

Altia[®] Design
Getting Started with
MATRIX_x



April, 2004

Altia, Inc. © 1992-2003

5030 Corporate Plaza Drive #200
Colorado Springs, CO 80919

This document contains information which is protected by copyright. All rights are reserved. Reproduction, adaptation, or translation without prior written permission is prohibited, except as allowed under the copyright laws.

Restricted Rights Legend

Use of this manual and flexible disk(s), tape cartridge(s), or CD-ROM(s) supplied for this product is restricted to this product only. Additional copies of the program may be made for security and backup purposes only. Resale of the programs or files in their present form or with alterations is expressly prohibited.

Trademarks

Altia® is a registered trademark of Altia, Inc.

Borland® is a registered trademark of Borland International, Inc.

HP-UX™ is a registered trademark of Hewlett-Packard Co.

IBM® is a registered trademark of IBM Corp.

Microsoft® and MS-DOS® are registered trademarks and Visual Basic™, Windows™, Win32s™ and Windows/NT™ are trademarks of Microsoft Corporation.

Silicon Graphics® is a registered trademark of Silicon Graphics, Inc.

SunOS® is a registered trademark and Solaris™ is a trademark of Sun Microsystems Computer Corp.

X Window System™ is a trademark of the Massachusetts Institute of Technology.

UNIX™ is a registered trademark of UNIX Systems Laboratories, Inc. or its successor.

AutoCode™, DocumentIt™, MATRIXx™, National Instruments™, NI™, ni.com™, RealSim™, SystemBuild™ and Xmath™ are trademarks of National Instruments Corporation.

All other product names mentioned herein are the trademarks of their respective owners.

Notice

The information contained in this document is subject to change without notice.

ALTIA,INC MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MANUAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Altia, Inc. shall not be liable for errors contained herein or direct, indirect, special, incidental, or consequential damages in connection with the furnishing, performance, or use of this material.



Contents



1.1	Introduction	1-1
1.2	Altia SystemBuild Tutorial	1-1
1.2.1	Use SystemBuild to Create a Simple Model	1-3
1.2.2	Use Altia Design to Create a Simple GUI	1-9
1.2.3	Connect Your Altia GUI to Your MATRIXx Model	1-13
1.2.4	Test Your Altia GUI/MATRIXx Model Interaction	1-17
1.2.5	Translating Existing PIC Files	1-18
1.2.6	Creating Connections to Existing Altia Objects	1-20
1.2.7	Tutorial Summary	1-23

1

Using Altia with MATRIXx

1.1 Introduction

Altia Design graphics and user interface software can be used in conjunction with NI's MATRIXx SystemBuild product. The MATRIXx interface to Altia Design allows the MATRIXx user to create new graphical panels with the state-of-the-art Altia Design editor. In addition, existing MATRIXx IA (Interactive Animation) picture (.pic) files can be converted to Altia Design (.dsn) files.

The Altia Design package includes an editor, runtime engine and numerous libraries of components for quickly creating a user interface to SystemBuild simulations. In addition to supplied components, users can modify existing components, import images and create their own custom components in the editor without programming. With these features, a modeler can quickly create a user interface prototype for product simulations that looks and behaves like the product's real user interface.

1.2 Altia SystemBuild Tutorial

The goal of this tutorial is to show the basic link between MATRIXx SystemBuild models and an Altia Graphical User Interface (GUI). In this tutorial, we will discuss every step necessary to develop a simple system that exercises this connection.

Before you begin this hour-long tutorial, make sure you have MATRIXx SystemBuild and Altia installed on your machine. Altia is included on the MATRIXx installation CD. If Altia is not already installed on your machine, PC users may install it by running the `setup.exe` file in the Altia directory on your CD. Be sure to install Altia in the `$NIHOME\altia` directory on your machine. If you have a previous version installed in `$NIHOME\altia`, it is strongly recommended that you uninstall this

previous version before installing a newer version. UNIX users should extract the Altia archive from the CD relative to their `$NIHOMEDIR` directory. If you need a new codeword to unlock Altia Design, please call Altia at (719) 598-4299.

- [Section 1.2.1](#), labeled **Use SystemBuild to Create a Simple Model**, will describe how to design a model that has a simple gain block which outputs to and is fed by an Altia Interface block.
- In [Section 1.2.2](#), labeled **Use Altia Design to Create a Simple GUI**, we will step through the procedures required to build a GUI that has a slider and a meter.
- The process of connecting the model to the GUI we have created will be addressed in [Section 1.2.3, Connect Your Altia GUI to Your MATRIXx Model](#). We will connect the two such that the slider bar is the input to the gain and the meter will show the gain's output.
- [Section 1.2.4](#), labeled **Test Your Altia GUI/MATRIXx Model Interaction**, will show you how to verify that your GUI and your model are connected correctly.
- [Section 1.2.5, Translating Existing PIC Files](#), describes how to translate an existing Interactive Animation (.pic) file into an Altia (.dsn) file. It also details how to modify a SystemBuild model to simulate with Altia instead of IA.
- The last section ([Section 1.2.6 - Creating Connections to Existing Altia Objects](#)), shows you how to create new connections to Altia objects.

1.2.1 Use SystemBuild to Create a Simple Model

The first thing you will need to do is decide on a name for a project directory. If the directory does not exist, create it now.

- **PC users** use the **File** menu in a file browser window to create a new folder.
- **UNIX users** you can use the `mkdir` command to create a new directory (e.g., `mkdir $HOME/tutor`).

1. Open Xmath and make sure you are in the project directory you have created by typing:

```
set directory="<full_path_to_project_directory>"
```

and pressing `<enter>` in the Xmath command line.

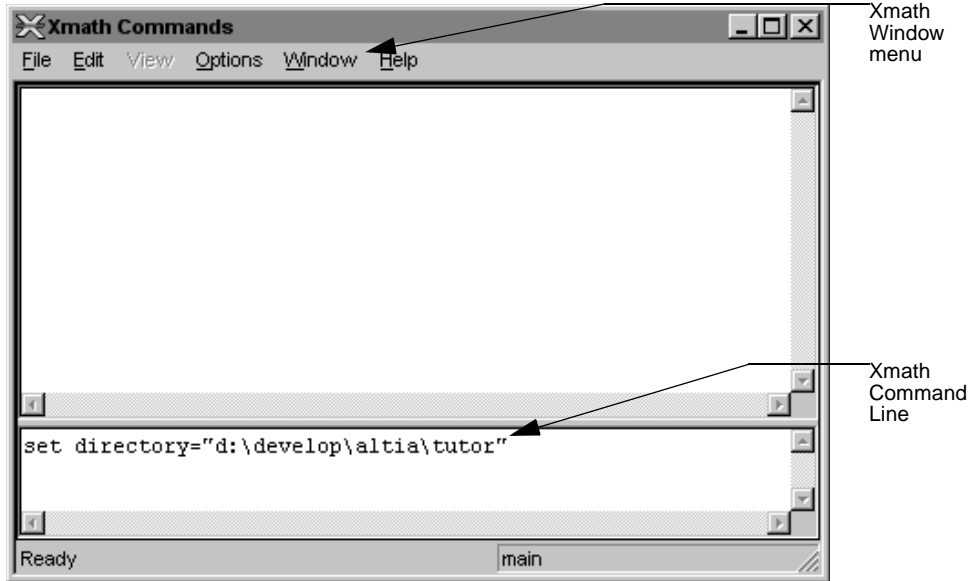


FIGURE 1-1 Xmath Window

2. Next, in the Xmath **Window** menu, click on **SystemBuild** to open up the Catalog Browser and SystemBuild editor utilities.

- The focus should be on the Catalog Browser, so in the Catalog Browser **File** menu, click on **New/SuperBlock**. This will open the SuperBlock Properties dialog box (See [Figure 1-2](#)) with the focus in the **Name** field.
- Type in a name for our SuperBlock (for example, `altiatut`) and change the number of outputs (located at the far right of the SuperBlock Properties dialog box) to one (1). Then click **OK** (at the bottom of the dialog).

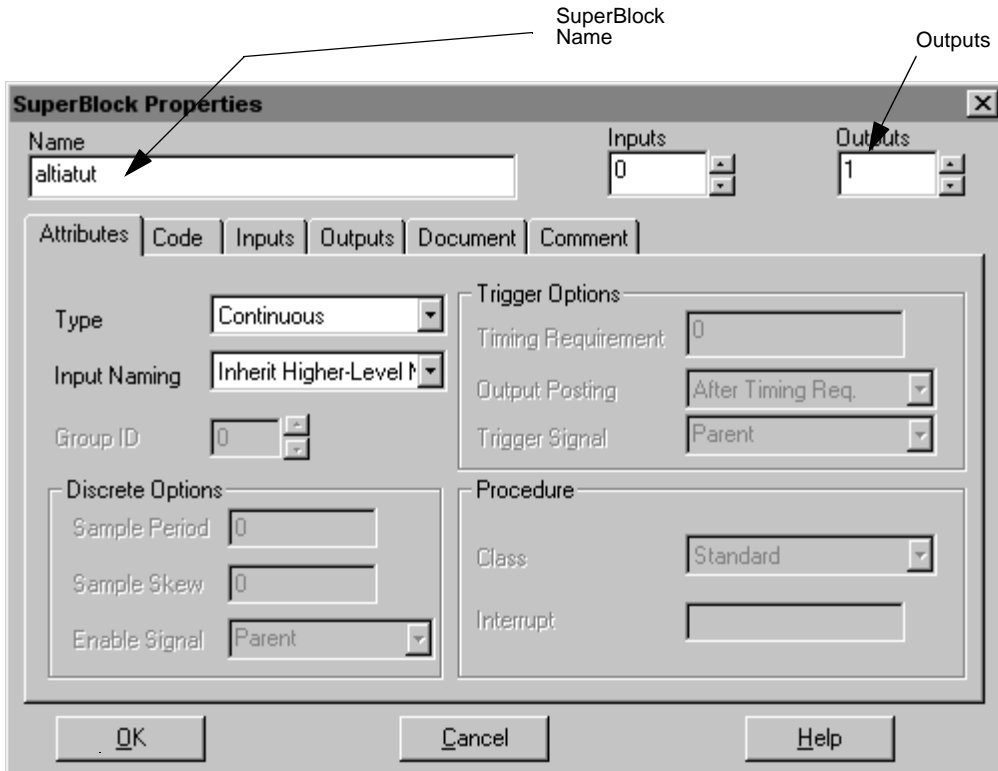


FIGURE 1-2 SuperBlock Properties Dialog Box

The SystemBuild editor will come up. Notice at the top of the SuperBlock editor work area that the name of our SuperBlock appears (`altiatut`) and it has 0 inputs and 1 output.

- We are now ready to add the components of our model. To do this, click on **Palette Browser** in the SystemBuild editor **Window** menu. The Palette Browser will open to the SuperBlock selections by default.

6. Click on the SuperBlock icon and drag it into the left side of your SystemBuild editor work area.
7. Next, scroll down the Palette Browser until the Altia block is visible. Click on the Altia icon and drag it into the right side of your SystemBuild editor work area as shown below.

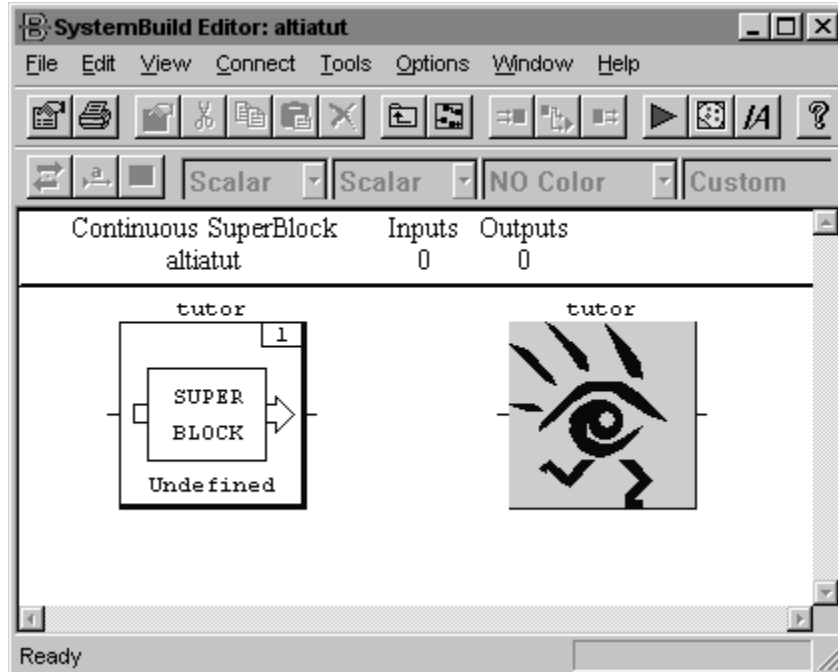


FIGURE 1-3 SystemBuild Editor Example

8. Next, we must give the two blocks identical names. Select the SuperBlock and press the **<enter>** key to open its property dialog. In the name field, type in **tutor** and click the **OK** button.
9. Change the name for the Altia block the same way. There is one additional parameter we need to change for the Altia block while its property dialog is open. Change the value of the **Input Direct Terms** parameter from **1** (one) to **0** (zero) and press **OK**.

10. Now, we need to add some functionality to the new SuperBlock within the SystemBuild editor. Double-click on the SuperBlock to focus into it. There should be nothing there initially. Let's add a Gain block.
11. Open (or switch to) the Palette Browser again. In the left pane of the Palette Browser window, click on the Palette below the SuperBlocks labeled Algebraic. Scroll until you can see the Gain icon. Click on the Gain icon and drag it onto your SystemBuild editor window.
12. Close the Palette Browser window as these are the only components that we will use for our simple simulation.

We must now define some of the parameters of these components.

- Click on the Gain component and press <enter> to view its properties. Name the Gain block **altia gain**. To make things a little more interesting, change the gain from 1.0 to 0.5 and click on **OK**.

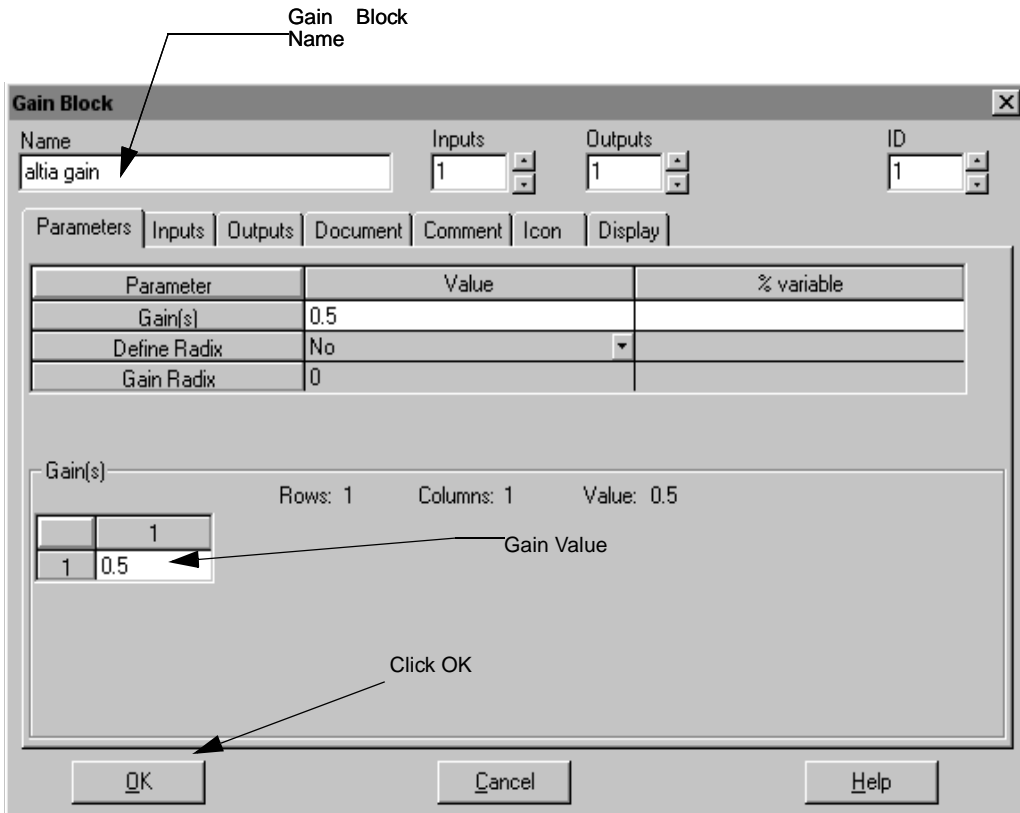
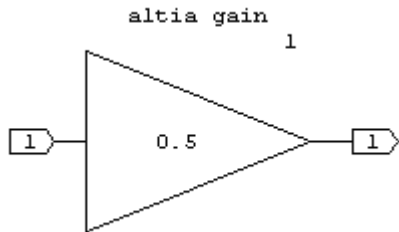



FIGURE 1-4 Gain Block

- After the properties of the Gain block have been set, it must be connected to its parental SuperBlock. To make the Gain block output connection, select the Gain block and then click on the **Connect** menu and choose **Outputs**. To make the Gain block input connection, select the Gain block and then click on the

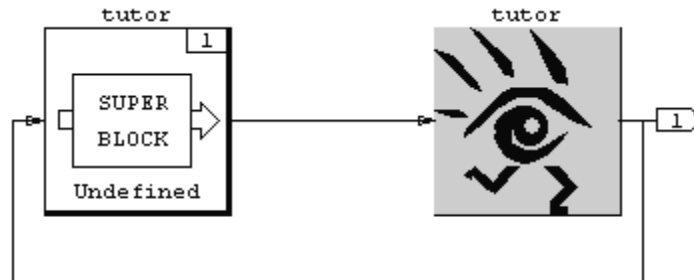
1


Connect menu and choose **Inputs**. The connected Gain block should resemble the following.



15. Press the **View Parent** button  to go back to the parent system which contains the SuperBlock and the Altia block.
16. Next, we will have to make connections within SystemBuild between the SuperBlock, which contains the Gain and the Altia Interface block. When we do this, connecting lines will appear that show which blocks are connected and the direction of the connections.
17. Press and hold the Control key while clicking on the SuperBlock component and then the Altia code block component. Then choose **Blocks** from the SystemBuild editor **Connect** menu. A connector from the SuperBlock to the Altia block will appear.
18. Next, select the Altia block first and then, while holding down Control, click on the SuperBlock. Once again choose **Blocks** from the SystemBuild editor **Connect** menu and you will see that the output of the Altia block is now connected to the input of the SuperBlock

19. There is one more thing we must do to finish out the simulation. We told SystemBuild that our top-level (`altiatut`) SuperBlock had one output. We must now define that output. To do so, click on the Altia block and, from the **Connect** menu, choose **Outputs**. You should notice that an external output indicator appears connected to the output of the Altia block. This indicator is typically labeled 1.



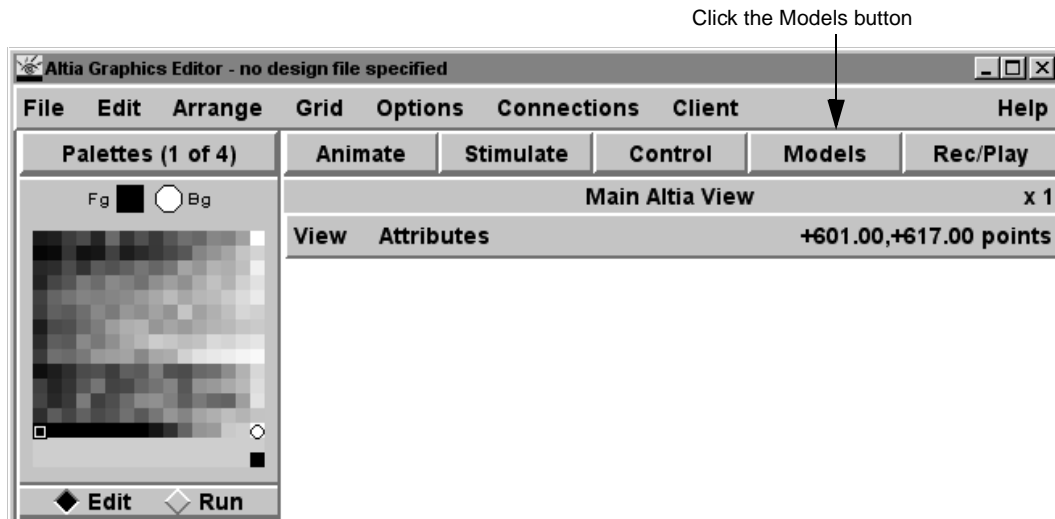
20. We are now finished designing our simple SystemBuild model. Save your model by returning to the Catalog Browser window and pressing the Save button . When asked what to name your model type in `tutor.dat` and click **OK**.

1.2.2 Use Altia Design to Create a Simple GUI

1. Open the Altia Editor by double-clicking on the Altia block in the SystemBuild editor window. This should cause the SystemBuild model to connect to the Altia Editor (a message to this effect will appear in the Xmath window). Every time you double-click on the Altia block in the SystemBuild editor, the connection with Altia will be re-established and the most recent I/O information will be provided to Altia.
2. Altia will open with a blank design file and name it `tutor.dsn` because the editor was started by double-clicking on the Altia block named `tutor`.

In this initial design we will only have a slider bar and an analog meter. Although, we could create these objects ourselves, it is easier to use objects from Altia's included model libraries.

3. To create the slider bar, click on the **Models** button which is located beneath the menu and above the main view.



4. In the Altia-Inquiry dialogue, change to the directory `$ALTIAHOME\models\` and choose the file `CONTROL.DSN`. Then click on the **Open** button.

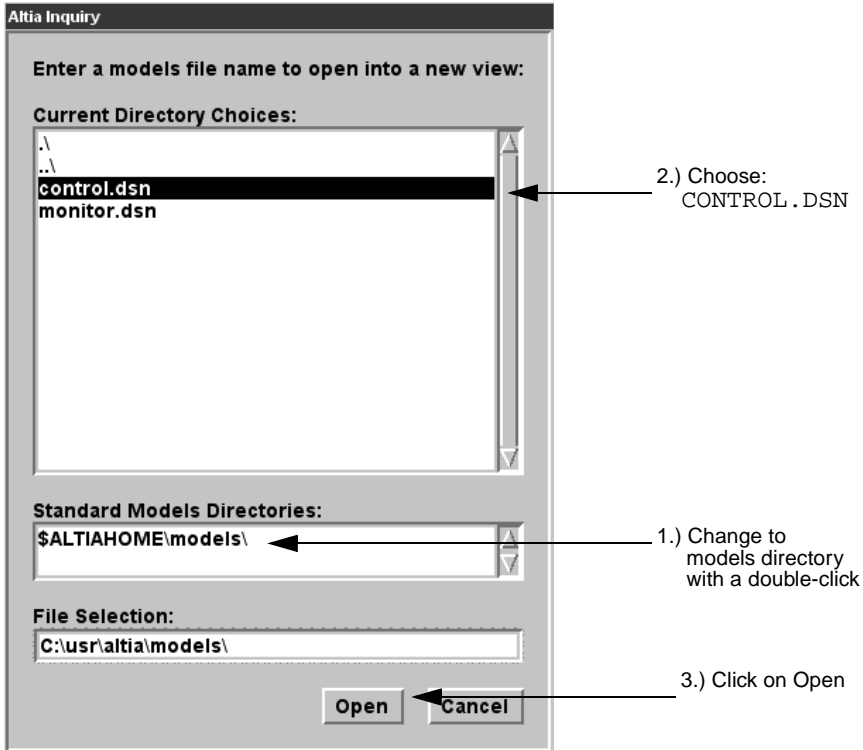
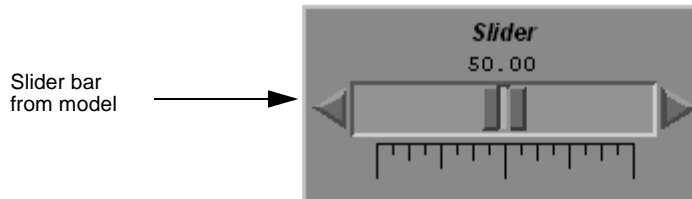


FIGURE 1-5 Altia-Inquiry Dialogue Box

5. A window will open which contains the control models. Press and hold the left mouse button on the slider image and drag it into the Main View of the Altia Editor. After you have placed the slider in the Altia Editor, close the controls Model View window.



6. To modify the properties of the slider, double-click on it to open the Property Dialog. Change the property labeled **Maximum** from 100.0 to 200.0. This will make our slider's range of values change from 0 - 100 to 0 - 200. We could also change many other slider attributes such as color scheme, title, number format, etc. For right now, we will just change its **Maximum** value.

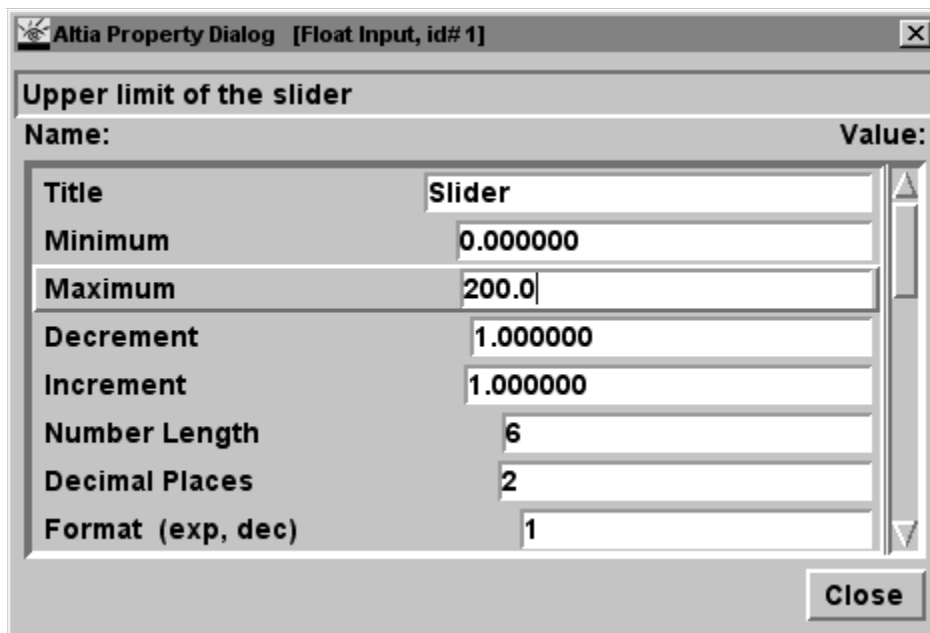
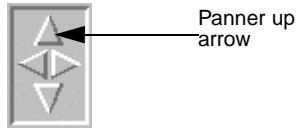


FIGURE 1-6 Altia Property Dialog Box

- The circular meter is located in the `MONITOR.DSN` model library. Open it in the same way that `CONTROL.DSN` was opened. To add the circular meter, you may need to pan up in the Models View until it is visible. To pan, click several times on the up arrow pan button located in the lower left-hand corner of the Models View window.



- Once you have found the circular meter, drag it into the Altia Editor Main View just below the slider. We will leave the circular meter's properties at their default values.



- Now, let's save our GUI to a design file. In the Altia Graphics Editor, choose **Save** from the **File** menu. We have now saved this simple design to the file `tutor.dsn`.

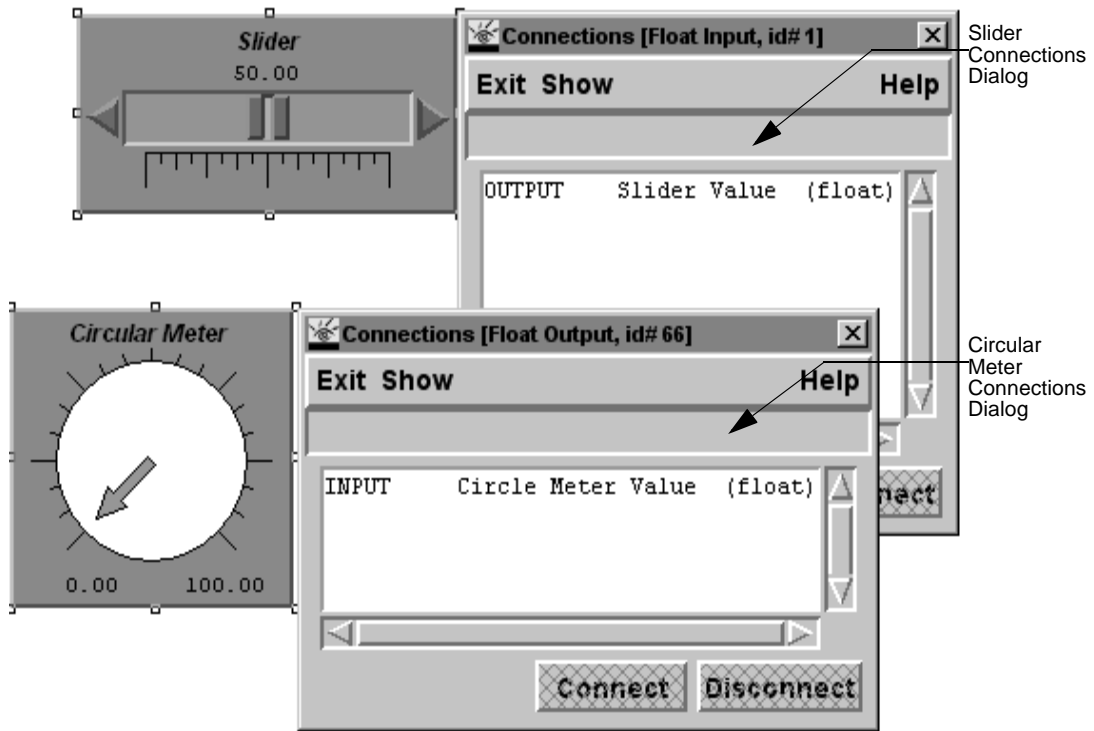
Although the graphics look neat, right now they don't do much because we have yet to connect them to anything. This is our next task.

1.2.3 Connect Your Altia GUI to Your MATRIXx Model

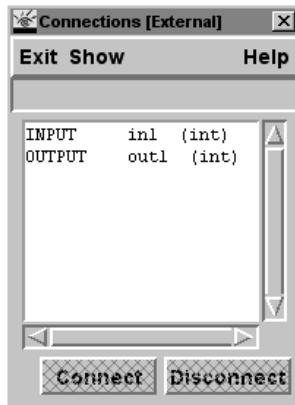
We must now connect the input/output labels of our SystemBuild model to the Altia GUI objects' labels so that the two programs may communicate. This is easily done with Altia.

- In the Altia Editor, select the slider and the circular meter by holding down the `<shift>` key and clicking on the slider and the meter. Next, from the **Connec-**

tions menu, choose **Selected Objects....** Two Connections dialogs will open (one for each selected object).

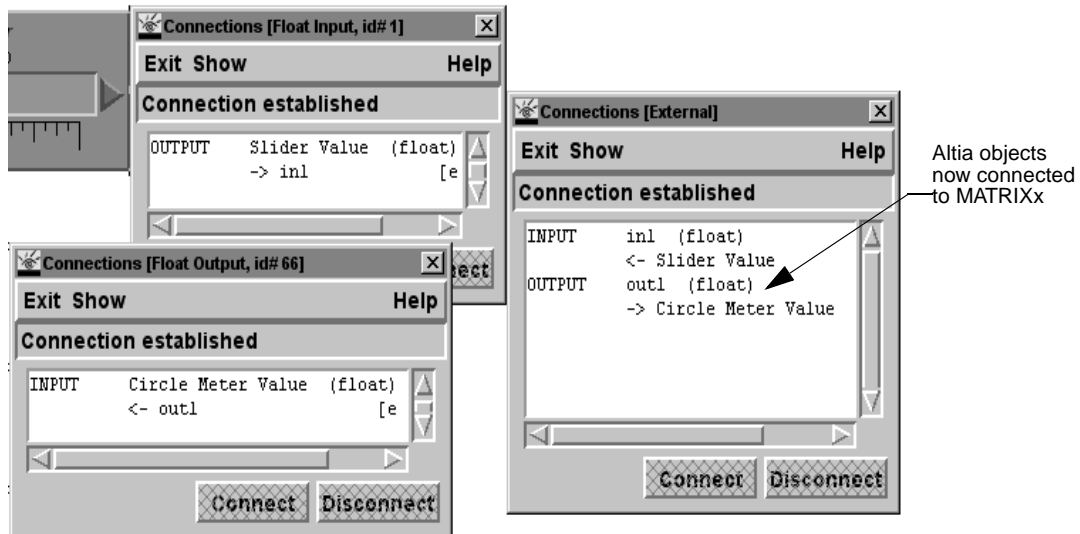


2. From the **Connections** menu, choose **External Signals...** to open the dialog which will allow us to connect to the MATRIXx signals.



3. Click on the signal in the slider Connections dialog labeled **OUTPUT SLIDER VALUE**, then click on the signal in the external Connection dialog labeled **INPUT in1**. When this is done, the **Connect** button should become available. Press it to connect the slider to the input of the SystemBuild model. The Connections dialogs should reflect the fact that the two objects are now connected.

- To connect the output of the model to the circular meter, click on the signal labeled **OUTPUT out1** in the external Connections dialog, then click on the signal labeled **INPUT Circle Meter Value** in the circular meter Connections dialog. Press the **Connect** button to connect the output of the model to the circular meter.



- The connections have now been established. Save the changes (connections) to your Altia design by choosing **Save** from the **File** menu. You must save whenever external connections are modified. Otherwise, the changes will not be recognized by a MATRIXx simulation. After saving, close all of the connections dialogs by choosing **Close All** from the **Exit** menu in one of the dialogs.

1.2.4 Test Your Altia GUI/MATRIXx Model Interaction

1. Select the SystemBuild editor window, and make sure no objects are selected. From the SystemBuild editor **Tools** menu, choose **Simulate**.
2. Click on the Time Vector/Variable field and enter `[0:1:10000]`' (Please note the single quote). Click on the **Interactive** checkbox and then click on **OK**.

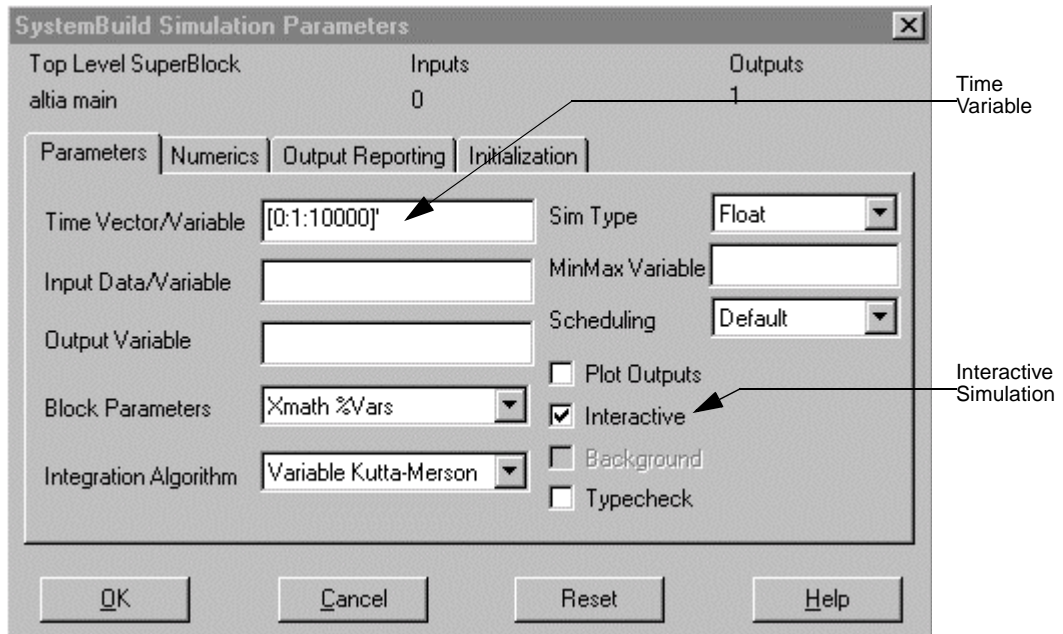


FIGURE 1-7 SystemBuild Simulation Parameters Window

3. The **Interactive Simulator** window will open showing the contents of our `altiatut` SuperBlock. We are ready to simulate. Your Altia Design editor window should still be open to your GUI design from the previous section. Change your focus to this window. If the Altia editor is not already opened, an Altia runtime-only window will automatically open for your GUI design.
4. In the Altia Design editor, you are probably in **Edit** mode (the radio buttons initializing **Edit/Run** modes are located just below the color palette on the left side of the Main View). Unselect all of the Altia objects and then switch the Altia Design editor to the **Run** mode by clicking on the **Run** radio button.

5. Select the SystemBuild Interactive Simulator window. From the **Simulation** menu, choose **Resume** and the simulation will begin.
6. Switch back to the Altia Design editor window. Click on the handle of the slider bar and drag the mouse from side to side. You will notice that the circular meter will change to reflect the status of the Gain output (which is 50% of the slider bar value). After a short while, the simulation will time out and the Altia objects will stop responding. You can **Reset** and **Resume** the simulation from the **Simulation** menu in the Interactive Simulator window.
7. When you are finished simulating, exit from the Interactive Simulator window by choosing **Exit** from the **File** menu. Close the Altia Design editor by choosing **Quit** from its **File** menu. If the simulation started its own Altia Runtime window named Main Altia View, close the window by selecting it and pressing **<Ctrl>-c** (the Control key and **<c>** key simultaneously).

1.2.5 Translating Existing PIC Files

If you already have an existing PIC file that you would like to import into Altia, then you must translate the PIC file. As an illustrative example, we will translate one of the demos that is shipped with MATRIXx.

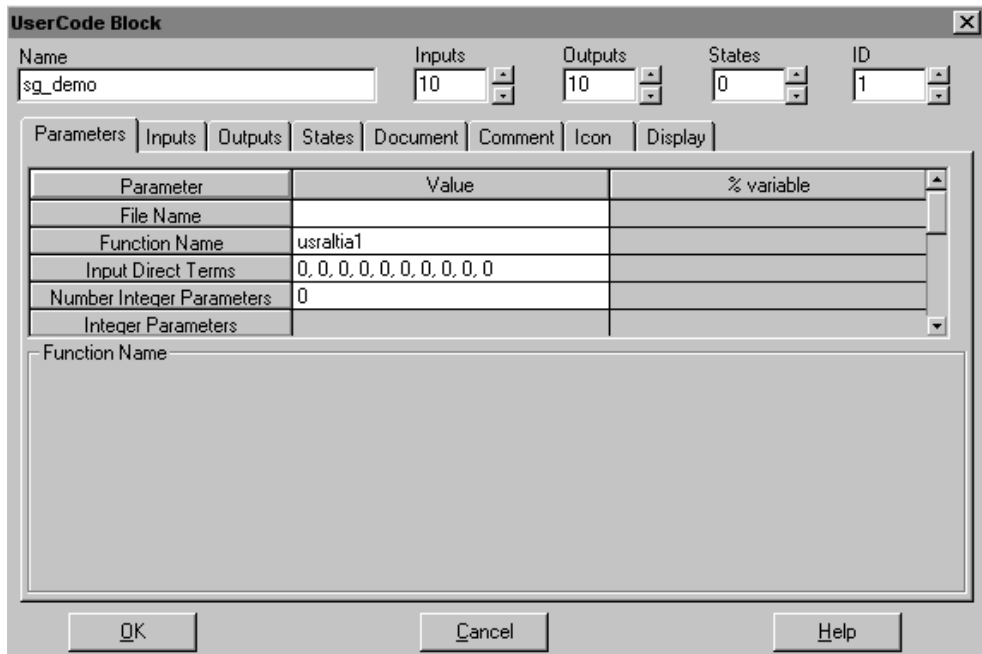
1. Create a new directory and name it **sg_demo**. Copy the **sg_demo.pic** and **sg_demo.rtf** files from the `$NIHOME\sysbld\demo\ia_demo` directory to the new directory.
2. Exit Altia and restart Xmath. Change to the new directory by typing

```
set directory="<full_path_to_new_directory>"
```

and pressing **<enter>** in the Xmath command line.
3. To translate the PIC file to a DSN file, simply type **pictodsn** on the Xmath command line. First, an error dialog will appear saying that the default file **pict.pic** could not be found. Click OK to dismiss this dialog. Next, a dialog will appear prompting you for a PIC file to translate because our new directory has no **animation.cfg** file. Type **sg_demo.pic** and press **<enter>**.
4. When the conversion is finished, a dialog will appear that says **Conversion complete!** This dialog will also tell you if there were any errors in converting the file and how many objects were translated.
5. Click **OK** to dismiss the dialog. If you look in your **sg_demo** directory, a new **sg_demo.dsn** file will have been created. You will also see **sg_demo.rtm** (the

Altia runtime configuration file for `sg_demo.dsn`) and `sg_demo.mxc` (the Altia-MATRIXx connections file for `sg_demo.dsn`).

6. Type **build** in the Xmath command line to open SystemBuild. Switch to the Catalog Browser window, and from the **File** menu, choose **Load**. In the **File name** box, type `sg_demo.rtf` and then press **<enter>**.
7. Double-click on the **S G Demo** SuperBlock in the Catalog Browser to open the SystemBuild editor. Select the User Code block by clicking on it and then press **<enter>**.
8. Change the block's Name to `sg_demo` and change the Function Name from `01` (or whatever it says) to `usraltia1` (the last letter of this name is a "one") and then click **OK**.



9. In the SystemBuild Editor window, double-click on the User Code block to open the Altia editor. You should see a design that is similar to the PIC file except that the IA graphics have been replaced with Altia versions.

If you do not wish to have new versions of the IA graphics in your Altia design, you may specify that option in the `pictodsn` command. To do so, type the following line exactly (including double quotes) on the Xmath command line:

```
pictodsn "--style old"
```

If you open the resulting `sg_demo.dsn` file with Altia (by first closing the existing Altia editor session and again double-clicking on the **User Code** block), you will see that it is indeed the same as the PIC file.

You can simulate with either version of `sg_demo.dsn` by choosing **Simulate** from the SystemBuild Editor **Tools** menu.

1.2.6 Creating Connections to Existing Altia Objects

As shown in the previous sections, the Altia objects taken from the `.dsn` files in the `$ALTIAHOME/models` directory already have connection information built into them. This is also true for objects from the `.dsn` files in `$ALTIAHOME/models/isiold`. What if you want to use the connections dialog on an older Altia object or an object that you have created yourself? Luckily, creating new connections to existing objects is simple.

1. Let's go back and open our `tutor.dsn` file in Altia.
2. Remove the dial gauge by clicking on the **Cut** button and then clicking on the dial gauge.
3. Open a new model library by clicking on **Models** and then going to the `$ALTIAHOME/models/more` directory. Then choose the `instrmnt.dsn` model file and click **Open**.
4. Pan up until you see the 0 - 100 needle gauge. Click on it and drag it into the Altia main view. Open the Animation editor by clicking on the **Animation** button and check to see what the animation name is on the slider (it will probably be `105_gauge`).
5. From the Altia **Connections** menu, choose **Selected Object** to show the connections of the slider. Right now, it will have none. In the Connections dialog, click on the **Edit** menu and choose **Add Connection**.
6. In the Edit Connection dialog, type the I/O Name **Needle Gauge** (this could be anything you want to identify the gauge). Next, type the Animation name observed in step 4 (probably `105_gauge`). To make things interesting, also type **x.5** in the **Options** text box (this will multiply incoming values from MATRIXx by one-half). Click **OK** to make the connection.

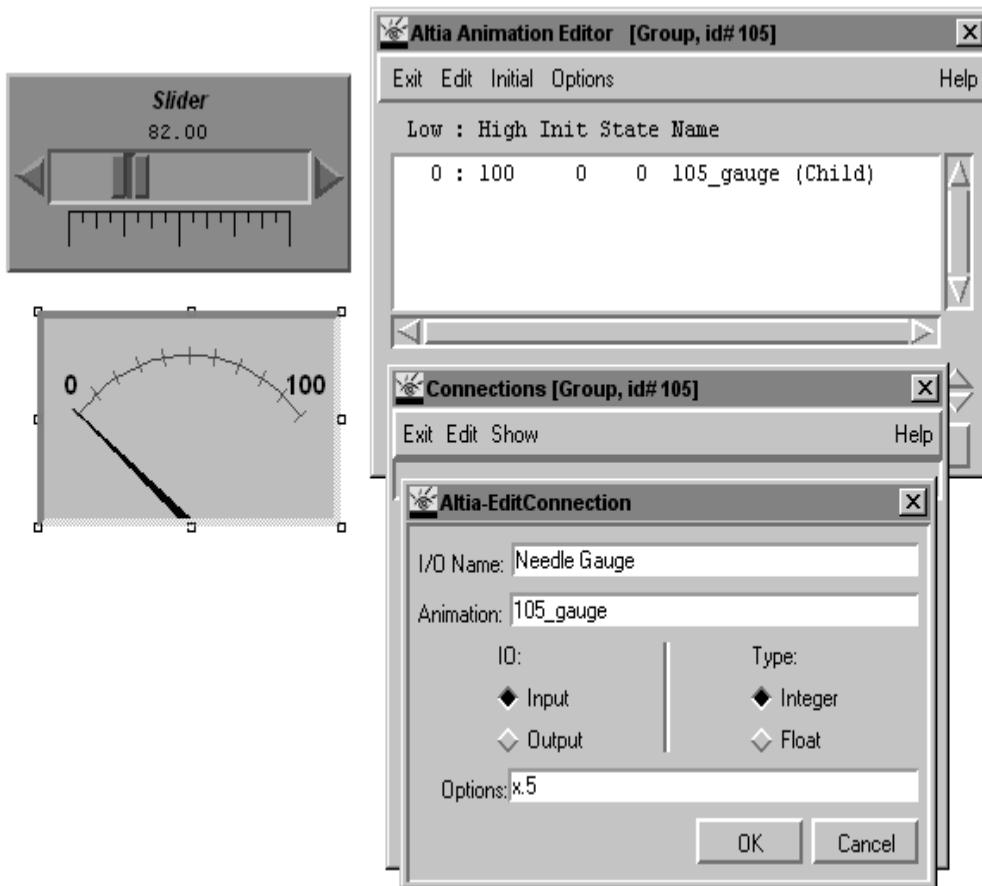
A short discussion about the **Options** field may be helpful at this point. The **Options** field gives you an opportunity to specify options recognized by the Altia block in SystemBuild. Entries in the **Options** field only have an effect when the object input or output is connected to a MATRIXx signal (which is the next step in our example).

The most commonly used option is the **x#** multiplier (for example, the **x.5** that you just entered). If a connection is an input into the object (which is the case with our example), then the value from MATRIXx will be multiplied by the specified number before it is sent to the object. For an output connection (such as on a slider), the value from the object is multiplied before it arrives as an input to the MATRIXx model.

Another useful option is the **continuous** setting for Altia object output connections. If an object's output is going to have many different values over a large range (which is common for a slider or knob), add the word **continuous** to the **Options** field. This will significantly improve data throughout between your Altia design and the SystemBuild Altia block. If you need to enter a multiplier and continuous setting in the same **Options** field, simply separate them with one or more spaces.

7. Now we can connect our new needle gauge to an external signal in much the same way as we did before. From the Altia **Connections** menu, choose **External Signals**. Next, in the External Signals dialog, click on the **OUTPUT** and in the

needle gauge connection dialog, click on **INPUT** then click on the **Connect** button.



8. Save your design (this is an important step so be sure to do it) and switch to the SystemBuild Catalog Browser. If you have not already done so, open the **altiatut** superblock by double-clicking on its entry in the Catalog Browser.
9. Start the simulation window (choose **Simulate** from the SystemBuild editor's **Tools** menu) and enter the same information as before (Time/Vector Variable = [0:1:10000]' and Interactive animation = On) then press **OK**.

10. Hit the **play** button in the Interactive Simulator window and the Altia runtime window will open and you can interact with the design. If you originally opened the Altia Design editor by double-clicking on the Altia block in your `altiatut` superblock, you won't get a new Altia runtime window. Instead, the simulation will connect to the editor! In this case, switch the Altia Design editor to the run mode by clicking on the **Run** radio button. Now you can use the slider in your drawing area to interact with the model simulation.

1.2.7 Tutorial Summary

In this tutorial, we have created an Altia GUI ([Section 1.2.2](#)) and a SystemBuild model ([Section 1.2.1](#)) from scratch. We then used the Connections dialogs in Altia to connect the GUI to the model ([Section 1.2.3](#)). In order to be sure that our connections were correct, we simulated the SystemBuild model and stimulated it using our Altia interface ([Section 1.2.4](#)).

Finally, we discussed how to translate existing PIC files to Altia ([Section 1.2.5](#)) and adding connection information to older Altia objects or custom objects ([Section 1.2.6](#)).

