Configuring a PXI system in Windows and Linux

The system we need to identify under Linux is a PXI-1031 chassis with a PXI-8196 embedded controller connected via MXI-3 to a 1045 chassis.

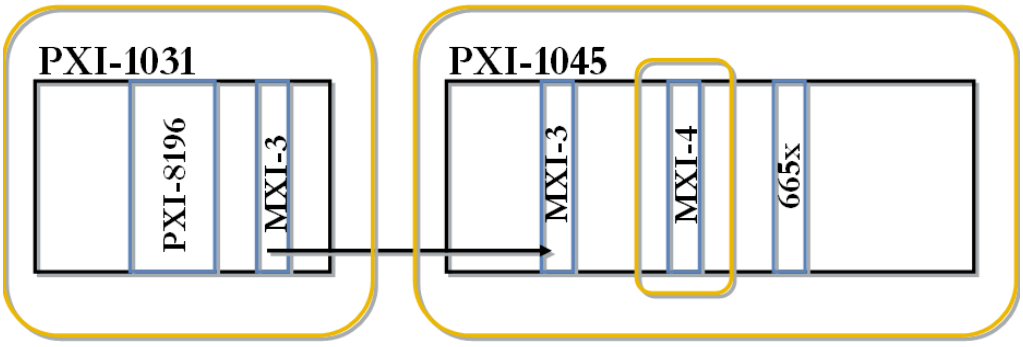

Figure 1.0. Example system

Below is the procedure used to configure the above system. The images illustrate how the system is configured under both Windows and Linux for comparison purposes.

Note: MXI-4 is the recommended product for PXI chassis-chassis communication. It provides error correction unlike MXI-3.

1.0 Display the PXI System

1. We first un-identify the system to start the identification from scratch
2. We then list all the components as shown below.

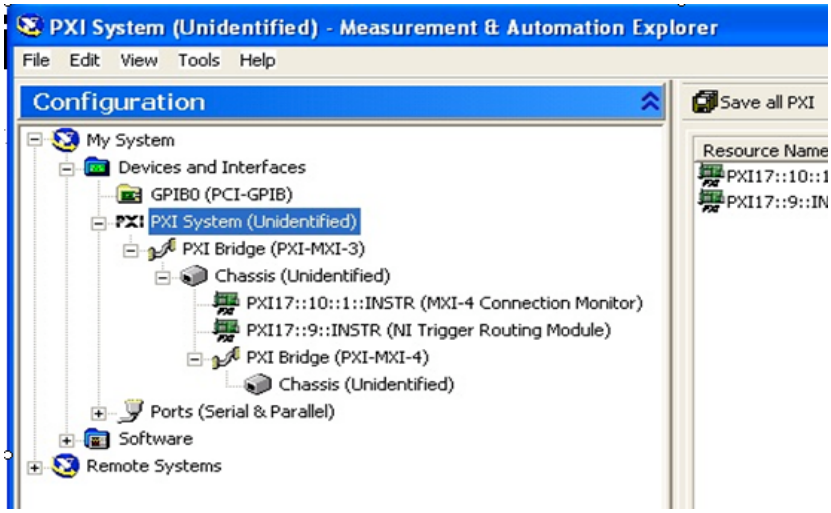

Figure 1.1. An unidentified system using MAX in Windows

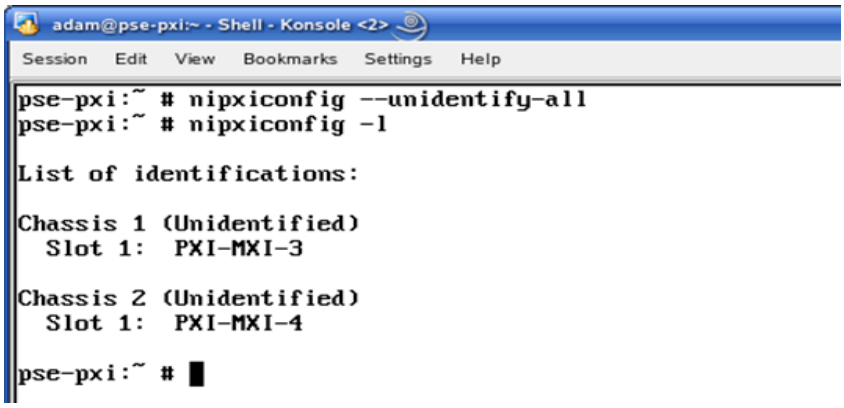

Figure 1.2. An unidentified system using nipxiconfig under Linux

2.0 Identify the Controller

3. Identify the controller (a PXI-8196 in this case)

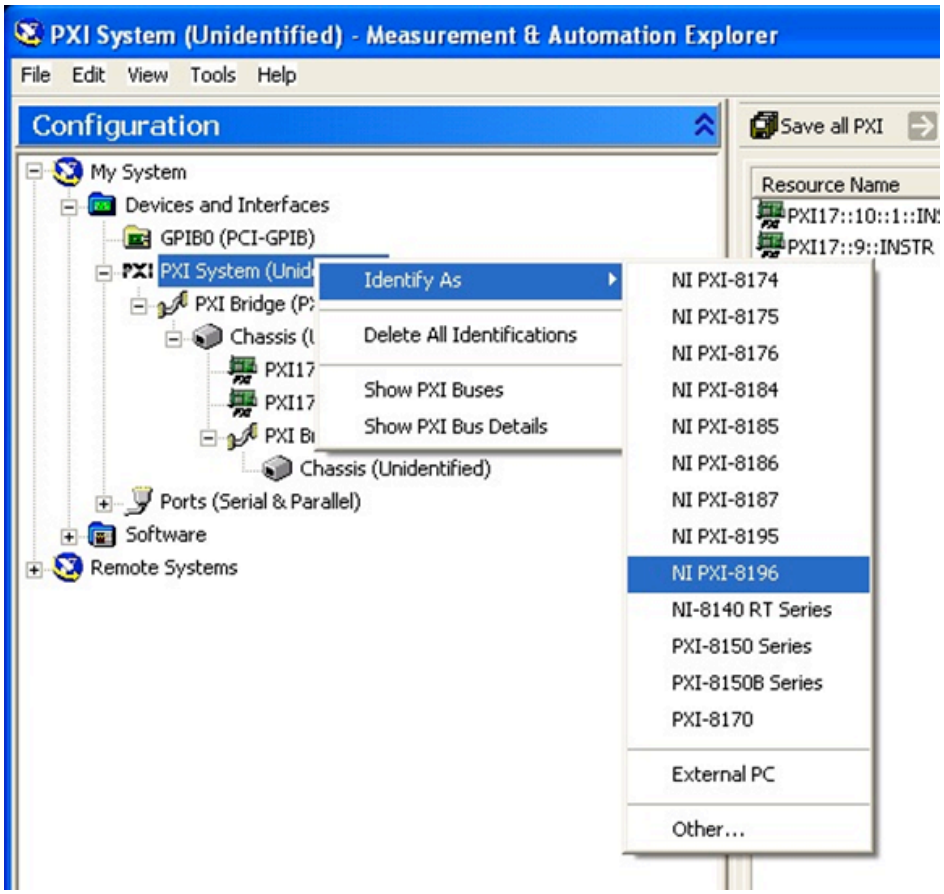

Figure 2.0. Identifying a controller using MAX under Windows

```
pse-pxi:~ # nipxiconfig --identify-controller "National Instruments" "NI PXI-8196"
pse-pxi:~ # nipxiconfig -l

List of identifications:

Chassis 1 (Unidentified)
  Slot 1:  PXI-MXI-3

Chassis 2 (Unidentified)
  Slot 1:  PXI-MXI-4

Chassis 3 (Unidentified)
  Slot 1:  NI PXI-8196
```

Figure 2.1. Identifying a controller using nipxiconfig under Linux

3.0 Identify the Chassis

4. After the controller is identified, identify the first chassis (chassis containing the PXI-8196 controller)

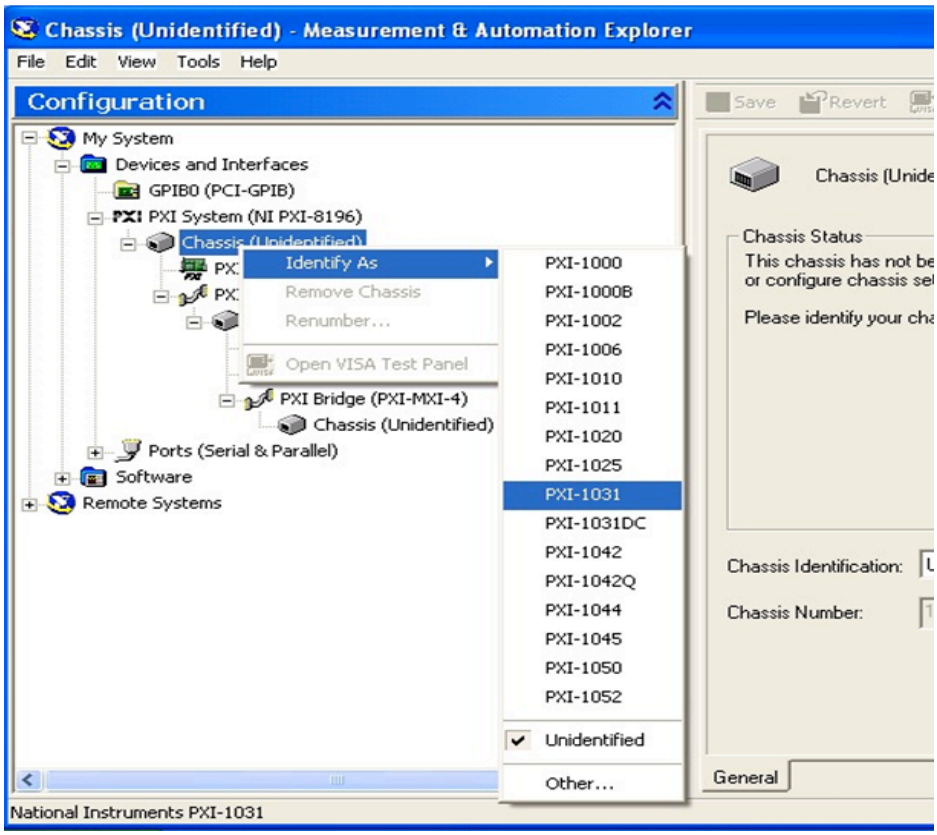

Figure 3.0. Identifying a chassis using MAX under Windows

Note: You can identify a chassis using the manufacturer name and model or by pointing MAX or nipxiconfig to the chassis.ini or controller.ini files. In MAX this is done by choosing **Identify as > other**. Using nipxiconfig, simply point the tool to the .ini file as in the example below.

```
pse-pxi:~ # ls /etc/natinst/nipxi/HardwareDescriptions/chassis/
.                                National Instruments PXI-1011.ini    National Instruments PXI-1042.ini
..                               National Instruments PXI-1020.ini    National Instruments PXI-1042Q.ini
National Instruments PXI-1000.ini   National Instruments PXI-1025.ini    National Instruments PXI-1044.ini
National Instruments PXI-1000B.ini  National Instruments PXI-1031.ini    National Instruments PXI-1045.ini
National Instruments PXI-1002.ini   National Instruments PXI-1031DC.ini  National Instruments PXI-1050.ini
National Instruments PXI-1006.ini   National Instruments PXI-1036.ini    National Instruments PXI-1052.ini
National Instruments PXI-1010.ini   National Instruments PXI-1036DC.ini  National Instruments PXI-1056.ini
pse-pxi:~ # nipxiconfig --identify-chassis 3 "/etc/natinst/nipxi/HardwareDescriptions/chassis/National Instruments PXI-1
031.ini"
pse-pxi:~ # nipxiconfig -l

List of identifications:

Chassis 1 (Unidentified)
  Slot 1:  PXI-MXI-3 from Chassis 3, Slot 3

Chassis 2 (Unidentified)
  Slot 1:  PXI-MXI-4

Chassis 3 (PXI-1031)
  Slot 1:  NI PXI-8196
  Slot 3:  PXI-MXI-3 to Chassis 1 (Unidentified)
```

Figure 3.1. Identifying a chassis using nipxiconfig for Linux (path method)

5. We now need to identify the second chassis linked through the MXI-3 interface.

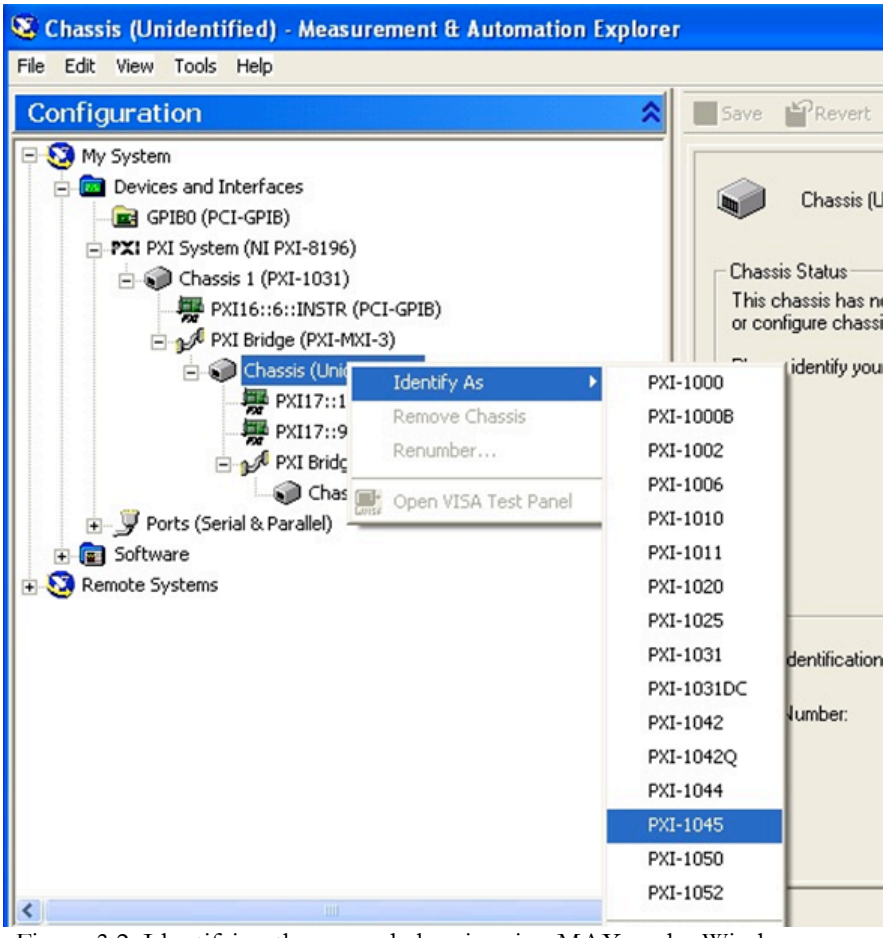

Figure 3.2. Identifying the second chassis using MAX under Windows

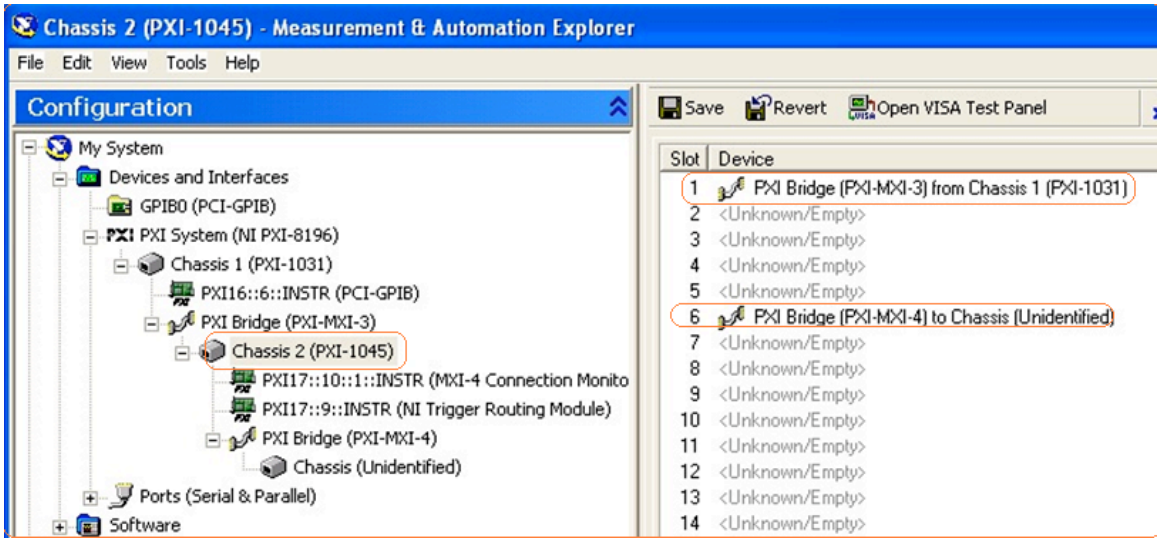

Figure 3.3. Both chassis and the embedded controller identified using MAX

```
pse-pxi:~ # nipxiconfig --identify-chassis 1 "National Instruments" PXI-1045
pse-pxi:~ # nipxiconfig -l brief

List of identifications:

Chassis 1 (PXI-1045)
  Slot 1:  PXI-MXI-3 from Chassis 3, Slot 3
Chassis 2 (Unidentified)
  Slot 1:  PXI-MXI-4 from Chassis 1, Slot 7
Chassis 3 (PXI-1031)
  Slot 1:  NI PXI-8196
```

Figure 3.4. Identifying the second chassis in nipxiconfig and displaying the brief system output

At this point, you have successfully configured and identified the demo system in Figure 1.0

4.0 Trigger Routing and Reservation (Optional)

Chassis 1 (chassis 2 under Windows) is a PXI-1045 which is an 18 slot chassis that has 3 segments. Triggers need to be routed across the segments explicitly when using drivers that do not provide automatic routing such as DAQmx. In Windows, we can do that through the Trigger tab of the PXI chassis. Similarly, we can use the nipxiconfig utility in Linux to route triggers across segments. Below is an example of routing triggers for the PXI-1045 chassis that was used in the above example. For more information on PXI timing and triggering, see KB 3TJDOND8 at ni.com.

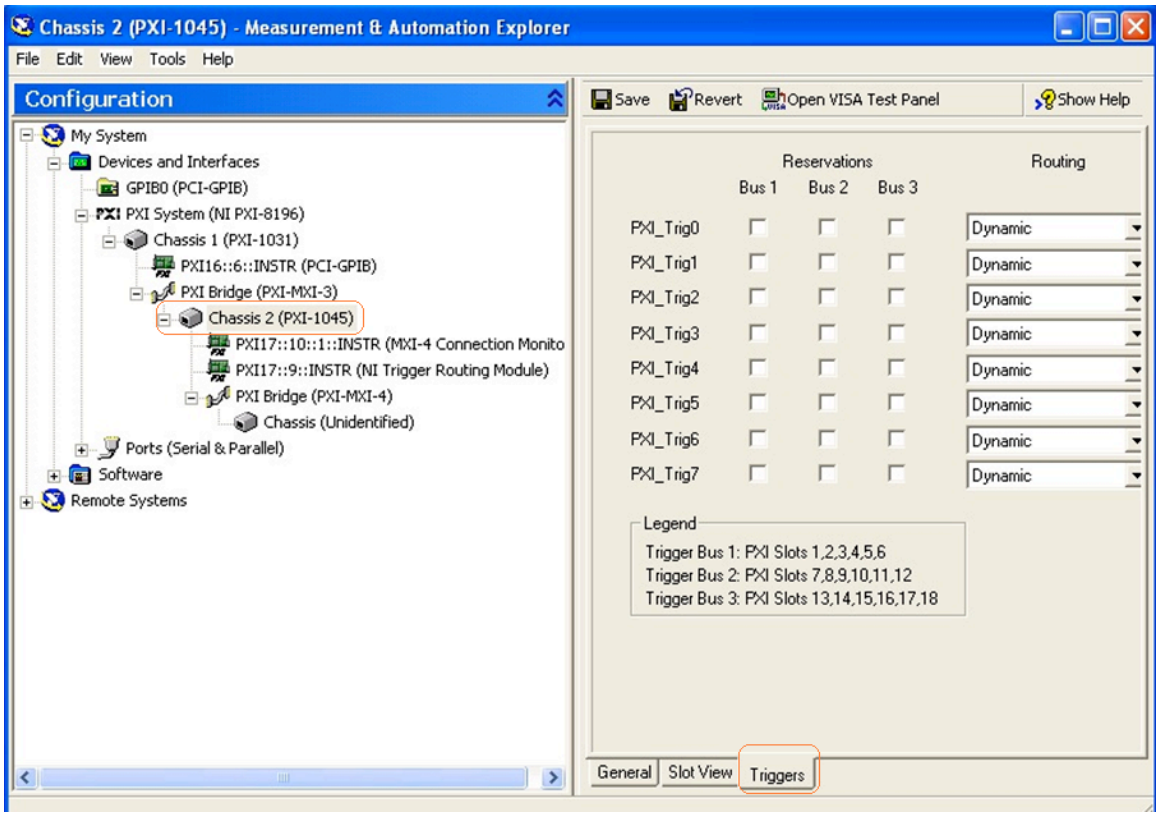

Figure 4.0. Trigger bus view using MAX under Windows

```
pse-pxi:~ # nipxiconfig --list-triggers 3

Chassis 3 (PXI-1031) PXI Trigger Configuration
            TriggerBus1
PXI_Trig0 -      [ ]
PXI_Trig1 -      [ ]
PXI_Trig2 -      [ ]
PXI_Trig3 -      [ ]
PXI_Trig4 -      [ ]
PXI_Trig5 -      [ ]
PXI_Trig6 -      [ ]
PXI_Trig7 -      [ ]
pse-pxi:~ # nipxiconfig --list-triggers 1

Chassis 1 (PXI-1045) PXI Trigger Configuration
            TriggerBus1    TriggerBus2    TriggerBus3
PXI_Trig0 -      [ ]            [ ]            [ ]
PXI_Trig1 -      [ ]            [ ]            [ ]
PXI_Trig2 -      [ ]            [ ]            [ ]
PXI_Trig3 -      [ ]            [ ]            [ ]
PXI_Trig4 -      [ ]            [ ]            [ ]
PXI_Trig5 -      [ ]            [ ]            [ ]
PXI_Trig6 -      [ ]            [ ]            [ ]
PXI_Trig7 -      [ ]            [ ]            [ ]
pse-pxi:~ #
```

Figure 4.1. Trigger bus view using nipxiconfig under Linux

Above figure shows the trigger bus for the PXI-1031, which is a single segment chassis and also for the PXI-1045, which has 3 segments as shown above. In order to reserve triggers in MAX, you place a check on the box that you intend to reserve. In this example, we are reserving **PXI_Trig1** in Chassis **2.**

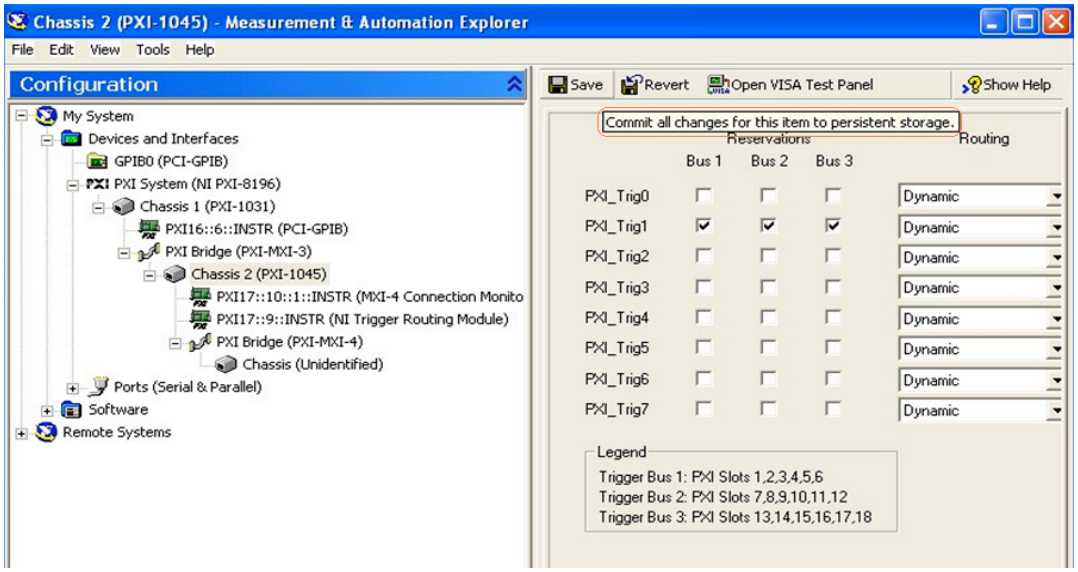

Figure 4.2. Reserving PXI_Trig 1 in chassis 2 under Windows

```
pse-pxi:~ # nipxiconfig --list-triggers 2

Chassis 2 (PXI-1045) PXI Trigger Configuration
            TriggerBus1    TriggerBus2    TriggerBus3
PXI_Trig0 -      [ ]            [ ]            [ ]
PXI_Trig1 -      [ ]            [ ]            [ ]
PXI_Trig2 -      [ ]            [ ]            [ ]
PXI_Trig3 -      [ ]            [ ]            [ ]
PXI_Trig4 -      [ ]            [ ]            [ ]
PXI_Trig5 -      [ ]            [ ]            [ ]
PXI_Trig6 -      [ ]            [ ]            [ ]
PXI_Trig7 -      [ ]            [ ]            [ ]
pse-pxi:~ # nipxiconfig --reserve-trigger 2 1
pse-pxi:~ # nipxiconfig --list-triggers 2

Chassis 2 (PXI-1045) PXI Trigger Configuration
            TriggerBus1    TriggerBus2    TriggerBus3
PXI_Trig0 -      [ ]            [ ]            [ ]
PXI_Trig1 -      [x]            [x]            [x]
PXI_Trig2 -      [ ]            [ ]            [ ]
PXI_Trig3 -      [ ]            [ ]            [ ]
PXI_Trig4 -      [ ]            [ ]            [ ]
PXI_Trig5 -      [ ]            [ ]            [ ]
PXI_Trig6 -      [ ]            [ ]            [ ]
PXI_Trig7 -      [ ]            [ ]            [ ]
pse-pxi:~ #
```

Figure 4.3. Reserving PXI_Trig 1 in chassis 2 under Linux

Now that **PXI-Trig1** is reserved, suppose that we want to route it from segment 3 towards slot 1. At the same time, we will route **PXI_Trig3** from the middle segment outwards.

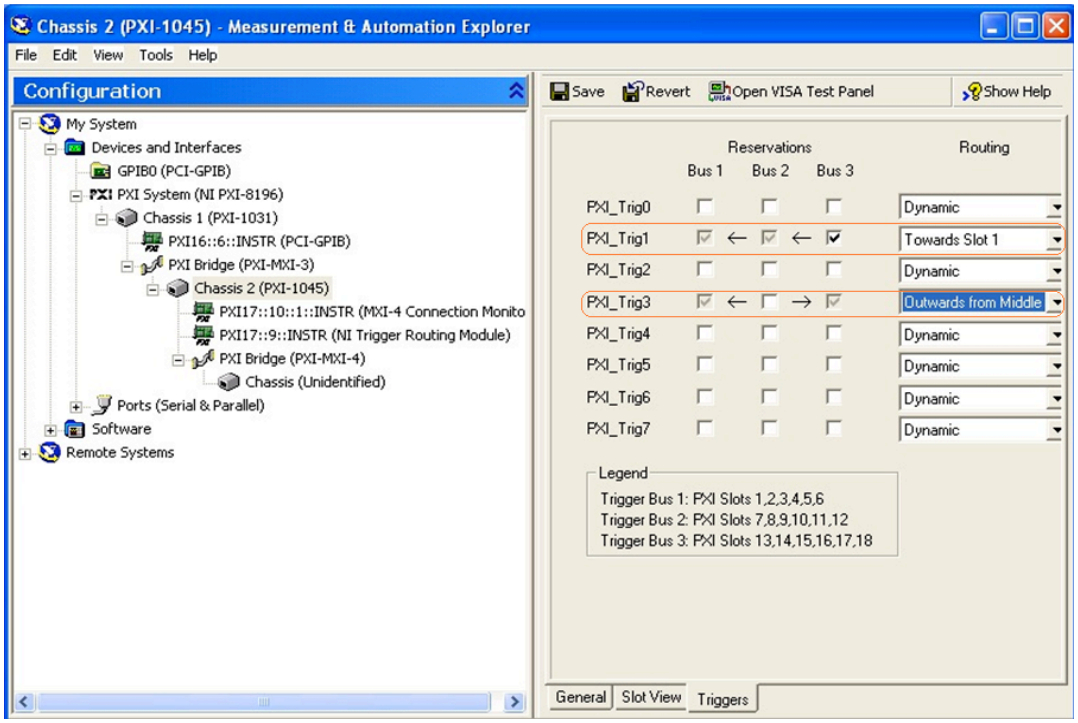

Figure 4.4. Route PXI Triggers using MAX under Windows

```
pse-pxi:~ # nipxiconfig --route-trigger 2 1 3
pse-pxi:~ # nipxiconfig --list-triggers 2

Chassis 2 (PXI-1045) PXI Trigger Configuration
            TriggerBus1    TriggerBus2    TriggerBus3
PXI_Trig0 -     [ ]            [ ]            [ ]
PXI_Trig1 -     [*]    <--     [*]    <--     [x]
PXI_Trig2 -     [ ]            [ ]            [ ]
PXI_Trig3 -     [ ]            [ ]            [ ]
PXI_Trig4 -     [ ]            [ ]            [ ]
PXI_Trig5 -     [ ]            [ ]            [ ]
PXI_Trig6 -     [ ]            [ ]            [ ]
PXI_Trig7 -     [ ]            [ ]            [ ]
pse-pxi:~ # nipxiconfig --route-trigger 2 3 2
pse-pxi:~ # nipxiconfig --list-triggers 2

Chassis 2 (PXI-1045) PXI Trigger Configuration
            TriggerBus1    TriggerBus2    TriggerBus3
PXI_Trig0 -     [ ]            [ ]            [ ]
PXI_Trig1 -     [*]    <--     [*]    <--     [x]
PXI_Trig2 -     [ ]            [ ]            [ ]
PXI_Trig3 -     [*]    <--     [ ]    -->     [*]
PXI_Trig4 -     [ ]            [ ]            [ ]
PXI_Trig5 -     [ ]            [ ]            [ ]
PXI_Trig6 -     [ ]            [ ]            [ ]
PXI_Trig7 -     [ ]            [ ]            [ ]
pse-pxi:~ #
```

Figure 4.5. Route PXI Triggers using nipxiconfig under Linux