Waveform external data file
external_filename            4029vec.dat                data
-- Waveform generation procedure
waveform_generator_procedure work.cd4029_wav.cd4029wav
```

Figure 3.5.1-1 continued. The CD4029 WAVES Header File.

```
BEGIN
RETURN
   NEW_FRAME_SET('U',FRAME_EVENT(ui, ETIME(700 ns))))+
   NEW_FRAME_SET('X',FRAME_EVENT(ud, ETIME(700 ns))))+
   NEW_FRAME_SET('0',FRAME_EVENT(lo_out, ETIME(700 ns))))+
   NEW_FRAME_SET('1',FRAME_EVENT(hi_out, ETIME(700 ns))))+
   NEW_FRAME_SET('Z',FRAME_EVENT(hz_out, ETIME(700 ns))));
   NEW_FRAME_SET('W',FRAME_EVENT(ud, ETIME(700 ns))))+
   NEW_FRAME_SET('L',FRAME_EVENT(wl_out, ETIME(700 ns))))+
   NEW_FRAME_SET('H',FRAME_EVENT(wh_out, ETIME(700 ns))))+
   NEW_FRAME_SET('-',FRAME_EVENT(ud, ETIME(700 ns))))+
END simout;
```

3.5.3 The Device Under Test Package

The device under test (or DUT) package contains the TEST_PINS declaration. This declaration is used to establish the order in which pin information will be read from the vector file. Figure 3.5.3-1 contains the DUT package for this example. Additionally, the name of this package has been changed to reflect the new design we are working with.

```
TYPE TEST_PINS IS (j1,j2,j3,j4,pe,ci,ud,bd,clk,q1,q2,q3,q4,co,
   vdd,vss);
```

3.5.4 The Waveform Generator Package

For the most part, the waveform generator package does not significantly change. The bulk of changes to this package are name and context clause changes to reflect the design being modeled.

3.5.4.1 The following name and context clause changes were made in this package. They include the library reference to the DUT package, the package name, the waveform generator procedure name and the vector data file name.

```
USE work.cd4029_dut.ALL;
PACKAGE cd4029_wav IS
PROCEDURE cd4029wav (SIGNAL wpl:INOUT WAVES_PORT_LIST)
FILE data_file:TEXT IS IN "4029vec.dat";
```

3.5.4.2 In addition to these changes, the "input_pins" and "output_pins" pinsets are declared. These pinsets call out all of the inputs and outputs to the model. Power and ground pins are included as inputs.

```
CONSTANT input_pins:PINSET:= NEW_PINSET ((j1,j2,j3,j4,pe,ci,ud,bd,clk,
```

```
--***************************************************************
--   (c) Copyright 1994 by the
--        Naval Air Warfare Center, Aircraft Division, Indianapolis
--   Source
--        Author(s):          Charles K. Rogers
--        Organization:       NAWC-ADI
--                            Code 306, MS-42
--                            6000 E 21st St
--                            Indianapolis,  IN  46219-2189
--                            Phone:   317-353-3579
--                            EMail:   ROGERSC1@po2.nawc-ad-indy.navy.mil
--        Reference:          VHDL  MODELING  GUIDE
--        Project:            SHARP  TIREP
--   DESC Certification
--        Status:             TBD
--***************************************************************
--   Revision History
--        Version:            1.0
--             Date:          26 May 1994
--             Comments:      Original Release
--***************************************************************
--   Module Description
--        File:               wav_frmp.wav
--        Module  Name(s):    wav_frame package
--        Constraints:        none
--        Limitations:        none
--        I/O  Format(s):     none
--        Purpose and Use:    This VHDL package provides the value dictionary
--             which is used in conjunction with the vector table.
--        Notes:              none
--***************************************************************
--   Standard Libraries/Packages
--        Waves_Interface     waves interface package
--   Associated Packages (order of analysis implied)
--        wav_logic           simulator value dictionary package
--   Component  Models
--        none
--   Platform:                486/33MHz PC
--   Software/Version:        V-System for Windows, Version 3.0
--***************************************************************
--   Design Specification Elements
--        simfrm                        function
--        simin                         function
--        simout                        function
--***************************************************************
LIBRARY waves;
USE work.wav_logic.ALL;
USE work.WAVES_INTERFACE.ALL;
PACKAGE wav_frame IS
   FUNCTION simfrm RETURN FRAME_SET;
```

Figure 3.5.2-1. The Wav_Frame Package.

```
    FUNCTION simin    RETURN FRAME_SET;
    FUNCTION simout RETURN FRAME_SET;
    END wav_frame;
PACKAGE BODY wav_frame IS
--*************************************************************
-- The following functions are based upon the PIN_CODES string being
-- set to "UX01ZWLH-". If a different PIN_CODES string is employed,
-- then these functions would need to be modified accordingly.
-- Otherwise, there are no changes required to this package.
--*************************************************************
    FUNCTION simfrm RETURN FRAME_SET IS
        BEGIN
        RETURN
            NEW_FRAME_SET('U',FRAME_EVENT(ui, ETIME(0   ns))))+
            NEW_FRAME_SET('X',FRAME_EVENT(ud, ETIME(0   ns))))+
            NEW_FRAME_SET('0',FRAME_EVENT(lo, ETIME(0   ns))))+
            NEW_FRAME_SET('1',FRAME_EVENT(hi, ETIME(0   ns))))+
            NEW_FRAME_SET('Z',FRAME_EVENT(hz, ETIME(0   ns))))+
            NEW_FRAME_SET('W',FRAME_EVENT(ud, ETIME(0   ns))))+
            NEW_FRAME_SET('L',FRAME_EVENT(wl, ETIME(0   ns))))+
            NEW_FRAME_SET('H',FRAME_EVENT(wh, ETIME(0   ns))))+
            NEW_FRAME_SET('-',FRAME_EVENT(ud, ETIME(0 ns))));
        END simfrm;
    FUNCTION simin RETURN FRAME_SET IS
        BEGIN
        RETURN
            NEW_FRAME_SET('U',FRAME_EVENT(ui, ETIME(0   ns))))+
            NEW_FRAME_SET('X',FRAME_EVENT(ud, ETIME(0   ns))))+
            NEW_FRAME_SET('0',FRAME_EVENT(lo_in, ETIME(0   ns))))+
            NEW_FRAME_SET('1',FRAME_EVENT(hi_in, ETIME(0   ns))))+
            NEW_FRAME_SET('Z',FRAME_EVENT(hz_in, ETIME(0   ns))))+
            NEW_FRAME_SET('W',FRAME_EVENT(ud, ETIME(0   ns))))+
            NEW_FRAME_SET('L',FRAME_EVENT(wl_in, ETIME(0   ns))))+
            NEW_FRAME_SET('H',FRAME_EVENT(wh_in, ETIME(0   ns))))+
            NEW_FRAME_SET('-',FRAME_EVENT(ud, ETIME(0 ns))));
        END simin;
    FUNCTION simout RETURN FRAME_SET IS
        BEGIN
        RETURN
            NEW_FRAME_SET('U',FRAME_EVENT(ui, ETIME(700   ns))))+
            NEW_FRAME_SET('X',FRAME_EVENT(ud, ETIME(700   ns))))+
            NEW_FRAME_SET('0',FRAME_EVENT(lo_out, ETIME(700   ns))))+
            NEW_FRAME_SET('1',FRAME_EVENT(hi_out, ETIME(700   ns))))+
            NEW_FRAME_SET('Z',FRAME_EVENT(hz_out, ETIME(700   ns))))+
            NEW_FRAME_SET('W',FRAME_EVENT(ud, ETIME(700   ns))))+
            NEW_FRAME_SET('L',FRAME_EVENT(wl_out, ETIME(700   ns))))+
            NEW_FRAME_SET('H',FRAME_EVENT(wh_out, ETIME(700   ns))))+
            NEW_FRAME_SET('-',FRAME_EVENT(ud, ETIME(700 ns))));
        END simout;
    END wav_frame;
```

Figure 3.5.2-1 continued. The Wav_Frame Package.

```
--***************************************************************
--   (c) Copyright 1994 by the
--        Naval Air Warfare Center, Aircraft Division, Indianapolis
--   Source
--        Author(s):           Charles K. Rogers
--        Organization:          NAWC-ADI
--                             Code 306, MS-42
--                             6000 E 21st St
--                             Indianapolis,  IN  46219-2189
--                             Phone:   317-353-3579
--                             EMail:   ROGERSC1@po2.nawc-ad-indy.navy.mil
--        Reference:           VHDL  MODELING  GUIDE
--        Project:             SHARP  TIREP
--   DESC Certification
--        Status:              TBD
--***************************************************************
--   Revision History
--        Version:             1.0
--             Date:           26 May 1994
--             Comments:       Original Release
--***************************************************************
--   Module Description
--        File:                4029dutp.wav
--        Module  Name(s):     package cd4029_dut
--        Constraints:         none
--        Limitations:         none
--        I/O  Format(s):      none
--        Purpose and Use:     This VHDL package provides the test_pins
--             declaration for this design.
--        Notes:               none
--***************************************************************
--   Standard Libraries/Packages
--        none
--   Associated Packages (order of analysis implied)
--        none
--   Component  Models
--        none
--   Platform:                 486/33MHz PC
--   Software/Version:         V-System for Windows, Version 3.0
--***************************************************************
--   Design Specification Elements
--        TEST_PINS             enumerated type declaration
--***************************************************************
PACKAGE cd4029_dut IS
--***************************************************************
-- The TEST_PINS data type is composed of the port signal names for the
-- VHDL module to be tested by the testbench.    The order of the signals
-- appearing in this declaration must be the same as the order of the
-- signals in the supporting vector data file.
```

Figure 3.5.3-1. The CD4029 DUT Package.

```
--*****************************************************************
      TYPE TEST_PINS IS (j1,j2,j3,j4,pe,ci,ud,bd,clk,q1,q2,q3,q4,co,
            vdd,vss);
END cd4029_dut;
```

Figure 3.5.3-1 continued. The CD4029 DUT Package.

```
      vcc,gnd));
CONSTANT output_pins:PINSET:= NEW_PINSET ((q1,q2,q3,q4,co));
```

3.5.5 The Vector File

The vector file contains the patterns which will be asserted on the model in order to test it. Both stimulus and response information is included in this file. In actuality, two vector files are required for a design. One file will be used to validate the model (i.e. will run without errors on a good model). The second vector file will exercise the model by intentionally causing errors to be generated. Sample vector files are included as Figure 3.5.5-1 (error free) and Figure 3.5.5-2 (error seeded).

3.5.6 The Testbench

The testbench is the final file to be addressed in this model. It is the testbench which applies stimulus to the model and checks the model's responses. The testbench for CD4029 counter is contained in Figure 3.5.6-1. The testbench is a free-running model which will execute for the time duration specified by the user or until the external vector file is exhausted. At the conclusion of the simulation, the testbench will report the success or failure of the simulation.

3.5.6.1 There are a number of changes which must be performed in the testbench file. The first task is to change the name of the testbench to one appropriate for the design. For the CD4029 design example, the name "cd4029_tst" was selected.

3.5.6.2 The next step is to call out the "cd4029_eds" component and configuration. This is accomplished as follows.

```
COMPONENT cd4029_eds
      GENERIC(wi_j1,wi_j2,wi_j3,wi_j4,wi_pe,wi_ud,wi_bd,wi_ci,wi_clk,
            wi_q1,wi_q2,wi_q3,wi_q4,wi_co:TIME:=0 ns;
            user_operating_point: POINT:=(selection=>prop_del,
            temp=>op_temp,    SUPPLY=>(0=>op_volt));
            x_generation: BOOLEAN:= TRUE;
            m_generation: SEVERITY_LEVEL:= WARNING);
      PORT(bd,pe,ci,ud,clk,j1,j2,j3,j4:IN STD_LOGIC;
            co,q1,q2,q3,q4: OUT STD_LOGIC;
            vdd,vss:IN STD_LOGIC);
            END COMPONENT;
FOR ALL:cd4029_eds USE ENTITY work.cd4029_eds;
```

```
--*****************************************************************
--    (c) Copyright 1994 by the
--          Naval Air Warfare Center, Aircraft Division, Indianapolis
```

Figure 3.5.4-1. The CD4029 Waveform Generator Package.

```
--    Source
--          Author(s):              Charles K. Rogers
--          Organization:             NAWC-ADI
--                                  Code 306, MS-42
--                                  6000 E 21st St
--                                  Indianapolis,  IN  46219-2189
--                                  Phone:   317-353-3579
--                                  EMail:   ROGERSC1@po2.nawc-ad-indy.navy.mil
--          Reference:              VHDL   MODELING   GUIDE
--          Project:                SHARP   TIREP
--    DESC Certification
--          Status:                 TBD
--****************************************************************
--    Revision History
--          Version:                1.1
--                Date:             10 August 1994
--                Comments:         Converted "fileend" from a signal to a
--                      variable in support of WAVES standard.
--          Version:                1.0
--                Date:             26 May 1994
--                Comments:         Original Release
--****************************************************************
--    Module Description
--          File:                   4029wavp.wav
--          Module  Name(s):        cd4029_wavpackage
--          Constraints:            none
--          Limitations:            none
--          I/O  Format(s):         none
--          Purpose and Use:        This VHDL package provides the waveform
--                generator procedure for the CD4029 counter.
--          Notes:                  none
--****************************************************************
--    Standard Libraries/Packages
--          Waves_Interface         waves interface package
--          Waves_Objects           waves objects package
--          textio                  standard text i/o package
--    Associated Packages (order of analysis implied)
--          wav_logic               simulator value dictionary package
--          wav_frame               simulator code map package
--          4029_dut                WAVES test pin package
--          4029vec.dat             WAVES vector data file
--    Component  Models
--          none
--    Platform:                     486/33MHz PC
--    Software/Version:             V-System for Windows, Version 3.0
--****************************************************************
--    Design Specification Elements
--          input_pins              constant
--          output_pins             constant
```

Figure 3.5.4-1 continued. The CD4029 Waveform Generator.

```
--      cd4029wav          procedure
--      data_file          file
--*****************************************************************
USE work.wav_logic.ALL;
USE work.WAVES_INTERFACE.ALL;
USE work.wav_frame.ALL;
USE work.cd4029_dut.ALL;
USE work.WAVES_OBJECTS.ALL;
PACKAGE cd4029_wav IS
--*****************************************************************
-- The following pinsets must be declared. These pinsets are used
-- to distinguish inputs from outputs in the vector file.
--*****************************************************************
   CONSTANT input_pins:PINSET:= NEW_PINSET ((j1,j2,j3,j4,pe,ci,ud,bd,clk,
      vdd,vss));
   CONSTANT output_pins:PINSET:= NEW_PINSET ((q1,q2,q3,q4,co));
--*****************************************************************
-- The procedure "cd4029wav" is named specifically for the subject
-- design in this example. The procedure name must be changed in
-- both the package declaration and the package body.
--*****************************************************************
   PROCEDURE cd4029wav (SIGNAL wpl:INOUT WAVES_PORT_LIST;
      VARIABLE fileend:OUT BOOLEAN);
   END cd4029_wav;
USE std.TEXTIO.ALL;
PACKAGE BODY cd4029_wav IS
   PROCEDURE cd4029wav (SIGNAL wpl:INOUT WAVES_PORT_LIST;
      VARIABLE fileend:OUT BOOLEAN) IS
--*****************************************************************
-- The filename for the "data_file" needs to be changed here to
-- identify the vector file for the design.
--*****************************************************************
      FILE data_file:TEXT IS IN "4029vec.dat";
      VARIABLE X:FILE_SLICE:= NEW_FILE_SLICE;
      CONSTANT single_frame_set_array:FRAME_SET_ARRAY:=
         NEW_FRAME_SET_ARRAY(simin,input_pins)+
         NEW_FRAME_SET_ARRAY(simout,output_pins);
      VARIABLE td:TIME_DATA:= NEW_TIME_DATA(single_frame_set_array);
      BEGIN
      fileend:= FALSE;
      LOOP
         READ_FILE_SLICE (data_file,X);
         EXIT WHEN X.end_of_file=TRUE;
         APPLY (wpl,X.codes.ALL,DELAY(X.fs_time),td);
         END LOOP;
      IF X.end_of_file=TRUE THEN fileend:=TRUE;
         END IF;
      END cd4029wav;
   END cd4029_wav;
```

Figure 3.5.4-1 continued. The CD4029 Waveform Generator.

```
%*****************************************************************
%   (c) Copyright 1994 by the
%        Naval Air Warfare Center, Aircraft Division, Indianapolis
%   Source
%        Author(s):            Charles K. Rogers
%        Organization:         NAWC-ADI
%                              Code 306, MS-42
%                              6000 E 21st St
%                              Indianapolis, IN 46219-2189
%                              Phone:  317-353-3579
%                              Email:  rogersc1@po2.nawc-ad-indy.navy.mil
%        Reference:            VHDL Modeling Guide
%        Project:              SHARP  TIREP
%   DESC Certification
%        Status:               TBD
%*****************************************************************
%   Revision History
%        Version:              1.0
%            Date:             26 May 1994
%            Comments:         Original Release
%*****************************************************************
%   module description
%        File:                 4029vec.dat
%        Module Name(s):       none
%        Constraints:          none
%        Limitations:          none
%        I/O Format(s):        none
%        Purpose and Use:      This file contains the functional vector
%            set for the cd4029 counter.    This vector set will run error
%            free on a valid model of the cd4029.    This file will run for
%            46,000 ns.
%        Notes:                none
%*****************************************************************
%   Standard Libraries/Packages
%        none
%   Associated Packages (order of analysis implied)
%        none
%   Component Models
%        none
%   Platform:                  486/33MHz PC
%   Software/Version:          V-System for Windows, Version 3.0
%*****************************************************************
%   Design Specification Elements
%        entire file
%*****************************************************************
% pin order is
% j1,j2,j3,j4,pe,ci,ud,bd,clk,q1,q2,q3,q4,co,vdd,vss
000000000UUUUU00 :1000 ns;
0000000000000010;
```

Figure 3.5.5-1. An Error-Free Vector File.

```
0000101000000110;
0000001000000110;
0000001011000110;
0000001001000110;
0000001010100110;
0000001000100110;
0000000000100110;
0000000011000110;
0000000001000110;
0000000010000010;
0000010000000110;
0000000000000010;
0000000011001110;
0000000001001110;
0000000010001110;
0000000000001110;
0000000011110110;
0000001101110110;
0000001110001110;
0000001100001110;
0000001111001110;
0000001101001110;
0000001110101110;
0000001100101110;
0000001111101110;
0000001101101110;
0000001110011110;
0000001100011110;
0000001111011110;
0000001101011110;
0000001110111110;
0000001100111110;
0000011111111110;
0000001111111010;
0000001101111010;
0000001110000110;
0000001100000110;
1010001100000110;
1010101101010110;
1010001101010110;
0101001101010110;
0101101100101110;
0101001100101110;
0000001100101110;
```

Figure 3.5.5-1 continued. An Error-Free Vector File.

```
%*****************************************************************
%   (c) Copyright 1994 by the
%        Naval Air Warfare Center, Aircraft Division, Indianapolis
%   Source
%        Author(s):              Charles K. Rogers
%        Organization:           NAWC-ADI
%                                Code 306, MS-42
%                                6000 e 21st st
%                                Indianapolis, IN 46219-2189
%                                Phone:   317-353-3579
%                                Email:   rogersc1@po2.nawc-ad-indy.navy.mil
%        Reference:              VHDL Modeling Guide
%        Project:                SHARP  TIREP
%   DESC Certification
%        Status:                 TBD
%*****************************************************************
%   Revision History
%        Version:                1.0
%             Date:              26 May 1994
%             Comments:          Original Release
%*****************************************************************
%   Module Description
%        File:                   4029vec.dat
%        Module Name(s):         none
%        Constraints:            none
%        Limitations:            none
%        I/O Format(s):          none
%        Purpose and Use:        This file contains an error-seeded vector
%             set for the cd4029 counter.    This vector set will test
%             synchronous and asynchronous constraints for the subject
%             model.  This vector set runs for 46,000 ns and will generate
%             10 constraint violations.
%        Notes:                  none
%*****************************************************************
%   Standard  Libraries/Packages
%        none
%   Associated Packages (order of analysis implied)
%        none
%   Component Models
%        none
%   Platform:                    486/33MHz PC
%   Software/Version:            V-System for Windows, Version 3.0
%*****************************************************************
%   Design Specification Elements
%        entire file
%*****************************************************************
% pin order is
% j1,j2,j3,j4,pe,ci,ud,bd,clk,q1,q2,q3,q4,co,vdd,vss
000000000UUUUU00 :1000 ns;
```

Figure 3.5.5-2. An Error Seeded Vector File.

```
0000000000000010;
% test pe minimum high pulse width
0000101000000110:5ns;
0000001000000110:1000ns;
% test clk minimum high pulse width
0000001010000110:5ns;
0000001001000110:1995ns;
0000001010100110:1000ns;
0000001000100110;
0000000000100110;
0000000011000110;
% test clk minimum low pulse width
0000000001000110:10ns;
0000000010000010:1000ns;
% test ci setup time
0000010000000110:3ns;
0000000010000110:1000ns;
0000000010000110;
0000000000000010;
0000000011001110;
0000000001001110;
0000000010001110;
% test bd/ud setup time
0000001101001110:20ns;
0000001111001110:1000ns;
0000001101001110;
0000001110101110;
0000001100101110;
0000001111101110;
0000001101101110;
0000001110011110;
0000001100011110;
0000001111011110;
0000001101011110;
0000001110111110;
0000001100111110;
0000011111111110;
0000001111111010;
0000001101111010;
0000001110000110;
0000001100000110;
1010001100000110;
% test pe setup time
1010101101010110:15ns;
1010001111010110:1000ns;
0101001101010110;
0101101100101110;
0101001100101110;
0000001100101110:7947ns;
```

Figure 3.5.5-2 continued. An Error Seeded Vector File.

```
--***************************************************************
--  (c) Copyright 1994 by the
--      Naval Air Warfare Center, Aircraft Division, Indianapolis
--  Source
--      Author(s):            Charles K. Rogers
--      Organization:         NAWC-ADI
--                            Code 306, MS-42
--                            6000 E 21st St
--                            Indianapolis,  IN  46219-2189
--                            Phone:   317-353-3579
--                            EMail:   ROGERSC1@po2.nawc-ad-indy.navy.mil
--      Reference:            VHDL  MODELING  GUIDE
--      Project:              SHARP  TIREP
--  DESC Certification
--      Status:               TBD
--***************************************************************
--  Revision History
--      Version:              1.1
--          Date:             10 August 1994
--          Comments:         Added process to call waveform generator
--              procedure which provides for the detection of the end
--              of vector file.
--      Version:              1.0
--          Date:             26 May 1994
--          Comments:         Original Release
--***************************************************************
--  Module Description
--      File:                 4029tstt.vhd
--      Module  Name(s):      cd4029_tst/arch_waves_app_tb
--      Constraints:          none
--      Limitations:          none
--      I/O  Format(s):       none
--      Purpose and Use:      This VHDL module describes the testbench for the
--          CD4029 counter.
--      Notes:                none
--***************************************************************
--  Standard Libraries/Packages
--      std_logic_1164        standard multi-value logic package
--      waves_objects         WAVES objects package
--  Associated Packages (order of analysis implied)
--      eia_567               standard eia package
--      cd4029_ds             cd4029 design specification package
--      wav_logic             simulator value dictionary package
--  Component  Models
--      cd4029_eds            the electronic data sheet model
--  Platform:                 486/33MHz PC
--  Software/Version: V-System for Windows, Version 3.0
--***************************************************************
--  Design Specification Elements
```

Figure 3.5.6-1. The CD4029 Testbench.

```
--      wire delays          generic map
--      operating points     generic map
--      cd4029_eds           model under test component
--      w_signals            waveform generator stimulus/response signals
--      p_signals            model response signals
--      to_1164              function
--      translate            process
--      errchk               process
--*****************************************************************
LIBRARY ieee;
LIBRARY eia;
USE ieee.STD_LOGIC_1164.ALL;
USE work.wav_logic.ALL;
USE work.WAVES_OBJECTS.ALL;
USE work.cd4029_wav.ALL;
USE eia.EIA_567.ALL;
USE work.cd4029_ds.ALL;
ENTITY cd4029_tst IS
--*****************************************************************
-- Following are the input and output routing delays defined for this
-- design. This listing should be altered to reflect the design being
-- modelled.
--*****************************************************************
   GENERIC (wi_j1,wi_j2,wi_j3,wi_j4,wi_pe,wi_ci,wi_ud,wi_bd,wi_clk,
      wi_q1,wi_q2,wi_q3,wi_q4,wi_co: TIME := 0 ns;
--*****************************************************************
-- Following is a standard set of operating points. This listing
-- needs to be updated if additional operating points are supported
-- by the design.
--*****************************************************************
-- Valid operating points are:
--    prop_del=tnom, op_temp=27 degrees_c, op_volt=(5 v)
--    prop_del=tnom, op_temp=27 degrees_c, op_volt=(10 v)
--    prop_del=tmin, op_temp=-55 degrees_c, op_volt=(5 v)
--    prop_del=tmin, op_temp=-55 degrees_c, op_volt=(10 v)
--    prop_del=tmax, op_temp=125 degrees_c, op_volt=(5 v)
--    prop_del=tmax, op_temp=125 degrees_c, op_volt=(10 v)
--    prop_del=tzero, op_temp=27 degrees_c, op_volt=(5 v)
--*****************************************************************
      prop_del: OPERATING_SELECTION := TNOM;
      op_temp: TEMPERATURE := 27 degrees_c;
      op_volt: VOLTAGE := 5 v;
--*****************************************************************
-- The x_generation global variable controls 'X' state generation by
-- the "async_checker" and "sync_checker" modules when an
-- assert condition is violated.  If true, then 'X' states are generated
-- on assertion violations.  If false, then 'X' states will not be
-- generated.
--*****************************************************************
```

Figure 3.5.6-1 continued. The CD4029 Testbench.

```
            x_generation: BOOLEAN := TRUE;
--*****************************************************************
-- The m_generation variable is used to control reports provided by
-- assertion statements in the "sync_checker" and "async_checker
--  modules.   If the severity_level of the m_generation variable is less
-- than or equal to the serverity_level of an assertion violation, then
-- the assertion report will be generated.   Otherwise, the report will
-- not be provided.       Valid severity_level's are note, warning, error
-- or failure.
--*****************************************************************
            m_generation: SEVERITY_LEVEL := WARNING);
    END cd4029_tst;
ARCHITECTURE arch_waves_app_tb OF cd4029_tst IS
--*****************************************************************
-- The model under test will need to be declared as a component
-- and instantiated in the body of the testbench.
--*****************************************************************
    COMPONENT cd4029_eds
        GENERIC(wi_j1,wi_j2,wi_j3,wi_j4,wi_pe,wi_ud,wi_bd,wi_ci,wi_clk,
            wi_q1,wi_q2,wi_q3,wi_q4,wi_co:TIME:=0  ns;
            user_operating_point:  POINT:=(selection=>prop_del,
            temp=>op_temp,    SUPPLY=>(0=>op_volt));
            x_generation: BOOLEAN := TRUE;
            m_generation: SEVERITY_LEVEL := WARNING);
        PORT(bd,pe,ci,ud,clk,j1,j2,j3,j4: IN STD_LOGIC;
            co,q1,q2,q3,q4: OUT STD_LOGIC;
            vdd,vss:IN STD_LOGIC);
            END COMPONENT;
    FOR ALL:cd4029_eds USE ENTITY work.cd4029_eds;
--*****************************************************************
-- The following signals are used for stimulus/response information
-- provided from the vector file through the waveform generator.
-- These signals are based upon the design being tested.
--*****************************************************************
    SIGNAL w_j1,w_j2,w_j3,w_j4,w_pe,w_ci,w_ud,w_bd,w_clk,w_q1,w_q2,
        w_q3,w_q4,w_co,w_vdd,w_vss:STD_LOGIC;
--*****************************************************************
-- The following signals are used for response information from the
-- model being tested. These signals are required for each of the
-- model outputs.
--*****************************************************************
    SIGNAL  p_q1,p_q2,p_q3,p_q4,p_co:STD_LOGIC;
    SIGNAL waves_data:WAVES_PORT_LIST;
    SIGNAL errcnt:INTEGER:=0;
    SIGNAL filend:BOOLEAN;
    CONSTANT  user_operating_point:POINT:=(selection=>prop_del,temp=>op_temp,
        SUPPLY=>(0=>op_volt));
--*****************************************************************
-- It is only necessary to change this function if a different
```

Figure 3.5.6-1 continued. The CD4029 Testbench.

```
-- VALUE_DICTIONARY is used.
--****************************************************************
   FUNCTION to_1164 (value:LOGIC_VALUE) RETURN STD_LOGIC IS
      BEGIN
      CASE value IS
         WHEN ui => RETURN 'U';
         WHEN ud => RETURN 'X';
         WHEN lo => RETURN '0';
         WHEN hi => RETURN '1';
         WHEN hz => RETURN 'Z';
         WHEN wl => RETURN 'L';
         WHEN wh => RETURN 'H';
         WHEN lo_in => RETURN '0';
         WHEN hi_in => RETURN '1';
         WHEN hz_in => RETURN 'Z';
         WHEN wl_in => RETURN 'L';
         WHEN wh_in => RETURN 'H';
         WHEN lo_out => RETURN '0';
         WHEN hi_out => RETURN '1';
         WHEN hz_out => RETURN 'Z';
         WHEN wl_out => RETURN 'L';
         WHEN wh_out => RETURN 'H';
         END CASE;
      END to_1164;
   BEGIN
--****************************************************************
-- The waveform generator procedure name needs to be altered to
-- correspond with the design being tested.
--****************************************************************
   waves:PROCESS
      VARIABLE fileend:BOOLEAN;
      BEGIN
      cd4029wav(waves_data,fileend);
      filend<=fileend;
      WAIT;
      END PROCESS;
--****************************************************************
-- The translate process needs to be altered to reflect the signals
-- provided through the waveform generator.
--****************************************************************
   translate: PROCESS(waves_data)
      BEGIN
      w_j1 <= to_1164(LOGIC_VALUE'val(waves_data.wpl(1).l_value));
      w_j2 <= to_1164(LOGIC_VALUE'val(waves_data.wpl(2).l_value));
      w_j3 <= to_1164(LOGIC_VALUE'val(waves_data.wpl(3).l_value));
      w_j4 <= to_1164(LOGIC_VALUE'val(waves_data.wpl(4).l_value));
      w_pe <= to_1164(LOGIC_VALUE'val(waves_data.wpl(5).l_value));
      w_ci <= to_1164(LOGIC_VALUE'val(waves_data.wpl(6).l_value));
      w_ud <= to_1164(LOGIC_VALUE'val(waves_data.wpl(7).l_value));
```

Figure 3.5.6-1 continued. The CD4029 Testbench.

```
        w_bd   <=  to_1164(LOGIC_VALUEval(waves_data.wpl(8).l_value));
        w_clk   <=  to_1164(LOGIC_VALUEval(waves_data.wpl(9).l_value));
        w_q1   <=  to_1164(LOGIC_VALUEval(waves_data.wpl(10).l_value));
        w_q2   <=  to_1164(LOGIC_VALUEval(waves_data.wpl(11).l_value));
        w_q3   <=  to_1164(LOGIC_VALUEval(waves_data.wpl(12).l_value));
        w_q4   <=  to_1164(LOGIC_VALUEval(waves_data.wpl(13).l_value));
        w_co   <=  to_1164(LOGIC_VALUEval(waves_data.wpl(14).l_value));
        w_vdd   <=  to_1164(LOGIC_VALUEval(waves_data.wpl(15).l_value));
        w_vss   <=  to_1164(LOGIC_VALUEval(waves_data.wpl(16).l_value));
        END PROCESS;
--*****************************************************************
-- The model under test will need to be declared as a component
-- and instantiated in the body of the testbench.
--*****************************************************************
    z1:cd4029_eds GENERIC MAP(wi_j1,wi_j2,wi_j3,wi_j4,wi_pe,wi_ud,wi_bd,
            wi_ci,wi_clk,wi_q1,wi_q2,wi_q3,wi_q4,wi_co,
            user_operating_point,x_generation,m_generation)
        PORT MAP(w_bd,w_pe,w_ci,w_ud,w_clk,w_j1,w_j2,w_j3,w_j4,p_co,
            p_q1,p_q2,p_q3,p_q4,w_vdd,w_vss);
--*****************************************************************
-- The following process monitors the output signals provided from
-- the waveform generator. This process is triggered by a transaction
-- on any of these signals. When triggered, the process will compare
-- compare all outputs from the model with the expected outputs from
-- the vector file. Any errors will be logged. Alter this process to
-- operate on the corresponding output signals for the design being
-- tested.
--*****************************************************************
    errchk: PROCESS(w_q1'transaction,w_q2'transaction,
        w_q3'transaction,w_q4'transaction,w_co'transaction,filend)
        BEGIN
        IF filend THEN
            IF  errcnt>0  THEN
                ASSERT FALSE REPORT "testbench failed" SEVERITY FAILURE;
                ELSE ASSERT FALSE REPORT "testbench passed" SEVERITY NOTE;
                END IF;
            ELSIF NOW>0  ns  THEN
            ASSERT  w_q1=p_q1  REPORT "error in q1" SEVERITY WARNING;
            ASSERT  w_q2=p_q2  REPORT "error in q2" SEVERITY WARNING;
            ASSERT  w_q3=p_q3  REPORT "error in q3" SEVERITY WARNING;
            ASSERT  w_q4=p_q4  REPORT "error in q4" SEVERITY WARNING;
            ASSERT  w_co=p_co  REPORT "error in co" SEVERITY WARNING;
            IF  w_q1/=p_q1  OR  w_q2/=p_q2  OR  w_q3/=p_q3  OR  w_q4/=p_q4  OR
                w_co/=p_co   THEN       errcnt<=errcnt+1;
                END IF;
            END IF;
        END PROCESS;
    END  arch_waves_app_tb;
```

Figure 3.5.6-1 continued. The CD4029 Testbench.

3.5.6.3 Next, an appropriate internal signal set is declared. The convention employed here is that a "w_" prefix represents a signal which is generated by the waveform generator procedure, while signals prefixed "p_" are outputs generated by the model being tested.

```
SIGNAL w_j1,w_j2,w_j3,w_j4,w_pe,w_ci,w_ud,w_bd,w_clk,w_q1,w_q2,
    w_q3,w_q4,w_co,w_vdd,w_vss:STD_LOGIC;
SIGNAL p_q1,p_q2,p_q3,p_q4,p_co:STD_LOGIC;
```

3.5.6.4 The waveform generator procedure call is then updated to reflect the procedure used for this design.

```
waves:cd4029wav(waves_data,filend);
```

3.5.6.5 Now, the "translate" process is updated to reflect the I/O of the design. Since these I/O are all provided through the waveform generator procedure, the signals used are all prefixed with the "w_".

```
translate: PROCESS(waves_data)
    BEGIN
    w_j1  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(1).l_value));
    w_j2  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(2).l_value));
    w_j3  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(3).l_value));
    w_j4  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(4).l_value));
    w_pe  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(5).l_value));
    w_ci  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(6).l_value));
    w_ud  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(7).l_value));
    w_bd  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(8).l_value));
    w_clk <= to_1164(LOGIC_VALUE'val(waves_data.wpl(9).l_value));
    w_q1  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(10).l_value));
    w_q2  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(11).l_value));
    w_q3  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(12).l_value));
    w_q4  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(13).l_value));
    w_co  <= to_1164(LOGIC_VALUE'val(waves_data.wpl(14).l_value));
    w_vdd <= to_1164(LOGIC_VALUE'val(waves_data.wpl(15).l_value));
    w_vss <= to_1164(LOGIC_VALUE'val(waves_data.wpl(16).l_value));
    END PROCESS;
```

3.5.6.6 Here, the EDS component is instantiated. Notice that the input signals are those provided by the waveform generator, while the output signals (with the "p_" prefix") will be checked against the vector file.

```
z1:cd4029_eds GENERIC MAP(wi_j1,wi_j2,wi_j3,wi_j4,wi_pe,wi_ud,wi_bd,
        wi_ci,wi_clk,wi_q1,wi_q2,wi_q3,wi_q4,wi_co,
        user_operating_point,x_generation,m_generation)
    PORT MAP(w_bd,w_pe,w_ci,w_ud,w_clk,w_j1,w_j2,w_j3,w_j4,p_co,
        p_q1,p_q2,p_q3,p_q4,w_vdd,w_vss);
```

3.5.6.7 Finally, the "errchk" process is updated to perform the output signal comparison and to keep track of any errors encountered during the course of the simulation.

```
errchk: PROCESS(w_q1'transaction,w_q2'transaction,
    w_q3'transaction,w_q4'transaction,w_co'transaction,filend)
    BEGIN
    IF filend THEN
        IF errcnt>0 THEN
            ASSERT FALSE REPORT "testbench failed" SEVERITY FAILURE;
            ELSE ASSERT FALSE REPORT "testbench passed" SEVERITY NOTE;
            END IF;
        ELSIF NOW>0 ns THEN
        ASSERT w_q1=p_q1 REPORT "error in q1" SEVERITY WARNING;
```

```
          ASSERT w_q2=p_q2  REPORT "error in q2" SEVERITY WARNING;
          ASSERT w_q3=p_q3  REPORT "error in q3" SEVERITY WARNING;
          ASSERT w_q4=p_q4  REPORT "error in q4" SEVERITY WARNING;
          ASSERT w_co=p_co  REPORT "error in co" SEVERITY WARNING;
          IF w_q1/=p_q1 OR w_q2/=p_q2 OR w_q3/=p_q3 OR w_q4/=p_q4 OR
             w_co/=p_co  THEN     errcnt<=errcnt+1;
             END IF;
          END IF;
       END PROCESS;
```

3.6 Order of Analysis

Once all appropriate files have been updated to reflect the new design, the files are analyzed as listed below. Once all files have been analyzed, the testbench may be exercised (simulated). The listings are indented to reflect the dependence of one file on another.

3.6.1 The EDS Model

The order of analysis for the EDS model is as follows. This listing assumes that the IEEE-STD-1164 package already resides on the platform. If not, then this package will need to be analyzed first.

```
eia_567p.vhd
   i_driver.vhd
   o_driver.vhd
   async_c.vhd
   sync_c.vhd
4029cor.vhd
   4029ds_p.vhd
      4029eiap.vhd
         4029eds.vhd
```

3.6.2 The WAVES Dataset

The order of analysis for the WAVES dataset is as follows. This listing assumes that the WAVES_SYSTEM and WAVES_STANDARD packages have been previously analyzed. If not, then these packages will need to be analyzed before proceeding with the following files. Normally, the DUT package is shown as being analyzed after the "wav_frame" package, but before the WAVES_OBJECTS" package. This order is not necessary however since the DUT package is not dependent on any other packages.

```
4029dutp.wav
wav_logp.wav
   wav_int.wav
      wav_frmp.wav
         4029obj.wav
            4029wavp.wav
```

3.6.3 The Testbench

The testbench file is the last file analyzed for the model

```
4029tstt.vhd
```

This page intentionally left blank