DesignWorks™ for Windows

Version 3.1

Simulator and Associated Tools

Reference Manual

September 1994

406 - 960 Quayside Drive
New Westminster, B.C., Canada, V3M 6G2
(604) 522-6200

Table of Contents

Table of Contents

Chapter I - Introduction

The DesignWorks Digital Simulation package consists of a number of tools that work interactively within the DesignWorks MEDA™ Framework:

Simulator is the digital simulation engine and is required for all simulation functions.

SimLoad associates internal simulation models with parts as they are used.

Timing displays timing diagrams.

Test Vector used to enter and run test vectors.

Chapter Organization

The DesignWorks manual is divided into the following chapters:

Chapter/Title	**Comment**
Chapter II - Getting Started	Describes installation and initial startup.
Chapter III - Simulation Details	Describes simulation details including signals, devices, triggers and initialization.
Chapter IV - Schematic Changes to Support Simulation	Describes schematic related issues including signal naming, power and ground, simulation models, probing as well as changes to the Schematic tool to support simulation
Chapter V - The Simulator Tool	Describes control of the simulator via the simulator window.
Chapter VI - The SimLoad Tool	Describes automatic or manual loading of simulation models.
Chapter VII - The Timing Tool	Describes displaying of timing waveforms.
Chapter VIII - The Test Vector Tool	Describes the creation and execution of test vectors.
Chapter IX - Primitive Devices	Describes information on the simulation characteristics of the primitive device types.
Appendix A - Timing Text Data Format	Describes the text format used for clipboard operations in the Timing tool.
Appendix B - Simulation Attribute Fields	Describes the attribute fields used by the various simulation tools.

Appendix C - Device Pin Types	Describes the available pin types and their effect on simulation.
Appendix D - Primitive Device Pin Summary	Describes pin orders and options for primitive devices.

Notes Regarding Copyright

The DesignWorks Simulator software and manual are copyrighted products. The software license entitles you to use the software on a single machine, with copies being made only for backup purposes, unless a specific license extension has been purchased. Any unauthorized copying of the program or documentation is subject to prosecution.

Note regarding trademarks

A number of product trademarks are referred to in this manual. Full credit for these is given in the last section.

Chapter II - Getting Started

This chapter describes installation and initial startup of the DesignWorks Digital Simulator and associated tools.

NOTE: Since the DesignWorks package and Microsoft Windows are constantly being upgraded, there may be recent changes and additional information supplied in the release notes with this package. It is important that these notes be reviewed prior to installation as they may contain information that supersedes those given here.

It is assumed that the user is already familiar with general Microsoft Windows operations, such as copying files, creating directories, etc. If not, then please see the Microsoft Windows user's guide.

IMPORTANT: Make a backup copy

Before proceeding, make a backup copy and a working copy of the DesignWorks disks, then put the originals away for safekeeping.

Installation

For more detailed information on the files included with the system or directions for installing the current version, see the release notes supplied with the current software version.

Installing the Simulation Tools

The following table summarizes the code modules making up the DesignWorks Simulator package:

Tool	**Status**
Simulator	Required
SimLoad	Optional
Timing	Optional
Test Vector	Optional

In order to be loaded when DesignWorks starts up, these tools must be in the same directory as the DesignWorks Schematic Module (not the DesignWorks program directory), or in the directory specified in the TOOLSFOLDER statement in the DesignWorks initialization file, normally "Tools".

By default, the installer will install the program files into the correct location.

Installing the Device Models

Simulation models for the standard digital library components are supplied in the form of "external sub-circuit models", described elsewhere in this manual. These should normally be located in a directory called Models under the DesignWorks directory.

See the Installation Notes supplied with the package for specific installation instructions.

Memory Usage

Each tool occupies a certain amount of memory space regardless of whether it is being used or not. If DesignWorks is operating on a system with a minimal amount of memory, it may be desirable to keep unused tools in another directory where they will not be found and loaded by DesignWorks . When they are actually needed, they can be moved temporarily into the Tools directory.

While operating, the Simulator and associated tools can consume large amounts of memory depending upon the size of the design and the amount of data being displayed.

Chapter III - Simulation Details

This chapter describes simulation details including signals, devices, triggers and initialization.

General Information on Simulation

DesignWorks has the ability to perform a realistic simulation of any digital circuit. Obviously, though, any simulation of any system must be limited in detail and must make certain assumptions. In particular, when simulating digital circuits, it must be understood that real circuits are never completely "digital" in nature, and in fact have many "analog" properties which affect how they operate.

DesignWorks is primarily intended to assist with the logical design of a circuit, and does not take into account factors such as line loading, power supply noise, rise and fall times, output drive, etc. As more of these factors are taken into account, the simulation becomes slower and less interactive, which defeats the purpose for which the DesignWorks simulator was created.

Type of Simulation

DesignWorks performs a discrete simulation of the signal changes in a logic circuit, meaning that signal levels and time change only in steps, rather than continuously. The program does not attempt to analyze your circuit, but simply tracks signal level changes through the devices. Thus, circuits with feeback loops or other delay-dependent features will be simulated correctly as long as they don't rely on particular analog characteristics of devices.

The simulation is "event driven", an event being a change in level of a signal. Each time an event occurs, a list is made of all the devices whose inputs are affected by that event. Any other events occurring at the same time are similarly evaluated, and affected devices added to the list. A type-specific routine is then called for each device on the change list in order to determine what output changes are going to occur. These changes are added to the event list, their time of occurrence depending upon the device delay. No computation is performed for times when no event occurs, so that device delay settings and clock values have no effect on how fast the simulation is performed.

DesignWorks performs strictly a digital simulation. It does not take into account factors such as fan-out (i.e. the number of inputs connected to a given output), line length (capacitance), asymmetrical output drive, etc., except in as much as these affect delay time.

Simulation Memory Usage

When a circuit is opened or created by DesignWorks, the circuit data is retained completely in the memory of your machine. Since the total memory available is fixed (until you buy your next memory expansion!), this places some limits on circuit size and simulation.

Each time a signal changes state, an "event" record is created in memory. If the signal is not being displayed in the Timing tool, then this record is deallocated again after the signal change has occurred. If the signal is being displayed, then the record is retained in memory until that change has scrolled off the left-hand side of the Timing window. As a result, the memory used by event records will increase when the number of displayed signals is increased, the resolution of the timing display is decreased or the "retain time" setting is increased.

Time Units

DesignWorks uses 32-bit signed integer arithmetic to calculate all time values used in the simulation. It is usually convenient to think of these values as being in nanoseconds, but the actual interpretation is left up to the user.

The simulation will wrap if any time value approaches the 32-bit integer limit.

Signal Simulation Characteristics

Signal States

DesignWorks uses thirteen different device output states in order to track conditions within your circuit. These states can be broken into three groups, as follows.

Forcing States (denoted by suffix .F):

LOW.F

HIGH.F

DONT01.F

DONT0Z.F

DONT1Z.F

CONF.F

Resistive States (denoted by suffix .R):

LOW.R

HIGH.R

DONT01.R

DONT0Z.R

DONT1Z.R

CONF.R

High Impedance:

HIGHZ

Note that the Forcing/Resistive distinction is used only to resolve conflicts between multiple outputs connected to the same signal. The final value stored or displayed for a given signal line can only be one of five possibilities:

LOW

HIGH

DONT

CONF

HIGHZ

Description of States

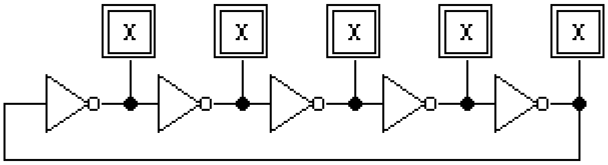
The High and Low states are the normal ones expected in a binary circuit, but are not sufficient to realistically simulate circuit operation, so the High Impedance, Don't Know and Conflict states are added. There will always be some cases where the simulation will not correctly mimic what would appear in a real circuit, and some of these cases are discussed below. In particular, if a circuit takes advantage of some analog property of a specific device, such as inputs that float high, known state at power-up, input hysteresis, etc., it will in general not simulate correctly.

High Impedance

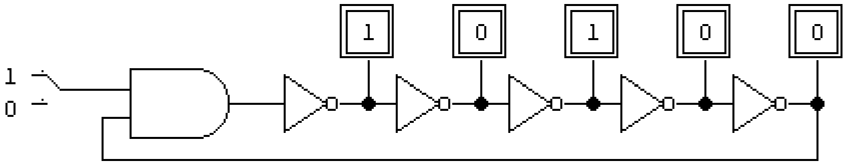
This state ("Z" on a binary probe) is used for cases when no device output is driving a given signal line. This may occur for an unconnected input, or for a disabled "three-state" or Open-collector" type device. If a device input is in the High Impedance state, it is treated as unknown for the purposes of simulation, even though in a real circuit the device may assume a high or low state, depending on the circuit technology used.

Don't Know

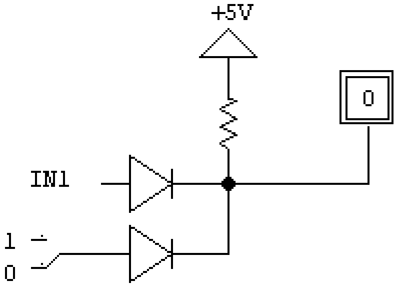
The Don't Know state is used in cases where the actual result in a real circuit would depend in the circuit technology used, on random chance, or on analog properties of the device not predictable using a strictly digital simulation. For example, if the following ring oscillator circuit is created in DesignWorks, all signals will be permanently unknown, since each depends on the previous one, which is also unknown. In actual hardware, this circuit may oscillate, or may settle into an intermediate logic level, which would not be defined in a digital circuit.



For the purposes of simulation, all circuits must have some provision for initialization to a known state. In most cases, circuits can be initialized using the Clear Unknowns command or by setting the initial value attribute, both described elsewhere in the manual. Alternatively, circuitry can be added to allow a reset to be done, as in the following modification to the ring oscillator:



A problem arises in simulating circuits with multiple open collector devices, such as a bus line, illustrated below:



In this circuit the upper device has an unconnected input at IN1 and therefore outputs a “Don’t Know” value. The lower device has a low input and therefore outputs a low value. In order to correctly resolve this situation the simulator needs to distinguish between a “Don’t Know” output from a normal “totem-pole” type output and a “Don’t Know” from an open collector, open-drain, or other single-drive output. In this case the upper device will produce a DONT0Z output which resolves correctly to a LOW on the output regardless of the state of IN1, using the rules described above.

Conflict

The "Conflict" state ("C" on a binary probe) results when two device outputs are connected and are of different or unknown state taking into account the rules described above.

State Display

The Timing tool displays the various signal states in different colors.

Stuck-at Levels

The DesignWorks simulator implements "stuck-at" levels to assist in setting initial simulation states, testing for faults, etc. When a signal is in a "stuck" state, it will not change state, regardless of changes in devices driving the line.

When the stuck status is set, the signal will retain the value it had at the time until some user action forces a change. When the stuck status is removed, the signal will return to the value determined by the devices driving the line.

Setting Stuck Levels

A signal can be placed in a Stuck High or Stuck Low state by any of the following means:

- applying the name "0" or "1" to the signal.
- typing "H" or "L" while viewing the signal value with the probe tool.
- using the Stick High or Stick Low control in the Simulators Stick Signals dialog.
- setting a "stuck" option in the Test Vector or other external simulation tool.

Each of these methods is described in more detail in the relevant sections of this manual.

Clearing Stuck Levels

The stuck status can only be cleared by one of the following user actions:

- typing the spacebar while viewing the signal using the probe tool.

- clearing the "stuck" state using the Simulators Stick Signals.
- clearing the "stuck" state using the Test Vector or other external simulation tool.

Resolution of Multiple Device Outputs

The DONT0Z and DONT1Z values are used primarily to handle cases of open collector or open emitter devices with unknown inputs (see more information below). Most other types of devices produce the DONT01 output when a value cannot be calculated.

In cases where two or more device outputs are connected together and each one drives the line with a different value, the following rules are used to resolve the actual value on the line.

- the forcing/resistive distinction is only used to resolve outputs from multiple devices. The final value used for display and simulation purposes is one of the forcing values or HIGHZ.
- a forcing drive always overrides a resistive drive or HIGHZ (i.e. the signal takes on the value of the forcing drive, ignoring all resistive drives and HIGHZs).
- a resistive drive always overrides HIGHZ.
- DONT0Z.F and LOW.F produce LOW.
- DONT1Z.F and HIGH.F produce HIGH.
- any other combination of conflicting forcing drives produces CONF.
- DONT0Z.R and LOW.R produce LOW.
- DONT1Z.R and HIGH.R produce HIGH.
- any other combination of conflicting resistive drives produces CONF.

Resistive vs. Forcing Drive

All primitive devices in DesignWorks output a forcing drive level except for the Resistor primitive device. The function of the Resistor device is to convert a forcing drive on one side into a resistive drive on the other. This can be used to modify the output of any existing device type by placing a resistor in series with it. Note that DesignWorks does not model analog properties of devices, so the resistor does not have a resistance value in the analog sense. In particular, there is no interaction between resistor and capacitor symbols to produce delay in lines. The delay effect can be simulated by setting a delay value for the resistor.

Probe Tool

The Probe tool is used to interactively examine and change values on individual signals and pins in the circuit schematic. When the probe tip is clicked and held on a signal or pin, the cursor will show the current value on the signal or pin and track changes that occur as the simulation progresses.

See "Probing a Signal" (page 31) in Chapter IV - Schematic Changes to Support Simulation for more information.

Busses

Busses (i.e. groups of signals represented by a single line on the schematic) have no particular significance to the simulator. The value of a bus is completely determined by the values of the individual signals it contains. The simulator performs no operations on the bus itself.

NOTE: A bus can be displayed in the Timing window using the Add To Timing command, but this is really equivalent to displaying all the internal signals individually and then grouping them.

Bus Pins

Bus pins, like busses, have no particular significance to the simulator. The value of a bus is completely determined by the values of the individual pins it contains. The simulator performs no operations on the bus pin itself. Bus pins are not supported on primitive device types.

Device Simulation Characteristics

Device and Pin Delay

Primitive Device Delay

Primitive devices (i.e. those with a program-defined simulation model) have a single delay value which can be set to any integer value from 0 to 32767. This delay is applied when any input change causes any output change. In addition, a pin delay in the range 0 to 32767 can be set on any input or output pin. Pin delays can be used to set arbitrary path delays through the device. See more information on pin delays below.

The initial delay value is set to 1 when the device is created, but this can be changed later using the Parameters command in the Options menu in the Simulator window. This delay applies whenever any input change causes an output change. There is no provision in the built-in simulation models for different delay values on low-to-high and high-to-low transitions. The Clock and I/O devices have no delay characteristic.

See “Working With Hierarchical Blocks” (page 25) in Chapter IV - Schematic Changes to Support Simulation for more information on settings of delays in hierarchical designs.

Sub-Circuit Device Delay

Sub-circuit devices inherit their delay characteristics from their internal circuit and have no "device delay" characteristic of their own. Simulation Parameters cannot be directly used on a sub-circuit device, although pin delays can be set separately on each instance of a sub-circuit device to customize path delays.

Pin Delays

Any input or output pin on any device (including port connectors and sub-circuit devices) can have a pin delay associated with it. Pin delays normally default to 0 time units, but can be in the range 0 to 32767.

A pin delay acts like a "buffer" device with the given delay inserted in line with the pin. On an input pin, the device simulation model will not see a change in signal value until after the pin delay has elapsed. On an output pin, the pin delay is added to the overall device delay for any changes scheduled on that pin.

Effect of zero delay

A delay value of zero is permitted in a DesignWorks device, but this setting should be used only with an understanding of how the simulation is implemented as it can result in unexpected side-effects.

On a given pass through the simulation routine, all the events on the list which occur at the current time are scanned and then the new outputs for all affected devices are calculated. If any of these devices has a zero delay setting then this will result in more changes being placed on the event list at the current time. However, all these changes emerging from zero-delay devices will not be evaluated until the next pass through the simulator. This is done to allow for user interaction with the simulation.

Stepping interactively through a circuit with zero-delay elements results in all these value changes to be updated on the screen, even though "simulation time does not advance. If a signal changes value then reverts to its original state within the same time step, this will be displayed as a zero-width spike in the timing window.

If a zero-delay feedback loop exists in a circuit, the signal changes will be simulated and any probes on the schematic will be updated at each pass through the simulator. However the events at the head of the list will always have the same time value associated with them and the simulated time will never advance. Updating of the timing window will stop until some delay is inserted into the loop.

Where Delays are Stored

Delays are stored as decimal integers in text form in attribute fields associated with each device or pin. For devices, the delay field is called "Delay.Dev", for pins it is "Delay.Pin" An empty or invalid string will be interpreted as the default value, usually 1 for devices and 0 for pins.

Some special-purpose devices, such as the Clock, One Shot and SetupHold primitive devices take two delay characteristics. In this case, two integers separated by a comma should appear in the Delay.Dev field.

Note: The Simulation Parameters dialog should be used to control these attribute fields rather than the Schematic tools attribute dialog.

See each of these primitive types in Chapter IX - Primitive Devices (page 107) for more information.

A number of additional fields are pre-defined in DesignWorks for storage of alternate delay values. These fields are not used directly by the simulator, but are only provided as a convenient place to store these values. The Save to Attribute and Set from Attribute functions in the Simulation Parameters dialog can be used to copy values to or from alternate delay fields.

The following alternate fields are pre-defined.

Field Name	**Function**
Delay.Dev.Typ Delay.Pin.Typ	Typical device/pin delay
Delay.Dev.Min Delay.Pin.Min	Minimum device/pin delay

Delay.Dev.Max Delay.Pin.Max	Maximum device/pin delay

Device Storage State

In DesignWorks, primitive storage devices (e.g. flip-flops, counters and registers) do not store their current state internally. The device state is completely determined by the values on the signals attached to the output pins. Thus, the following factors will affect the operation of these devices:

- Conflicting or overriding values on the output signals (e.g. a stuck state) will override the last device state calculated by the model.
- Device and pin delays will influence the calculation of a new device state. E.g. if the period of a clock applied to a counter is less than the total delay through it, an erroneous count sequence will result.

If desired, this behavior can be modified by placing the primitive devices in a sub-circuit device and setting appropriate pin types and delays on the parent device to "buffer" the outputs.

NOTE: These comments do not apply to RAM or bidirectional switch primitives, both of which store internal state information independent of the values of the attached signals.

See "Working With Hierarchical Blocks" (page 25) in Chapter IV - Schematic Changes to Support Simulation for more information.

Input Signal Values

Unless an alternate input value mapping is specified (see below), for all device types except switches, signal values "High-impedance" and “Conflict" are treated as “Don't Know” when applied to a device input. When a device is first created, all input signals take the “High Impedance" state and outputs are set depending on their type, normally to the "Don't Know" state. Thus, an unused input pin will appear as an unknown input to a device, which may affect its output level.

As with real circuits, all unused inputs should be connected to a high or low level as appropriate. This can be done by naming the pin signal either "0" or "1", using a power or ground symbol, or by using a pullup resistor to set a high level.

Note: “0” and “1” only work for simulation and are NOT equivalent to power and ground for netlist purposes.

Input Value Mapping

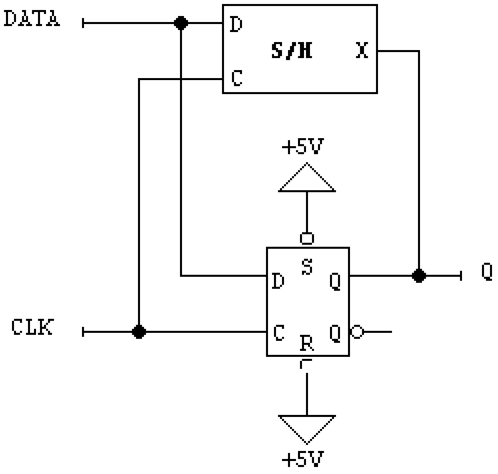
An alternate input value can be specified for a design by placing a specially-formatted string in the Sim.InputMap attribute field for the design. This is intended primarily to model specific logic families that have a known response to high-impedance inputs. The mapping applies only to primitive devices and is global to the design that it is specified in. Note that this mapping does not occur at inputs to sub-circuit devices but does affect the primitives that they contain.

> *See Appendix B - Simulation Attribute Fields (page 135) for more information on the format of the Sim.InputMap field.*

Setup and Hold Times

All standard device types having an edge-triggered clock, such as the D and JK-flip-flops, register and shift register, have an effective setup time of 1 unit and a hold time of zero units. That is, if the data and clock inputs change simultaneously, the old value of the data input will be used.

Setup and hold times can be checked by attaching a "SetupHold" primitive device to the inputs of the clocked device to be checked, as follows:



The SetupHold device puts out a highZ value until a setup or hold violation occurs when it switches to a “Don’t Know” state. Thus, the output of the SetupHold device can be paralleled with the flip-flop so that the output line will enter a conflict state when an error occurs.

See Chapter IX - Primitive Devices (page 107) for more information on the SetupHold device.

Device Pin Types

Every device pin has a characteristic known as its *pin type*, e.g. input or output. The pin type is set when the part entry in the library is created and cannot be changed for individual device pins on the schematic. Correct pin type settings are crucial to correct and efficient operation of the simulator.

The pin type is used by the simulator to determine the direction of signal flow and which output values are allowable on a given output pin.

See Appendix C - Device Pin Types (page 141) for more information on the available pin types and how they affect the simulation.

See Chapter X - Device Symbol Editing in the DesignWorks/Schematic Reference Manual for procedures on setting pin types when creating a symbol.

Device Pin Inversion

The logic of any pin on any device can be inverted by placing a non-empty value in the Invert.Pin attribute field of the pin. When this is done any value passing into or out from that pin will be inverted. This applies to primitive types as well as sub-circuit devices. The following table summarizes the level mappings that occur.

External Signal Value	Internal Signal Value
LOW.H	HIGH.H
LOW.L	HIGH.L
HIGH.H	LOW.H
HIGH.L	LOW.L
All others	Unchanged

NOTES:

1) The logical inversion of the pin is completely independent of the graphical representation of the pin. I.e. using the "inverted pin" graphic in the DevEditor tool does not invert the pin logic in the simulator. The Invert.Pin field must be set to have this effect.

2) Although pin inversion can be specified independently for each device on the schematic we do not recommend modifying these settings after a device has been placed on the schematic. This can create the confusing situation of two devices with the same name and symbol but different logical characteristics.

See also:

• “Pin Delays and Inversion” (page 28) in Chapter IV - Schematic Changes to Support Simulation for information on pin inversion in sub-circuit blocks.

• Chapter IX - Primitive Devices (page 107) for information on how pin inversion can be used with specific primitive types.

• Appendix D - Primitive Device Pin Summary (page 147) for more information on pin usage and pin inversion.

• Chapter X - Device Symbol Editing in the DesignWorks/Schematic Reference Manual for procedures on setting pin attributes when creating a symbol.

Triggers

The DesignWorks Simulator has a powerful trigger capability analogous to a "word recognizer" on a logic analyzer. Any number of triggers can be set to perform various actions when certain time and signal value conditions are met. These actions include:

- Drawing reference lines in the timing window.
- Stopping the simulator.
- Enabling or disabling the timing display.
- Generating a beep from the computers speaker.
- Enable checking for another trigger.

Any number of triggers can be set up with different activation conditions. For display purposes, triggers are named T1, T2, T3, etc. To assist in setting up complex time-related conditions, trigger T_N can be used to enable T_{N+1}.

> *See Chapter V - The Simulator Tool (page 37) for more information on setting triggers.*

Simulation Clearing and Initialization

The DesignWorks Simulator provides a number of mechanisms to assist in setting initial values and restarting a simulation.

The Clear Simulation Operation

The Clear Simulation operation is invoked by clicking on the Reset control in the Simulator window. This operation performs the following steps:

- Other tools (such as Timing and Test Vector) are notified and perform their own processing.
- All signal change events on the queue are disposed of, whether pending or historical.
- Any clocks in the design are re-initialized.
- If any signal or pin initial values are specified, they are set up. See below for information on setting initial values.
- All devices are queued for immediate reevaluation.

The Clear Unknowns Operation

The Clear Unknowns operation is invoked by clicking on the Clear X control in the Simulator window. This operation performs the following steps, stopping as soon as all unknown states are removed from the design:

- Any pending signal change that would result in an unknown state is removed from the queue.
- Any primitive type with storage capability (i.e. flip-flop, register or counter) that has a “Don't Know” output value is cleared, either to its specified initial value (if any) or to zero.
- The input mapping for the design is temporarily set so that all unknown input values map to zero.
- A single device that currently has an unknown output state is randomly selected and queued for reevaluation. The simulator is cycled repeatedly as long as the number of unknown states in the design decreases. This step is then repeated until the number of unknowns ceases to diminish.
- The input mapping for the design is restored to its original state.

If this operation does not clear the design to an appropriate state, refer to the other techniques discussed below.

NOTE: Designs with "hard" unknowns, such as unconnected inputs or conflicting outputs will **not** be successfully cleared by this procedure. All device inputs should be specified to a known value if not driven by other devices.

Setting Initial Values

Initial values can be specified for signals and pins which will be applied by the Clear Simulation and Clear Unknowns operations, as described in the preceding sections.

For both object types, the initial value is entered into an attribute field, either Initial.Sig or Initial.Pin. The allowable values consist of a single character chosen from the following table.

Character	**Value**
0	LOW
1	HIGH
Z	HIGHZ
X	DONT01

All other values will be ignored.

NOTES:

1) It is left completely to the user to decide if the specified initial values make sense. E.g. No checking is done to determine if a given device output value is the reasonable result of the device's current inputs.

2) Devices do not have initial value settings since their values are completely determined by the state of their output pins. See Pin Initial Values below.

Signal Initial Values

The initial value for a signal is stored in the Initial.Sig attribute field using the format described in the previous section. When a Clear Simulation operation is invoked, the initial value specified is placed on the signal without regard for the current output levels of devices driving the signal. The given value will stay on the signal until some device driving the signal changes state or some other user action changes it.

NOTE: If a pin initial value is specified for any output pin driving the signal, the signal value will be overridden.

Pin Initial Values

The initial value for a pin is stored in the Initial.Pin attribute field using the format described earlier. Initial values can only be specified for output or bidirectional pins and will be ignored on input pins.

When a Clear Simulation operation is invoked, the initial value specified is placed on the pin without regard for the current inputs affecting the device. The given value will stay on the pin until the device model schedules a state change or some other user action changes it.

This page has been intentionally left blank.

Chapter IV - Schematic Changes to Support Simulation

This chapter describes schematic related issues including signal naming, power and ground, simulation models, probing as well as changes to the Schematic tool to support simulation

Schematic Simulation Issues

Hierarchy Mode

The DesignWorks Schematic tool supports three hierarchy modes: Flat, Physical and Pure. The following table summarizes these modes in terms of their significance for simulation.

Mode	**Comment**
Flat	Despite its name, Flat mode does allow devices to have internal circuits for simulation purposes, although it will discourage access to them. Flat mode also changes the way device names are assigned and reports are generated, but these issues do not affect the simulator.
Physical	This mode allows unrestricted access to internal circuits for simulation purposes.
Pure	Simulation is not supported in Pure mode as only a single definition copy of device type data is kept in memory, regardless of the number of times the device is used. Therefore, there is nowhere to store the simulation states associated with each instance.

Working With Hierarchical Blocks

The simulator does not impose any new rules on working with hierarchical blocks, but there are some effects of editing a design with active simulation that should be noted.

See Chapter VII - Hierarchical Design in the DesignWorks/Schematic Reference Manual for additional information.

Editing an Open Internal Circuit

A number of issues arise if the same device type or hierarchical block appears multiple times in a design and one copy of the type or block is opened for editing (i.e. using the Push Into command or by double-clicking on the device):

- The Schematic tool creates a separate, temporary type definition for the open device when it is opened. Any simulation values viewed or changed, or any circuit changes made, will apply only to that one device instance while it remains open.
- When the open internal circuit is closed, the action taken depends on edits that have taken place. If no edits have been made or changes have been made only to instance data (i.e. simulation values or attributes marked as "keep with instance"), then the other blocks of the same type will not be affected by the close. If any definition data (i.e. any graphical or structural change to the circuit or any definition attribute change) has changed then a dialog is displayed prompting for the action to take. If the Update control is selected then all instance data (i.e. signal values, etc.) in other blocks of the same type will be lost. Instance data will be completely replaced by the values from the edited block.

The Port Interface

The connection between a pin on a parent device symbol and the corresponding signal in the internal circuit is quite complex from a simulation standpoint. In order for this connection to act like a "hard wire" between the two levels, the following conditions must be met:

- the *pin type* on the parent device symbol must be "bidirectional".
- the *pin type* of the corresponding port connector in the internal circuit must be "bidirectional".
- the *pin delays* on both the pin on the parent device and the pin on the port connector must be zero.
- no *pin inversion* must be specified either on the parent device pin or the port connector pin.

Any other combination of settings will result in some degree of isolation or "buffering" between the two levels. I.e. The observed signal value on the signal in the internal circuit may be different from that on the parent pin.

NOTE: When a symbol is created in the DevEditor tool, all pins default to type "input", i.e. they will not drive any attached signal. When creating a hierarchical block symbol for simulation purposes, the pin types must be set to appropriate values.

The effects on these various settings are summarized in the following sections.

Parent Device Pin Type

Any signal value driven out of a parent pin by an internal circuit may be translated according to the pin type on the parent device. These effects are summarized in the following table.

Pin Type	**Effect**
Input	This will prevent that pin from ever driving the attached signal, regardless of drives in the internal circuit.
Output / Three-state	This will pass the sum of the internal drives up to the parent pin without any translation. Signal value changes on the signal attached to the parent pin will not be passed to the internal circuit.
Open collector / Open emitter	Any drive level from the internal circuit will be translated according the capability of the pin type. See Appendix C - Device Pin Types (page 141) for more details.
Bidirectional	All changes on the internal signal are passed to the parent pin and vice versa.
Other types	Other types, such as Tied High and Tied Low are not recommended.

NOTE: Although it may be tempting to set all pins to "bidirectional", this is not recommended since it significantly increases simulation overhead and increases the difficulty of isolating circuit drive problems.

Port Connector Pin Type

The pin type on the port connector is also used to translate the value of any incoming signal changes, in a manner similar to the parent pin type. Normally, the pin type setting on a port connector should complement the setting of the parent pin as follows:

Parent Pin Type	**Port Connector Name**	**Port Connector Pin Type**
Input	Port In	Output
Bidirectional	Port Bidir	Bidirectional
All others	Port Out	Input

Other settings on the port connector pin are not recommended.

Pin Delays and Inversion

The normal pin delay and inversion settings can be applied to the port interface. A non-empty value in the Invert.Pin attribute field will cause any signal values passing in either direction to be inverted. An integer value in the Delay.Pin attribute will cause the specified delay to be inserted in line with level changes passing in either direction.

NOTES:

1) It is recommended that pin delay and inversion settings be applied only to the pin on the parent device and not to the port connector in the internal circuit. Attribute settings on the port connector are more difficult to verify and edit since the port connector is a "pseudo-device" and some schematic editing operations will be disabled.

2) Changes made in Invert.Pin and Delay.Pin attribute field after a device has been placed on the schematic, will affect only that one device instance. Default values can be set in these attribute fields when the symbol is created in DevEditor tool.

Power and Ground Connectors

Power and Ground connector symbols do not have any inherent simulation signal drive unless their pin type has been set to "Tied High" or "Tied Low", as appropriate. Most of the positive-supply symbols provided with this DesignWorks have "tied high" settings, while others will be "tied low". The symbols provided with older DesignWorks releases may not have any drive setting, resulting in a high impedance level on these signals. This can be remedied by either:

- Replacing any one or all of the ground or power symbols with one that has the appropriate setting.
- Forcing a Stuck High or Stuck Low level onto the signal using the probe tool, Test Vector tool or other external means. Note that because all like-named ground or power segments are logically connected, this only needs to be done on any one segment.

Special Signal Names 0 and 1

The signal names "0" and "1" are recognized by the simulator as special. If any signal is named "0", it will be given a Stuck Low value. If "1" is found, it will be given a Stuck High value. These values can be cleared or changed using the probe tool, if desired.

See “Probing Signals” (page 31) in Chapter IV - Schematic Changes to Support Simulation for more information.

Simulation Models

In order to completely simulate a design, every symbol must have an associated simulation model. In DesignWorks, simulation models can take one of the following forms:

- Primitive Devices - These types have "hard-wired" program code to evaluate input and output changes. They include the gates, flip-flops and other devices as described in Chapter IX - Primitive Devices, as well as the user-definable PROM and PLA primitives.
- Sub-circuit Devices - The simulation function of a sub-circuit device is completely determined by its internal circuit (except for the addition of pin delays and inversion). The definition of a device sub-circuit can be stored with the part in a library (referred to as an "internal sub-circuit") or can be stored in a separate design file on disk (an "external sub-circuit"). The sub-circuit itself can contain any combination of primitive devices or other sub-circuits (except itself, of course!) nested to any desired depth.

Whenever any device type is to be simulated, all information about the device must be loaded into memory. Unless internal circuits or code models are explicitly purged from the design, they will become a permanent part of the design and will be saved with the file.

Primitive Devices on the Schematic

Primitive devices are provided in various primitive libraries and can be used at any time as part of a schematic, whether or not the simulator is installed. However, these libraries are not intended to match any real logic families and do not have any part name, pin number or packaging information associated with them.

See Chapter IX - Primitive Devices (page 107) for more information on creating and using primitive types.

Simulation Pseudo-Devices

The simulation pseudo-devices (i.e. those in the Primitive I/O library) are handled specially by the Schematic tool. In general, the symbol, pin types or other characteristics of these devices cannot be modified. In addition, they are treated differently from normal device symbols in the following ways:

- By default, these devices are flagged "omit from report", meaning that they will not appear in any netlist or bill of materials produced by the Report tool. This setting can be changed using the Schematic tool's Get Info command.
- These symbols will not be assigned a name when placed on a schematic. A name can be manually assigned, if desired.
- The switch and keyboard types respond to a normal mouse click by changing state, rather than being selected. To select one of these devices hold the shift key down while clicking on it.

Using External Sub-Circuit Models

Most of the models for industry-standard device types (e.g. 74XX, 4000, etc.) are provided as external sub-circuits. These models are simply design files containing an equivalent circuit for the device. The circuits are designed to match with the symbol libraries provided with the Schematic package.

External sub-circuit models can be loaded either interactively, i.e. as each part is used, or in batch mode after the design is created. Both of these functions are provided by the SimLoad tool.

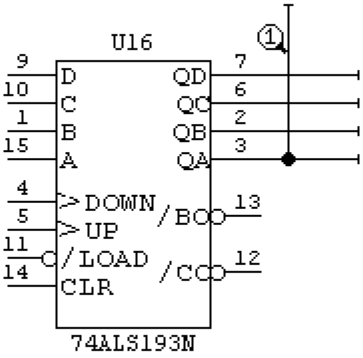
> *See Chapter VI - The SimLoad Tool (page 53) for more information.*

Probing Signals / Schematic Tool Palette

When the Simulator Tool is installed, a probe tool in available in the palette window displayed by the Schematic tool. Selecting the Probe tool places the schematic in Probe mode, allowing signal and pin values to be displayed and injected. When the probe tip is clicked and held on a signal or pin, the cursor will show the current value on the signal or pin and track changes that occur as the simulation progresses.

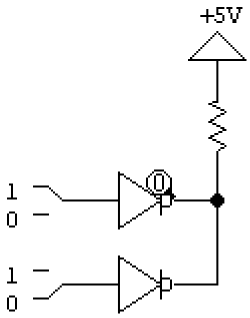
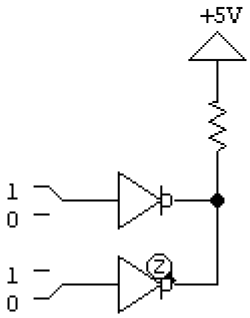
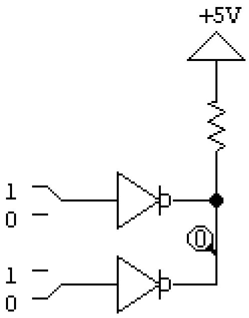
Probing a Signal

Only the signal under the cursor at the time of the click is examined. Moving the mouse while the button is pressed does not change the signal being viewed.



Probing a Pin

If the probe tip is clicked on a device pin close to the device body, the probe shows the driving level of that pin, rather than the state of the attached signal. This can be used to resolve drive conflicts in multiple drive situations, as in the following example:

Pin Drive on Upper Device	**Pin Drive on Lower Device**	**Combined Signal Value**
		

NOTE: The probe display does not distinguish between low and high drive levels.

Injecting a Value Using the Probe Tool

While the mouse button is held, press the appropriate key to inject a new value onto a signal:

Key	Injected Value
0	LOW.F
1	HIGH.F
X	DONT01.F
C	CONF.F
Z	HIGHZ
L	LOW.F stuck
H	HIGH.F stuck
space	Unstick

If a stuck value is forced onto a signal, it will not change state until the stuck value is cleared by some user action, regardless of device outputs driving the line. If a non-stuck value is forced, the signal value will revert to its appropriate new level when any change occurs on a device output driving the line.

The spacebar "unstick" command causes the signal to revert to its driven value.

Note: values cannot be injected onto pins.

See “Stuck-at Levels” (page 10) in Chapter III - Simulation Details for more information.

Schematic Menu Changes

When the simulator is installed, two new menu items appears in the Schematic menu.

Simulation Menu

Parameters...

This command displays the Simulation Parameters dialog and is enabled when at least one device or pin is selected in the schematic.

> *See “Simulator Procedures / Simulation Parameters” (page 43) in Chapter V - The Simulator Tool for more information.*

Timing Menu

Add Automatically

When signals are created or named they will be automatically added to the Timing window (if displayed) when this command has a check mark beside it.

Add to Timing

This command adds all selected signals to the Timing window (if displayed).

Add as Group

This command adds all selected signals as a group to the Timing window (if displayed).

> *See Chapter VII - The Timing Tool (page 61) for more information.*

This page has been intentionally left blank.

Chapter V - The Simulator Tool

Introduction

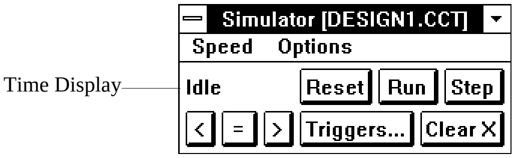
The chapter describes control of the simulator via the Simulator tool.

Starting the Simulator

By default, the simulator starts running as soon as a circuit is loaded. To display the simulator control window, select “Simulator” from the Tools menu in the DesignWorks window

Simulator Window

The simulator window is a floating window. It is displayed when the Simulator tool is selected from the “Tool” menu in the DesignWorks window OR when a new timing window is created. It can be hidden by selecting “Close” from the system menu in the top left corner of the window.



Item	Comment
Time Display	The status area in the top left corner of the simulator window displays the current simulation time as the simulator progresses or “Idle” if the simulator is not running.
Reset	Removes all scheduled signal events, sets all devices, signals and pins to their specified initial values (if any) and recalculates output values for all circuit elements. In addition, it clears the timing window (if open) and resets the Test Vector tool (if open).
Run	Causes the simulator to execute at the fastest possible speed.
Step	Causes the simulator to execute one time step. If the simulator is running then this control causes the simulator to stop after the next event time.

Clear X	Clears all flip-flop, counter and register primitives to the zero state and attempts to remove all unknown signal values from the circuit. Certain circuit conditions may prevent signals from being placed in a known state such as unconnected inputs that have not been set to a known level, storage devices, such as RAMs that have an unknown stored value and/or any simulation models that do not produce a known output when all inputs are known.
Triggers...	Displays the trigger control dialog. See Simulator Procedures / Triggers (page 40) for more information
< (Zoom In)	Increases the horizontal display resolution in the Timing window (if it is displayed).
> (Zoom Out)	Decreases the horizontal display resolution in the Timing window (if it is displayed).
= (Clear Zoom)	Sets the horizontal display resolution in the Timing window (if it is displayed) to its initial defaults.

Simulator Window Menu Commands

Speed Menu

The Speed menu is used to control the simulation speed, i.e. the amount of delay inserted between simulation steps. Simulation speed can be set individually for each open design.

Stop

This command stops the simulator immediately. No simulation processing is done when the simulator is in this state.

Run

This command starts the simulator at the fastest possible speed.

Single Step

This command simulates one time step. To perform the single step, the simulator looks at the time value associated with the next signal change event in the queue, simulates the effect of that and all following events scheduled at the same time, then returns to the stopped state. The actual time value of a single step depends on the nature of the circuit.

Other Simulation Speeds

The intermediate speed settings between Stop and Run insert various amounts of delay between executing successive simulation time steps. These can be used to slow the simulation progress for convenient observation.

Options Menu

Parameters...

This command displays the Simulation Parameters dialog and is enabled when at least one device or pin is selected in the schematic.

> *See "Simulator Procedures / Simulation Parameters" (page 43) in Chapter V - The Simulator Tool for more information.*

Simulation Scope...

This command displays the Simulation Scope dialog.

> *See "Simulator Procedures / Simulation Scope" (page 50) in Chapter V - The Simulator Tool for more information.*

Stick Signals...

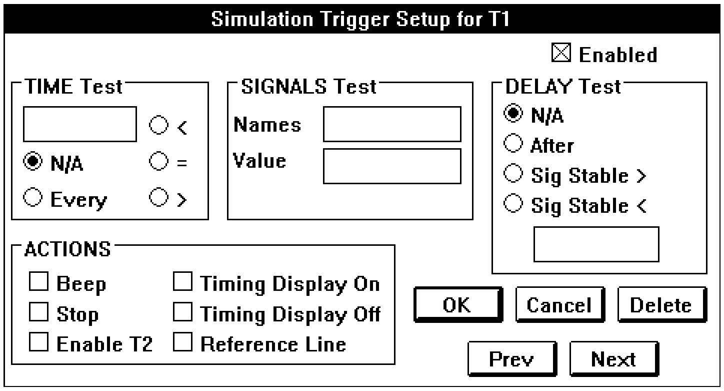
This command displays the Stick Signals dialog.

> *See "Simulator Procedures / Stick Signals" (page 51) in Chapter V - The Simulator Tool for more information.*

Simulator Procedures

Triggers

Selecting the triggers control in the Simulator window displays the following dialog:



Simulation Trigger Setup for T1

☒ Enabled

TIME Test: ● N/A ○ Every ○ < ○ = ○ >

SIGNALS Test: Names, Value

DELAY Test: ● N/A ○ After ○ Sig Stable > ○ Sig Stable <

ACTIONS: □ Beep □ Stop □ Enable T2 □ Timing Display On □ Timing Display Off □ Reference Line

Trigger Conditions

A trigger is activated when all three sets of conditions are met:

- The time condition, i.e. the current simulator time value is less than, equal to, greater than, or a multiple of, a given value.
- Signal value condition, i.e. one or more signals are at specified levels.
- The delay condition, i.e. the trigger is activated after a certain delay, or the signal condition exists for greater or less than a specified amount of time.

Trigger Enabling

When the "Enabled" switch is on, the trigger is "armed" and the selected actions will take place as soon as the trigger's conditions are met. If this switch is off, this trigger is disabled until enabled by the previous trigger or by this switch being selected.

Time Test Controls

The controls related to the time condition are summarized in the following table.

Control	Comment

Time Value	Enter the time value as a decimal integer. The meaning of this value is determined by the switches below it.
N/A	This control specifies that the time condition should be considered to be always true. The time value is ignored.
Every	This control specifies that the trigger will be activated every time the simulator time equals a multiple of the specified value.
<, =, >	These controls indicate that the trigger will be activated when the simulation time is less than, equal to, or greater than the given value, respectively.

Signal Test Controls

The controls related to the signal condition are summarized in the following table.

Control	**Comment**
Names	Enter the names of one or more signals whose values will be compared to the hexadecimal integer value typed in the Value field. One or more signals can be entered using the following formats: CLK The single signal CLK D7..0 The signals D7 (MSB), D6, D5 ... D0 IN1 OUT3 The signals IN1 and OUT3
Signals Value	Enter the signal comparison value as a hexadecimal integer. This value is converted to binary and compared bit-for-bit with the signals named in the Names field. The rightmost signal name is compared with the least significant bit of the value, etc.

Delay Condition Controls

The controls related to the time condition are summarized in the following table.

Control	**Comment**
N/A	This control specifies that the delay condition should be considered to be always true. The delay value is ignored.
After	With this control enabled, the trigger will be activated the specified amount of time after the time and signal value conditions are met, regardless of whether they continue to be true.
Sig Stable >	With this control enabled, the trigger will be activated after the specified delay, as long as the time and signal conditions are still true.

Sig Stable <	With this control enabled, the trigger will be activated if the time and signal conditions are true for less than the specified delay. I.e. Activation occurs when the time and signal conditions cease to be true if they have been true for less than the specified value.
Delay Value	Enter the delay value as a decimal integer.

Trigger Actions Controls

When a trigger is activated, any combination of the displayed actions can be invoked.

Control	**Comment**
Beep	Generates a single system beep.
Stop	Stops the simulator immediately.
Enable T_{N+1}	Enables the next numbered trigger.
Timing Display On	Turns on the timing waveform display.
Timing Display Off	Turns off the timing waveform display.
Reference Line	Draws a vertical reference line at this time on the timing waveform display.

Navigation Controls

The navigation controls perform the following functions:

Control	**Comment**
OK	Accept changes for this trigger and close the dialog.
Cancel	Discard changes for this trigger and close the dialog.
Delete	Delete this trigger and close the dialog. All subsequent triggers, by number, are decremented.
Prev	Accept changes for this trigger and display the previous trigger (if one exists).
Next	Accept changes for this trigger and display the next trigger.

Simulation Parameters

The Simulation Parameters is a general method of setting device and pin delays and options. If no devices or pins are selected in a circuit then Simulation Parameters menu command will be disabled. This command displays the

Simulation Parameters dialog which depends on the types of devices currently selected:

Selection	**Dialog**	**Notes**
A single CLOCK device	Clock Parameters	Only one clock device can be set at a time.
A single SETUPHOLD device	SetupHold	Only one SetupHold device can be set at a time.
A single ONESHOT device	One Shot	Only one One Shot device can be set at a time.
Any other combination of one or more devices or pins	General Delay	Any selected CLOCK, ONESHOT, SETUPHOLD, sub-circuit or other non-delay devices will be ignored for device delay calculations.

NOTES:

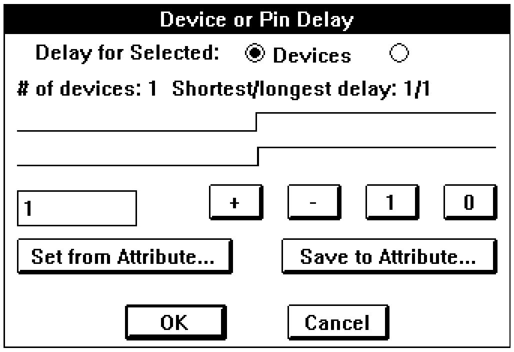
1) The device delay of a sub-circuit device cannot be set as its general delay characteristics are determined by its internal circuit. If any sub-circuit devices are selected, they will be ignored for device delay purposes. Pin delays on sub-circuit devices can be set to modify the path delay through a particular pin. Delays on the devices or pins inside a sub-circuit device are not affected by any settings on the parent device using this command.

2) The Simulation Parameters rely on numeric information in a specific format being present in the Delay.Dev or Delay.Pin attribute fields. Any invalid information in these fields will be ignored and default values used.

See Appendix B - Simulation Attribute Fields (page 135) for more information on the usage of specific attribute fields for simulation parameters.

General Delay Dialog

When selection contains devices and pins with delay characteristics, the following dialog is displayed:



Device or Pin Delay

Delay for Selected: ● Devices ○

# of devices: 1 Shortest/longest delay: 1/1

1

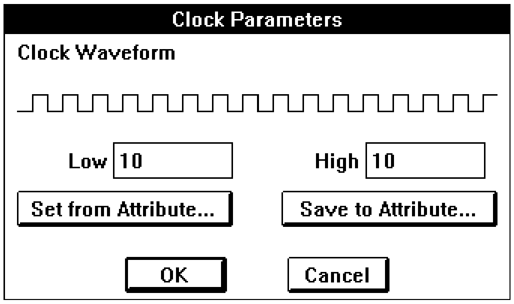
Item	Comment
Devices	When this control is enabled, the other controls display and set the *device delay* characteristic of the devices currently selected in the circuit. Items such as CLOCK or sub-circuit devices which have no device delay characteristic will be skipped.
Pins	When this control is enabled, the other controls display and set the *pin delay* characteristic of the pins currently selected in the circuit.
Number of selected devices/pins	Shows the number of devices or pins that will be affected by changes made in this dialog.
Shortest / Longest delay	Shows the shortest and longest delays found in any of the selected devices or pins. (Note that each device or pin has only a single integer delay value associated with it.)
Delay Text	If all selected devices or pins have the same delay value, it is shown here. If a variety of values exist among the selected items, this item will be empty. Typing a new value (between 0 and 32767) will set all items to the given value.
+	This control will add 1 to the delays in all selected items, to a maximum value of 32767.
-	This control will subtract 1 from the delays in all selected items, to a minimum value of zero.
1	This control will set the delay in all selected items to 1.
0	This control will set the delay in all selected items to 0. .
Set From Attribute	This control will display the Set From Attribute dialog, allowing the delays in all selected objects to be loaded from a selected attribute field in each object. This dialog is described below.

Save To Attribute	This control display the Save To Attribute dialog, allowing the delays in all selected objects to be saved to a selected attribute field in each object. This dialog is described below.

See Chapter III - Simulation Details (page 5) for more information on the meaning and usage of device and pin delays.

Clock Parameters Dialog

When a single clock device is selected, the following dialog is displayed:

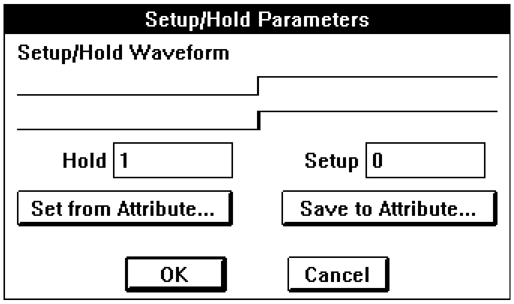


Item	Comment
Low	Enter the low time setting of the selected clock device. Allowable settings are in the range 1 to 32767.
High	Enter the high time setting of the selected clock device. Allowable settings are in the range 1 to 32767.
Set From Attribute	This control will display the Set From Attribute dialog, allowing the parameters in the selected clock device to be loaded from a selected attribute field. This dialog is described below.
Save To Attribute	This control will display the Save To Attribute dialog, allowing the clock parameter in the selected device to be saved to a selected attribute field. This dialog is described below.

See Chapter IX - Primitive Devices (page 107) for more information on how to set the startup delay and initial value of a CLOCK device by setting the pin delay and inversion on the output pin.

SetupHold Parameters Dialog

When a single SETUPHOLD device is selected, the following dialog is displayed:



Setup/Hold Parameters

Setup/Hold Waveform

Hold: 1

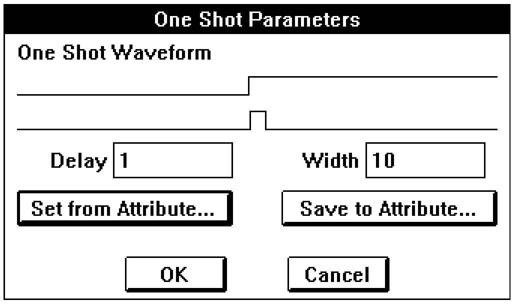
Setup: 0

Item	Comment
Setup	Enter the setup time setting of the selected device. Allowable settings are in the range 0 to 32767.
Hold	Enter the hold time setting of the selected clock device. Allowable settings are in the range 0 to 32767.
Set From Attribute	This control displays the Set From Attribute dialog, allowing the parameters in the selected device to be loaded from a selected attribute field. This dialog is described below.
Save To Attribute	This control displays the Save To Attribute dialog, allowing the setup and hold parameters in the selected device to be saved to a selected attribute field. This dialog is described below.

See Chapter IX - Primitive Devices (page 107) for more information on how to set the initial value of a SETUPHOLD device by setting the inversion on the output pin.

One Shot Parameters Dialog

When a single ONESHOT device is selected, the following dialog is displayed:


One Shot Parameters

One Shot Waveform

Delay 1

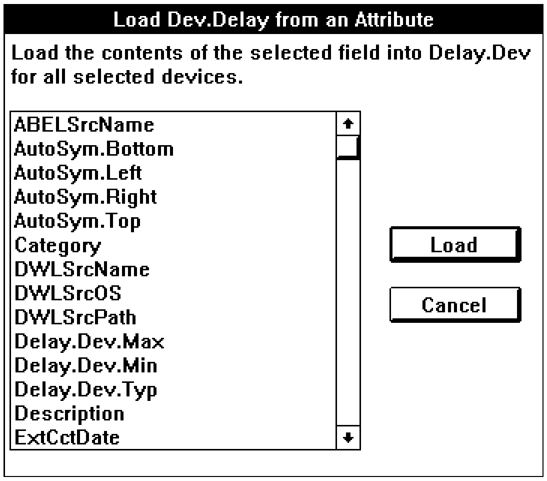
Width 10

Item	Comment
Delay	Enter the delay time setting of the selected device. Allowable settings are in the range 0 to 32767.
Width	Enter the width time setting of the selected clock device. Allowable settings are in the range 1 to 32767.
Set From Attribute	This control displays the Set From Attribute dialog, allowing the parameters in the selected device to be loaded from a selected attribute field. This dialog is described below.
Save To Attribute	This control displays the Save To Attribute dialog, allowing the parameters in the selected device to be saved to a selected attribute field. This dialog is described below.

See Chapter IX - Primitive Devices (page 107) for more information on how to set the initial value of a ONESHOT device by setting the inversion on the output pin.

Loading Delays from Attributes

Each of the simulation parameters dialogs described above has Set from Attribute and Save to Attribute controls. Selecting either of these controls displays a dialog similar to the one below:



Load Dev.Delay from an Attribute

Load the contents of the selected field into Delay.Dev for all selected devices.

Set From Attribute

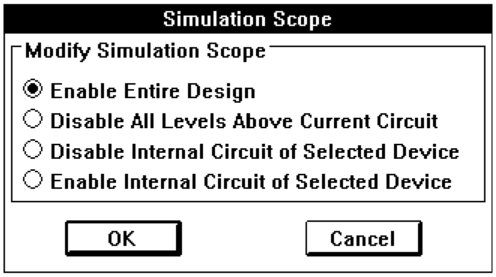
Selecting an attribute field in the list and clicking on the Load control will cause the contents of the selected field to be copied to the Delay.Dev or Delay.Pin field of all selected devices or pins. The copied value will completely replace the old contents. If the selected field is empty in a given object, the Delay.Dev or Delay.Pin field is left unchanged in that object.

Save to Attribute

Selecting an attribute field in the list and clicking on the Save control will cause the contents of the Delay.Dev or Delay.Pin field to be copied to the selected field in all selected devices or pins. The copied value will completely replace the old contents, even if the source value is empty.

Simulation Scope

Selecting the simulation scope command from the Simulator window menu displays the following dialog:



Simulation Scope

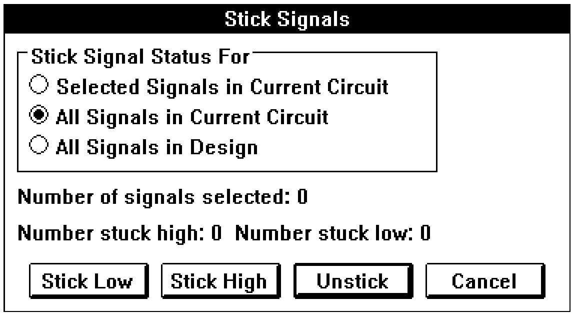
Modify Simulation Scope

● Enable Entire Design
○ Disable All Levels Above Current Circuit
○ Disable Internal Circuit of Selected Device
○ Enable Internal Circuit of Selected Device

Action	Comment
Enable Entire Design	Removes all previous disables and enables simulation of all parts of the design.
Disable All Levels Above Current Circuit	The current circuit (i.e. the one displayed in the frontmost circuit window) is set to be the topmost level of the hierarchy that will be simulated. Simulation is disabled in all other hierarchy levels and all driving port connectors at this level will drive at a high impedance level.
Disable Internal Circuit of Selected Device	The internal circuit of the device selected on the schematic will be disabled. All the output pins on the device will be set to a high impedance drive level.
Enable Internal Circuit of Selected Device	The internal circuit of the device selected on the schematic will be re-enabled. Its outputs will revert to the state determined by the internal circuit.

Stick Signals

Selecting the stick signals command in the Simulator window menu displays the following dialog:



Stick Signals

Stick Signal Status For

○ Selected Signals in Current Circuit

● All Signals in Current Circuit

○ All Signals in Design

Number of signals selected: 0

Number stuck high: 0 Number stuck low: 0

Control	**Comment**
Selected Signals in Current Circuit	If this control is enabled, the Stick High, Stick Low or Unstick buttons will apply only to signals currently selected in the schematic.
All Signals in Current Circuit	If this control is enabled, the Stick High, Stick Low or Unstick buttons will apply to all signals in the circuit represented in the topmost schematic window, including circuit pages in the same circuit. If this is part of a hierarchical design, only this circuit level is affected.
All Signals in Design	If this control is enabled, the Stick High, Stick Low or Unstick buttons will apply to all signals in all parts of the current design.
Number of signals selected	The number of signals that will be affected by any changes made in this dialog.
Number stuck high	The number of signals in the selected scope that are currently stuck at a high level.
Number stuck low	The number of signals in the selected scope that are currently stuck at the low level.
Stick Low	This control closes the dialog and applies a "stuck low" value to all selected signals.
Stick High	This control closes the dialog and applies a "stuck high" value to all selected signals.
Unstick	This control closes the dialog and unsticks all selected signals, allowing them to return to their driven value.

See Chapter III - Simulation Details (page 5) for more information on stuck signals.

Chapter VI - The SimLoad Tool

Introduction

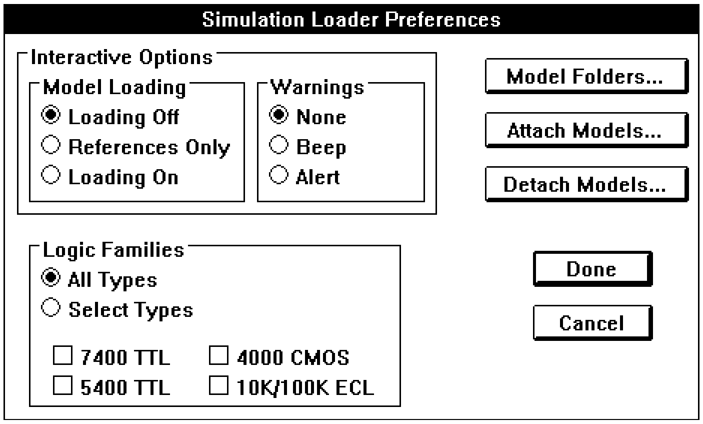
This chapter describes automatic or manual loading of simulation models using the SimLoad tool.

Starting SimLoad

To start SimLoad, select it from the Tools menu in the DesignWorks window. This opens the SimLoad window.

The SimLoad Window

The SimLoad window is a dialog. It is displayed when the SimLoad tool is selected from the “Tool” menu in the DesignWorks window. It can be hidden by selecting Cancel or Done from the window controls.



Simulation Loader Preferences

Interactive Options

Model Loading
● Loading Off
○ References Only
○ Loading On

Warnings
● None
○ Beep
○ Alert

Logic Families
● All Types
○ Select Types

□ 7400 TTL □ 4000 CMOS
□ 5400 TTL □ 10K/100K ECL

SimLoad Window Controls and Fields

Status

Most of the models for industry-standard device types (e.g. 74XX, 4000, etc.) are provided as external sub-circuits. These models are simply design files containing an equivalent circuit for the device. The circuits are designed to match with the symbol libraries provided with the Schematic package.

External sub-circuit models can be loaded either interactively, i.e. as each part is used, or in batch mode after the design is created. SimLoad defaults to the inactive state, i.e. no loading occurs.

Two levels of loading action are available:

- References Only - This method can be used to check for the existence of the model file without actually loading it. In this case, each time a new part is used, SimLoad attempts to locate a model for it and sets the external circuit reference attributes to the location of the file. The file itself is not loaded, i.e. the device will not have a sub-circuit. The sub-circuit can be loaded later using the Attach External Sub-Circuit command in the Sub-Circuit sub-menu in the Schematic tool's Options menu.
- Loading On - In this mode, SimLoad locates the external model file and immediately loads and attaches it to the part. The external reference attributes are also set, allowing for later updating if the model file is changed.

NOTE: The SimLoad action in both the above cases is the same. In "Loading On" mode, the SimLoad tool directs the Schematic tool to enable its "Auto-Load External Sub-Circuits" mode. This mode can also be controlled directly in the Sub-Circuit sub-menu in the Schematic tool.

Warnings

This section allows the warning level to be set in the event that a model cannot be located during interactive loading.

Control	**Comment**
None	No warning is issued.
Beep	A standard beep tone will be issued each time a new device type is used for which a model could not be located.
Alert	A warning dialog will be displayed whenever a model is not located.

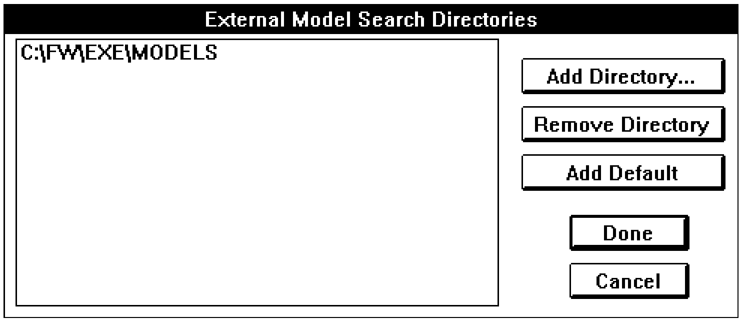
It is possible to determine at any time which devices have an external model associated with them by viewing the contents of the ExtCctName attribute field. This can be done using the Schematics Attributes dialog, the Browser tool (select the ExtCctName field in the Secondary field list), or using the Report tool. A special report format file called "External Circuit Form" is provided with the simulator for this purpose.

Logic Families

This section can be used to restrict the types of devices that will have models attached to them. When the "All Types" control is selected, every device that is used that is a sub-circuit primitive type and does not have an internal sub-circuit will result in a search for a model to attach. When the Select Types control is selected, only the specific standard families indicated will be checked for external models.

Model Directories...

By default, the SimLoad tool searches for simulation models in the directories displayed in this dialog:



External Model Search Directories

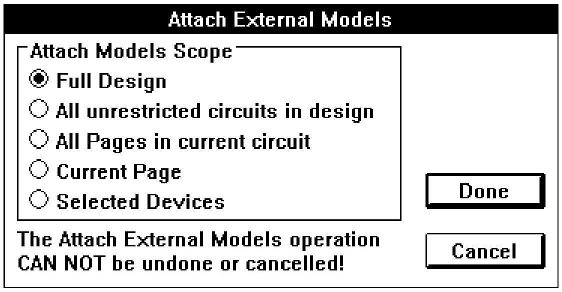
C:\FW\EXE\MODELS

To add a new directory, click on the Add Directory control. This will display a standard directory selection dialog. To remove a directory from the search list, simply select it in the list and click on the Remove Directory control. The default directory can be added by clicking on the Add Default control.

Note: The default directory must be included in the list of directories in order for it to be searched.

Attach Models...

SimLoad allows batch loading of external models. The Attach Models control will display the following dialog:



Attach External Models

Attach Models Scope

● Full Design

○ All unrestricted circuits in design

○ All Pages in current circuit

○ Current Page

○ Selected Devices

The Attach External Models operation CAN NOT be undone or cancelled!

The Attach Models Scope section allows the scope of the search to be specified. When the Done control is clicked, the devices in the given scope are checked, external references are created or updated, where possible, and the external circuits are loaded and attached. After the operation is complete, a summary dialog will be displayed indicating the number of models found and loaded.

NOTE: This operation can result in a dramatic increase in the amount of memory used by the design. If insufficient memory is available, a message will be displayed.

Detach Models...

SimLoad's Detach Models operation performs the reverse operation to Attach Models. Devices in the selected scope are scanned and the internal circuit is removed from every device having an external reference. The external reference attributes are left intact, allowing for later reloading of the external models.

SimLoad Procedures

Checking Individual External References

The Attach External Sub-Circuit command in the Sub-Circuit sub-menu of the Options menu in the Schematic tool is useful for checking or editing the external reference attributes of a single device. Select the device in question in the schematic, then select this command. It will show the current status of the external reference and allow it to be updated.

Batch Updating or Reloading Models

When an external sub-circuit is loaded, the Schematic tool stores the "Last Modified Date" of the file in the ExtCctDate attribute field of the device. This allows it to check the status of external files relative to a device's internal circuit. Devices that have existing sub-circuits that were loaded from external models, or that have an external reference but no internal circuit, can be updated using the Update External Sub-Circuits command in the Sub-Circuit sub-menu of the Options menu in the Schematic tool. This command will report if all selected items are already up to date.

Creating External Sub-Circuit Models

External sub-circuit models are simply normal DesignWorks design files with port connectors added to match the pins on the parent device. Following is a brief summary of the considerations affecting sub-circuits:

- In order to be loaded automatically using the SimLoad tool, the file must be named according to the conventions outlined later in this chapter.
- A port connector should be placed in the sub-circuit for each pin on the parent symbol, including no-connect pins. Failure to do this will result in the Schematic tool reporting an error each time the sub-circuit is opened and closed.
- Each port connector must be named to match the corresponding pin name on the parent device.
- The type of each port connector should match the type of the corresponding pin on the parent device. E.g. An input pin on the parent device should match with a "Port In", or other appropriate type.
- The sub-circuit can contain any combination of other devices, including nested sub-circuits to any depth. The only restriction is that the sub-circuit may not contain the parent type or any other type that would create a recursive nesting of blocks.

See “Working With Hierarchical Blocks” (page 25) in Chapter IV - Schematic Changes to Support Simulation for more information.

See Chapter VII - Hierarchical Design in the DesignWorks/Schematic Reference Manual for other considerations relating to the creation of sub-circuits.

Customizing Delays for Logic Families

For cases such as the 74XX families, only one model is provided for all logic families (e.g. 74ALS, 74LS, 74HC, etc.).

NOTE: These models provide logical functionality only. No attempt is made to match the delay characteristics of the individual parts.

Delays can be manually customized to match a specific part type by either:

- Modifying the sub-circuit model (either in the model file or after it has been used in the design), OR
- Setting appropriate pin delays in the parent device.

Model File Naming Conventions

When searching for an external sub-circuit model to attach to a device, the SimLoad tool generates a file name based on the type name, i.e. the name as it appears in the parts palette. The specified search directories are then scanned for the generated file name.

File names are generated differently for each common logic family type, using the following rules:

- File names always end with ".EXT" (note that file names are not case-sensitive).
- Any text in parentheses is ignored, since these are assumed to be package codes.
- If the first two characters of the name are "54" or "74" the a 54XX or 74XX logic family is assumed. Any alphabetic characters are skipped (i.e. the family type) and the file name "74_nnn.EXT" is generated, where "nnn" refers to any remaining digits that are found.
- If the first character of the name is "4" a 4000-series part is assumed. Any alphabetic characters are skipped and the file name "4_nnn.EXT" is generated, where "nnn" refers to any remaining digits that are found.
- If the first character of the name is "1", then a 10K or 100K ECL part is assumed and the name "10_nnn.EXT" or "100_nnn.EXT" is generated, as appropriate.
- For all the above cases, if the name ends with "M", it is assumed to be a "monolith" symbol (i.e. with multiple gates in one symbol) and the M replaces the underscore character. e.g. "74M00.EXT".
- For all the above cases, if the name ends with a gate unit extension (e.g. "74ALS240.a"), then the unit letter replaces the underscore. e.g. "74a240.EXT".
- For all other cases, the type name (with text in parentheses stripped) is used verbatim (up to 8 characters).

Chapter VII - The Timing Tool

Introduction

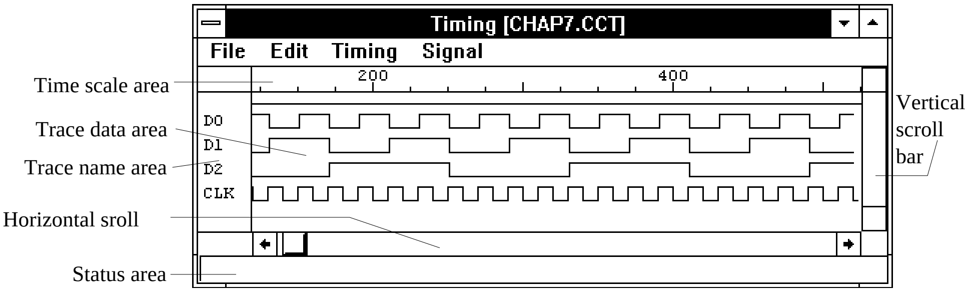
This chapter describes displaying of timing waveforms via the Timing Tool.

Starting the Timing Tool

To start the Timing tool, select it from the Tools menu in the DesignWorks window. Only one Timing window can be displayed per design, regardless of the number of hierarchy levels in the design.

Timing Window

The timing window is displayed when the Timing tool is selected from the "Tool" menu in the DesignWorks window. It can be hidden by selecting "Close" from the system menu in the top left corner of the window.



Item	Comment
Time Scale	Located just below the timing window's title bar, the time scale is used to establish the absolute timing of value changes in the trace area. The scale is dependent upon the timing resolution (set using the < and > controls in the simulator window). The time scale is also used to set insertion points and selection intervals for use in editing functions.
Trace Data Area	Displays the simulation results.
Trace Name Area	Displays the list of trace names corresponding to the timing traces on the right. Traces can be repositioned by dragging them vertically in this area. In addition, a pop-up menu can be displayed by pressing the right mouse button on a signal name.

Status Area	This area shows progress information for timing commands.
Horizontal Scroll Bar	This control allows scrolling of the trace area. The horizontal scroll bar is disabled when the simulation is running.
Vertical Scroll Bar	This controls allows scrolling of both the trace name and data areas.

Timing Window Menu Commands

File Menu

Import Timing (Text) ...

This command clears the timing window, prompts for a timing text file, opens the file and, finally, pastes the data into the trace area. This is equivalent to selecting the Clear Simulation command, then using the Paste command to place the file data at time zero. See the section “Paste Rules” (page 72) for more information.

NOTE: This command does not display or remove any traces in the timing window. It only reads signal event data and associates it with matching traces. If any traces are named in the file that are not currently displayed, you will be warned and that data will be skipped.

See Appendix A - Timing Text Data Format (page 131) for a description of the file format.

Export Timing (Text) ...

This command prompts for a save file then saves all the displayed trace data to a supplied file. This file can be used for external purposes, or can be reloaded as a setup for a new simulation using the Import Timing (Text) command.

See Appendix A - Timing Text Data Format (page 131) for a description of the file format.

Print Timing...

This command displays the print dialog to initiate printing of the timing windows contents using the current printer setup. The current display will be divided into as many pages as required.

Page Setup...

This command displays the page setup dialog which is used for the Print Timing command. This page setup can be different than the schematic diagram page setup.

Edit Menu

Undo

This command undoes the last editing operation in the Timing window. Unlike the Schematic tool, Timing supports only a single Undo and no Redo operation.

Copy

This command copies the selected timing data to the clipboard in picture and text format. The data is stored in both picture form and text data form (i.e. Test Vector command format).

> *See "Timing Procedures / Copying and Pasting" (page 71) in Chapter VII - The Timing Tool for more information.*
>
> *See Appendix A - Timing Text Data Format (page 131) for a description of the file format.*

Paste

This command pastes the timing text data from the clipboard onto the selected area of the timing window. The selected time interval is deleted and then the new data is inserted. i.e. data following the selection interval will be moved forward by the width of the selection interval, then back by the width of the pasted data.

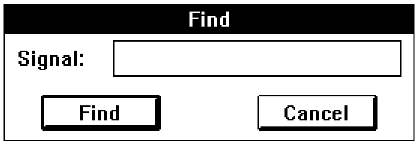
> *See "Timing Procedures / Copying and Pasting" (page 71) in Chapter VII - The Timing Tool for more information.*
>
> *See Appendix A - Timing Text Data Format (page 131) for a description of the file format.*

Select All

This command selects all traces and the entire time interval of the timing display.

Find ...

This command allows you to locate a signal or group in the timing display by its trace name. The following dialog will be displayed:



The name of any displayed trace can be typed into the Signal control. When the Find button is clicked, the item will be located and selected (if necessary, the window will be scrolled vertically to display it). The search is done by the displayed trace name in the timing window, not the original name of the item in the schematic.

The following "wildcard" characters can be used in this search:

Wildcard	Comment
*	Matches any string of zero or more characters. For example "D*" will match "D", "D0", "D12", "DogGone", etc. "*" by itself matches anything.
#	Matches any string of 1 or more digits. E.g. "XA#" will match "XA0", "XA123", etc.
?	Matches any single character. For example, "CTRL?" will match "CTRLX", "CTRL/", etc.

The wildcard characters can be used in combination, as in "DATA#??" which will match "DATA123", "DATA1SQ", etc.

Timing Menu

Display On

This command enables updating of the timing display.

Display Off

This command disables updating of the timing display. Events are saved but are not drawn into the timing window. This allows simulation to proceed at a substantially faster rate. Updating of the timing window can be re-enabled either manually or by a trigger.

Normal Size

This command sets the horizontal display resolution to its initial defaults.

Enlarge

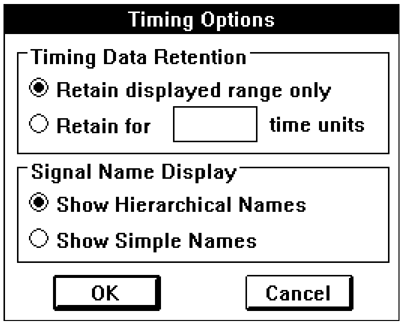
This command increases the horizontal display resolution in the timing window.

Reduce

This command decreases the horizontal display resolution in the timing window.

Timing Options ...

This command displays the following dialog:



Timing Options

Timing Data Retention

● Retain displayed range only

○ Retain for ☐ time units

Signal Name Display

● Show Hierarchical Names

○ Show Simple Names

Timing Data Retention

These options allow you to determine how much signal event data is retained in memory when a simulation is run.

Each time a signal level change occurs DesignWorks creates a record in memory containing a reference to the signal, time, new value and source of the change. In a large simulation these records can consume enormous amounts of memory. This data can be retained for the following purposes:

- for use in refreshing the timing window should it become hidden then redisplayed.
- for use in timing window editing operations such as taking the output from one circuit and using it as stimulus for another.

Data can be retained only for signals displayed in the Timing window. Signal event data for all other signals is discarded immediately after it is no longer required for simulation.

The option "Retain for Displayed Range Only" is the normal default and results in data being discarded immediately after the corresponding point on the timing display scrolls off the left hand side. This results in minimal memory usage. The setting is equivalent to entering 0 in the retain time box.

The option "Retain for x Time Units" allows you to keep the signal event data for the specified amount of time after it scrolls off the left side of the screen. If this results in a memory shortage occurring then the simulation will stop and a message will be displayed.

Signal Name Display

This option lets you choose how names of signals in sub-circuits will be displayed in the timing window. The option "Show Hierarchical Names" will display the complete "path" to the signal, i.e. prefixed with the names of all parent devices. The option "Show Simple Names" will show only the basic signal name in the trace name area of the window.

Signal Menu

Get Info ...

For groups, this command displays a dialog allowing reordering of the signals in the group. This affects the way the combined hexadecimal value is shown in the timing display.

For individual signals, a signal info dialog is displayed.

Go To Schematic

This command selects the signal or groups in the schematic module corresponding to the first highlighted signal or groups in the timing window then brings the required schematic window forward.

Remove

This command removes the selected signals or groups from the timing window.

Group

This command combines all the selected traces into a single display group. If any of the selected traces were already grouped, they are in effect Ungrouped first and then recombined with other selected items into a single new group.

Ungroup

This command breaks all signals in selected groups into individual traces.

Timing Window Pop-Up Menu

Clicking the right mouse button in the trace name area of the timing window will display the timing pop-up menu.

Get Info ...

For groups, this command displays a dialog allowing reordering of the signals in the group. This affects the way the combined hexadecimal value is shown in the timing display.

For individual signals, a signal info dialog is displayed.

Go To Schematic

This command selects the signal or groups in the schematic module corresponding to the first highlighted signal or groups in the timing window then brings the required schematic window forward.

Remove

This command removes the selected signals or groups from the timing window.

Group

This command combines all the selected traces into a single display group. If any of the selected traces were already grouped, they are in effect Ungrouped first and then recombined with other selected items into a single new group.

Ungroup

This command breaks all signals in selected groups into individual traces.

Collect

This command brings all selected items together in the display underneath the topmost selected item. This is used to bring associated signals closer together for easier comparison of timing, etc.

To Top

This command sends all selected traces to the top of the timing window.

To Bottom

This command sends all selected traces to the bottom of the timing window.

Timing Procedures

Repositioning Signals

Any collection of selected trace names and their corresponding timing data can be repositioned within the list by clicking on the desired trace names in the trace name area (hold the shift key down while clicking to select multiple items) and, keeping the mouse pressed, dragging the vertical line to its new location. Releasing the mouse button will cause the list to be revised with both trace names and trace data traces in their new positions. Alternatively, the To Top, To Bottom and Collect commands in the pop-up menu can be used.

Displaying Groups

The Timing tool allows multiple signal lines to be grouped into a single trace with values displayed in hexadecimal.

Creating a Group Trace

A group trace can be created by any of the following methods:

- Select traces by clicking on the desired names (hold the shift key down while clicking to select multiple items). Use the right mouse button on any item then select the Group command in the pop-up menu.
- Select any collection of signals in the schematic diagram, then select the Add As Group command from the Timing menu in the Schematic window.

NOTE: If any of the selected traces is already displayed, it will not be added to the group.

- Select a bus in the schematic diagram, then select either the Add To Timing or the Add As Group command in the Timing menu. Busses are added as a group by default. They can then be ungrouped if desired, using the Ungroup command in the timing pop-up menu.

Order Within a Group

For the purposes of displaying a hexadecimal value for a group, the order of signals within the group is important. When a group is created, the following rules are used to establish the order:

- If the signal name has a numeric part (e.g. "D12" or "WRDAT4X", then the numeric part is used to sort the signals. The lowest numbered signal will be the least significant bit of the group value. Any unnumbered signals will be in the most significant bit positions.
- Otherwise, the signal's existing position is used, i.e. traces that appeared higher in the timing window will be more significant.

The order of signals within a group can be changed using the Get Info command on a group trace. This is displayed by selecting the Get Info command in the timing pop-up menu or by double-clicking on the trace name.

Entering a Group Name

When a group is first created, a group name is automatically generated from the names of the enclosed signals. This name can be edited using the Get Info command in the timing pop-up menu.

NOTE: The group name is lost when an Ungroup operation is done.

Copying and Pasting

To select timing data for copy operations the simulation must be stopped. To do this:

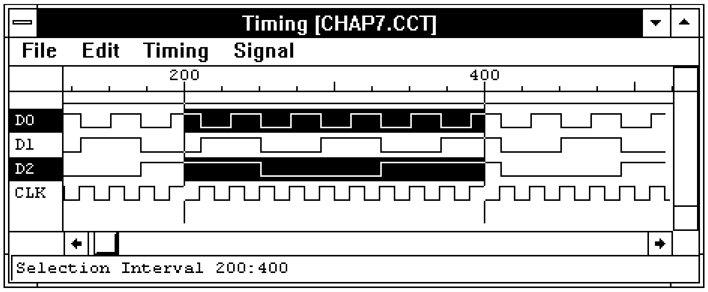
- Click in the trace area or timing scale area of the Timing window.
- Click on the Step control in the Simulator window.
- Select Stop from the Speed menu in the simulator window.

There are two methods of selecting areas for edit operations.

Separate Trace Name and Interval Selection

Select traces to be included by holding the shift key down while clicking in the trace name area, then select the time interval is selected by clicking and dragging in the time scale. This allows you to select non-contiguous traces in the display.

- Click on the desired name in the trace name area to highlight and select it. To select more than one name, hold the shift key down and click on the labels.
- To set the selection interval, click and hold down the mouse button in the time scale at either end of the desired interval. Drag left or right until the desired interval is enclosed. When the mouse button is released the select interval is set and two selection interval lines will appear. If any of the trace names were selected, the timing signal within the selected interval will be highlighted in the time display.



Selection Interval 200:400

NOTE: Clicking and releasing the mouse button at one spot will create a zero-width interval. This can be used to insert Pasted data without deleting any existing data.

Drag Selection

This method allows you to select a group of trace names and a time interval in a rectangular area of the timing window.

To do a drag selection, click and hold the mouse button at any corner of the rectangular area you wish to select. Drag diagonally across the desired area. When the mouse button is released, the enclosed time interval and traces will be selected.

NOTE: The selection operations in the timing window have no effect on selections in the schematic window.

Selecting All Traces or All Time

To select a specific time interval in all traces on the diagram:

- Use the Select All command in the Edit menu to select the entire diagram.
- Drag select an interval in the time scale area without clicking in the trace data area.

To select all time intervals for specific traces:

- Deselect all traces (see below) then hold the shift key down and click on the desired trace names to be selected.

Deselecting

Click in the trace name area above or below the name list to deselect all signals.

Paste Rules

The following rules are used for matching the data on the clipboard with the selected interval in the timing window:

- Data is always pasted by name, i.e. the name of a signal in the clipboard data will be matched by the same-named signal in the timing window. Neither the order of the signals in the clipboard data or the selected status of traces in the timing window is significant. To paste data from one signal to a signal with a different name, it is necessary to paste it first into the Test Vector or into a text editor, modify the names, then paste it back.
- The Paste operation affects only signals named in the clipboard data, regardless of the selection in the timing window.

- The Paste operation will not locate signals in the schematic that are not currently displayed in the timing window. No new traces will be added by this operation.
- If the time interval selected in the timing window is non-zero width then the selected interval is deleted and all later events on pasted signals are moved forward. A time interval equal to the width of the clipboard data is then inserted and the new data pasted into this interval.

Chapter VIII - The Test Vector Tool

Introduction

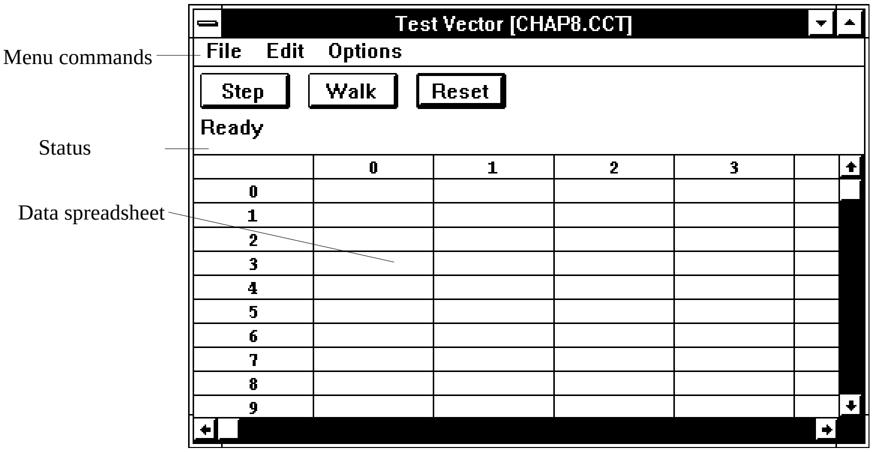
This chapter describes the creation and execution of test vectors using the Test Vector Tool.

Starting the Test Vector Tool

To start the Test Vector tool, select it from the Tools menu in the DesignWorks window. More than one Test Vector window can be displayed per design.

Test Vector Window

The Test Vector window is displayed when the Test Vector tool is selected from the “Tool” menu in the DesignWorks window. It can be hidden by selecting “Close” from the system menu in the top left corner of the window.



Item	Comment
Status Display	Shows the current operational status of this Test Vector window.
Step	The Step button causes Test Vector to interactively execute the next line of the data spreadsheet, i.e. the line which contains the currently selected cell. If the line selected is past the last data line, then a beep will be issued and no action will be taken.

Walk	The Walk button causes the Test Vector to step through the data spreadsheet as quickly as possible in interactive mode, starting with the currently selected cell. To start from the beginning of the program, it is necessary to select a cell in the first row of the spreadsheet, or first click the Reset button.
Reset	The Reset button removes all data that was created by the program from the spreadsheet and selects the top-left cell. All user-entered data (i.e. headers, commands, expected values, etc.) are left intact.

Test Vector Menu Commands

File Menu

Open Test File ...

This command loads data for the spreadsheet from a text file. The format of this file is the same as that commonly used for transferring data between spreadsheet applications:

- the text from the top cell in the leftmost column is written first, followed by a tab character, followed by the contents of the next cell to the right, followed by a tab, etc.
- each line is terminated by a carriage return character.

Save Test File ...

This command saves the contents of the data spreadsheet to a text file in the format described above.

Set Trace File ...

This command creates a text file and writes a line of text data matching the current line of the spreadsheet each time a step is done in interactive or monitor mode. This allows you to create a permanent record of simulation data. This file can be opened and closed at any time during the simulation without disrupting any simulation data.

Save to Design Attr ...

This command behaves identically to Save Test File command except that the destination for the data is a design attribute field instead of a file. The Test Vector data will then be stored with the design file the next time the design itself is saved.

This command will display a list of the fields defined for the current design allowing one attribute to be selected as the destination. All existing data in the selected field will be lost. Attribute fields must be less than 32,000 characters.

Load From Design Attr...

This command corresponds to the Open Test File command except that the data is loaded from a design attribute field instead of a file.

Edit Menu

Cut

This command depends upon how data is selected in the spreadsheet.

If a cell is being actively edited (i.e. a portion of the text is selected or the I-beam cursor is flashing in it) then only the selected text will be copied to the clipboard and the selected text will be deleted.

If a group of cells has been selected using the shift key, then the contents of all selected cells will be copied to the clipboard then the cells will be cleared.

> *See "Test Vector Procedures / Clipboard Operations" (page 89) in Chapter VIII - The Test Vector Tool for more information on the format of the data.*

Copy

This command is identical to the Cut command except that the selected items are not cleared.

Paste

This copies the data on the clipboard to the spreadsheet. Data is assumed to be in the clipboard format described above, i.e. any number of cells can be pasted in one operation.

Note the following rules for the Paste operation:

- The size of the selected area on the spreadsheet has no effect on the Paste operation. The top-left cell in the selected group will be the top-left corner of the area affected by the Paste. However, the number of cells affected by the Paste is determined entirely by the data on the clipboard.
- Tab or carriage return characters cannot be pasted into Test Vector cells since they are used to delimit cell data.
- If a cell is being actively edited at the time of the Paste (i.e. a portion of its text is selected or the blinking I-beam cursor is located in it) then the text on the clipboard up to the first tab will replace the selected text in that cell.
- Data is always copied and pasted as viewed on the screen without regard for the "across" or "down" data orientation Test Vector setting.
- Clipboard operations cannot be undone.

Clear

This command clears all data from the selected cells.

Note: This operation cannot be undone.

Select All

This command selects the entire spreadsheet area.

Fill Down

This command copies the data from the topmost cell in each selected column to all selected cells underneath it. It has no effect on the clipboard.

Fill Right

This command copies the data from the leftmost selected cell in each row to all selected cells to the right of it. It has no effect on the clipboard.

Insert Rows

This command inserts above the topmost selected row a number of rows equal to the number currently selected. All lower rows are moved down.

NOTE: This affects the entire width of the spreadsheet, regardless of the width of the selected area.

Delete Rows

This command deletes the entire width of each row touched by the current selection or containing the cell currently being edited. All data in the selected rows will be lost and all lower rows will be moved up. THIS CANNOT BE UNDONE!

Insert Cols

This command inserts left of the leftmost selected column a number of columns equal to the number currently selected. The columns to the right are shifted right.

NOTE: This affects the entire height of the spreadsheet, regardless of the height of the selected area.

Delete Cols

This command deletes the entire height of each column touched by the current selection or containing the cell currently being edited. All data in these columns will be lost and columns to the right will be shifted left. THIS CANNOT BE UNDONE!

Options Menu

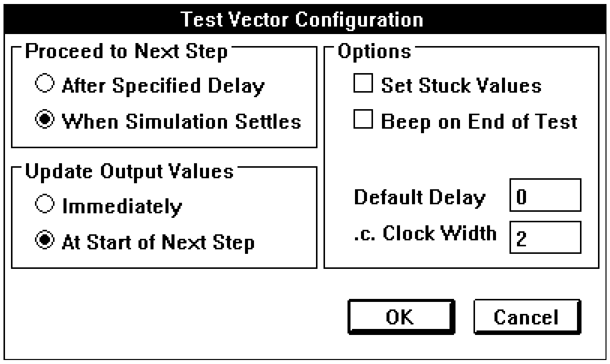
Link to Circuit

This command causes the Test Vector to become associated with the current circuit, i.e. the circuit displayed in the frontmost document window. This is intended primarily for use in hierarchical designs where sub-circuits are being opened and closed.

See “Test Vector Procedures / Hierarchical Designs” (page 91) in Chapter VIII - The Test Vector Tool for more information on using hierarchical designs.

Configuration...

This command displays a dialog allowing the Test Vector tool to be configured:



Test Vector Configuration

Proceed to Next Step
○ After Specified Delay
● When Simulation Settles

Options
□ Set Stuck Values
□ Beep on End of Test

Update Output Values
○ Immediately
● At Start of Next Step

Default Delay 0
.c. Clock Width 2

Proceed to Next Step

These two buttons allow you to determine whether the Test Vector waits for the simulator to complete all processing before proceeding or goes ahead at a fixed pace without regard for other activity.

Control	Comment
After Specified Delay	With this setting, the test program will proceed to the next step after the delay specified on that row (if any) or after the Default Delay specified in this box. This setting is most appropriate when using the Test Vector to generate continuous patterns or using it with a design that generates its own clock signals.
After Simulation Settles	With this setting, the Test Vector in effect "goes to sleep" until the DesignWorks simulator has completely settled, i.e. events have been evaluated and no new ones have been created by the circuit. Note that, depending on the circuit, this may take an arbitrary amount of time. If the simulated circuit has any oscillating elements then this settling will never occur. This setting is preferable for cases where the Test Vector is generating all inputs and clocks, since it allows the test program to be independent of delays in the design.

Update Output Values

These two buttons allow you to determine when the outputs from the design are evaluated.

Control	**Comment**
Immediately	The expected outputs are checked and actual outputs are displayed at the same time step as the inputs were set up, i.e. these will be the values resulting from the previous vector.
At Start of Next Step	Outputs on each line are checked immediately before the next step is executed, i.e. after all simulation values have settled from this step or the specified delay has elapsed. Thus, the output values shown on each row will be the result of the inputs specified in that row.

Set Stuck Values

When this checkbox is set, all signal values set up by the Test Vector are "stuck", i.e. they override any outputs from existing devices in the circuit and can only be changed by further Test Vector outputs or other user settings.

Beep at End of Test

When this switch is enabled, a standard beep tone will be issued whenever the end of the test program is reached.

Clock Width

This editable text item determines the width of the clock pulse generated by a ".C" or ".N" signal value in the "Specified Inputs" column. A zero value is not allowed.

Default Delay

This editable text item operates slightly differently depending on other settings:

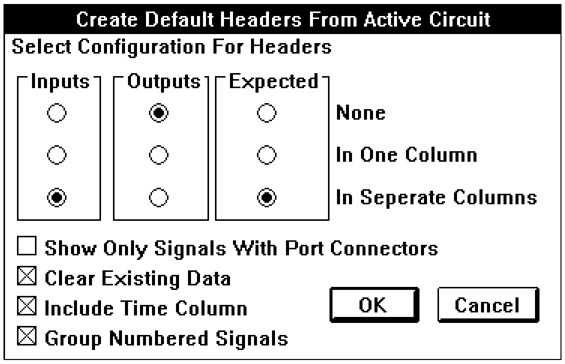
- If the "After Specified Delay" step option is selected or a "Copy to Timing" operation is being done, this will be the default delay between steps and can be overridden by a $DELAY command in the test program. A value of zero is not allowed.
- If the "When Simulation Settles" option is selected, this will be the minimum step size. I.e. The time between steps will be the larger of this

value or the amount of time required for the circuit to settle. A value of zero is allowed in this mode.

Get Header...

This command examines the linked circuit and finds all undriven inputs and final outputs in the circuit and creates a default header in the first row of the spreadsheet area. These can then be edited manually if needed.

The following dialog will be displayed:



Create Default Headers From Active Circuit

Select Configuration For Headers

Inputs	Outputs	Expected	
○	●	○	None
○	○	○	In One Column
●	○	●	In Seperate Columns

☐ Show Only Signals With Port Connectors
☒ Clear Existing Data
☒ Include Time Column
☒ Group Numbered Signals

Column Format Options

The nine buttons in the Inputs/Outputs/Expected grid allow you to select how header entries are created for input and output signals.

NOTE: "Outputs" and "Expected" both refer to the output signals, but offer different treatment of them. It is allowable to included the same signals in both places.

These options are summarized in the following table.

	Inputs	**Outputs**	**Expected**
None	No input columns ($INPUT) are included	No actual output columns ($OUTPUT) are included	No expected output columns ($EXPECTED) are included
In One Column	All inputs are included in a single column, e.g. $I A B C D	All outputs are displayed in a single column, e.g. $O A B C D	All expected outputs are checked in a single column, e.g. $E A B C D
In Separate Columns	Individual inputs or groups are shown in separate columns	Individual outputs or groups are shown in separate columns	Individual expected outputs or groups are checked in separate columns

Other Default Header Options

The following table summarizes the remaining options in the Default Header dialog:

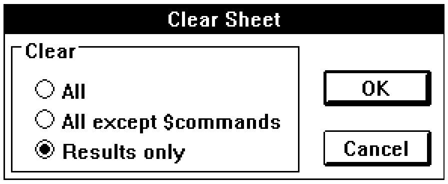
Control	**Comment**
Show Only Signals with Port Connectors	This option is intended for use with sub-circuit blocks that have Port Connector symbols attached to all external pin connections.
Clear Existing Data	If this box is checked, all data currently in the spreadsheet area will be cleared before creating the new header. THIS CANNOT BE UNDONE!
Include Time Column	This option will insert a column at the left with a $TIME header. This will display time values as the test program executes.
Group Numbered Signals	When this option is enabled, sequentially numbered signals (e.g. FC0 FC1 FC2) will be grouped together using the "[]" grouping operator, e.g. "[FC2..0]". They can then be specified with a single hexadecimal value. The lowest numbered signal is assumed to be the least significant bit of the value. This applies to all Inputs, Outputs or Expected columns.

Resize Columns

This command adjusts the width of all columns so that the widest data in the column just fits.

Clear Sheet...

This command clears the entire data spreadsheet area, offering the following options:



Clear Sheet

Clear

○ All
○ All except $commands
● Results only

Control	Comment
All	All cells in the spreadsheet are cleared
All Except $Commands	Any cell starting with a "$" character will be left untouched.
Results Only	All cells that were filled in by the program will be cleared and all others left untouched.

Copy to Timing

This command converts test vector data to simulation events. If the DesignWorks simulator is not stopped, then simulation will proceed immediately after events are set up.

NOTES:

1) The meanings of the some of the commands in the Command column of the spreadsheet are changed in this mode. For example, an infinite repeat loop is not allowed and will be interpreted as a single iteration.

2) This command does not affect the clipboard.

Monitor Mode

This command selects monitor mode operation. In this mode, any command or specification that would set up a signal value change in the circuit under test is ignored. Each time any signal specified as an output changes state, a new line is displayed in the spreadsheet area.

Selecting this command again returns to normal mode.

Update Display

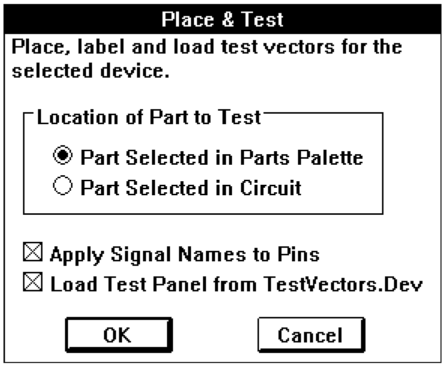
When this item is checked, the spreadsheet is updated with new data after each operation. This provides best user feedback on the progress of the test, but substantially slows down the process. This control only affects the appearance of the display. All other controls still operate interactively. This switch can be turned on and off at any time during operation to check on the progress of the test.

Place and Test Device...

This command provides the ability to load a device from a library and ready it for testing in a single operation. It is intended to allow a quick verification or analysis of a library part.

NOTE: Most of the parts in the Primitive libraries shipped with the Simulator contain test vectors which can be used to verify their operation or as examples for creating your own device test procedures. Simply click on the part in the Parts Palette once so that it is selected, then select this command.

This command will operate either on a part type selected in the currently displayed library in the Parts palette, or it will operate on a selected device already placed in the current circuit. It displays the following dialog:



Place & Test

Place, label and load test vectors for the selected device.

Location of Part to Test

● Part Selected in Parts Palette
○ Part Selected in Circuit

☒ Apply Signal Names to Pins
☒ Load Test Panel from TestVectors.Dev

Control	Comment
Place Part Selected in Parts Palette	If this button is selected, then the part currently selected in the Parts Palette will be placed in the center of the current schematic page. The circuit is assumed to be empty, but no checking is done.
Operate on Selected Device in Circuit	If this button is selected, then the operations selected below will be performed on the device that is currently selected in the schematic.
Apply Signal Names to Pins	If this option is checked, a visible signal name will be applied to each pin, corresponding to the name of the pin. This name can then be referred to in the test vectors.
Load Test Vector from TestVectors Attribute	If this option is selected then the spreadsheet area will be cleared and new data loaded from the TestVectors.Dev attribute field in the device.

Save Test to Device

This command causes the entire spreadsheet area to be saved to the TestVectors.Dev attribute field of the device selected in the schematic.

See "Test Vector Menu Commands / Save to Design Attr" (page 77) in Chapter VIII - The Test Vector Tool for the format and limitations of this data.

Test Vector Procedures

Multiple Test Vector Windows

Multiple Test Vector windows with completely independent operating modes, sets of test vectors and circuit associations can be created. For example, one window could be used to generate a repeating test pattern while another monitors outputs on a set of signals in a different sub-circuit block.

This is done by simply selecting the Test Vector item from the Tools menu in the DesignWorks window while another Test Vector window is already open. However, it is recommend that you become quite familiar with general Test Vector operation first.

Manipulating the Data Spreadsheet

The data spreadsheet section is used to enter your "test program", i.e. a sequence of signal values to apply to the circuit under test, and to view the resulting signal changes.

Time normally progresses downward, i.e. a single row of the spreadsheet represents one simulator time step. Once all these values are set and commands executed, the next line is selected.

Data Entry

Data is entered by double clicking in the desired cell and typing directly onto the spreadsheet. Cells can contain signal value data and commands which indicate which signals to deal with and control execution of the test program. Commands always start with a "$" character to distinguish them from data.

In the simplest case, you can enter a sequence of signal values in one column and view the resulting outputs in another column. You can step through the test data interactively using the "Step" or "Walk" buttons, or set up all the events specified in your test program in one block, using the Copy To Timing menu command.

Each of the signal value columns should begin with a "header", i.e. a command listing the signals to be used in that column and whether they are inputs or outputs. If no header is specified in a given column, that column will be ignored and its contents are effectively comments.

Selecting Multiple Cells

Any rectangular block of cells can be selected by clicking in the cell at one corner of the block then, while holding the shift key down, clicking at the opposite corner. The selected cells can then be operated on using the commands in the Edit menu. Drag selection is not supported.

Keyboard Operations

The keyboard can be used to move within cells and between cells:

Key	**Comment**
Escape	Cancel changes made during this edit session (only valid during cell editing) and return to non-edit mode.
Enter (or Shift Enter)	Accept changes in the current cell (if editing), move to the next cell lower (higher) and begin editing this cell.
Tab (or Shift Tab)	Accept changes in the current cell (if editing), move to the next cell to the right (left) and begin editing this cell.
Arrow Keys	Move within a cell (if editing) or between cells.

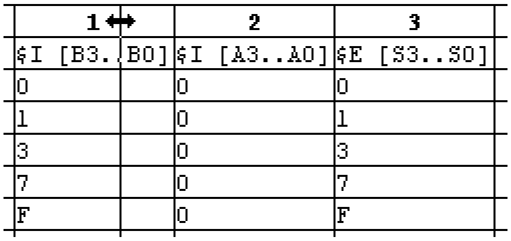
Clipboard Operations

The standard Edit menu operations Cut, Copy, Paste and Clear can be used on any block of selected cells or on a segment of text from a single cell.

See "Test Vector Procedures / Clipboard Operations" (page 89) in Chapter VIII - The Test Vector Tool for more information on these commands.

Sizing Rows & Columns

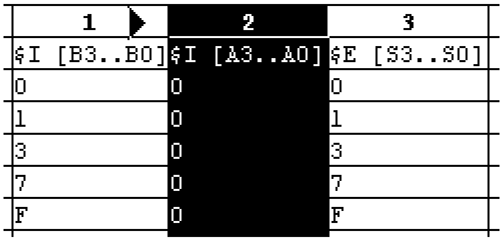
Rows and columns may be resized by moving the cursor into the row or column heading area and positioning it over the horizontal (row) or vertical (column) dividing line. When correctly positioned, the cursor will change shape. Press the mouse button and drag the line to size the row or column as desired.



1	2	3
$I [B3..B0]	$I [A3..A0]	$E [S3..S0]
0	0	0
1	0	1
3	0	3
7	0	7
F	0	F

Moving Rows & Columns

Rows and columns are reordered by moving the cursor into the row or column heading area and positioning it over the row or column heading to move. Press the mouse button and drag the label to its new position using the arrow to determine where the final “drop” position will be.



1	2	3
$I [B3..B0]	$I [A3..A0]	$E [S3..S0]
0	0	0
1	0	1
3	0	3
7	0	7
F	0	F

Clipboard Operations

Selecting Multiple Cells for Clipboard Operations

The clipboard operations Cut, Copy and Paste can operate on the selected characters in a single cell or on any rectangular group of cells in one operation. To select a group of cells, click at the top-left corner of the group, then hold the shift key down while clicking at the bottom-right corner. All the cells in the group will be highlighted.

NOTE: The Paste does not require a group to be selected. It always pastes all the data on the clipboard starting at the top-left cell currently selected. The size of the selected area is not significant.

Exchanging Clipboard Data With The Timing Tool

The Timing tool uses a clipboard data format which is a subset of the command structure supported by Test Vector. Therefore, data copied from a timing window can always be pasted into a Test Vector window. Conversely, a Test Vector test program can be copied and pasted into the Timing window as long as it is restricted in format. In general, the Timing tool ignores all control commands such as $REPEATand $WAIT, as well as any expected output specifications.

See Appendix A - Timing Text Data Format (page 131) for a description of the file format.

Data Format for Clipboard Operations

The Cut, Copy and Paste commands operate on text data in the following form:

- text for top-left selected cell, followed by a tab character, followed by text for next cell to the right, followed by a tab, etc. to the end of the selected cells in the row.
- the last cell in a row is followed by a carriage return character instead of a tab.
- if a selected cell is empty, the preceding and following tab characters will indicate its place.

Test Vector Modes

The Test Vector tool has three basic modes of operation:

- In "interactive mode" (the default), the test vectors and instructions entered in the spreadsheet are evaluated, new signal values are set up in the simulated circuit, and resulting outputs are displayed before proceeding. This mode gives a great deal of interaction and control over the simulation, but at the expense of execution speed. Interactive mode is controlled using the "Step" and "Walk" buttons.
- In "monitor mode", the Test Vector acts as a passive display window for simulation results. In this mode, the circuit is checked after each simulated time step and any desired signal values are displayed in the spreadsheet. Monitor mode is invoked by selecting the Monitor Mode menu command.
- In "copy to timing" mode the data in the data spreadsheet are evaluated and a sequence of signal value changes are set up in the simulator corresponding to all the test vectors. The Test Vector then allows the simulator to proceed to evaluate the scheduled inputs without further interaction. This method of setting up simulation inputs allows the simulator to proceed at full speed, but gives no user interaction during the simulation (of course, the normal DesignWorks interaction with the circuit and the timing window are still possible). This process is invoked by selecting the Copy to Timing command.

Hierarchical Designs

Test Vector can be used with hierarchical designs, but each Test Vector window can be associated with only one circuit or sub-circuit at any given time. Multiple Test Vector windows can be open at one time and associated with different sub-circuits. Hierarchical signal names (i.e. prefixed with the names of parent devices) are not supported.

Test Vector-to-Circuit Association

When a Test Vector window is opened, it by default links to the current circuit (i.e. the one displayed in the frontmost document window). It will then search that one circuit for any signal names specified in the test program and perform operations only on that circuit.

It will remain linked to that circuit as long as the circuit is open, even if it is no longer the frontmost. If the circuit is closed, the Test Vector will become inactive until it is attached to a new circuit using the Link to Circuit command in the Options menu.

NOTE: Test Vector can only be used on circuits that are open for editing.

Interaction With Other Tools

The Test Vector can be used in conjunction with the Timing tool and other simulation display and control mechanisms, however the following points should be noted:

- Results will be unpredictable if two tools are attempting to modify the value of the same signal. This can be caused by the Test Vector modifying a signal that has also been drawn in the timing window, multiple Test Vectors driving the same signal, or attempting to change the value of a signal that is being driven by a device on the schematic.
- By default, the Test Vector proceeds to the next step in the test program only after all signal states in the design have settled (i.e. there are no signal change events pending). If the design contains any clock devices or oscillations, or if some other tool is creating signal change events, the design may never settle and the next step will never be executed.

Test Vector Commands

Row and Column Usage

In the spreadsheet area, each horizontal row usually corresponds to one "test vector", i.e. a set of signal stimuli and comparison data that is applied at a particular time step in a simulation. Until the limit of 32,767 rows is reached, you can always enter new data into the empty rows displayed at the bottom of the spreadsheet.

The cells in the spreadsheet have no fixed meaning. Their usage at any given time depends upon the last command that was encountered in that column.

Normally, we recommend using separate columns for inputs, outputs, and commands, but it is quite possible to mix them in a single column.

$Commands

Test Vector commands always consist of a "$" character followed by a keyword. Commands fall into two groups:

- *Definition Commands* indicate how the following cells in the same column are to be interpreted. These usually give a list of signal names whose values will be specified or observed. Definition commands have no immediate effect on the execution of the test program or on the circuit under test.
- *Control Commands* allow specific loop control, delay or other control actions to be specified.

NOTE: For compactness, command keywords can always be shortened to the minimum number of letters needed to distinguish them from other commands. In most cases, only one letter is needed, e.g. "$I" can be used instead of "$INPUTS". The minimum contraction is shown in the command list below.

Note the following rules concerning command execution:

- Commands have to be "executed" (i.e. stepped through) before they take effect. E.g. if you select the cell under a command and then step downwards from there, the command will have no effect.
- Non-command cells (i.e. anything not starting with a "$") will be ignored completely unless a previous definition command indicates how to interpret them.
- A definition command in a given column always completely overrides any previous definition command in the same column.
- If the circuit schematic is modified while a test program is in progress, all current signal lists are invalidated and no signal data will be interpreted or displayed until the next definition command is encountered.

Definition Commands

NOTE: The $DELAY command can be used either as a control command (if a delay value follows the command keyword) or definition command (if the $DELAY command appears alone in a cell). It appears in both sections below.

Command	**Short Form**	**Description**
$TIME	$T	Display current simulation time in the following cells.
$INPUTS signalList	$I	Specify the values of input signals.
$OUTPUTS signalList	$O	View the values of the given signals (can actually be used to view the "real" values of inputs as well).
$EXPECTED signalList	$E	Specify the expected values of the given signals
$WAIT signalList	$W	Pause the test program until a specified value is detected.
$DELAY	$D	Delay by the specified amount before executing this line.

Signal List Format

Many of the definition commands require that a list of signals be specified. The format of this list can be seen in the following examples:

CLK

specifies simply the single signal "CLK". In this case, each test vector will consist of a single entry, e.g. "0" or "1".

A B C D

specifies four separate signals A, B, C and D. In this case, each test vector will consist of four values separated by spaces, e.g. "0 1 1 0".

[A B C D]

uses the grouping operator [] to indicate that the four signals will be specified by a single hexadecimal number with A in the most significant bit position. E.g. the hex test vector "E" (or 1110 binary) would assign a 1 value to A, B, and C, and a 0 value to D.

NOTE: For compatibility with the Timing tool, Test Vector will allow a "group name" to be prefixed to a group, e.g. "CONTROL[A B C D]". This group name is not used internally and will be ignored. For this reason, there must be a blank between any preceding signal name and the opening "[" or the name will be ignored.

[D3..0]

uses the ".." operator to indicate a sequence of numbered signals. This case also uses the [] to indicate that the four signals D0, D1, D2 and D3 are to be treated as a single integer, in this case with D3 in the most significant position.

CLK CLR [D15..0] [FC2..0]

specifies four groups, the first being the signal CLK, the second being the signal CLR, the third being the numbered signals D0 through D15, and the last being the set FC0, FC1 and FC2. A typical test vector for this group might be:

.C 0 B3 5

In this case, the ".C" specifies that a clock pulse is to be applied to the CLK signal, the value 0 (low) to the CLR, the next sequential integer value to the 16 signals D0, D1, etc. and the hex value "5" (or 101 binary) to the signals FC2, FC1 and FC0. See definitions of these codes below.

The $INPUTS Command

All cells below this command in the same column are used to specify the input values to be set up in the circuit under test. As each line of the test program is read, the cell in this column is interpreted and new values are placed on the signal lines in the circuit. This command must be placed in the first row of this column to specify which signals are to be set up. The values specified in all subsequent rows are interpreted according to that header. See the information on header format above.

All non-command cells in this column are interpreted as signal value data. A data cell should contain one signal value item for each signal group specified in the most recent header (a signal group being either a single signal, or a group specified in brackets [] in the header. If the cell is empty or if less items are specified in the data cell than in the header, then unspecified items will be left unchanged from the previous time they were specified.

Each signal value item can be one of the items in the following table.

Value	**Comment**
A hexadecimal value	This will be interpreted as an integer with one bit for each signal in the corresponding header group. For individual signals, only the least significant bit is used, so the only values will be 0 or 1. For multiple-signal groups (i.e. specified in []'s in the header), the last signal in the group will take the value of the least significant bit and preceding signals will take the values of successively higher numbered bits. Unused high-order bits are ignored.
The value code ".C" or ".c"	This code indicates a positive clock pulse (i.e. a transition to the 1 level, followed by a transition to the 0 level), with a width as specified in the Configuration command. Note that if the signal is already at a 1 level, no rising transition will occur. For more information see the note below.
The value code ".N" or ".n"	This code indicates a negative clock pulse (i.e. a transition to the 0 level, followed by a transition to the 1 level), with a width as specified in the Test Options command. Note that if the signal is already at a 0 level, no falling transition will occur. See also the notes below.
The value code ".Z" or ".z"	This code indicates a high impedance value.
The value code ".X" or ".x"	This code indicates a “Don’t Know” value and can be used to test a circuits response to unknown inputs.
The increment operator "++" or "++n"	This code indicates that the integer value specified previously is to be incremented and the new value applied. If an integer is supplied immediately following the "++" then this number is added to the old input value, otherwise 1 is used.
The decrement operator "--" or "--n"	This code indicates that the integer value specified previously is to be decremented and the new value applied. If an integer is supplied immediately following the "--" then this number is subtracted from the old input value, otherwise 1 is used.

NOTE: Care must be taken regarding setup times when using the .C or .N clock pulse operators. E.g. .C generates a rising transition immediately which may not allow any setup time after values settled from the last step. This situation can be avoided in most cases by using .N to test positive-edge-triggered devices and .C for negative-edge-triggered devices. This ensures that a setup time at least the width of the clock pulse is allowed.

Command examples:

```
$I [D3..0]
$I CLK CLR LD/
$I FC0..2
```

The $EXPECTED Command

This command indicates that the following cells will contain the expected values for signals in the circuit under test. The values in the circuit are compared to those specified. The result of this comparison is shown as a mark character in each data cell. If the actual values equal the expected ones, then an "=" is displayed in the cell, otherwise a "#" is displayed. If no value appears in a data cell, then no comparison is done.

Each signal value item can be one of the items in the following table.

Value	**Comment**
A hexadecimal value	This will be interpreted as an integer with one bit for each signal in the corresponding header group. For individual signals, only the least significant bit is used, so the only values will be 0 or 1. For multiple-signal groups (i.e. specified in []'s in the header), the last signal in the group will take the value of the least significant bit and preceding signals will take the values of successively higher numbered bits. Unused high-order bits are ignored.
The value code ".X" or ".x"	This code indicates a “Don’t Know” value and can be used to test a circuits response to unknown inputs
The value code ".Z" or ".z"	This code indicates a high impedance value.
The value code ".DC" or ".dc"	This code indicates a "Don't Care", i.e. it will always match. Note that this is different than not specifying a value, in that this will display an "=" result, whereas no comparison is done if no value is given.

The comparison is done at the same time step that the inputs are set or after waiting for the simulation to settle. See the menu command “Configuration...” (page 80) for more information.

Command examples:

```
$E [ADDR0..15]
$E EQ
$E A B C D
```

The $OUTPUTS Command

This command specifies that the following cells should show the actual values read from the circuit and the result of the comparison with the expected values. Any cell in this column not containing a command will be overwritten when the outputs are read.

Command examples:

```
$O [ADDR0..15]
$O EQ
$O A B C D
```

The $TIME Command

This command specifies that the following cells should show the simulation time when execution of this step is started. Time values are displayed in decimal.

Command examples:

```
$TIME
```

The $WAIT Command

This command specifies that the following cells will contain signal value conditions to wait for before proceeding. This command is similar to $EXPECTED, except that instead of displaying an equal or non-equal status, the test program waits until an equal status occurs. Note the following rules:

- If the wait condition is already met, then the test program proceeds and the wait has no effect.
- The test program enters the wait state after any new input values specified in the row have been set up.
- If any delay is specified in the same row, then the delay is applied concurrently and it becomes, in effect, a minimum wait time.

Command examples:

```
$WAIT SYNC
$WAIT [D7..0]
```

The $DELAY Command

This command specifies that the following cells in the same column should be scanned for a decimal integer delay value. The following rules are applied:

- If a value is found, the specified delay is applied after setting up any input value specified in this row. I.e. This specifies the duration of any values applied on this line.
- If multiple $DELAYS appear in one row, then only the longest will apply.
- If no value appears in this column, then the default delay will apply.

Command examples:

$DELAY

NOTE: If any value appears after the $DELAY keyword, then it is taken to be a control command and the following cells in the column will not be interpreted as delays. See the description of this format below.

Control Commands

Command	**Short Form**	**Description**
$REPEAT or $REPEAT nnn	$R	Repeat rows up until the next $END either indefinitely or nnn times.
$END	$EN	Marks the end of a REPEAT loop.
$DELAY nnn	$D	Delay for nnn time units after applying the input values specified in this row.
$STOP	$S	Stop execution after this row.

$REPEAT Command

The $REPEAT/$END construct allows you to specify any integer number or an infinite repetition of a group of test vectors. The format of the $REPEAT command can be seen in the following example:

	$IN CLK	$IN CLR	$IN [D3..0]	$OUT [Q3..0]
	0	0	0	
	0	1	0	
	0	0	0	
$REP 32	0	0	++	
$END	1	0		

In this case, all vectors from the line containing the $REPEAT down to and including the line containing the $END will be repeated 32 times. If the repetition factor is omitted, it will be taken as infinite in interactive mode, and 1 in timing setup mode.

$REPEATs can be nested to any desired depth. If the $END is omitted, the last line in the data spreadsheet that has been modified is taken as the end of the loop.

$DELAY Command

When the $DELAY command appears with a value after it, it is taken as a control command and the specified delay is applied after execution of the current row. See the description of the $DELAY definition command above for more details.

Command example:

```
$D 15
```

$STOP Command

This command causes the test program to cease execution after applying any new input values specified in this row.

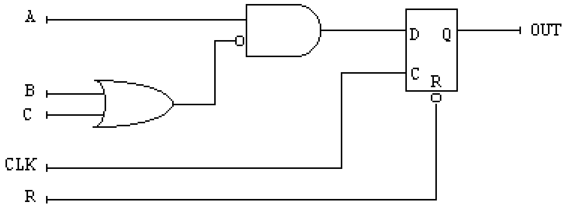
Test Vector Examples

This section provides some simple examples of Test Vector usage. For more complex examples, see the example files provided with the simulator release.

NOTE: Most of the devices in the Primitive device libraries contain stored test vectors and can be used in conjunction with the Place and Test Device command (page 85 in the Options menu for example purposes.

Testing a Circuit

In this example we will execute a set of test vectors for the following simple circuit made up of primitive devices:



Entering the Test Program

NOTE: This file is included with the Simulator release and can be loaded using the Open Test File command in the File menu.

With the above circuit open in the frontmost schematic window, follow these steps:

- Select the Test Vector item in the Tools menu. An empty Test Vector window will appear.
- Select the “Get Header” command in the Options menu.
- For this example, the "Include Time Column" switch was turned off, although you can optionally leave it on and shift all entries right one column.
- Click OK on the Default Header dialog. This will create the header entry for the first row.

- Enter the remaining cells as shown below. It may be convenient to enter the first few lines manually then use the Fill Down command in the Edit menu to set the default values in the remaining rows.

$I R	$I CLK	$I C	$I B	$I A	$E OUT
0	0	1	1	0	0
1	0	1	1	0	0
1	.N	1	1	0	0
1	.N	0	1	0	0
1	.N	0	0	0	0
1	.N	0	0	1	1
1	.N	1	0	1	0

We are now ready to execute the test program. You may wish to save the current entry using the “Save Test File” command in the File menu.

Executing the Test Program

To execute the test program:

- Click on the Reset button. This will select the top-left cell in the spreadsheet.
- Click on the Walk button. If the simulator is stopped, you will be prompted to start it, otherwise the program will be executed.

If all Test Vector and delay settings are the default values, this program will fail in the last step, due to a setup time problem. This will be explained in the next section.

Dealing With Delays and Setup Times

In this example, we are feeding inputs through a combinational logic array to an edge-triggered device. We must be aware that a delay must be allowed between changes at the combinational inputs and the application of the clock edge to the flip-flop.

In this case, the flip-flop is positive-edge-triggered. We have used the ".N" negative clock pulse notation since this in effect inserts a delay equal to the pulse width before the rising edge. This is because is creates a negative edge first (which is not significant to this flip-flop), then waits for the specified pulse width, then inserts the positive edge.

The default Test Vector clock pulse width is 2 units and the default delay in primitive devices is 1 unit. The last test vector in the above program causes a logic transition to pass through two gates, for a total delay of 2 units. Thus, the input transition at the D input to the flip-flop arrives at the same time as the positive edge of the clock. The D flip-flop model always uses the old value at the input, so the new value does not reach the Q output.

There are two solutions to this problem:

- Change the Test Vector clock pulse width using the "Configuration" command in the Options menu. This will affect all clock pulses generated by this Test Vector window.

NOTE: These settings are not saved with the test file and will return to default values the next time the program is restarted.

- Change the test program to set the clock input explicitly and insert a delay at this line, as in the following changes to the final line:

1	.N	0	0	1	1
$DEL 5	0	1	0	1	.DC
1	1	1	0	1	0

In this case, we are replacing the .N operator with two separate test vectors with explicit 0 and 1 values for the clock and a delay of 5 in between.

Test Vector as a Pattern Generator

Test Vector can also be used as a repeating test pattern generator or even as a replacement for a device or bus interface which is not present in the circuit design.

In this case, we will use the $WAIT command to synchronize the values generated by the Test Vector with the design under test. When the SYNC signal enters the 0 state, the D0..3 lines will be set to zero. When SYNC goes to 1, the Test Vector will cycle through a Gray code sequence and then loop back to the beginning.

Creating Fixed Step Sizes

By default, the Test Vector executes each step when the simulation settles after the previous one. This is appropriate when the test program is driving the circuit and evaluating the outputs, but in this case we want a fixed step size, regardless of other simulation activity. To do this:

- Select the Configuration command in the Options menu.
- Enter the value 10 into the Default Delay box.
- Select the After Specified Delay option.
- Click OK on this dialog.

The Test Vector will now proceed at fixed intervals whether or not the simulation has settled.

The Test Program

The following test program performs the functions outlined above:

	$WAIT SYNC	$I [D3..0]
$REP	0	
	1	0
		1
		3
		2
		6
		7
		5
		4
		C
		D
		F
		E
		A
		B
		9
$END		8

As a variation on this, removing the 0 and 1 entries from the $WAIT column or deleting the column altogether will cause the test program to loop continuously without synchronization.

This page has been intentionally left blank.

Chapter IX - Primitive Devices

This chapter provides information on the "primitive" or built-in device types in DesignWorks. These types are intended primarily for use in creating model circuits for higher-level macro devices. Because their simulation functions are hard-coded, they occupy much less memory space than macro devices and simulate more efficiently.

IMPORTANT NOTES:

1) In primitive devices, the association of logic functions to pins on a device symbol are made by pin order. When creating primitive devices using the DevEditor tool, you must be aware of the pin order requirements for the device type you are using. Refer to the description of each type in this chapter and to Appendix D - Primitive Device Pin Summary.

2) Bus pins are not supported on primitive device types.

The following table lists the available primitives and their functions.

Primitive Type	**Description**	**Variations using Same Primitive Type**	**Max. Size**
NOT	Inverter		1 input
AND	N-input AND gate	Any pin inversions	799 inputs
NAND	N-input NAND gate	Any pin inversions	799 inputs
OR	N-input OR gate	Any pin inversions	799 inputs
NOR	N-input NOR gate	Any pin inversions	799 inputs
XOR	N-input XOR gate		799 inputs
XNOR	N-input XOR gate		799 inputs
Transmission Gate	Transmission gate		1 bit
Three-State Buffer	Non-inverting N-bit 3-state buffer with optional common inverted enable	Buffer (non three state)	400 bits
Resistor	Digital resistor		
Multiplexer	M*N-to-M multiplexer		256 inputs
Decoder	1-to-N line decoder		256 inputs

Adder	N-bit adder with carry in and out	Incrementer	256 bits
Subtractor	N-bit subtractor with borrow in and out	Decrementer	256 bits
D Flip-Flop	D-type flip-flop	optional S & R	1 bit
JK Flip-Flop	JK flip-flop	T flip-flop, optional S & R	1 bit
Register	N-bit edge-triggered register		256 bits
Counter	N-bit synchronous counter	up/down	256 bits
Shift Register	N-bit shift register		256 bits
One Shot	Retriggerable one shot		
Clock	Clock oscillator		
Binary Switch	Debounced toggle switch		
SPST Switch	Open/closed single pole switch		
SPDT Switch	Double throw switch		
Binary Probe	Signal level display		
Hex Keyboard	Hexadecimal input device		
Hex Display	Hexadecimal digit display		
SetupHold	Setup/Hold time checker		
Unknown	Unknown value detector		

The following table lists devices supported primarily for compatibility with older version of DesignWorks. We do not recommend using these in new designs.

Device	**Description**
Pullup	Pullup resistor, single pin
D Flip-Flop ni	D-type flip-flop (non-inv S & R)
JK Flip-Flop ni	JK flip-flop (non-inv S & R)
Glitch	Glitch detector

SimStop	Simulation halt device

Gates and Buffers

The Primitive Gates library contains the primitive gates with a built-in simulation function. The NOT, AND, NAND, OR, NOR, XOR and XNOR devices behave according to the appropriate truth tables for such devices. Any gate input which is in the “Don’t Know”, “High Impedance” or “Conflict” state is treated as a “Don’t Know”. A gate with a “Don’t Know” input will not necessarily produce a “Don’t Know” output. For example, if one input on an AND gate is low, the output will be low, regardless of the state of the other input, as in the following truth table.

Gate Definition

The gate types, except NOT, can be created with any number of inputs (from 0 to 799) and are defined as shown in the following table.

Function	Output is...	Output is DONT if...
AND	LOW if any input is LOW, HIGH otherwise	Some input is DONT and no input is LOW
NAND	HIGH if any input is LOW, LOW otherwise	Some input is DONT and no input is LOW
OR	HIGH if any input is HIGH, LOW otherwise	Some input is DONT and no input is HIGH
NOR	LOW if any input is HIGH, HIGH otherwise	Some input is DONT and no input is HIGH
XOR	HIGH if an odd number of HIGH inputs and no DONTs	Any input is DONT
XNOR	HIGH if an even number (or zero) of HIGH inputs and no DONTs	Any input is DONT

Gate Pin Order

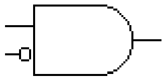
The NOT type must have exactly one input and one output, in that order. All other logic gate types can have any number of inputs followed by a single output, up to the maximum DesignWorks limit of 800 pins.

NOTE: Pin order is important in all primitive devices! When creating a gate type using the DevEditor tool , the output pin must be the last item on the pin list.

Pin Inversions

The logic of any pin on any device can be inverted by placing a non-empty value in the Invert.Pin attribute field of the pin.

For example, to create the following AND gate with one inverted pins:



the following steps must be taken in the DevEditor tool:

1) Create the desired graphic symbol using the DevEditor's drawing tools.
2) Place the three pins as shown. ORDER IS IMPORTANT! All primitive devices must have a specific pin order. For gates, all inputs come first and the output pin last.
3) In the pin list, double-click on the last pin (the output pin). This will display the Pin Info palette for that pin.
4) Set the pin type to Output, and, if desired, edit the pin name.
5) Press the Enter key on the keyboard to move to the next pin and edit the other pin names, if desired, and check that they are all set to Input.
6) Click the close box in the Pin Info palette.
7) In the pin list, click once to select the input pin that is to be inverted, then select the Pin Attributes command in the Options menu.
8) Select the All Fields option in the attributes dialog then select the Invert.Pin field.
9) Enter the value "1" for this field, then click Done. (The actual value doesn't matter, as long as it is non-empty.)
10) Select the Sub-circuit / Part Type command in the Options menu
11) Click on the Primitive Type button, then select the AND primitive type in the menu.
12) Close the Part Type dialog and save the part to a library in the usual manner.

NOTES:

1) The logical inversion of the pin is <u>completely independent</u> of the graphical representation of the pin. I.e. using the "inverted pin" graphic in the DevEditor tool <u>does not</u> invert the pin logic in the simulator. You must set the Invert.Pin field to have this effect.

2) Inverted gate types NAND and NOR can be created by using the NAND and NOR primitive type setting or by using the AND and OR settings and inverting the output pin (or the input pins, using DeMorgan's Theorem). The two methods will produce identical simulation results. There is a slight memory overhead but no execution speed overhead to using an inverted pin.

Transmission Gate

The transmission gate (X-Gate) device behaves as an electrically controlled SPST switch. When the control input is high, any level change occurring on one signal pin will be passed through to the other. Since it has no drive capability of its own it will behave differently than a typical logic device when a high-impedance or low drive level signal is applied to its signal inputs. Most other primitives, such as gates, interpret any applied input as either High, Low or Don't Know. The transmission gate, on the other hand, will pass through exactly the drive level found on its opposite pin. Thus, a high-impedance level on one pin will be transmitted as a high-impedance level on the other pin. Note that the simulation of this device may produce unpredictable results in extreme cases, such as an unbroken ring of transmission gates.

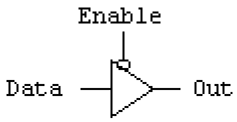
NOTE: No variations in number or order of pins are possible with the XGATE primitive type. It must have exactly one control pin and two bidirectional pins with pin order as described in Appendix D - Primitive Device Pin Summary.

Three-State Buffer

The three-state buffer has N data inputs, N data outputs, and an optional active-low enable input. If the enable input exists and is high, all outputs enter a "High-impedance" state. If the enable input doesn't exist, or is low, each output will follow the corresponding input if it is low or high, or produce a "Don't Know" level otherwise.

A single-input three-state buffer is shown in the following table:

Enable	Data	Out
0	0	0
0	1	1
1	0	Z
1	1	Z



Making Non-Inverting Buffers

The Buffer primitive type can also be used to make a non-inverting buffer, i.e. with its outputs always enabled, simply by omitting the enable input.

This can be used for the following purposes:

- To represent a non-inverting buffer or level translator in a design.
- To insert a delay in a signal path without affecting the logic of the signal.
- To create various types of open collector, open emitter or inverting buffers, when used in conjunction with different pin type and inversion settings on the outputs. NOTE: It is more efficient to use the NOT primitive type to make a simple inverter.

Resistor

The resistor device simulates the effects of a resistor in a digital circuit. It is more general than the Pullup Resistor device and can be used as a pullup, pulldown or series resistor. Whenever a signal level change occurs on either pin of the resistor, the device converts that level into a resistive drive level (see Chapter III - Simulation Details (page 5) for more information on drive levels). A high impedance drive on one end is transmitted as a high-impedance output to the other end. Note that DesignWorks does not simulate analog properties of devices so the resistor device does not have a resistance value in the analog sense and will not interact with capacitor symbols placed on the same line. The effect of resistance on line delay can be simulated by setting the delay of the resistor device.

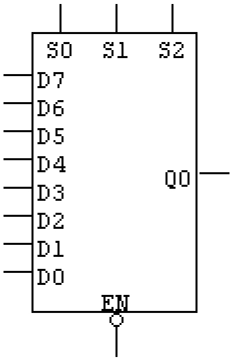
Logic Devices

Multiplexer

This is a device which selects one of N data inputs and routes it to a corresponding output line. There can be from 1 to 256 outputs, plus an optional enable input, as long as the total pin count does not exceed the 800 pin limit.

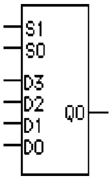
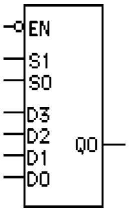
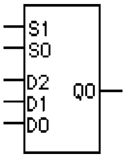
A typical 8-to-1 multiplexer obeys the following function table. X = Don't Care.

EN	S2	S1	S0	7	6	5	4	3	2	1	0	G
0	0	0	0	X	X	X	X	X	X	X	0	0
0	0	0	0	X	X	X	X	X	X	X	1	1
0	0	0	1	X	X	X	X	X	X	0	X	0
0	0	0	1	X	X	X	X	X	X	1	X	1
0	0	1	0	X	X	X	X	X	0	X	X	0
0	0	1	0	X	X	X	X	X	1	X	X	1
0	0	1	1	X	X	X	X	0	X	X	X	0
0	0	1	1	X	X	X	X	1	X	X	X	1
0	1	0	0	X	X	X	0	X	X	X	X	0
0	1	0	0	X	X	X	1	X	X	X	X	1
0	1	0	1	X	X	0	X	X	X	X	X	0
0	1	0	1	X	X	1	X	X	X	X	X	1
0	1	1	0	X	0	X	X	X	X	X	X	0
0	1	1	0	X	1	X	X	X	X	X	X	1
0	1	1	1	0	X	X	X	X	X	X	X	0
0	1	1	1	1	X	X	X	X	X	X	X	1
1	X	X	X	X	X	X	X	X	X	X	X	1



Multiplexer Pin Variations

A number of variations in multiplexer logic are possible with this primitive type, depending on which input and output pins are included. The following table summarizes the possible variations. Samples are shown with M=1 and N=2, but any combination of M and N can be used within the maximum pin limit of 800.

Number of Sections	**Number of Inputs/Section**	**Number of Select Inputs**	**Number of Enable Inputs**	**Sample Symbol**
M	2^N	N	0	
M	2^N	N	1	
M	$2^{N-1}+1$ .. 2^N	N	0*	

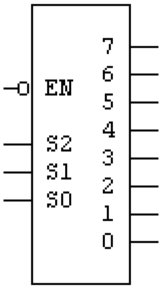
* If there are less than 2^N inputs per section there can be no enable input.

See Appendix D - Primitive Device Pin Summary (page 147) for specific pin order requirements for multiplexer types.

Decoder

The Decoder activates one of N outputs, depending on M select inputs, as follows (X = Don't Care):

EN	S2	S1	S0	7	6	5	4	3	2	1	0
0	0	0	0	1	1	1	1	1	1	1	0
0	0	0	1	1	1	1	1	1	1	0	1
0	0	1	0	1	1	1	1	1	0	1	1
0	0	1	1	1	1	1	1	0	1	1	1
0	1	0	0	1	1	1	0	1	1	1	1
0	1	0	1	1	1	0	1	1	1	1	1
0	1	1	0	1	0	1	1	1	1	1	1
0	1	1	1	0	1	1	1	1	1	1	1
1	X	X	X	1	1	1	1	1	1	1	1



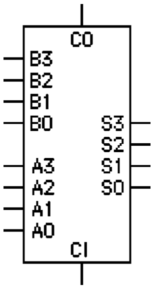
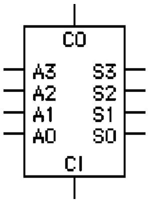
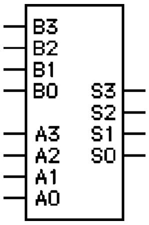
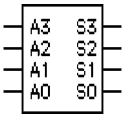
Adder/Incrementer

The N-bit Adder accepts one or two N-bit input arguments and (optionally) a 1-bit carry and outputs their N-bit sum plus an optional 1-bit carry out.

Multiple adders can be connected together by feeding the carry out from each stage to the carry in of the next more significant stage. The carry in to the least significant stage should be set to zero.

Adder Pin Variations

The adder primitive can be used in four variations, as summarized in the following table. Sample symbols are shown with 4 bit inputs, although any number of bits from 1 to 256 is permissible.

	Has B Inputs	**No B Inputs**
Has Carry In	S = A + B + Cin  CO B3 B2 B1 B0 S3 S2 S1 S0 A3 A2 A1 A0 CI	S = A + Cin 
No Carry In	S = A + B  B3 B2 B1 B0 S3 S2 A3 S1 A2 S0 A1 A0	S = A + 1 

In addition, the Carry Out pin can be independently included or omitted from any of these configurations.

See Appendix D - Primitive Device Pin Summary (page 147) for specific pin order requirements for adder/incrementer types.

Subtracter/Decrementer

The Subtracter primitive type behaves identically to the Adder type except that a subtract or decrement operation is performed, depending upon pin configuration.

D Flip-Flop

The D-type flip-flop is positive-edge triggered and obeys the following function table:

S	**R**	**D**	**Clock**	**Q**	**Q/**
0	0	X	X	1	1
0	1	X	X	1	0
1	0	X	X	0	1
1	1	0	Rises	0	1
1	1	1	Rises	1	0
Rises	Rises	X	X	X	X

In the above table X on the input side means "Don't Care" and on the output side means "Don't Know".

Flip-Flop Setup and Hold Times

None of the DesignWorks primitive types explicitly implement variable setup and hold times. However, all edge-triggered primitives have an effective setup time of 1 unit since they always use the input signal value existing before the current step. I.e. if the data input changes at the same time as the clock, the old data value will be used to determine the new output value. This effective setup time can be modified by specifying input pin delays on either the data or clock pins.

Checking for setup and hold violations can be done in one of two ways:

- Attaching a SetupHold primitive device to the inputs and output of the device under test. The combined device could be made into a sub-circuit if desired to combine it into a single symbol.
- Using the Trigger capability of the simulator to watch for value changes within a set amount of time.

Flip-Flop Initialization

Note that when a flip-flop is first placed in the schematic, it is in an unknown state and must be correctly initialized before it will produce predictable outputs. This can be done in the following ways:

- Adding circuitry to force an explicit reset.
- Using the Clear Unknowns command in the Simulator window to force an initial state before starting the simulation.
- Specifying an initial output value for both the Q and Q/ outputs in their respective Initial.Pin attributes. This will be applied every time a Clear Simulation command is executed.

D Flip-Flop Optional Pins

The D Flip-Flop primitive type has the following optional pins:

- The Q/ (Not-Q) output can always be omitted.
- The Set input alone or both the Set and Clear inputs can be omitted.

> *See Appendix D - Primitive Device Pin Summary (page 147) for specific pin order requirements for D Flip Flop types.*

D Latch

The D Latch primitive type is identical to the D Flip-Flop in function and pin specifications except that it is level-triggered instead of edge-triggered. I.e. The Q and Q/ outputs will follow the level of the D input as long as C is high.

JK Flip-Flop

The JK flip-flop is negative-edge triggered and obeys the following function table:

S	R	J	K	Clock	Old Q	New Q	New Q/
0	0	X	X	X	X	1	1
0	1	X	X	X	X	1	0
1	0	X	X	X	X	0	1
1	1	0	0	falls	0	0	1
1	1	0	0	falls	1	1	0
1	1	0	1	falls	X	0	1
1	1	1	0	falls	X	1	0
1	1	1	1	falls	0	1	0
1	1	1	1	falls	1	0	1
rises	rises	X	X	X	X	X	X

In the above table X on the input side means "Don't Care" and on the output side means "Don't Know".

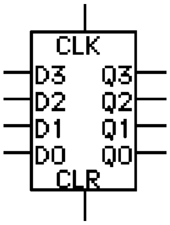
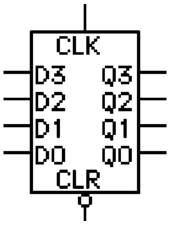
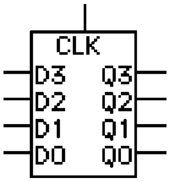
If any inputs are in an unknown state, the simulator will determine the output state where possible, or else set it to "Don't Know".

See Appendix D - Primitive Device Pin Summary (page 147) for specific pin order requirements for D Flip Flop types.

Register

This device implements an N-bit, positive-edge-triggered register with common clock and an optional active-high clear inputs.

The following table illustrates some pin variations available for the Register primitive type.

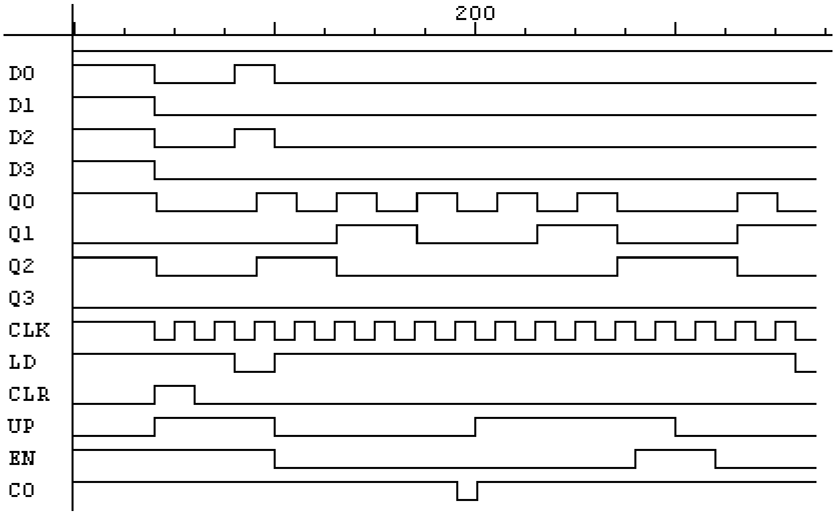
4-bit register with active-high clear	
4-bit register with active-low clear (using pin inversion)	
4-bit register without clear	

See Appendix D - Primitive Device Pin Summary (page 147) for specific pin order requirements for D Flip Flop types.

Counter

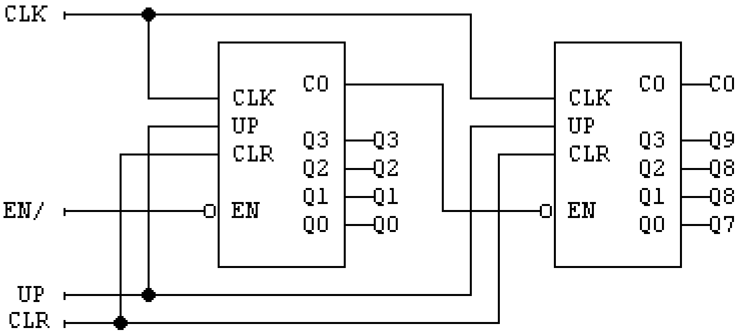
This device implements a N-bit, presettable, synchronous, positive-edge-triggered up/down counter with active-low enable. The load data inputs and most of the control inputs can be omitted for simplified versions.

The following timing diagram shows a typical count cycle. Note that the CO (Carry Out) output goes low when the count reaches 2^N-1 (when counting up) or 0 (when counting down) and rises again on the next count. This can be used to cascade multiple counters together, as shown. The CLR input clears the counter asynchronously (i.e. regardless of the state of the clock). The Count/Load input, when low, causes the data from the N data inputs to be passed to the outputs on the rising edge of the next clock. The Enable input disables counting when high, but has no effect on loading.



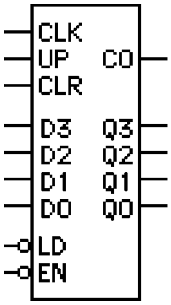
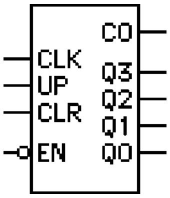
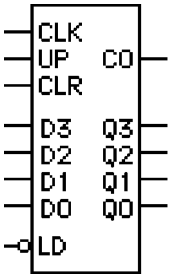
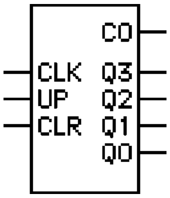
Cascading Multiple Counters

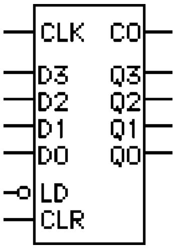
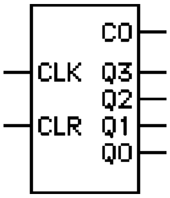
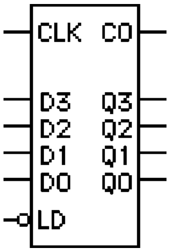
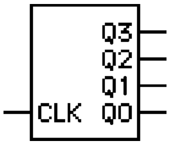
Counter primitives with the optional Enable and Carry Out pins can be cascaded to form larger synchronous counters as follows:



Counter Pin Variations

The following table summarizes the possible pin usage variations for the counter primitive type. The sample are shown with N=4, although the number of bits can be anywhere in the range 1 to 256.

Optional Inputs	**Including Load Inputs**	**Excluding Load Inputs**
CLR, UP/DN, ENABLE	 CLK UP CLR D3 D2 D1 D0 LD EN CO Q3 Q2 Q1 Q0	
CLR, UP/DN	 CLK UP CLR D3 D2 D1 D0 LD CO Q3 Q2 Q1 Q0	

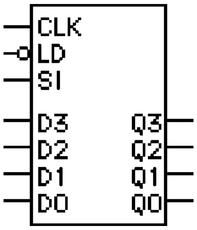
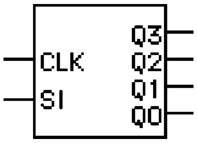
CLR	 CLK CO D3 D2 D1 D0 Q3 Q2 Q1 Q0 LD CLR	
none	 CLK CO D3 D2 D1 D0 Q3 Q2 Q1 Q0 LD	

NOTE: CO can be independently included or omitted in any of the above variations.

Shift Register

The shift register is an N-bit, positive-edge-triggered device with serial or optional parallel load. When the Shift/Load input is low, data from the N parallel data input lines is transferred to the outputs on the rising edge of the next clock. When Shift/Load is high, the next rising clock edge causes the value at the Shift In input (SI) becomes the new value for output Q0, Q0 shifts to Q1, Q1 to Q2, etc., and the old value at the most significant output is lost.

The following table shows the shift register primitive with and without parallel inputs.

With Parallel Load	**Without Parallel Load**
 CLK LD SI D3 D2 D1 D0 Q3 Q2 Q1 Q0	

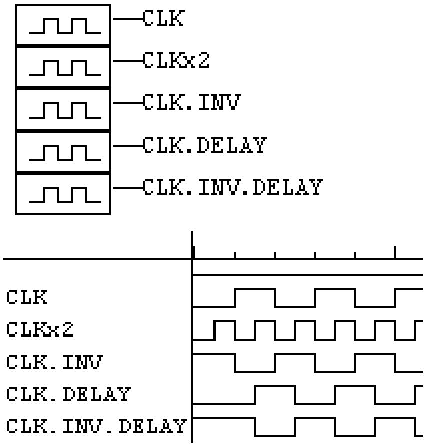
NOTE: The Shift Register primitive cannot be created without data outputs (i.e. a parallel-in serial-out register) because the flip-flop values are stored on the output pins. Primitive devices have no internal state storage. See Chapter III - Simulation Details (page 5) for more information.

Clock

The clock oscillator is used to generate a repeating signal to activate other devices. When it is first created, the clock output pin will be low, then after a delay time called the "Low Time" it will change to the high state. After a further delay called the "High Time" the signal will revert to low, and the cycle will repeat. The low and high times are initially set to 10, but can be modified using the Parameters command in the Options menu in the Simulator window. Any number of Clocks may exist at once with independent delay times.

Creating Synchronized or Offset Clocks

When the Clear Simulation command is selected (or the Restart button on the Timing palette pressed), all clocks in the design are restarted. Clock outputs will be set to the low state and the timer for the low period will be restarted. Clock high and low times, combined with pin inversion and pin delay settings can be used to precisely determine the relationship between two clock outputs. The following circuit example summarizes these options.



Signal	Low Time	High Time	Invert.Pin	Pin Delay
CLK	10	10		0
CLKx2	5	5		0
CLK.INV	10	10	1	0
CLK.DELAY	10	10		5
CLK.INV.DELAY	10	10	1	5

Setting Clock Values

To set the high and low times for a clock, first select the device in question (by setting the cursor to "Point" mode and clicking the mouse button inside the device symbol), then choose the Parameters command in the Options menu in the Simulator window.

You will be presented with a dialog containing buttons for increasing or decreasing the high and low values. The minimum for either value is 1 and the maximum is 32767.

> *See “Simulation Parameters” (page 43) in Chapter V - The Simulator Tool for more information.*

One Shot

The one shot is used to generate an output pulse of a fixed length when it is triggered by the rising edge of the trigger input. Two parameters can be set for a one shot:

- the delay from the rising edge of the input to the start of the output pulse.
- the duration of the pulse.

The delay and duration times are initially set to 1 and 10, respectively, but can be modified using Parameters command in the Options menu in the Simulator window.

The One Shot device is retriggerable, meaning that the output pulse will not end until "duration" time units has passed since the last trigger input. Repeating the trigger input can cause the output pulse to be extended indefinitely.

> *See “Simulation Parameters” (page 43) in Chapter V - The Simulator Tool for more information.*

I/O Simulation Pseudo-Devices

Binary Switch

The Binary Switch device allows setting of a signal to a low or high level. When a switch is first created its output is at a low level. Pointing at the switch with the cursor in "Point" mode and clicking the mouse button causes the switch arm to move and the output to change to the opposite state. Any number of device inputs can be driven by a switch output. A switch has no delay characteristic since it has no inputs. The switch can be selected by holding the shift key down and clicking on the switch device.

SPST Switch

The SPST switch device simulates the actions of a simple open/closed switch in a digital circuit. When a switch is first created it is open and both connections present a high impedance logic level. Clicking on the switch with the cursor in "Point" mode causes the switch arm to close and the switch to "conduct". In terms of the digital simulation, this means that whatever logic level is present on each pin is transmitted to the other one.

An SPST switch has a default delay of zero but this can be set to any value from 0 to 32767 using the Parameters command in the Options menu in the Simulator window. The switch can be selected by holding the shift key down and clicking on the switch device.

SPDT Switch

The SPDT switch device operates in essentially the same manner as the SPST switch described above, except that it is always conducts between the single pin on one side and one of the two pins on the other. As with the other two switch types, clicking on it with the Point cursor changes the position of the contact. The switch can be selected by holding the shift key down and clicking on the switch device.

Binary Probe

The binary probe is a device for displaying the level present on any signal line. When the probe is first created, its input is unconnected and therefore in the "High-impedance" state, which will be displayed as a "Z". When the input pin is connected to another signal, the displayed character will change to reflect the new signal's current state. Any further changes in the signal state will be shown on the probe.   Possible displayed values are 0 (low), 1 (high), X (Don’t Know), Z (High Impedance) or C (Conflict).

Hex Keyboard

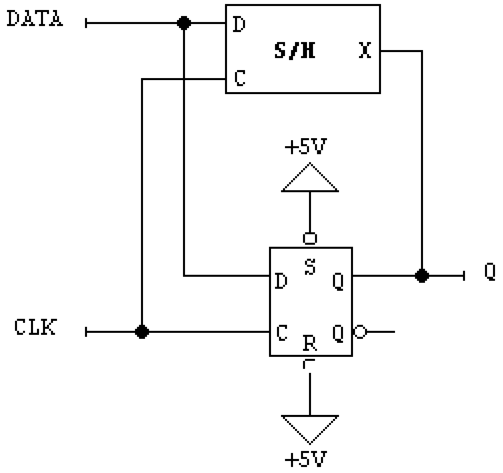
The hex keyboard outputs the binary equivalent of a hexadecimal digit on four binary lines. A "key" is pressed by positioning the tip of the arrow cursor in the desired key number and clicking the mouse button. The binary data on the output lines will change to reflect the new value and remain until the next key is pressed. On a fifth line, it outputs a signal which goes high momentarily and then low again when a key is pressed. The hex keyboard can be selected by holding the shift key down and clicking on the hex keyboard.

Hex Display

The hex display shows the hexadecimal equivalent of its four binary inputs. If any of the inputs is unknown, high-impedance or conflict, then an X will be displayed.

SetupHold

Setup and hold times can be checked by attaching a "SetupHold" primitive device to the inputs of the clocked device to be checked, as follows:



Each time the "D" input changes, the SetupHold device marks the current time. On the next rising edge of the "C" input, the current time is compared to the last transition on the "D" input. If the difference is less than or equal to the Setup time specified in the Simulation Parameters dialog, the output switches to a

Don't Know state and remains in this state until a non-violating input transition occurs. Similarly, for each clock rising edge, the time is marked and compared to the time at the next transition on the "D" input. If this difference is less than the Hold time given, the output is set to Don't Know.

The SetupHold device puts out a highZ value until a setup or hold violation occurs when it switches to a don't know state. Thus, the output of the SetupHold device can be paralleled with the flip-flop so that the output line will enter a conflict state when an error occurs.

Unknown Detector

The Unknown Detector device detects any unknown value (i.e. not 0 or 1) on its input and puts out a high value after the delay time specified in the Simulation Parameters dialog. When the input returns to a 0 or 1 state, the output becomes zero. This can be used in conjunction with triggers to detect invalid conditions in a circuit and stop the simulator.

Appendix A - Timing Text Data Format

When a Copy or Cut operation is done on a selected area in the Timing window, two types of data are placed on the system clipboard:

- A picture of the selected area, in standard PICT format.
- A text description of the signal value changes occurring in the selected area.

This Appendix describes this text data format.

General Description

The clipboard text format used by the Timing tool is actually a subset of the command structure supported by the Test Vector tool. Thus, it is always possible to paste timing data into the Test Vector. Copying data from the Test Vector tool to the timing diagram is also possible with some restrictions. The Timing tool does not support control commands like $REPEAT/$END which are available in the Test Vector.

NOTE: The Test Vector's Copy to Timing command is an alternate way of moving Test Vector data to the timing window. This bypasses the clipboard and does support control commands.

Specific Description

The following rules describe the timing text data format:

- The data is pure ASCII text with no special binary codes except for standard tab and carriage return characters.
- The format of the data is based on the common "spreadsheet" text data format, i.e. each text item is followed by a tab character, except for the last one on a line which is followed by a carriage return.
- Every line has the same number of text items on it.
- The first line of the text (i.e. up to the first carriage return) is a "header" which indicates the meaning of the items on the following lines, by position.
- The lines following the header are signal value lines. Each line represents one time step. A complete data line is written out each time any value on the line changes. No line is written out for time steps in which none of the represented signals changed value.

Header Format

The header consists of a series of "commands", each starting with a $ sign, which describe the meaning of the corresponding data items on the following lines.

The header always contains the command "$T" (denoting a time column), followed by a tab character, followed by "$D" (denoting a delay column). The remaining items depend on the traces that were selected in the timing diagram.

NOTE: The Timing tool always places the time and delay items in the order given here, although it will accept data with them in any order, or even completely missing. Since time and delay are redundant, either one is sufficient. If they are both missing a default delay value will be used.

Single Signal Items

An individual signal is specified by the text $I (for input) followed by a space, followed by the name of the signal. If the signal contains any blanks or control characters, it will be enclosed in quotation marks.

Grouped Items

Grouped items are denoted by the text "$I" followed by a blank, followed by the name of the group, followed immediately (without any spaces) by a list of the signals in the group, contained in square brackets. Any group or signal name which contains blanks or control characters will be enclosed in quotation marks.

NOTE: The Test Vector tool does not use group names, but will ignore any text immediately preceding the "[]" grouping brackets. For this reason, there must not be a space between the group name and the "[" or the group name will be interpreted by the Test Vector as a signal name.

Data Line Format

Each line following the header must contain one data item for each item in the header line. Thus, the first two items will always be:

- The time at which the events on this line take place. The Timing tool places the absolute time at which the events occurred (i.e. corresponding to the time scale on the diagram) in this column. However, when the data is pasted, the times are considered to be relative to the time of the first data line. This is a decimal integer which may take on any 32-bit unsigned value.

- The delay from this step to the next step. This is redundant information, since it can be derived from the times in the first column. It is provided for compatibility with Test Vector and for improved flexibility in exporting to outside software systems.

NOTES:

1) If the delay and time columns do not match, the longest time is used.

2) The delay on the last line has special significance because it indicates the delay from the last signal change to the end of the selected interval. When pasting, this value is used to determine how much time to insert.

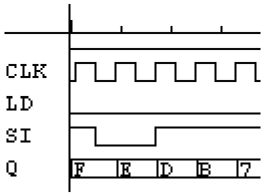
The following items on a line will be signal or group values matching the items in the header.

- For groups or individual signals, the special values ".X" (for DONT01) and ".Z" (for HIGHZ) indicate that all signals covered by this data item are in the given state.
- Individual signals not in Don't Know or High Impedance states will be either "0" or "1".
- Grouped signals which are not all unknown or all high impedance will be specified by a hexadecimal value. The least significant bit of the value corresponds to the rightmost signal in the group list. Special character "X" may be substituted for a hex digit if any of the four signals represented by that digit is unknown, or "Z" if all the signals represented by that digit were high impedance.

Timing Text Example

Following is an example of the timing text data and the corresponding timing diagram:

$T	$D	$I Q[Q0 Q1 Q2 Q3]	$I SI	$I LD	$I CLK
87410	2	F	1	0	0
87412	10	F	1	0	1
87422	10	F	0	0	0
87432	1	F	0	0	1
87433	9	E	0	0	1
87442	10	E	0	0	0
87452	1	E	1	0	1
87453	9	D	1	0	1
87462	10	D	1	0	0
87472	1	D	1	0	1
87473	9	B	1	0	1
87482	10	B	1	0	0
87492	1	B	1	0	1
87493	9	7	1	0	1
87502	4	7	1	0	0



Appendix B - Simulation Attribute Fields

The tools making up the Simulator package make use of a number of attribute fields to store device, signal and pin parameters and simulation setup information. The following table summarizes the usage of these fields.

The following information is provided in the attribute table:

Field Name	The name of the field as it appears in the Define Attributes dialog.
Type	The type of objects this field is used in.
Tool	The name of the tool that uses this field. If this is blank, then the field is provided as a convenience for the user but not "hard-wired" to any program function.
Inst/Def	Specifies whether the field data is kept with the definition or instance of a sub-circuit. I.e. Instance fields can have a different value in each copy of a sub-circuit.
Description	How the field is used.

Field Name	**Type**	**Tool**	**Inst/Def**	**Description**
ABELSrcName	Device	FromABEL	Def	Contains the name of the ABEL source file used to generate this part definition.
Delay.Dev	Device	Sim	Def	Specifies device delay. For most devices, a single decimal integer 0 to 32,767. For Clock, One Shot and SetupHold devices, two integers separated by commas. Should be set using the Simulation Parameters command and not edited manually.
Delay.Dev.Max	Device		Def	Maximum device delay. This is provided for use with the Load From Attribute button in the Simulation Parameters command. Not used internally.
Delay.Dev.Min	Device		Def	Minimum device delay. See Delay.Dev.Max.
Delay.Dev.Typ	Device		Def	Typical device delay. See Delay.Dev.Max.

Delay.Pin	Pin	Sim	Inst	A decimal integer specifying pin delay in the range 0 to 32767. Should be set using the Simulation Parameters command and not edited manually.
Delay.Pin.Max	Pin		Def	Maximum pin delay. See Delay.Dev.Max.
Delay.Pin.Min	Pin		Def	Minimum pin delay. See Delay.Dev.Max.
Delay.Pin.Typ	Pin		Def	Typical pin delay. See Delay.Dev.Max.
DWLSrcName	Device	FromABEL	Def	Contains the name of the ".dwl" file used to generate this part definition.
DWLSrcPath	Device	FromABEL	Def	Contains the "path name" of the ".dwl" file used to generate this part definition, i.e. names of all nested directorys.
ExtCctDate	Device	SimLoad/ Schematic	Def	The "last modified" date of the external circuit file, stored as an integer in 1/60 sec since Jan. 1, 1904. Should not be edited manually. See ExtCctName.
ExtCctName	Device	SimLoad/ Schematic	Def	The file name of the external circuit file. Should normally be set using the SimLoad tool or Attach External Sub-circuit command.
ExtCctPath	Device	SimLoad/ Schematic	Def	The "path name" of the external circuit file, i.e. names of all nested directories. See ExtCctName.
Initial.Pin	Pin	Sim	Inst	The initial value for the pin on Clear Simulation. Output pins only. One character: 0, 1, X or Z.
Initial.Sig	Signal	Sim	Inst	The initial value for the signal on Clear Simulation. One character: 0, 1, X or Z.
Invert.Pin	Pin	Sim	Def	Any non-empty value indicates pin inversion.

Sim.InputMap	Design	Sim	Def	Used to specify alternate input mapping for primitive devices. Must be exactly five characters, each one 0, 1 or X. First character specifies the value to substitute for a low input value, second for a high input value, third for high impedance, fourth for Don't Know, fifth is Conflict. e.g. "011XX" will substitute a high value for high impedance. Default is "01XXX
Sim.TrigSave	Design	Sim	Def	Used by the Simulator to store the current trigger setup. Should not be edited manually.
TestVectors.Cct	Design		Def	Used to store sets of test vectors for the design. Not used internally.
TestVectors.Dev	Device	Test Vector	Def	Used to store test vectors for the device. Loaded and saved by the Place and Test Device and Save Test to Device commands in Test Vector.
Timing.Save	Design	Timing	Def	Used to save the current status of the timing window. Should not be edited manually.

This page has been intentionally left blank.

Appendix C - Device Pin Types

Every device pin has a characteristic known as its *pin type*. The pin type is set when the part entry in the library is created and cannot be changed for individual device pins on the schematic.

> *See the DesignWorks/Schematic User's Manual for information on how to set the pin type while creating a device symbol.*

What Pin Types are Used For

For many general schematic editing purposes, the pin type will be unimportant and can be ignored. However, pin type settings are important in the following cases:

- The pin type of each pin is used by the simulator to select what type of output values are generated by a pin. e.g. An open collector output will not generate a HIGH drive level.
- Pin type information is required in many netlist file formats for FPGA layout and digital simulation.
- Correct pin type settings allow the ErrorFind tool to check for fanout and multiple drive situations.
- Other future analysis tools may use this information for timing and loading analysis.

Pin Types Table

The following table lists the function of each of the pin types available in DesignWorks. The Output Value Mapping column specifies how output values specified by the model are mapped to actual pin drive values.

Pin Type	**Description**	**Initial Value**	**Output Value Mapping**
Input	Input - this is the default for pins created using the DevEditor tool. This setting is used for all pins on discretes except those with some digital function. No output value can be placed on an input pin.	HIGHZ	No output drive allowed.
Output	Output - always enabled.	DONT01	None

TriState	Output - can be disabled (i.e. high-Z). Note: The three-state capability only exists for specific primitive types that have a three-state enable pin. For other types this will behave like OUT.	DONT01	None
Bidirectional	Bidirectional	DONT01	None
Open Collector	Open collector output - i.e. pulls down but not up	DONT0Z	HIGH maps to HIGHZ.
Bus	Bus pin - This does not represent a physical signal but is a graphical representation of a group of internal pins, each having its own type. Bus pins cannot have values and are not supported on primitive device types.	None	None.
Tied Low	Output - always driving low	LOW	All values converted to LOW.
Tied High	Output - always driving high	HIGH	All values converted to HIGH.
Latched Input	Input to a transparent latch - this is used for calculating cumulative setup and hold times.	HIGHZ	No output drive allowed.
Latched Output	Output from a transparent latch - this is used for calculating cumulative setup and hold times.	DONT01	Same as OUT.
Clocked Input	Input to an edge-triggered latch - this is used for calculating cumulative setup and hold times.	HIGHZ	No output drive allowed.
Clocked Output	Output from an edge-triggered latch - this is used for calculating cumulative setup and hold times.	DONT01	Same as OUT.
Clock Input	Clock input - this is used for calculating cumulative setup and hold times.	HIGHZ	No output drive allowed.
Open Emitter	Open emitter output - i.e. can pull up but not down.	DONT1Z	LOW maps to HIGHZ.
No Connection	A no-connect pin	HIGHZ	No output drive allowed.

Device Pin Type and Simulator Efficiency

Incorrect device pin type settings can have a major impact on simulation speed, even in cases where they wouldn't affect the correctness of the results.

Bidirectional Pins

The use of bidirectional pins should be avoided unless specifically required by circuit logic. On primitive types, any value change on a signal attached to a bidirectional pin will cause the device model to be called to reevaluate the device. On sub-circuit devices, the simulator must make several passes through all circuit levels which may affect the value of the signal or be affected by it. Setting a pin on a sub-circuit device to be an input or output greatly reduces this overhead.

Output Pins

If a device pin will only ever be used to drive the attached signal and the device cannot be affected by changes in value on the pin, then it should be an output type. Changes in the value of a signal attached to an output pin do not cause the device model to be called for reevaluation. This is particularly significant for sub-circuit devices.

Input Pins

Pins with an input type setting can never place a drive value on the attached signal. On sub-circuit devices this provides an important hint to the simulator that internal value changes on the attached signal will not affect any other circuit level.

This page has been intentionally left blank.

Appendix D - Primitive Device Pin Summary

In DesignWorks primitive device types, the function of each pin is determined by its type (i.e. input or output) and its sequential position in the device's pin list (i.e. as seen when the part is opened in the DevEditor). Pin name is not significant. Each type has specific rules about the ordering of pins. Failure to adhere to these rules will result in incorrect simulator operation.

NOTE: Bus pins are not supported on primitive device types.

For many primitive types, certain control inputs and outputs can be omitted to create simplified device types. E.g. On flip-flop types, the Set and Reset inputs can be omitted. The table below shows which combinations of inputs are allowable and the required order.

Pin Inversion

In addition to the pin function options shown in the table, any pin on any device can be inverted by specifying a value in the Invert.Pin attribute field. Any non-empty value will cause the pin logic to be inverted.

Pin Function Table

The following table lists the pin functions and orders for all primitive device types. In some cases, a number of pins can be optionally omitted, so rules are given rather than enumerating all possible combinations.

See Chapter IX - Primitive Devices (page 107) for more information.

Primitive Type	Limitations	Pin Names and Types	Possible Pin Orders
NOT	Exactly 1 input and 1 output	IN - in OUT - out	1) IN OUT
AND, NAND, OR, NOR, XOR, XNOR	N inputs: 1 <= N <= 799	$IN_0..IN_{N-1}$ - in OUT - out	1) $IN_0..IN_{N-1}$ OUT
X-Gate	Exactly 2 ports and 1 enable	X1 X2 - bidir EN - in	1) X1 EN X2

Three-State Buffer	N data inputs N data outputs 1 <= N <= 256	$IN_0..IN_{N-1}$ - in $OUT_0..OUT_{N-1}$ - out EN - in	1) $IN_0..IN_{N-1}$ $OUT_0..OUT_{N-1}$ EN 2) $IN_0..IN_{N-1}$ $OUT_0..OUT_{N-1}$
Resistor	Exactly 2 pins	X1 X2 - bidir	1) X1 X2
Multiplexer	L select inputs M output bits 1 <= M <= 256 N inputs/ output 2^{L-1} <= N <= 2^L i.e. the number of inputs per output bit can be less than the number of select input combinations	$S_0..S_{L-1}$ - in $IN_{0,0}..IN_{N-1,M-1}$ - in * EN - in § $OUT_0..OUT_{M-1}$ - out * $IN_{n,m}$ is the input routed to output m when select value is n. § An enable input can exist only if N == 2^L, otherwise the extra input is assumed to be a data input.	1) $IN_{0,0}..IN_{0,M-1}$ $IN_{1,0}..IN_{1,M-1}$ .. $IN_{N-1,0}..IN_{N-1,M-1}$ $S_0..S_{L-1}$ $OUT_0..OUT_{M-1}$ 2) $IN_{0,0}..IN_{0,M-1}$ $IN_{1,0}..IN_{1,M-1}$ .. $IN_{0,0}..IN_{N-1,M-1}$ $S_0..S_{L-1}$ EN $OUT_0..OUT_{M-1}$ § § Option 2 only if N == 2^L
Decoder	L select inputs M output bits 1 <= M <= 256 AND 2^{L-1} <= M <= 2^L i.e. the number of output bits can be less than the number of select input combinations	$S_0..S_{L-1}$ - in EN - in $OUT_0..OUT_{M-1}$ - out	1) $OUT_0..OUT_{M-1}$ $S_0..S_{L-1}$ 2) $OUT_0..OUT_{M-1}$ $S_0..S_{L-1}$ EN
Adder, Subtracter	N output bits N "A" operand inputs required N "B" operand inputs optional 1 <= N <= 256	$A_0..A_{N-1}$ - in $B_0..B_{N-1}$ - in CIN - in $SUM_0..SUM_{N-1}$ - out COUT - out	1) $A_0..A_{N-1}$ $B_0..B_{N-1}$ $SUM_0..SUM_{N-1}$ CIN COUT * * $B_0..B_{N-1}$ CIN & COUT can be omitted in any combination

D Flip-Flop, D Latch	Must have at least D and CLK inputs and Q output	S - set in D - D in C - clock in R - reset in Q - out NQ - inverted out	1) S D C R Q NQ 2) S D C R Q 3) D C R Q NQ 4) D C R Q 5) D C Q NQ 6) D C Q
JK flip-flop	Must have at least CLK input and Q output	S - set in J - J in K - K in C - clock in R - reset in Q - out NQ - inverted out	1) S J C K R Q NQ § 2) S T C R Q NQ *§ 3) T C R Q NQ *§ 4) C R Q NQ § 5) C Q NQ § § NQ can always be omitted * T = J & K tied together
Register	N output bits N input bits 1 <= N <= 256	IN$_0$..IN$_{N-1}$ - in CLK - in CLR - in OUT$_0$..OUT$_{N-1}$ - out	1) IN$_0$..IN$_{N-1}$ OUT$_0$..OUT$_{N-1}$ CLK CLR 2) IN$_0$..IN$_{N-1}$ OUT$_0$..OUT$_{N-1}$ CLK
Counter	N output bits N input bits (optional) 1 <= N <= 256	IN$_0$..IN$_{N-1}$ - in CLK - in LD - in CLR - in UP - in EN - in OUT$_0$..OUT$_{N-1}$ - out COUT - out	1) IN$_0$..IN$_{N-1}$ OUT$_0$..OUT$_{N-1}$ CLK LD CLR UP EN COUT 2) IN$_0$..IN$_{N-1}$ OUT$_0$..OUT$_{N-1}$ CLK LD CLR UP COUT 3) IN$_0$..IN$_{N-1}$ OUT$_0$..OUT$_{N-1}$ CLK LD CLR COUT 4) IN$_0$..IN$_{N-1}$ OUT$_0$..OUT$_{N-1}$ CLK LD COUT 5) IN$_0$..IN$_{N-1}$ OUT$_0$..OUT$_{N-1}$ CLK COUT Note: IN$_0$..IN$_{N-1}$ & COUT can always be omitted
Shift Register	N output bits N input bits 1 <= N <= 256	IN$_0$..IN$_{N-1}$ - in CLK - in LD - in CIN - in OUT$_0$..OUT$_{N-1}$ - out	1) IN$_0$..IN$_{N-1}$ OUT$_0$..OUT$_{N-1}$ CLK LD CIN 2) OUT$_0$..OUT$_{N-1}$ CLK CIN

One Shot		CLK - in CLR - in Q - out NQ - out	1) CLK CLR Q NQ 2) CLK CLR Q
Clock Osc	Exactly one output pin	CLK - bidir	1) CLK
Binary Switch	Exactly one pin	SW - bidir	1) SW
SPST Switch	Exactly 2 pins	X1 X2 - bidir	1) X1 X2
SPDT Switch	Exactly 3 pins	X1 X2 COM - bidir	1) X1 X2 COM
Binary Probe	Exactly 1 pin	PR	1) PR
Hex Keyboard	4 or 5 pins	X_0..X_3 - bidir STROBE - out	1) X_0..X_3 STROBE 2) X_0..X_3
Hex Display	Exactly 4 pins	X_0..X_3 - in	1) X_0..X_3
SetupHold	Exactly 3 pins	CLK - in DATA - in Q - out	1) CLK DATA Q
Unknown Detector device	Exactly 2 pins	D - in Q - out	1) D Q

Credits

DesignWorks, LogicWorks and MEDA are trademarks of Capilano Computing Systems Ltd.

DesignWorks was designed by Chris Dewhurst and coded by Chris Dewhurst, Scott Farquharson, Don Gamble, Geoff Harris, Addam Smith, David Taylor and Neil MacKenzie of Capilano Computing Systems Ltd.

DesignWorks was written using Borlands C++ from Borland International of Scotts Valley, California. DesignWorks was implemented using the XVT Portability Toolkit from XVT Software Inc of Boulder, Colorado.

Microsoft is a trademark of Microsoft Corporation of Bellevue, Washington.

ABEL is a trademark of Data I/O Corporation of Redmond, Washington.

Other trademarks used in this manual are property of their respective holders.

Technical Support

Any problems or questions can be referred to:

Technical Support,
Capilano Computing Systems Ltd.,
406 - 960 Quayside Drive,
New Westminster, B. C.
Canada, V3M 6G2
Phone: (604) 522-6200
Fax (604) 522-3972

AppleLink: D0816
Internet: capilano_computing@wimsey.com

Index