Notes for simulating digital circuits with ELDO
Input files used by ELDO, Transistor Scaling, Forces, and Plotting
rev 2

DA-IC and ELDO Files

Two files are used as input to ELDO: design_name.cir and design_name.spi that will hereafter be referred to as a .cir or .spi file. The .cir contains ELDO commands that control simulation, specify circuit nodes whose data will be displayed, establish forces on the inputs, etc. The .spi file holds the circuit netlist and is read into the simulator with a .INCLUDE statement in the .cir file.

The .cir file is re-written after you enter data or change settings in several of the menus accessed via the simulation panel in da-ic. Thus while it could be edited with a text editor chances are it will be overwritten as you prepare for simulation. However, from the simulation panel select Options > Additional and the window shown in figure 1 below will open to allow additional commands to be entered for insertion either at the start of or end of the .cir file. The commands you enter will be remembered in a da-ic configuration file and thus anytime da-ic re-writes the .cir file your commands will be included.

User action(s) that cause the .spi file to be written depend on how the user configures the simulation environment. From the simulation panel select Session > Environment and the panel shown in figure 7 of this document appears. Note the four Auto-Run Simulation Setup options on the left. The last two that begin with Netlist cause the .spi file to be re-written every time a simulation is run. Thus any edits you make to the .spi fill will be overwritten by the netlister and your hard work of editing is lost. Thus I recommend the second option, i.e. Run Simulation and Display Waveforms. Note however that you will have to click the Netlist button right above Run Eldo on the simulation pallette to get a netlist to simulate with Eldo or get a netlist to edit.

Scaling

Transistor sizing (width and length) is specified in units of lambda in the schematics that are used for both simulation and as input to schematic driven IC layout. Lambda dimensions are used in the layout and so specifying widths and lengths in lambda works just fine. However, for simulation, transistor sizes need to be in units of meters (or microns) to obtain correct simulations. Thus lambda's must be converted using the appropriate scaling factor.

Spice has a circuit description command named .SCALE (note the leading period) that can be used to specify particular model parameters that need to be scaled and by how much. The syntax of this command is (using a MOSFET as an example):

```
        .SCALE M  W .35   L .35   AD .123   PD .35   AS .35   PS .35
```

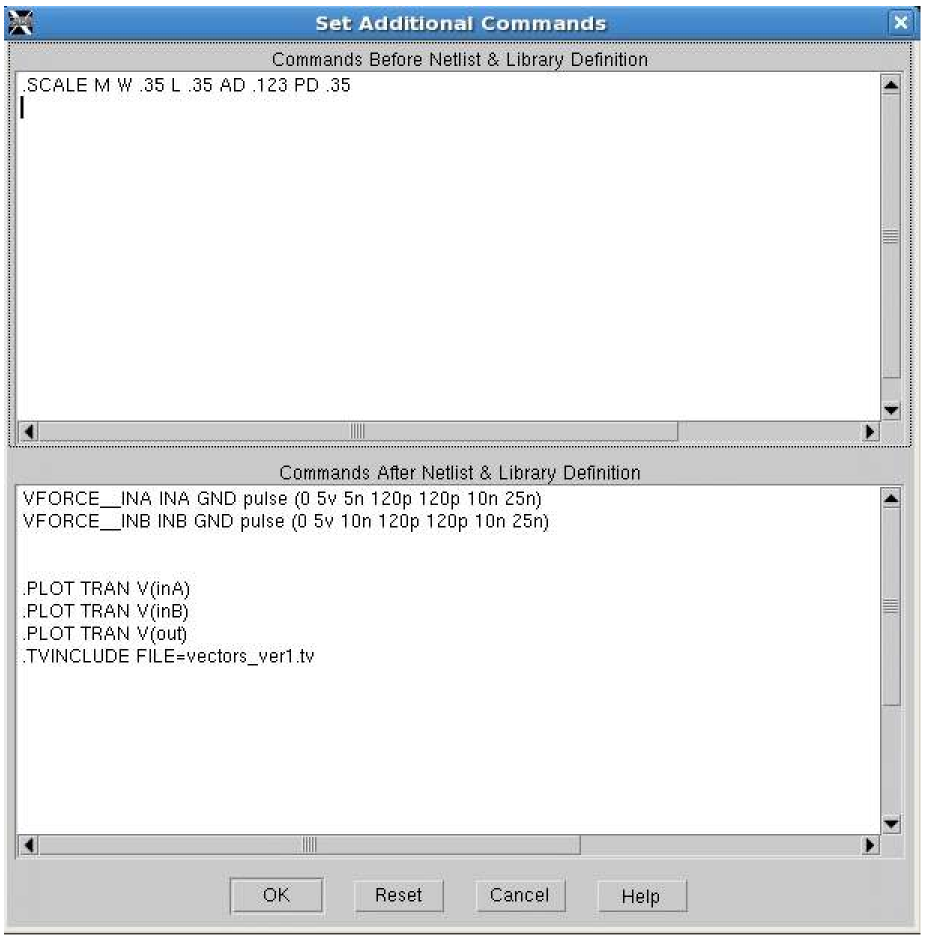
Where the first field after scale indicates the type of component who's parameters will be scaled (various formats are allowed to specify the intended component(s) subject to scaling. See the Eldo User Manual). The M here indicates a MOSFET transistor (either N or P type). Following the component type designator are data pairs composed of a parameter name and the scaling factor for that parameter. The specified parameter will be multiplied by the scaling factor. For example, a 5 lambda wide transistor has will have a 5 x .35 or 1.75um width.

The .SCALE command should be placed in the upper part of the Options > Additional   window as shown in figure 1 below.

Forces

Forces can be handled in one of two ways. The first way is using the graphical force manager accessed from the simulation pallette button Forces > Manager which opens up the manager window and various forces can be defined. However, if you don't explicitly save your setup then when you exit da-ic and start it up again the knowledge about your forces will be gone and you have to create them again. To save your session, again from the Simulation pallette, select Session > Save Session as Default and click yes when prompted about the sim_setup file already existing (or you can do a save-as and create a unique instance of setup). When you restart daic in the future select: Session > Restore Setup From   and another pop-up box will ask to confirm this. Answer yes. The forces defined in the previous session that you saved should now be restored.

A second way to handle forces is to not define them via the Forces button on the simulation pallette and instead place the appropriate statements in the Options > Additional   window as shown in the lower window of figure 1 below (pulse statements shown here)..


Set Additional Commands
Commands Before Netlist & Library Definition

```
.SCALE M W .35 L .35 AD .123 PD .35
```

Commands After Netlist & Library Definition

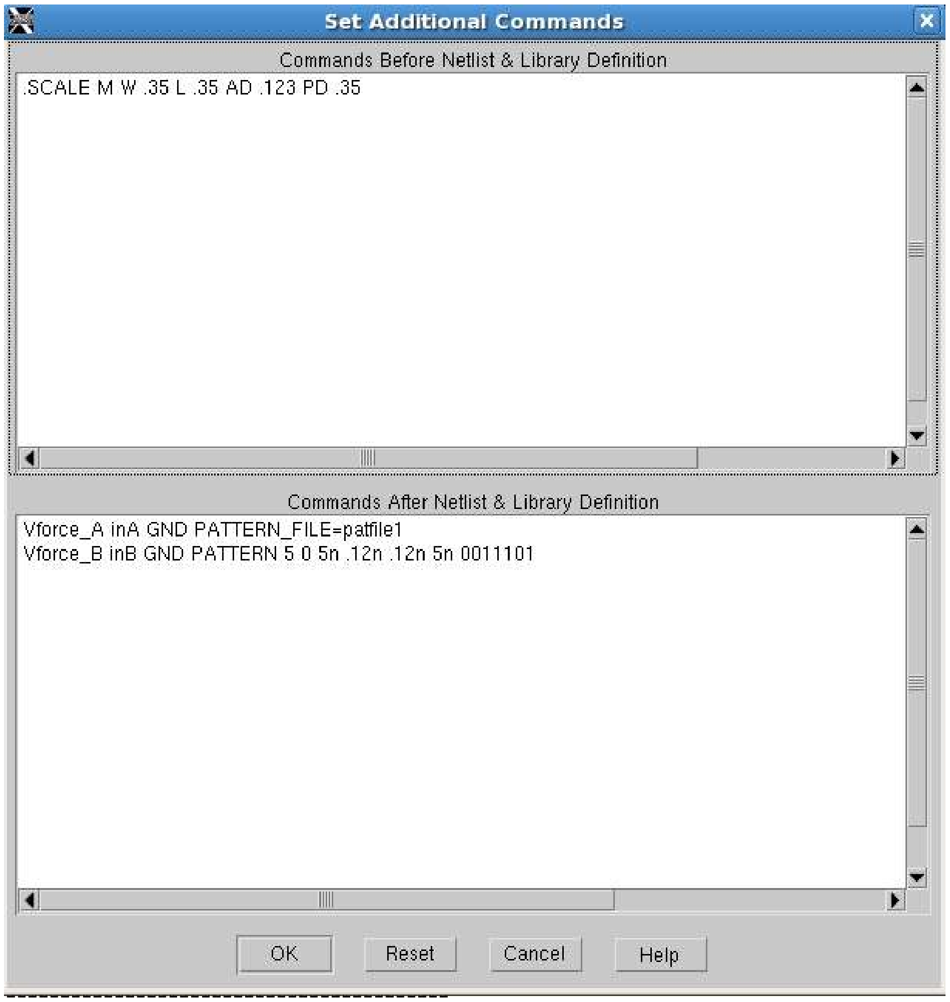
```
VFORCE__INA INA GND pulse (0 5v 5n 120p 120p 10n 25n)
VFORCE__INB INB GND pulse (0 5v 10n 120p 120p 10n 25n)


.PLOT TRAN V(inA)
.PLOT TRAN V(inB)
.PLOT TRAN V(out)
.TVINCLUDE FILE=vectors_ver1.tv
```

Figure 1 - Set Additional Commands window

Alternative Force Commands

When working with digital type circuits the pulsed voltage source is generally not useful if the signal isn't a single pulse or isn't periodic. There are two other independent voltage sources that work well for digital circuits where the input voltages can be represented by logic 0's or 1's.

Figure 2 below shows two pattern voltage sources that have been placed in the Additional Commands window.. See the handout on SPICE netlist syntax for details of those sources.


Set Additional Commands
Commands Before Netlist & Library Definition
Commands After Netlist & Library Definition
OK
Reset
Cancel
Help

```
.SCALE M W .35 L .35 AD .123 PD .35
```

```
Vforce_A inA GND PATTERN_FILE=patfile1
Vforce_B inB GND PATTERN 5 0 5n .12n .12n 5n 0011101
```

Figure 2 Set Additional Commands - Pattern voltage sources shown

The first pattern statement references patfile1 which is an ascii text file with the same information that would be placed directly into the pattern command although for a long sequence it is better to have it in a file. Here is the contents of pattfile1:

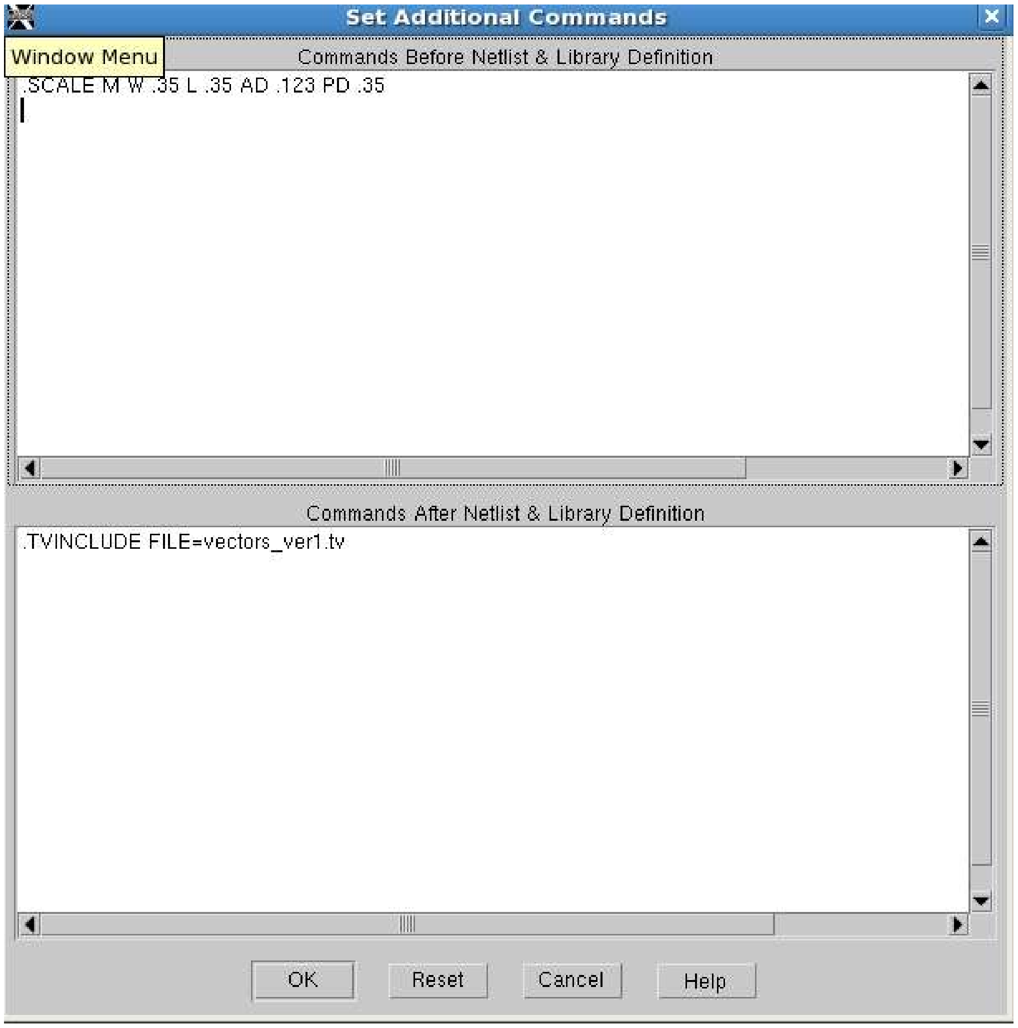
```
**** Pattern for the A input of a NAND2 gate
5, 0, 5n, 120p, 120p, 5n, 0101010
```

The pattern for the B input could also be specified to come from a file. However, only one pattern can be contained in a single file.

The second way that a pattern of 1's and 0's can be presented as input to the simulator is using a .TVINCLUDE statement which specifies what is called a test_vector file.. This statement would replace the other Force statements for this example circuit although you can use a mix of Force statements and test vector inputs but not driving the same circuit node(s).

A test vector is a set of bits that represent multiple inputs of a digital circuit at a particular instant of time and may include within the "vector" a set of bits that represent the outputs that should be observed for the given inputs. A test vector file contains a list of vectors and a time at which each vector should be applied to the circuit inputs. The simulator applies each vector and then checks the output to see if outputs match the expected outputs. The simulation transcript will contain a report about how many vectors didn't match and the time stamp of when the match failed. That allows long vector sequences, like going through all the bit combinations of data that could be applied to an ALU and confirming that all combinations work. The vector file could be generated with a script.

Figure 3 below shows the use of a .TVINCLUDE command and figure 4 a sample vector file.

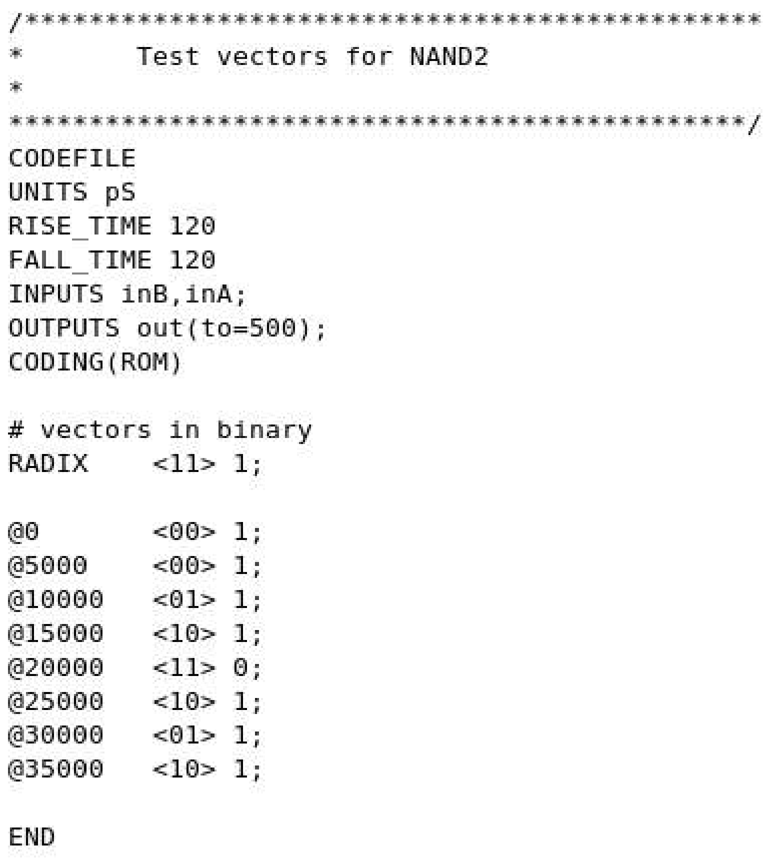

Set Additional Commands
Window Menu
Commands Before Netlist & Library Definition
Commands After Netlist & Library Definition
OK
Reset
Cancel
Help

```
.SCALE M W .35 L .35 AD .123 PD .35
```

```
.TVINCLUDE FILE=vectors_ver1.tv
```

Figure 3 Set Additional Commands - showing .TVINCLUDE

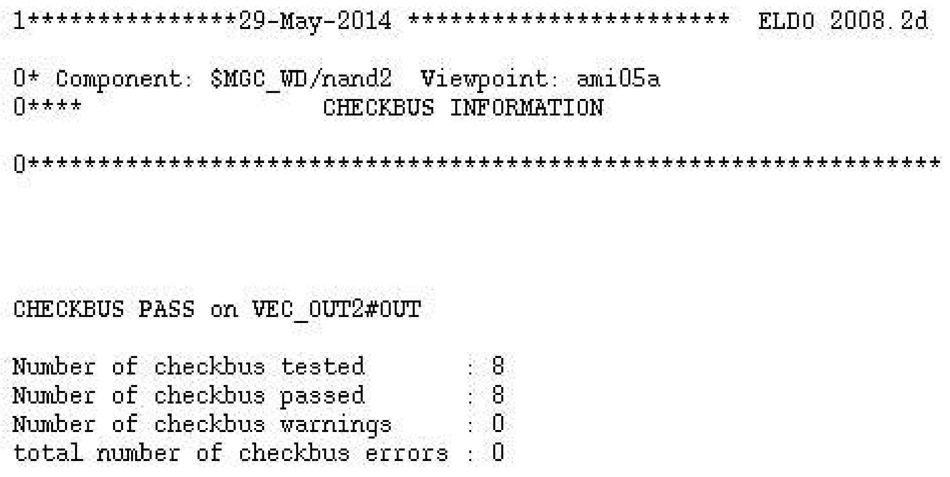
NOTE: In figure 1 both PULSE type voltage sources and .TVINCLUDE test vector inputs to the same input nodes is shown. It doesn't work to drive an input with two sources at once. Use one or the other. Also, the yellow box in figure 3 is extraneous. Ignore it.



```
/*********************************************
*       Test vectors for NAND2
*
*********************************************/
CODEFILE
UNITS pS
RISE_TIME 120
FALL_TIME 120
INPUTS inB,inA;
OUTPUTS out(to=500);
CODING(ROM)

# vectors in binary
RADIX    <11> 1;

@0       <00> 1;
@5000    <00> 1;
@10000   <01> 1;
@15000   <10> 1;
@20000   <11> 0;
@25000   <10> 1;
@30000   <01> 1;
@35000   <10> 1;

END
```

Figure 4 - Example test vector file.

Example report text (from the .chi file) follows in figure 5.



```
1***************29-May-2014 ***********************  ELDO 2008.2d

0* Component: $MGC_WD/nand2  Viewpoint: ami05a
0****                 CHECKBUS INFORMATION

0*************************************************************



CHECKBUS PASS on VEC_OUT2#OUT

Number of checkbus tested       : 8
Number of checkbus passed       : 8
Number of checkbus warnings     : 0
total number of checkbus errors : 0
```

Figure 5

The “Number of checkbus tested” means the number of test vectors including that at time = 0;

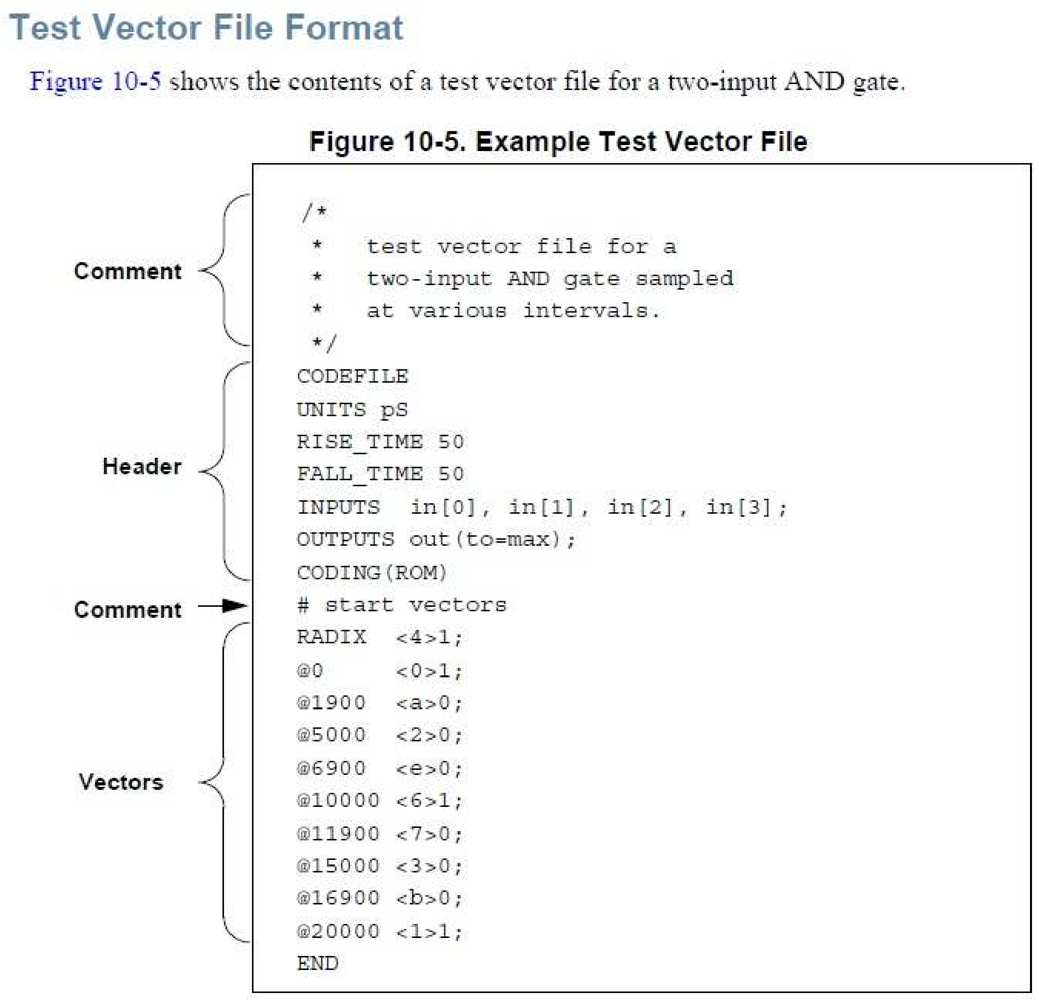
Plotting Simulation Results

Specifying the circuit nodes that you wish to have plotted can be done via the da-ic graphical user interface or by directly placing plot commends into the .cir file using the Set Additional Commands window. See figure 1 for example plot commands. If you use the graphical interface then also use Session > Save_Setup_as_Default to save your plotting definitions along with any other commands you entered into the Set Additional Commands window..

Appendix

See the ELDO users manual (/home/classes/engr434/docs/eldo_ur2008.pdf) pages 885 to 891 regarding details of the .TVINCLUDE command and vector file format. See pages 319 to 321 regarding details of the PATTERN function.

Here is the example test vector file from the Eldo user manual but I don't think it is correct for the gate it claims to be testing. However, it is useful for defining the parts of the file:



Test Vector File Format

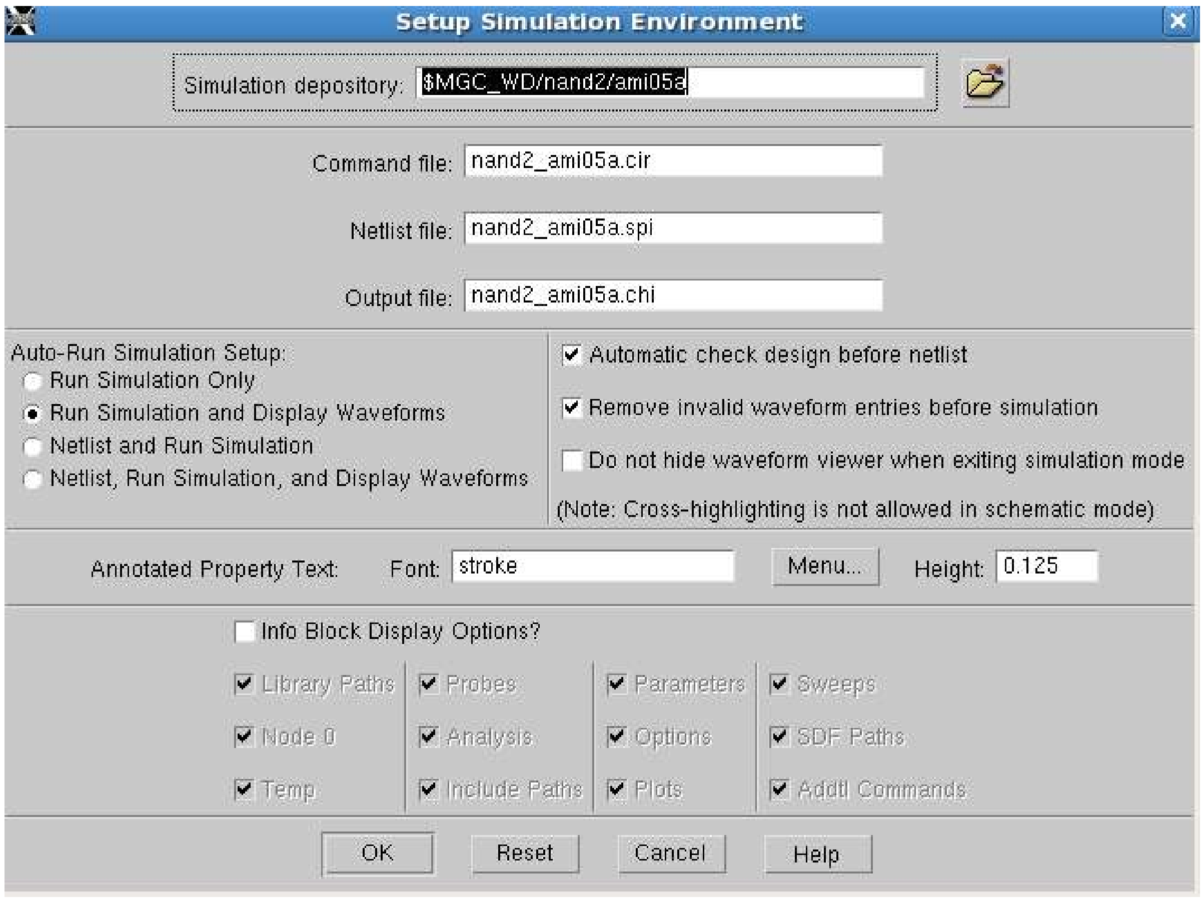
Figure 10-5 shows the contents of a test vector file for a two-input AND gate.

Figure 10-5. Example Test Vector File

```
/*
 *   test vector file for a
 *   two-input AND gate sampled
 *   at various intervals.
 */
CODEFILE
UNITS pS
RISE_TIME 50
FALL_TIME 50
INPUTS  in[0], in[1], in[2], in[3];
OUTPUTS out(to=max);
CODING(ROM)
# start vectors
RADIX  <4>1;
@0     <0>1;
@1900  <a>0;
@5000  <2>0;
@6900  <e>0;
@10000 <6>1;
@11900 <7>0;
@15000 <3>0;
@16900 <b>0;
@20000 <1>1;
END
```

Figure 6 - Excerpt from the Eldo User Manual

Figure 7 below shows the menu for set up control of when the .spi file is written (the first time) or re-written additional times (overwriting any former version):


Setup Simulation Environment

Simulation depository: $MGC_WD/nand2/ami05a

Command file: nand2_ami05a.cir

Netlist file: nand2_ami05a.spi

Output file: nand2_ami05a.chi

Auto-Run Simulation Setup:
- ○ Run Simulation Only
- ● Run Simulation and Display Waveforms
- ○ Netlist and Run Simulation
- ○ Netlist, Run Simulation, and Display Waveforms

- ✓ Automatic check design before netlist
- ✓ Remove invalid waveform entries before simulation
- ☐ Do not hide waveform viewer when exiting simulation mode

(Note: Cross-highlighting is not allowed in schematic mode)

Annotated Property Text: Font: stroke Menu... Height: 0.125

☐ Info Block Display Options?

✓ Library Paths	✓ Probes	✓ Parameters	✓ Sweeps
✓ Node 0	✓ Analysis	✓ Options	✓ SDF Paths
✓ Temp	✓ Include Paths	✓ Plots	✓ Addtl Commands

Figure 7 - Session > Setup Simulation Environment.

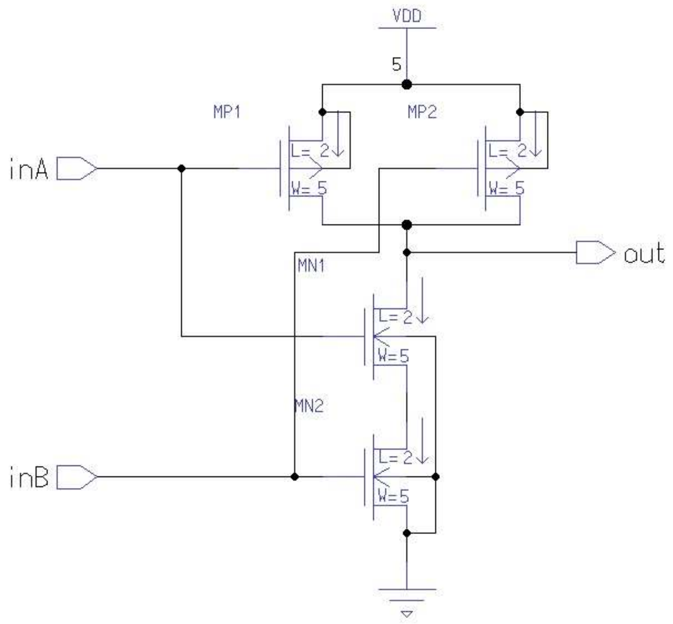


Figure 8 - NAND gate circuit used for illustration

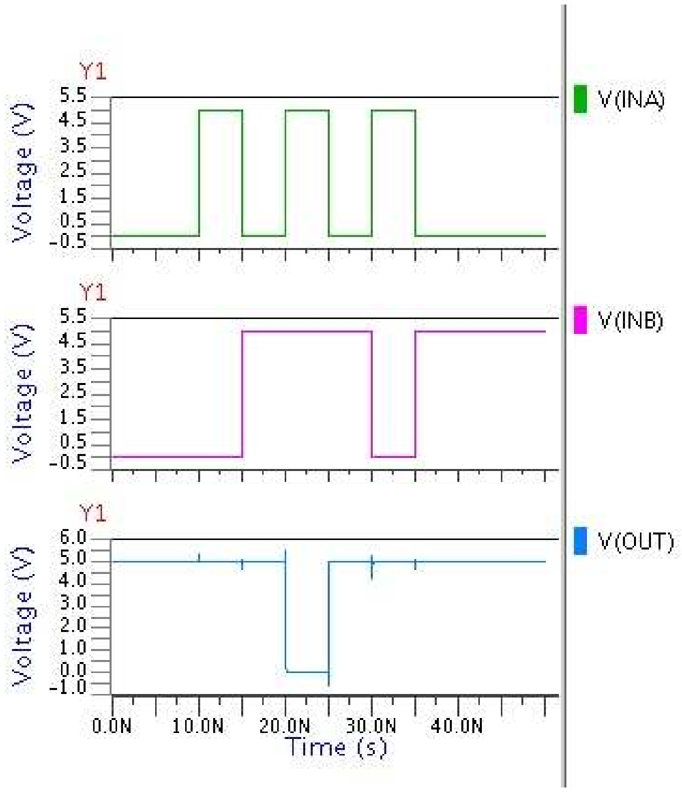


Figure 9 - Waveforms, selected by the plot commands in figure 1, generated by the pattern signal shown in figure 2. The test vector file in figure 4 creates the same wave forms.