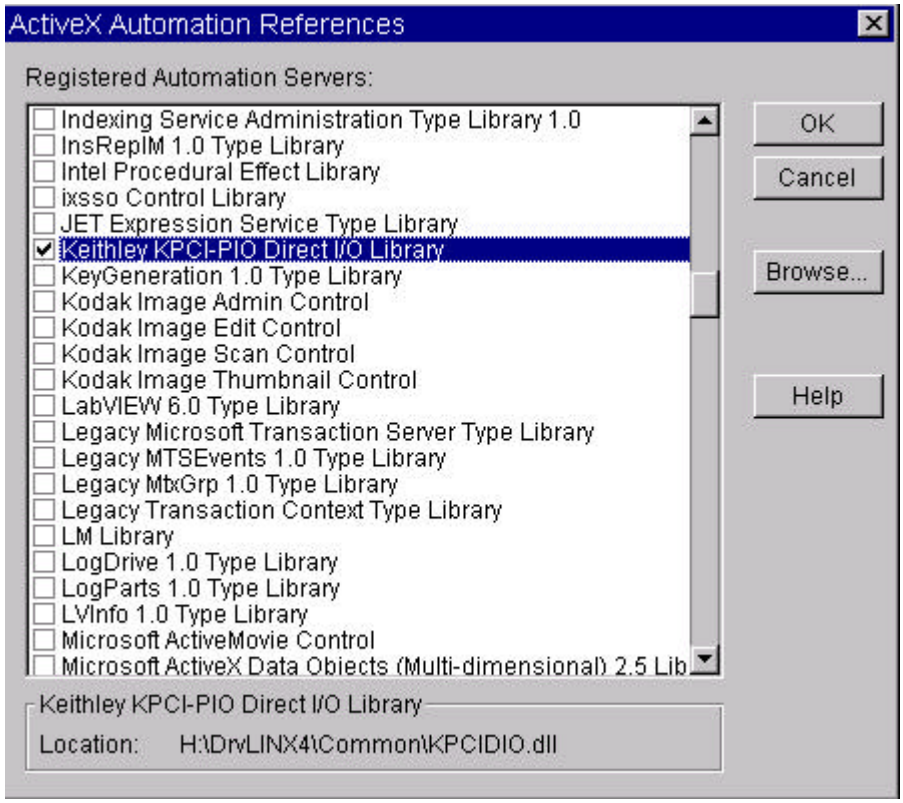
Using the Direct I/O COM API of DriverLINX with Agilent VEE 5.0

DriverLINX drivers for digital I/O boards provides two programming interfaces:
1. Service Request API for hardware independence with other KPCI or PIO boards
2. Direct I/O COM API for simple, fast interaction with the registers of the digital I/O hardware

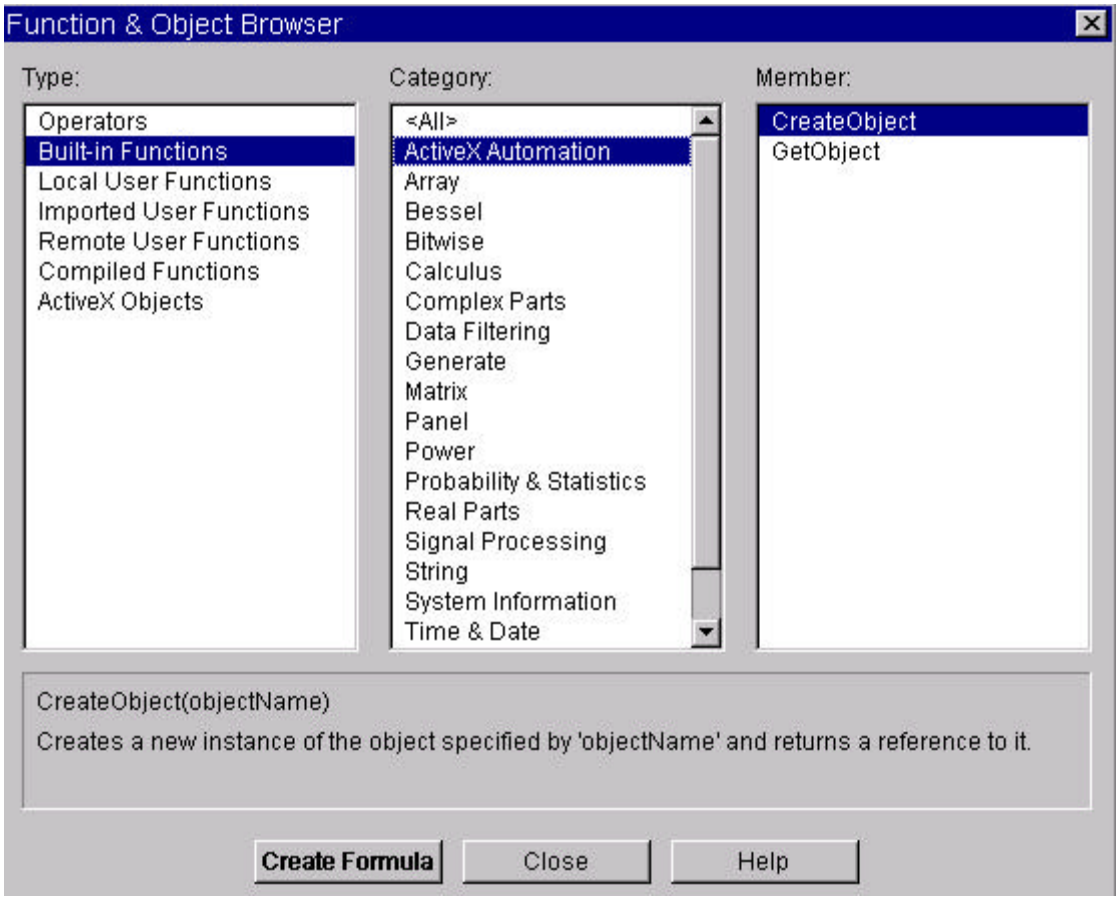
Below is a step by step guide for use of the Direct I/O COM object API. The completed project is also available as an example program in the download center.

Step 1: Make sure VEE is set for Standard compatibility mode (in Default Preferences) for support of ActiveX. Then, include a reference to the object: Select ActiveX Automation (not Control) References from the Device menu. The dialog below will be displayed. Scroll down the alphabetical list and place a check in the box for the Keithley KPCI-PIO Direct I/O Library.



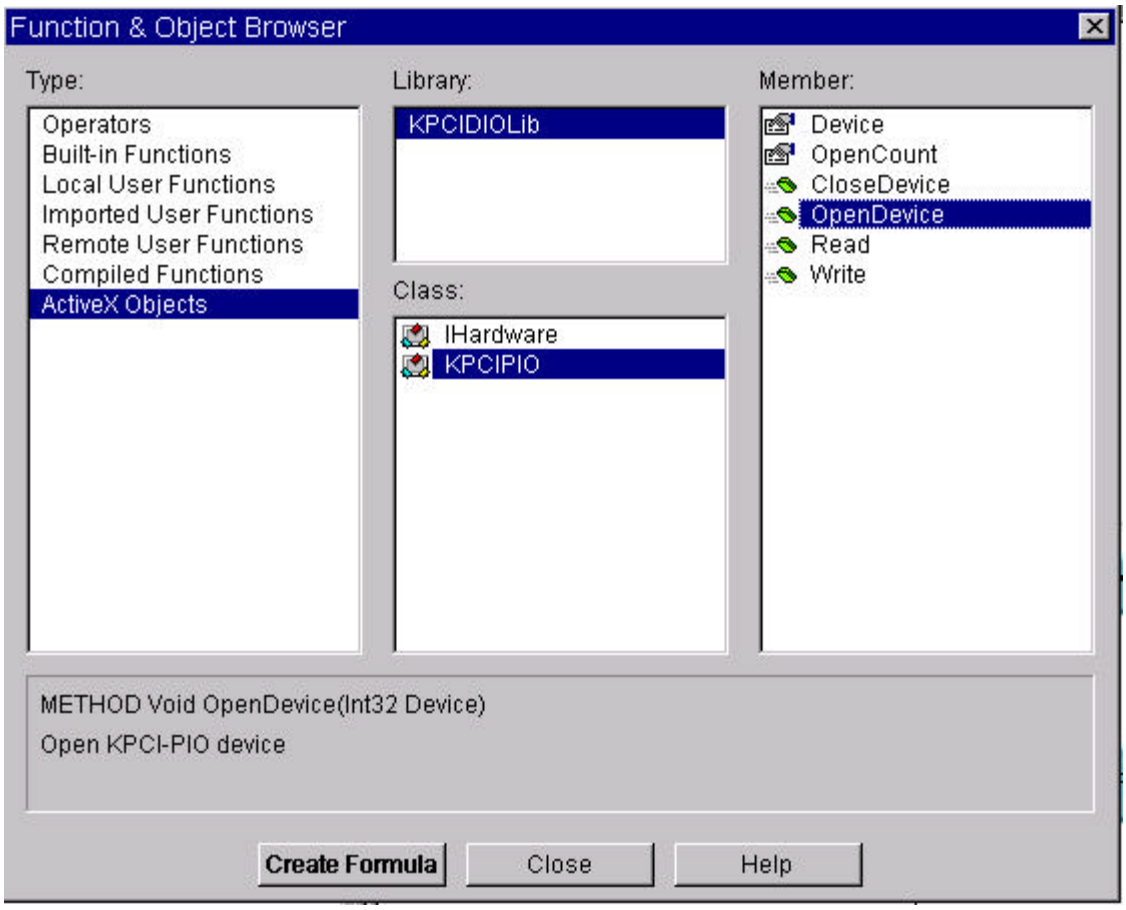
Step 2: Declare an ActiveX Automation Variable. This will allow VEE to perform type checking later in the code. It does not create an instance of the object. From the menus, select Data|Variable|Declare Variable. The name can be anything that makes sense to you; below it is set to DirectIO. Set the type to object. Check the Specify Object Type box. Notice the object has two classes: KPCIPIO and IHardware. KPCIPIO provides an 8255 interface: a control register and one register for each of three 8-bit digital ports. The IHardware class provides 32-bit register access to the KPCI-PIOxx card. The balance of this application note will refer to the KPCIPIO class only. Select the KPCIPIO class (not the IHardware class).

Step 3: Create an object. Click the function and object browser icon to bring up the dialog below. Highlight the three areas shown: Built-in Functions, ActiveX Automation, CreateObject member. Then click the Create Formula button.

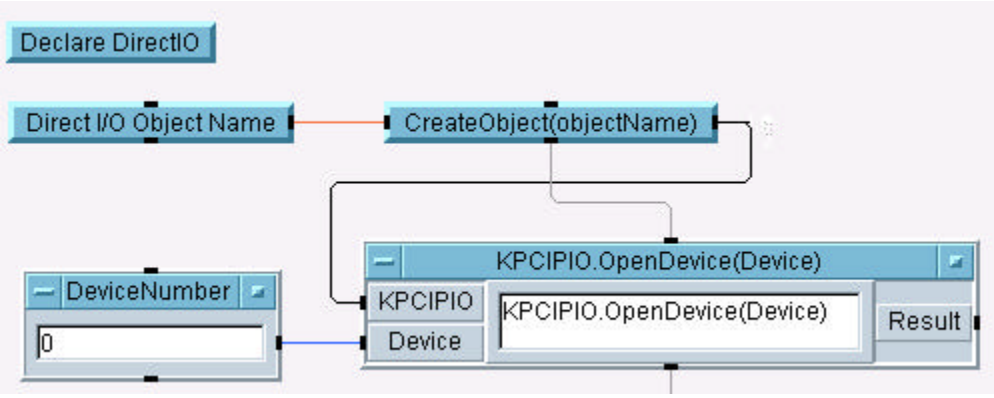


This will place a function object into the VEE project, however, to use this function to create our object we need to know the object name. A text constant is used to pass KPCIPIO.1 which is the object name of the Direct I/O COM object (from the Windows registry) for the KPCI-PIO24 and KPCI-PIO96 boards. The result terminal of this function will be used will all subsequent function blocks that call a method of the object.

Step 4: Use the methods of the object. Again, click the function and object browser icon to bring up the dialog below. Highlight the four areas shown: ActiveX Objects, KPCIPIO class and the OpenDevice member. Then click the Create Formula button.



The screen shot below shows the OpenDevice function block in the VEE project. The output of the CreateObject function is wired to this function. This OpenDevice method also requires a device number parameter. This device number corresponds to the device number assigned in the DriverLINX Configuration Panel. Device Number can be any value between 0 and 5 (default is 0).



To complete the program, the read and write methods of the object would be used to configure the ports for input or output direction and to control or read their values. Before program exit, the CloseDevice method should be called.

You will need to know something of the 8255 chip to know at what offsets (address) and what values to write. Below is a brief summary:

Port A is at an offset of 0
Port B is at an offset of 1
Port C is at an offset of 2

The control register is at an offset of 3.

Individual bits of the control register control the input or output direction of the individual ports. Write a 0 to the bit to make the corresponding Port an output; write a 1 to make it an input. All ports are inputs by default at PC power-up or reset. If using only digital inputs, then the control register is not important for your application.

bit 0.....4 lower bits of Port C
bit 1.....Port B
bit 2.....not used, always zero
bit 3.....4 upper bits of Port C
bit 4.....Port A
bit 5.....not used, always zero
bit 6.....not used, always zero
bit 7.....not used, **always 1**

To make all ports outputs, write a hex value of 80 to the control register and then write to the registers for the individual ports to control their logic 0 or 1 state.

Consult chapter 7 of the DriverLINX Tutorial Manual or the Using DriverLINX with your KPCI-PIO Series manual for more information.