NI Scan Engine Performance Benchmarks

Overview
This document provides a summary of NI Scan Engine performance benchmarks for CompactRIO hardware with the NI Scan Engine installed. These benchmarks pertain only to this specific feature and do not represent loop rates and processor loads of typical applications using the same number of channels or operating at the same loop rates. For details on the CompactRIO Scan Mode features, refer to the NI Scan Engine documentation in the LabVIEW Real-Time 8.6 help or the Scan Interface documentation in the NI-RIO 3.0 help.

Single-Point Benchmarks

The single-point benchmarks in this document are based on a simple PID control loop to provide an indication of Scan Mode performance for a closed loop system. The test setup and procedure is largely based on the methodology described in the “Benchmarking Single-Point Performance on National Instruments Real-Time Hardware” tutorial (see Benchmarking Single-Point Performance).

Hardware
NI 9012 Real-Time Controller
NI 9205 Analog Input Module
NI 9264 Analog Output Module

Software
LabVIEW 8.6 Real-Time Module
NI-RIO 3.0
NI-RIO IO Scan 1.0.0 (on the controller)

Test Application
Analog Input + PID + Analog Output (T2b) – see figure 1 below for example VI

Refer to Table 1 in the Benchmarking Single-Point Performance tutorial
Figure 1. Single-channel PID test application using the Synchronize to Scan Engine timing source to set the Timed Loop rate.

Fastest Loop Rate

Fastest loop rate represents the fastest loop the systems could attain without losing data.

<table>
<thead>
<tr>
<th>Number of Channels</th>
<th>Scan Mode (Hz)</th>
<th>FPGA Mode (Hz)</th>
<th>Compact FieldPoint (Hz)</th>
<th>Machine Control Architecture (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1000s</td>
<td>7100</td>
<td>605</td>
<td>435</td>
</tr>
<tr>
<td>16</td>
<td>1000</td>
<td>3500</td>
<td>313</td>
<td>not tested</td>
</tr>
<tr>
<td>80</td>
<td>399</td>
<td>974</td>
<td>n/a</td>
<td>not tested</td>
</tr>
</tbody>
</table>

Table 1. Fastest loop rate for T2 test with different HW/SW setup
The maximum NI Scan Engine rate supported on CompactRIO controllers is 1kHz.

Figure 2 illustrates how scan mode performance scales with the number of channels in the system.

Figure 2. Fastest Loop rate for Scan Mode for PID test setup

CPU Usage for Scan Mode

CPU usage is an important metric of system performance. Most real-world applications involve more complex logic and data processing than the simple PID control loop performed in these tests. Therefore, it is important to understand how much CPU time would be left for additional processing given various scan rates and channel counts. Scan Mode is designed to run periodically so depending on the rate, the actual CPU usage will vary. Table 2 shows CPU usage for the PID test setup at various rates.

<table>
<thead>
<tr>
<th>Number of Channels (AI/AO pair)</th>
<th>100 Hz</th>
<th>500 Hz</th>
<th>1000 Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10%</td>
<td>52%</td>
<td>55%</td>
</tr>
<tr>
<td>16</td>
<td>16%</td>
<td>47%</td>
<td>95%</td>
</tr>
<tr>
<td>32</td>
<td>18%</td>
<td>57%</td>
<td>n/a</td>
</tr>
<tr>
<td>64</td>
<td>27%</td>
<td>96%</td>
<td>n/a</td>
</tr>
</tbody>
</table>
Understanding NI Scan Engine Performance

There are two main areas of interest when analyzing the performance of a Scan Mode application: 1) the I/O Scan thread, and 2) the individual I/O Variable nodes on the block diagram.

I/O Scan Thread

Time taken by the I/O Scan thread varies depending on the I/O modules in use. Figure 3 compares the I/O scan execution times of a selected set of NI-RIO digital (NI-94xx) and analog (NI-92xx) I/O modules. All I/O modules, except NI-9264, used for this test support 32 I/O channels.

Figure 3 I/O Scan thread cost for different I/O module types (Above Time-Critical Priority)
Based on this data, we can make the following assumptions when trying to understand the I/O Scan time for a given hardware setup:

1. Digital modules (e.g. NI-94xx) consume less I/O Scan time than analog modules (e.g. NI-92xx) as they don’t require any scaling or calibration.

2. Input-only modules for a given data type (e.g. NI-9205) take less time than output modules of the same data type (e.g. NI-9264), as there is no processing needed to write output data.

3. Depending on the type and number of modules used, the total execution time of the I/O Scan thread generally ranges from 200 to 500 µs.

Note that figure 3 shows ~60 µs of overhead with no I/O modules in use. This represents the fixed I/O Scan thread overhead. If you are not using any of the Scan Mode features, we recommend that you uninstall Scan Mode software from the target to avoid this fixed cost.

Each I/O variable node allows you to enable or disable the timestamp feature. Enabling timestamps on any I/O variable under a module adds overhead to the I/O Scan time for that module. Figure 4 shows the difference in I/O Scan time for NI 9205 modules with and without timestamp enabled. The timestamp cost varies by module, but in general enabling timestamps can add 10-15% overhead to the I/O scan time of a given module. The timestamp feature is disabled by default.
I/O Variable Node

In addition to the execution time of the I/O Scan thread itself, each evaluation of an I/O variable node in a LabVIEW VI takes time to execute. Figure 5 shows the total execution time of Boolean and Double I/O variable nodes for both read and write operations. The trend for total I/O variable node access time is clearly linear.

![I/O Variable Node - Total Execution Time (Scanned Access Mode)](chart.png)

Figure 5. Total execution time of I/O Variable nodes

Figure 6 shows the average execution time of each I/O variable node for the same setup used to collect data for figure 5. As shown below, the execution time per node varies from 7 μs to 9 μs, with the average execution time around 8 μs. This data is an approximation of the average execution time per I/O variable node as the number of nodes on the block diagram increases. This data is based on averages collected over several thousand iterations and is adjusted to account for system jitter and benchmarking overhead. In general, 10 μs per I/O variable node is a reasonable estimate of execution time using scanned access mode.
Figure 6. Average execution time per I/O variable node increases slightly as the total number of variable nodes increases.

It is important to note that these benchmarks are taken with the I/O variable error clusters wired together. Due to details of the LabVIEW compiler, the I/O variable nodes execute two to three times faster when executed sequentially.

I/O variable nodes execute faster using scanned access mode than direct access mode. When you access an I/O variable using direct access mode, LabVIEW must traverse the software stack all the way down to the hardware driver, whereas scanned access simply involves accessing local memory. Use scanned access mode for synchronous access to a group of I/O channels. Use direct access mode to access an I/O channel asynchronously from the I/O scan. Although the execution time of direct access mode is slower than that of scanned access mode, you can take advantage of the asynchronous quality of direct access mode to access individual I/O channels at either a faster or a slower rate than the I/O scan. Figure 7 below compares scanned versus direct mode execution times for I/O variable nodes of the Double data type.
I/O Variable Network Publishing Overhead

I/O variables support network publishing for remote data access and monitoring. Figure 8 compares the system CPU usage for various network publishing rates.
The test setup used for the network publishing benchmarks included continuously changing analog input data from NI 9205 modules. The test used NI Distributed System Manager probes to monitor the input data on all the I/O modules and to record the total CPU usage for each scenario. The results reflect the total CPU usage with the I/O scan running at the default period of 10 ms and network publishing either disabled for all I/O variables or enabled at various network publishing rates.

Network publishing overhead is typically negligible if no remote client is connected to the target or if the I/O data is not changing quickly. But as figure 8 shows, the CPU usage can become significant when monitoring a large number of continuously changing analog I/O channels. Network publishing is enabled by default for I/O variables. However, you can disable network publishing for individual I/O variables using the Shared Variable Properties dialog box. You also can disable network publishing for multiple I/O variables at once using the Multiple Variable Editor. The "Publishing Disabled" series in figure 8 represents the baseline CPU usage of the system with only the I/O scan running and network publishing disabled for all I/O variables.

To optimize your system’s CPU usage, you can either slow down the network publishing rate by adjusting the Network Publishing Period on the Scan Engine page of the RT Target Properties dialog box, disabling network publishing for I/O variables that you are not monitoring remotely, or making sure you don’t have a remote client connected to any I/O module unless it is using the I/O data for remote access or monitoring.
Estimating NI Scan Engine Cost For Your System

As noted in the previous sections, I/O Scan thread overhead depends on the number and type of I/O modules used in your system. For most systems, the I/O Scan thread time will range from 200-500 µs.

An I/O variable node itself can take about 10 µs of execution time. The total cost of I/O variable nodes depends on the number of nodes present in your application and the rates of the loops containing the nodes.

You can use the Real-Time Execution Trace Toolkit to estimate the total execution time of the I/O Scan thread for your system. See figure 8 for an example trace session showing the I/O Scan running at Above Time Critical priority. The data shown in figure 3 and figure 4 is an average of several iterations of the data collected from similar traces and is adjusted to account for system jitter and benchmarking overhead.

![Real-Time Execution Trace Toolkit](image)

Figure 9. I/O Scan Execution trace for a system with eight NI-9205 modules

Related Resources

White Paper: Using NI CompactRIO Scan Mode with NI LabVIEW Software