In this example, we tried to pass to the AppendToGraph function a reference to a wave that did not exist. To find the source of the error, you should use Igor's debugger and set it to break on error (see **Debugging on Error** on page IV-213 for details).

Runtime Errors From Built-in Functions

Normally, when an error occurs in a built-in function, the built-in function does not post an error but instead returns 0, NaN or an empty string as the function result. As a debugging aid, you can use the rtFunctionErrors pragma to force Igor to post an error. See **The rtFunctionErrors Pragma** on page IV-55 for details.

Custom Error Handling

Normally when an error occurs, Igor aborts function execution.

Sophisticated programmers may want to detect and deal with runtime errors on their own. See **Flow Control for Aborts** on page IV-48 for details and examples.

Coercion in Functions

The term "coercion" means the conversion of a value from one numeric precision or numeric type to another. Consider this example:

```
Function foo(awave)
   WAVE/C awave

   Variable/C var1

   var1 = awave[2]*cmplx(2,3)
   return real(var1)
End
```

The parameter declaration specifies that awave is complex. You can pass any kind of wave you like but it will be coerced into complex before use. For example, if you pass a real valued integer wave the value at point index 2 will be converted to double precision and zero will be used for the imaginary part.

Operations in Functions

You can call most operations from user-defined functions. To provide this capability, WaveMetrics had to create special code for each operation. Some operations weren't worth the trouble or could cause problems. If an operation can't be invoked from a function, an error message is displayed when the function is compiled. The operations that can't be called from a function are:

```
AppendToLayout     Layout       Modify             OpenProc
PrintGraphs        Quit         RemoveFromLayout
Stack              Tile
```

If you need to invoke one of these operations from a user-defined function, use the Execute operation. See **The Execute Operation** on page IV-201.

While Modify can not be called from a function, ModifyGraph, ModifyTable and ModifyLayout can.

You can use the **NewLayout** instead of Layout, the **AppendLayoutObject** instead of AppendToLayout, and the **RemoveLayoutObjects** instead of RemoveFromLayout.

External operations implemented by very old XOPs also can not be called directly from user-defined functions. Again, the solution is to use the Execute operation.