Bergmans Mechatronics LLC

LabSocket-E

User Guide

LabSocket-E v1.0

Jan 2017

BML Document BML-2017-100.3

Table of Contents

1. Introduction

LabSocket enables remote access to LabVIEW applications from desktop or mobile web browsers without the need for browser plugins or a client-side run time engine. The system automatically creates a web page representation of a LabVIEW application ("Target VI") front panel and dynamically synchronizes the contents of the web page and front panel.

A key technology used in the system is the HTML5 WebSocket interface that enables continuous, bi-directional communication with a web browser. The use of this interface contributes to the name of the system: "LabSocket = LabVIEW + WebSocket".

LabSocket-Embedded, or "LabSocket-E", is a version of LabSocket that is designed for remote access to application operating on National Instruments Real-Time (RT) platforms such as myRIO, sbRIO, FlexRIO, CompactRIO, CompactDAQ and legacy PXI devices. LabSocket-E has received Compatible with LabVIEW certification by National Instruments.

This User Guide describes the setup and operation of the LabSocket-E software. The guide assumes that the user is familiar with LabVIEW. Users of cRIO-903x devices are assumed to have a basic level of Linux experience.

1.1 Downloading and Activating the LabSocket-E VIPM File

LabSocket-E is distributed in the form of a VI Package Manager file that is available for download via VI Package Manager or over the web. Download details and information on activation of the software is provided in Section 4.1.

1.2 Licensing, Executables and Real-Time Deployments

The LabSocket-E licensing agreement allows the software to be installed on a single developer platform. From this single development platform, developers may create an unlimited number of Real-Time applications that incorporate the LabSocket-E software and may deploy these applications to an unlimited number of real-time devices.

Full licensing terms can be found under "**Help > LabSocket-E License...**" after LabSocket-E installation.

1.3 Direct Connection Between Browser and RT Platform

In contrast to the desktop version of LabSocket, LabSocket-E does not require the use of a LabSocket Server Virtual Machine. Instead, LabSocket-E is able to

establish a direct WebSocket connection between one or more browsers and an RT platform. This change simplifies the setup and deployment of LabSocket-E applications.

1.4 Client Mapping

LabSocket-E is currently configured for "Basic" client mapping mode. In this mode, one or more browsers connect to the same instance of a Target VI. The same data is displayed in each browser and each browser is able to control the Target VI.

The desktop version of LabSocket supports MultiClient mapping mode. In this mode, each browser that connects to the system is mapped to a unique instance of the Target VI. This mapping mode may be added to LabSocket-E at a later date.

1.5 Acknowledgement

Thanks to MediaMongrels Ltd for developing the "WebSockets" software package for LabVIEW. This package is employed by LabSocket-E to establish direct WebSocket connections with web browsers. It is automatically installed during the LabSocket-E installation process.

1.6 Contact

For assistance with system setup or questions about the system, please contact:

John Bergmans, Bergmans Mechatronics LLC
e-mail: jbergmans@bergmans.com
Phone: 1-714-474-8956 (8 AM to 6 PM US Pacific Time)
Skype: johnbergmans

2. System Requirements

2.1 Browsers

Browsers that use the LabSocket system must be compatible with the HTML5 WebSocket interface standard. Most modern mobile and desktop browsers are compatible with this standard. A complete list of WebSocket-compatible browsers is shown at: http://caniuse.com/websockets

No browser plug-ins or client-side run-time engine are required.

2.2 LabVIEW Development Platform

LabVIEW	LabVIEW 2014 32-bit Full Development System or later. Support for earlier versions of LabVIEW may be available upon request.
Minimum Hardware	2.3 GHz CPU and 4 GB memory
Operating System	Windows 7 or later

2.3 Real-Time Platform

LabSocket-E is designed for use on any National Instruments RT platform, including myRIO, sbRIO, FlexRIO, CompactRIO, CompactDAQ and legacy PXI devices.

As of this writing, LabSocket-E has been successfully operated on the following platforms.

- cRIO-9012
- cRIO-9030
- cRIO-9063
- cRIO-9074
- cRIO-9081
- myRIO-1900
- sbRIO-9606
- sbRIO-9651

2.4 Ports Used by LabSocket-E

Ports 80 and 61614 on the RT platform must be accessible to the browser.

3. Theory of Operation

LabSocket-E operates in two modes (Figure 3.1). In “Deployment" mode, the developer uses the ***Client Code Generator*** tool on the development PC to automatically create HTML and JavaScript (JS) code representing the Target VI front panel and a binary configuration file. The Generator is then used to deploy the HTML and JS code, other browser support files and the configuration file to the Real-Time (RT) platform.

The “LabSocket-E Start” VI, which invokes the LabSocket ***Synchronizer*** module, is added to the block diagram of the Target VI. The developer deploys the Target VI to the RT platform using LabVIEW Project

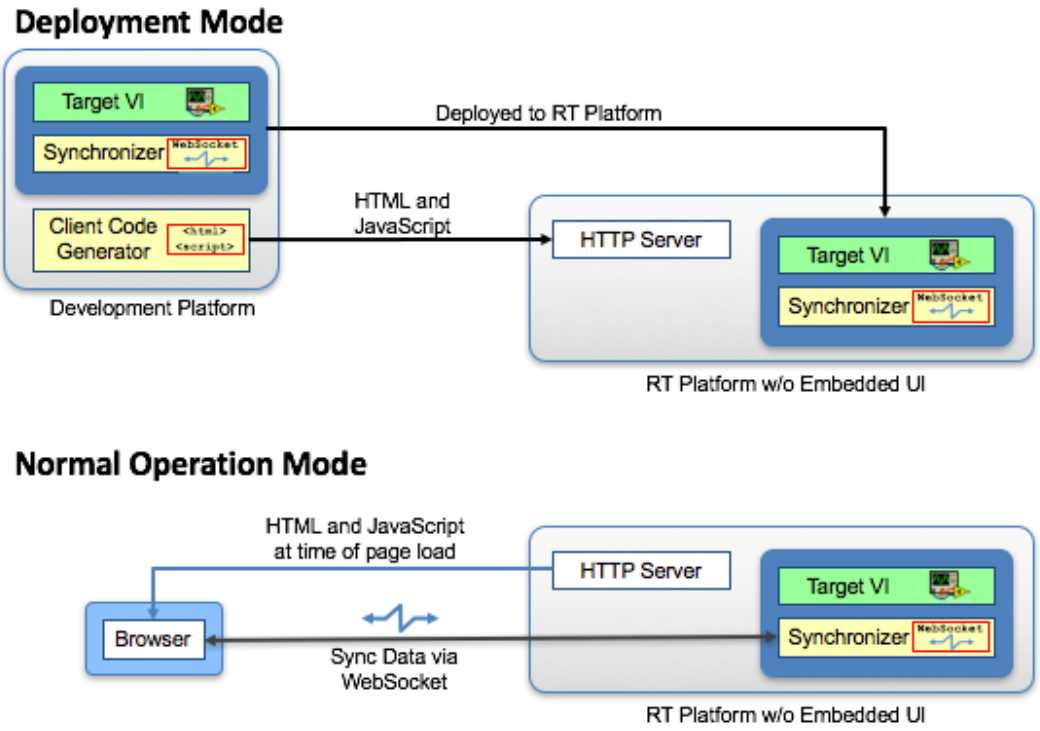


Figure 3.1. LabSocket-E System Overview

In “Normal Operation” mode, a remote browser loads the HTML and JS code from the HTTP server of the RT platform. The JS code establishes a direct WebSocket connection to the Synchronizer module on the RT platform. The Synchronizer then ensures that the values of the controls and indicators in the front panel and the browser match at all times.

If more than one browser connects to the RT platform, each browser displays identical content and each browser is able to control the Target VI.

The core of the Synchronizer contains a Primary Processing Loop that reads the Target VI front panel values and sends updates to the browser(s). This loop also updates the Target VI front panel values based on data received from the browser(s). Section 8.4 provides details on how to optionally adjust the timing of and monitor the Primary Processing Loop.

4. Setup Instructions

This section provides step-by-step instructions on the installation of the LabSocket-E software on the developer's PC as well as minor modifications that may be required to the RT platform.

4.1 LabSocket-E Software Installation

1. Start VI Package Manager on the LabVIEW Host Platform.
2. Install the LabSocket-E package version 1.0.x.x. Also install the dependencies that are automatically selected by this package.

   Alternately, the LabSocket-E VIPM file (labsocket_e-1.0.x.x.vip) may be downloaded from one of the following URLs and installed using VI Package Manager. Install the dependencies that are automatically selected by LabSocket-E.
   a. LabVIEW Tools Network
      http://www.ni.com/gate/gb/GB_EVALTLKTLABSOCKET/US
   b. labsocket.com website
      http://labsocket.com/download.html#LabSocketEEvaluationResources

3. Exit LabVIEW if it currently running.
4. Restart LabVIEW.

LabSocket-E may now be used with real-time platforms in Interactive mode for up to 30 days. To enable the capability to operate in Headless mode for up to 7 days, activate LabSocket-E using the evaluation credentials:

License ID: 3353857
Password: demo

LabSocket-E will stop functioning in evaluation mode after either i) 30 days following installation or ii) 7 days following the trial activation.

For indefinite operation of LabSocket-E, please purchase a license through either:

- the LabSocket Purchase Page (http://labsocket.com/purchase.html), or,
- the National Instruments' LabVIEW Tools Network (http://ni.com/labviewtools/labsocket)

Upon receipt of an order, activation credentials that enable indefinite operation of the software will be e-mailed to the customer.

Note

cRIO-903x Users - With the installation of an additional library, LabSocket-E is capable of dynamic synchronization of picture elements and high resolution replication of any indicator element. See Appendix A for instructions on installation of this library.

4.2 WebDAV Server Installation (VxWorks and PharLap Only)

LabSocket requires that WebDAV server be installed on the real-time platform. This software is installed by default on NI Linux Real-Time platforms.

For VxWorks and PharLap platforms this software may be installed as follows:

1. Launch Measurement and Automation Explorer.
2. Expand Remote Systems and select the RT platform on which to install WebDAV.
3. Right-click on Software and select Add/Remove Software.
4. Use the Wizard to install WebDAV Server on the RT platform.

4.3 Real-Time Platform Configuration

It is recommended that the LabSocket-E browser client code be deployed to a sub-directory within the root directory of the RT platform HTTP server. The use of such a sub-directory ensures that the LabSocket-E browser client code will not interfere with the existing "System Configuration" web page located in the HTTP root directory.

NI Linux Real-Time Platforms

A terminal program to log into the RT platform to perform this modification is required. For Windows users, the free terminal program PuTTY is recommended. It may be downloaded from: http://www.putty.org/.

Note that before logging in to the RT platform, the Secure Shell Server of the RT platform must be enabled. Use Measurement & Automation Explorer (MAX) to enable the SSH server if it is not already enabled.

1. Log in as user "admin" to the RT platform using an ssh client such as PuTTY.
2. Navigate to the root directory of the HTTP server

```
cd /var/local/natinst/www
```

3. Create a new directory named “labsocket”:

```
mkdir labsocket
```

4. Change the group of the labsocket directory to “ni”:

```
chown admin:ni labsocket
```

5. Change the group permissions of the labsocket directory:

```
chmod g+w labsocket
```

6. Use the command “`ls -l`” to view the contents of the HTTP root directory. The permissions, owner and group settings for the labsocket directory should appear similar to the following:

```
drwxrwxr-x    2 admin    ni            4096 Dec  3 17:52 labsocket
```

7. Logout out of the RT platform

Phar Lap and VxWorks Platforms

Use the System Configuration page hosted by the RT platform to create the sub-directory.

1. Point a web browser to the IP address of the RT platform. The System Configuration page will appear in the browser.

2. Log in using the “admin” account. (Note that to use the Remote File Browser in the following steps, the admin account password must not be blank.)

3. Phar Lap platforms – Using the Remote File Browser,

    a. Navigate to the directory `c:\ni-rt\system\www`
    b. Create the directory `labsocket`

4. VxWorks platforms – Using the Remote File Browser,

    a. Navigate to the directory `/ni-rt/system/www`
    b. Create the directory `labsocket`

5. Log out of the System Configuration page and close the browser window.

5. Front Panel Elements

5.1 Supported Front Panel Elements

LabSocket-E supports the synchronization of the values of a subset of the available LabVIEW front panel elements. A list of supported elements is shown in Table 5.1. Each of these elements is demonstrated in the example VI "LSE Demo.vi" that is included with LabSocket-E (Section 7).

Table 5.1. Front Panel Elements Supported by LabSocket-E

a) All Real-Time Platforms

Element	Notes
String controls and indicators	Includes support for password-style text
Numeric controls and indicators	
Boolean controls and indicators	
Boolean controls and indicators with images	See Section 8.2
Picture ring controls and indicators	See Section 8.2
Enum controls and indicators	
Slider controls and indicators	
Waveform graphs	See Section 5.2
XY graphs	See Section 5.2
Tabs	
Text decorations	
Multicolumn Listbox controls and indicators	See Section 5.2
Static picture elements	See Section 8.3
#LS_no_sync preprocessor tag	Disables synchronization of an element. See Section 8.1
#LS_no_display preprocessor tag	Prevents display in browser of an element. See Section 8.1

b) Additional Elements Supported on cRIO-903x with Embedded UI On

Element	Notes
Dynamic picture elements	
#LS_image preprocessor tag	Enables high-resolution replication of any element. See Section 8.1 and Appendix A

Unsupported elements on the front panel of the Target VI may be present on the Target VI front panel and are ignored by LabSocket-E.

5.2 Front Panel Element Configuration Details

Unique Label Names

Each control and indicator on the Target VI front panel must have a unique label name. The Client Code Generator (Section 6.2) will create an error message and not output the browser client code if any elements with the same label name are found.

Boolean Control Mechanical Action

All Boolean controls must use one of the "Switch When..." mechanical action options. The Client Code Generator will create an error message and not output the browser client code if a Boolean control with a "Latch When..." mechanical action is found.

Multicolumn Listbox Data Type

Each Multicolumn Listbox (MCLB) control and indicator must have a Scalar data type. The Client Code Generator will create an error message and not output the browser client code if a MCLB with Array data type is found.

Plot Element Data Format

LabSocket-E requires that the input data for XY Graphs and Waveform Graphs is of the form shown in Table 5.2. Support for XY Graphs and Waveform Graphs is demonstrated in the example VI "LSE Demo.vi".

Table 5.2. Data Format for XY Graphs and Waveform Graphs

Element	Format
XY Graphs	Single cluster or 1D array of clusters containing, in this order: - X data (1D array of doubles) - Y data (1D array of doubles)
Waveform Graphs	Cluster containing, in this order - Initial time in seconds (double) - Time interval in seconds (double) - Plot data (1D or 2D array of doubles)

6. LabSocket-E Application Development

This section describes the process of developing real-time applications for Target VIs that use LabSocket-E. The steps described here assume that LabSocket-E installation and any necessary modifications to the RT platform are complete. These steps also assume that a LabVIEW Project has already been created and that the Target VI has been added to the Real-Time device in the project.

The example Target VI used in this section is the VI "LSE Demo.vi". This VI is included in the LabSocket-E software package and demonstrates the controls and indicators supported by LabSocket-E. Further details about this VI are provided in Section 7.

6.1 Target VI Modifications

To invoke the LabSocket-E Synchronizer on the RT platform, include the "LabSocket-E Start.vi" on the block diagram of the Target VI. After installation of LabSocket-E on the developer platform, this VI is available on the Tools Palette under **Addons > LabSocket-E** (Figure 6.1).

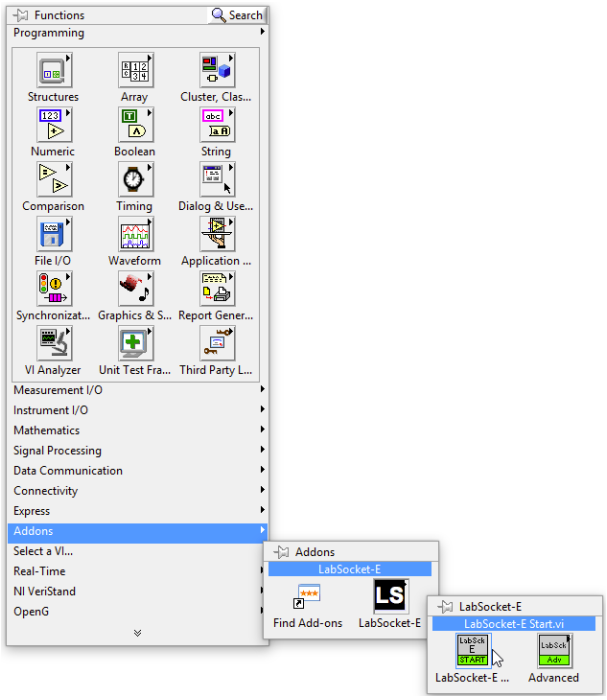


Figure 6.1. Selecting LabSocket-E Start.vi *from Functions Palette*

It is recommended that the LabSocket-E Start.vi be placed on the block diagram of the Target VI such that it is among the first, if not the first, code to execute

when the Target VI begins running. No input parameters are required for LabSocket-E.vi.

A portion of the block diagram of “LSE Demo.vi” containing the LabSocket-E Start.vi is shown in Figure 6.2

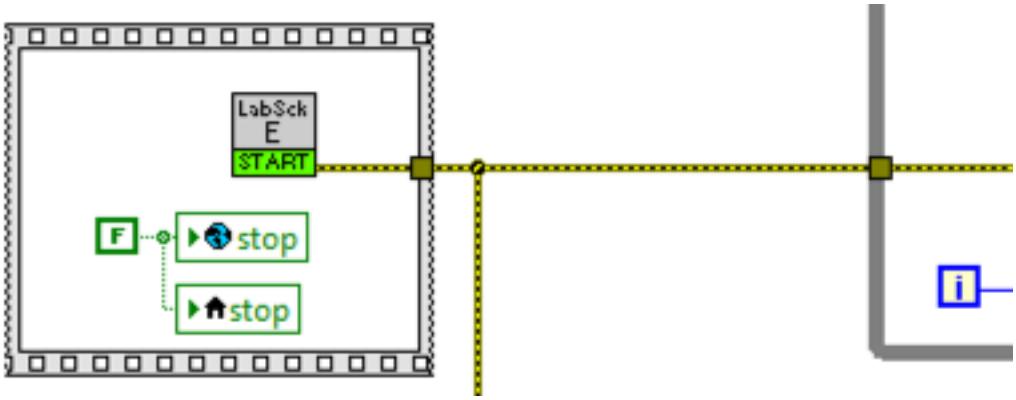


Figure 6.2. LabSocket-E Start.vi *on block diagram of “LSE Demo.vi”*

6.2 Generate and Deploy Browser Client

The LabSocket-E Client Code Generator performs two functions. First it generates the browser client and second it deploys the code to the real-time platform. Details on the use of the Client Code Generator to perform these functions are described here.

Part 1 Generate Browser Client

1. Select **Tools>LabSocket-E>LabSocket-E Client Code Generator…**. The LabSocket-E Client Code Generator window (Figure 6.3) will open.

2. In the "Local Platform" section:
    1. Select the Target VI that will be used to as for generating the browser client code. This VI will also later be deployed to and run on the RT platform.
    2. Identify the "Client Code Destination Directory" on the developer platform. This is the local destination for the browser client code.

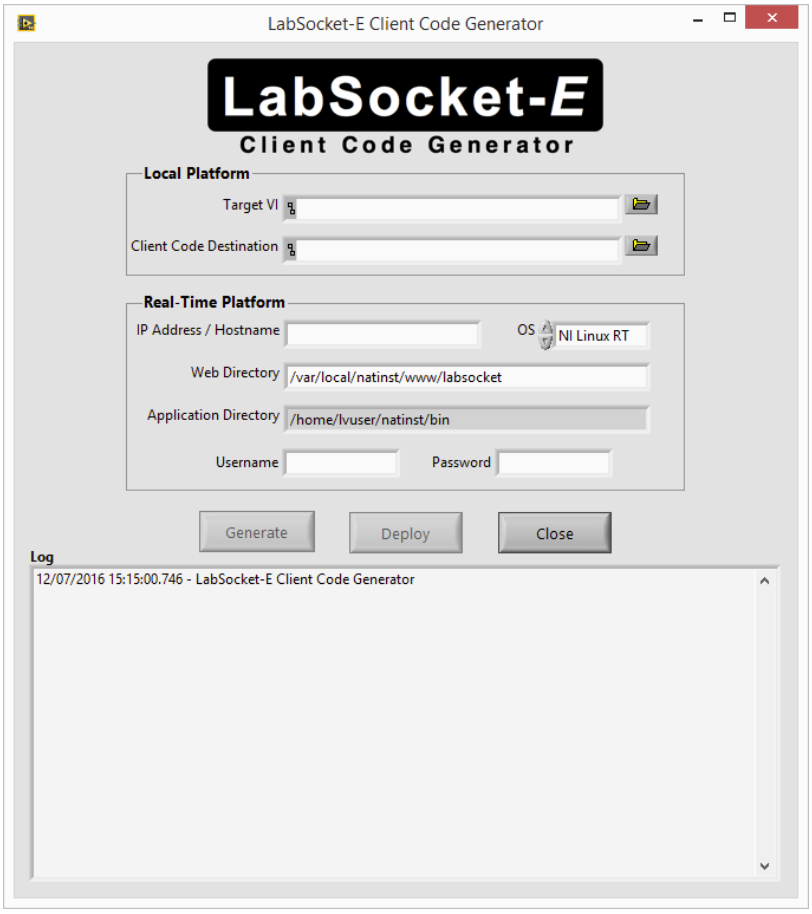


Figure 6.3. LabSocket-E Client Code Generator

Notes

1) The maximum allowable length of the path to Client Code Destination directory is 162 characters.

2) Other than the above path-length restriction, the Client Code Destination directory can be located anywhere on the development PC. A recommended location is a directory on the Desktop.

3) When the Client Code Generator generates the browse client code it will create the following directories and files in the Client Code Destination directory:
    a. "labsocket_config" – a directory containing a binary configuration file for the Target VI
    b. "labsocket_www" – a directory containing the HTML file and its companion JavaScript file for the browser client as well as other support files and directories

4) The Client Code Generator will present a warning to the operator during the automatic code generation process (Step 5, below) if any elements with identical label names are found.

3. In the "Real-Time Platform" section of the Client Code Generator:
    1. Enter the IP address or hostname of the Real-Time platform

    2. Select the Operating System for the Real-Time platform. The OS value automatically sets the Web Directory field. This field represents the destinations on the RT platform for the browser client code. This value may be modified but the default value is strongly recommended,

       The OS value also automatically sets the Application Directory field on the RT platform. This directory is the default location of LabVIEW for real-time applications. This value may not be modified in the current version of the Client Code Generator. Please contact Bergmans Mechatronics for an alternate deployment solution if a non-default Application Directory is required.

    3. Enter the Username and Password credentials for the RT platform

4. Press “Generate” to automatically generate the browser client code. When browser client code has been generated, the URL for the Target VI will be displayed in the Log Window (Figure 6.4).

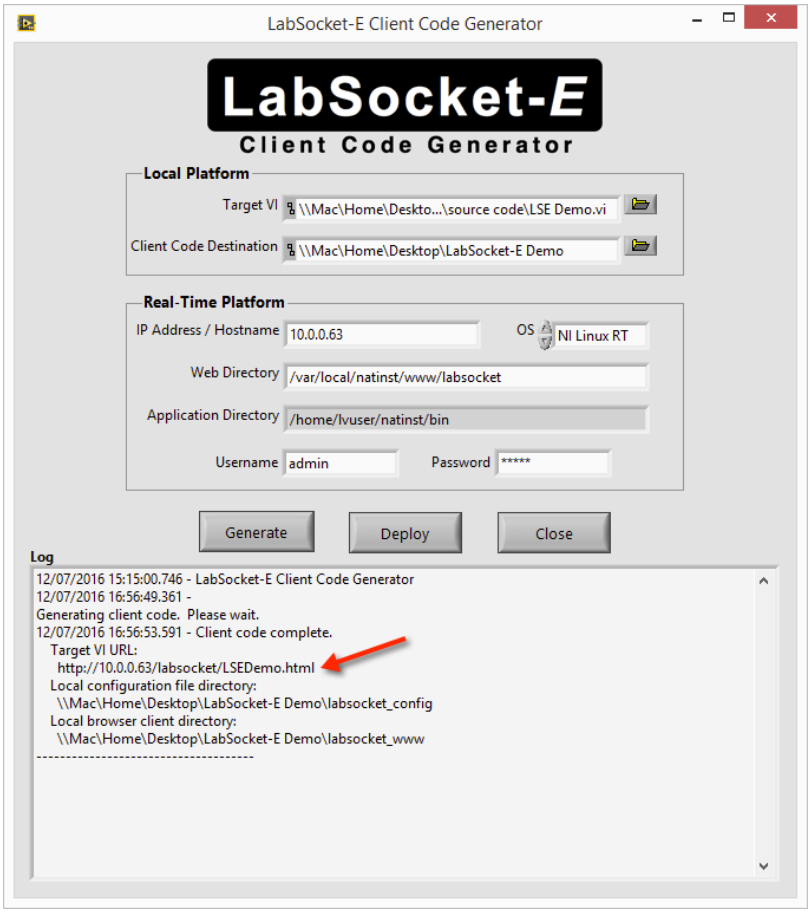


Figure 6.4. LabSocket-E Client Code Generator after pressing Generate button

Typical contents of the “labsocket_www” directory within the Client Code Destination directory following the automatic generation of the browser client code are shown in Figure 6.5. These contents are shown for informational purposes only. Normally, a developer does not need to view or edit the items in this directory.

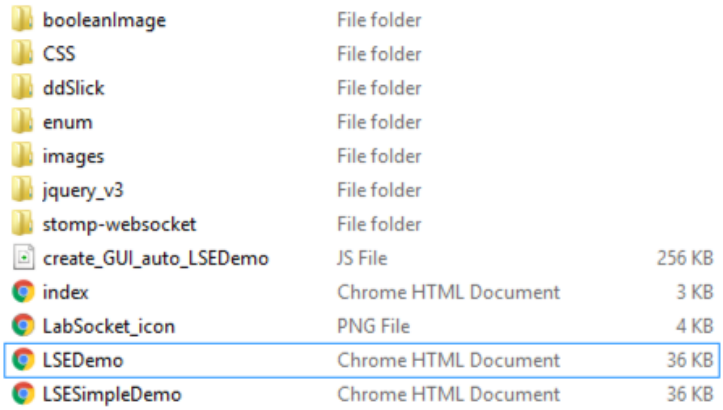


Figure 6.5. Typical contents of “labsocket_www” directory

Part 2 Deploy Browser Client

1. Press the "Deploy" button on the Client Code Generator to transfer the browser client code to the RT platform. The list of files transferred and other information about the deployment is shown in the Log window (Figures 6.6 and 6.7). A description of the files transferred to the RT platform is included in the Note below.

2. The Client Code Generator may now be closed by pressing the "Close" button, or it may remain open.

Note

On the initial client code deployment to an RT platform, the following group of files is sent to the RT platforms:

1. a binary configuration file named *TargetVIName*.cfg, where *TargetVIName* is based on the name of the Target VI
2. a file named *TargetVIName*.html. This is the main HTML file for the browser client. The name of this file is included in the URL identified by the Client Code Generator (See Section 6.2, Part 1, Step 4)
3. a companion JavaScript file for item 2, named create_GUI_auto_*TargetVIName*.js
4. a set of JavaScript library files
5. a file named "LabSocket_icon.png" that is used as the "favicon" image in the web browser tab
6. a file named "index.html". This is a test web page that can be used for diagnostic purposes of the RT platform HTTP server. If the HTTP server is functioning normally, this page will be available at http://*XX.XX.XX.XX*/labsocket/index.html, where *XX.XX.XX.XX* is the IP address of the RT platform.

On each subsequent deployment to the RT platform, Items 1-3 are sent to the RT platform. Any files among items 4-6 that are missing on the RT platform are also sent on subsequent deployments.

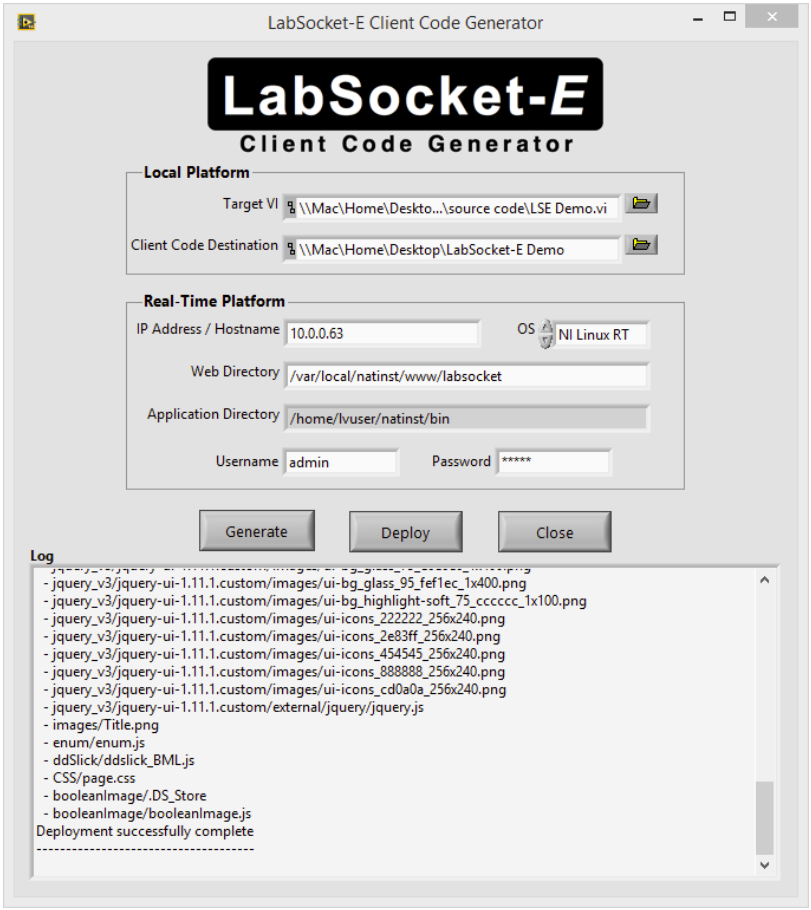


Figure 6.6. Client Code Generator window showing typical Log window display on initial deployment to an RT platform

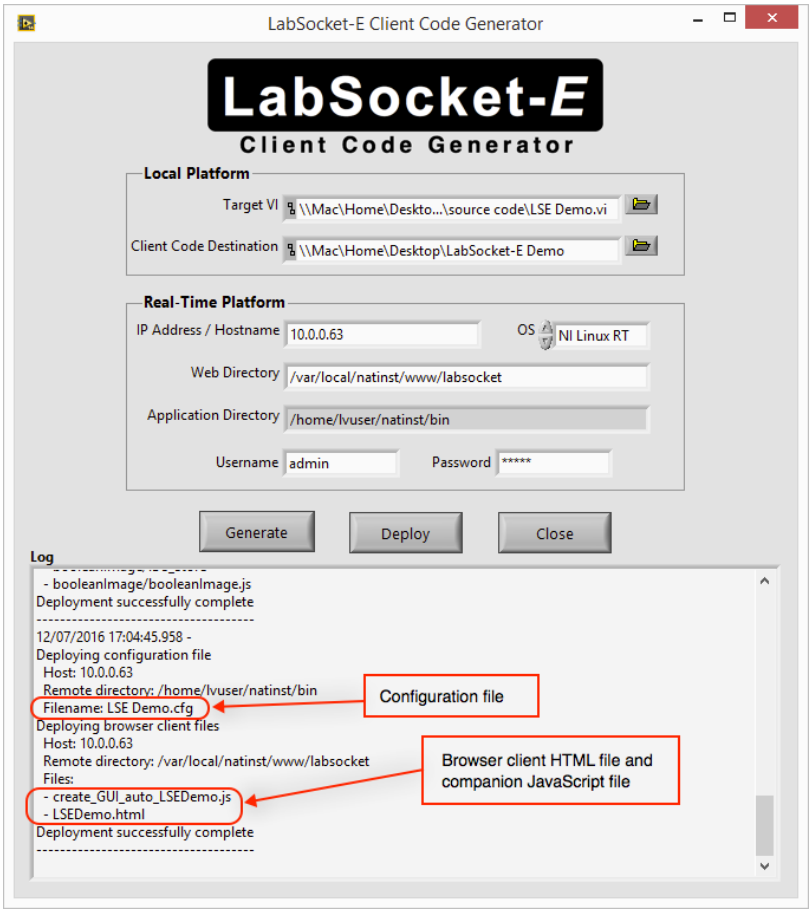


Figure 6.7. Client Code Generator window showing typical Log window display after a deployment to an RT platform subsequent to the initial deployment

6.3 Deploy the Real-Time Application

After the browser client has been deployed to the RT platform, the real-time application can be built and deployed to the RT platform as usual using LabVIEW Project. The application may then be run in either interactive or headless mode.

To access the application using the browser client, point a browser to the URL specified by the Client Code Generator (Section 6.2, Part 1, Step 4)

7. Example VI

The LabSocket-E VIPM package includes an example VI named "LSE Demo.vi" that demonstrates the capabilities the system. The project may be accessed by selecting **Help > Find Examples...** from the main LabVIEW window. Select the **Directory Structure** radio button. In the directory structure, double-click on the **LabSocket** directory, to locate the VI.

"LSE Demo.vi" (Figure 7.1) demonstrates support for:

- common controls and indicators supported by LabSocket-E, including string, numeric values, enums, picture rings, and Booleans with images;
- the #LS_no_display preprocessor tag that prevents a front panel element from being replicated in the browser (Section 8.1);
- XY Graphs, Waveform Graphs and MultiColumn ListBoxes;
- String element background color control; and,
- Dynamic picture element synchronization for cRIO-903x platforms (See Appendix A)
- LabSocket-E performance monitoring using the "LabSocket-E Monitor.vi" (Section 8.4)

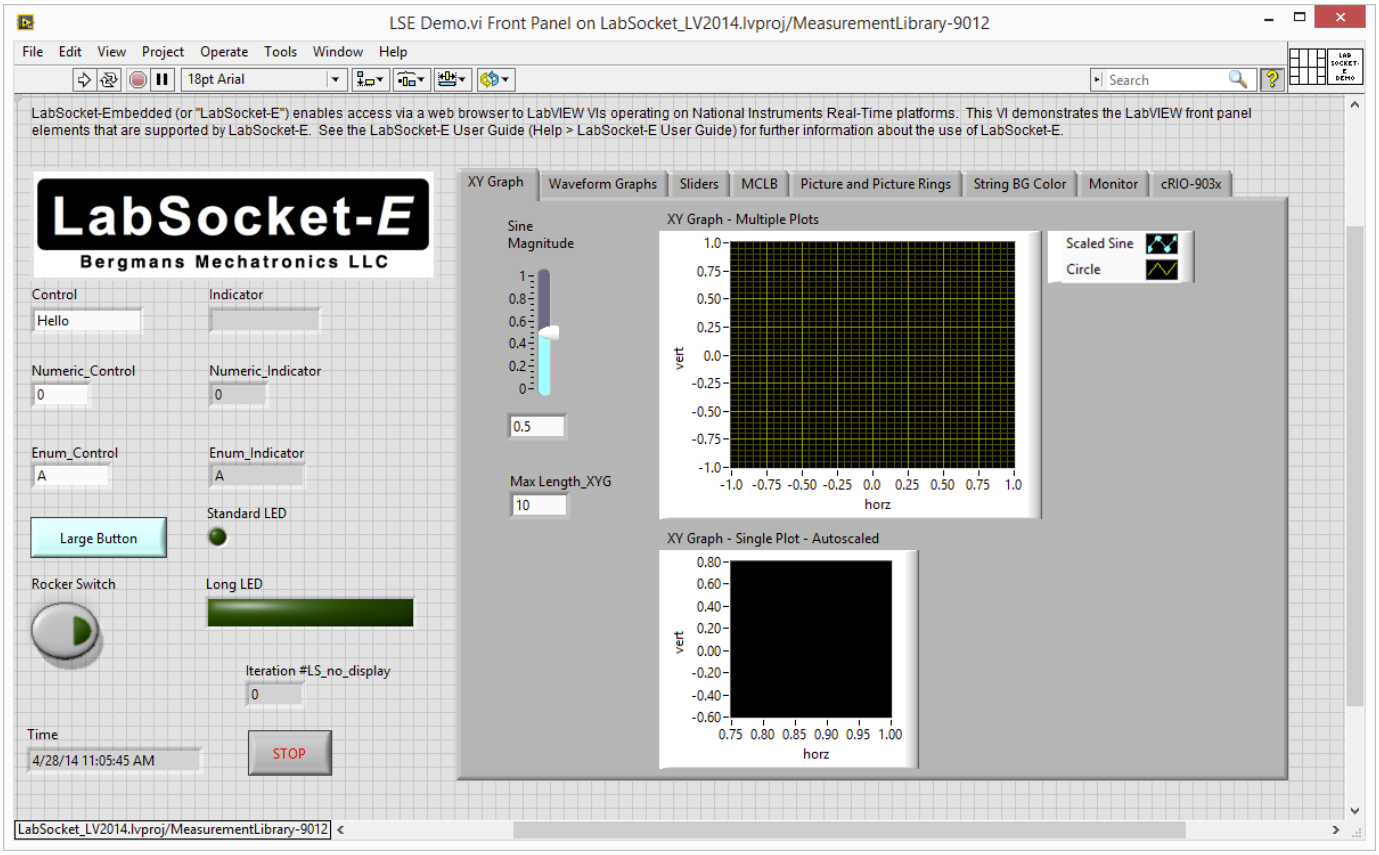


Figure 7.1. "LSE Demo.vi" *Front Panel*

8. Advanced Features

This section provides details on the following advanced features of LabSocket-E: i) preprocessor tags that allow fine tuning of the browser client; ii) picture ring support and high-fidelity replication of boolean elements; iii) insertion of custom JavaScript code into the browser client; and, iv) replication of PNG images in the browser.

Also included in this section are details on the Advanced VIs that enable the developer to tune and monitor the performance of LabSocket-E.

8.1 Preprocessor Tags

Preprocessor tags in the caption or label text of front panel elements may be used to specify how individual elements are to be replicated in the browser. These tags are summarized in Table 8.1.

Table 8.1. Preprocessor Tags

Tag	Description	Example Application
#LS_no_display	Do not replicate the target element in the browser	Prevent display of a Target VI stop button
#LS_no_sync	Prevents synchronization of the front panel element with its browser representation	Prevent tab element synchronization in an aoplication with mutiple simultaneous users
#LS_image(*compression,decimation)*	(cRIO-903x only) Display the target element as an image in the browser. Updated image is only sent to browser if image change detected. Optional arguments are: i) *compression* - image compression (0=no image compression, 9=maximum image compression); and, ii) *decimation* - number of iterations of main LabSocket processing loop to skip before checking for image update. NOTE - Only functional on cRIO-903x platforms with embedded UI enabled and OpenG ZIP LIbrary v4.1.0-b2 (or later) installed. See Appendix A.	Display an exact replica of a Graph element (eg. intensity or 3D plots)

The effect of the #LS_no_display tag is shown in the screenshot in Figure 8.1.

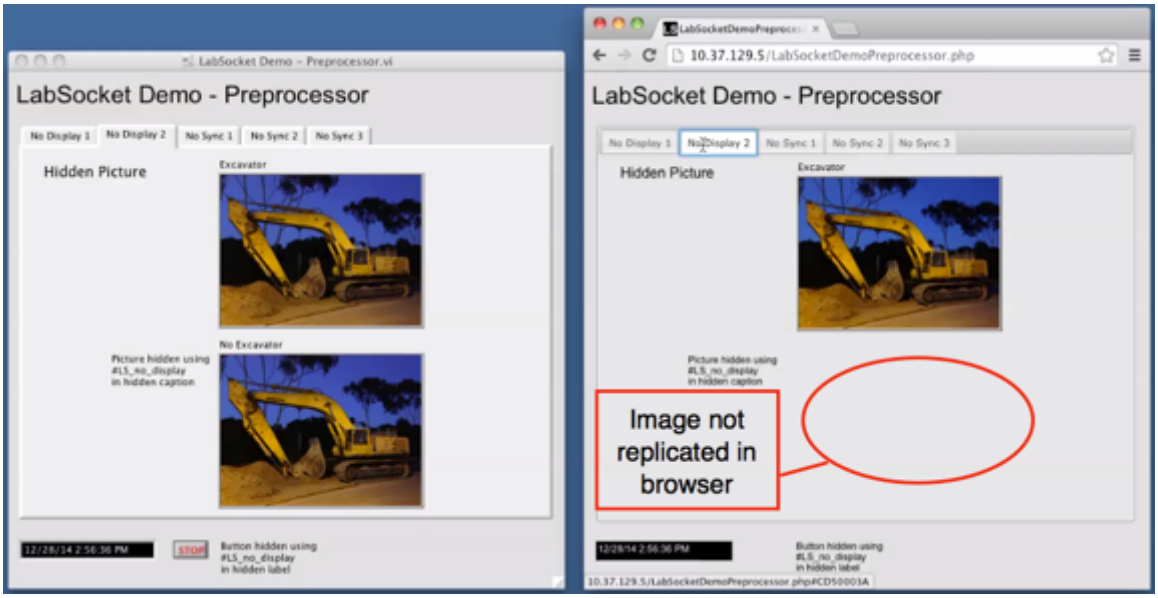


Figure 8.1. Screenshot demonstrating the use of "#LS_no_display" preprocessor tag to prevent replication of a picture (L- LabVIEW VI, R – Browser)

8.2 Picture Rings and High-Fidelity Boolean Elements

LabSocket supports replication of Picture Ring controls and indicators. This replication only occurs if the element is a type def, otherwise the element is not replicated. Figure 8.2 illustrates Picture Ring control and indicator replication.

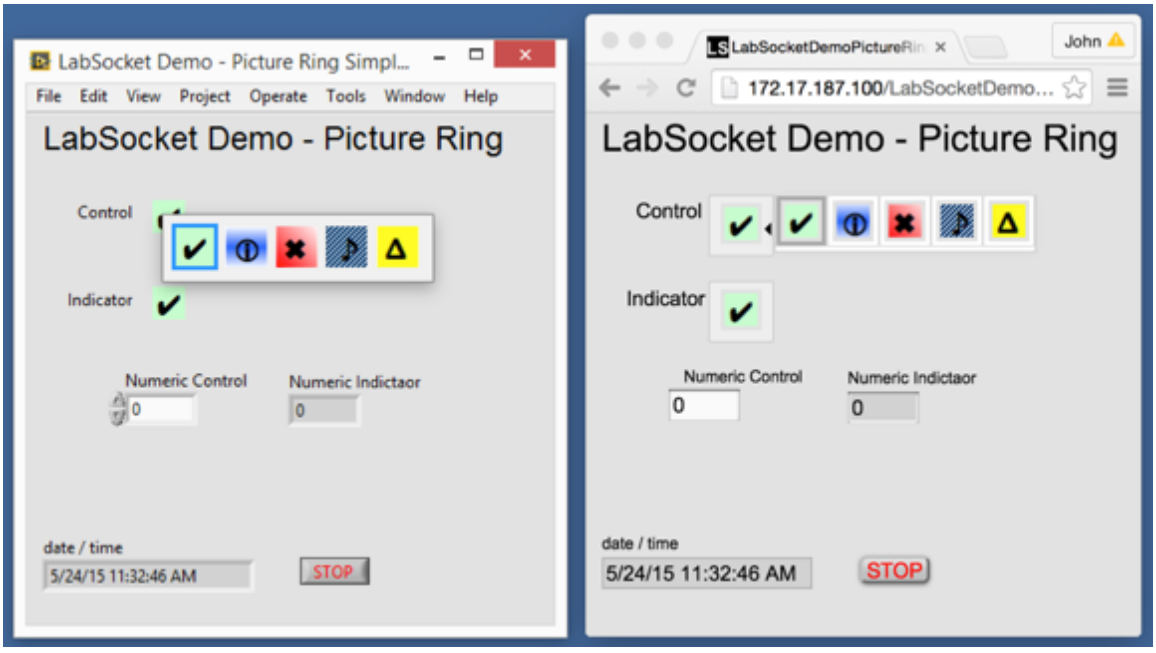


Figure 8.2. Screenshot demonstrating the replication of a picture ring control in the browser (L- LabVIEW VI, R – Browser)

LabSocket also supports high-fidelity replication of Boolean controls and element if the element is a type def (Figure 8.3).

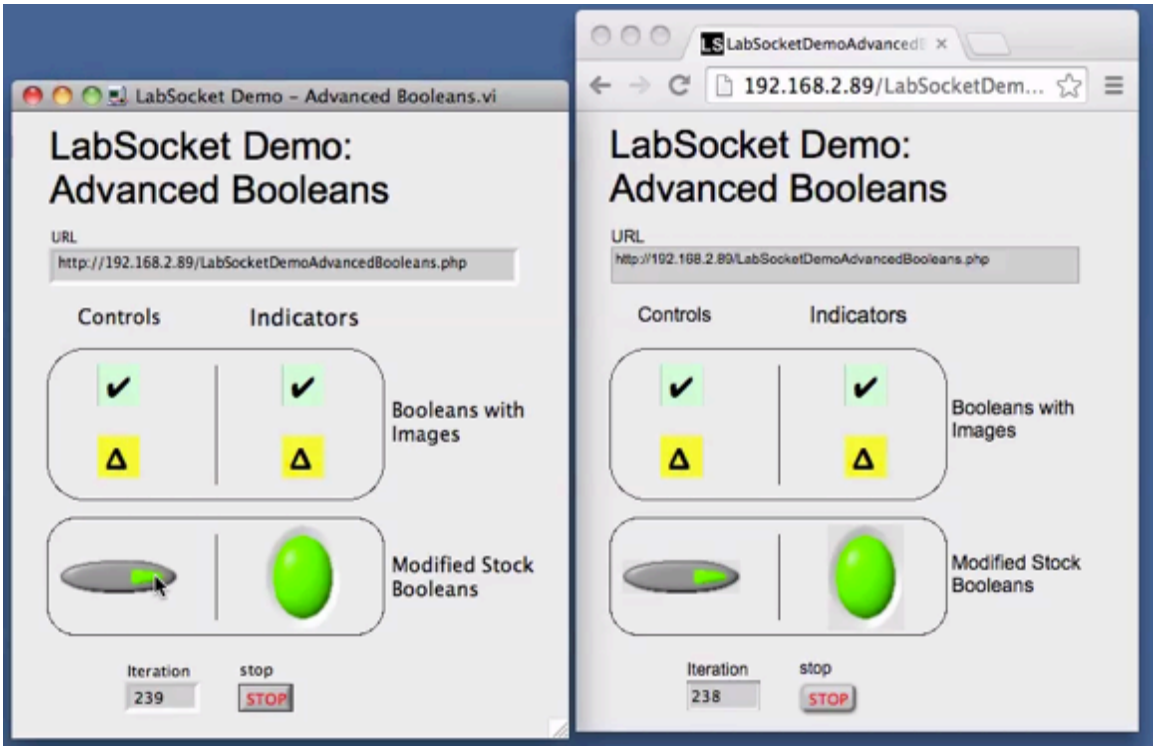


Figure 8.3. Screenshot demonstrating high-fidelity representation of Boolean elements (L- LabVIEW VI, R – Browser)

8.3 Working with Images

A "PNG to Picture Converter" tool is provided with the LabSocket system to enable images contained in PNG files to be easily rendered in the browser. The process for working with images is as follows.

1. From the menu bar, select **Tools>LabSocket-E>PNG to Picture Converter…**

2. Select the PNG file of interest.

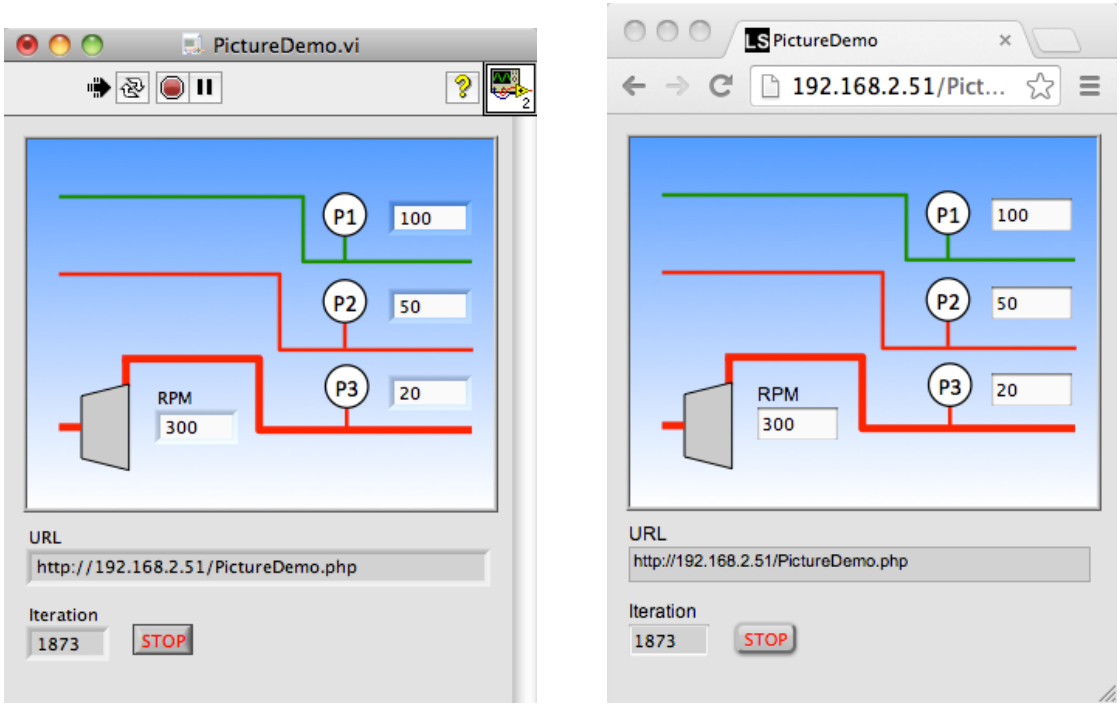
3. A new, unnamed VI will appear. Copy the picture control from this VI to the front panel of the Target VI.

4. Right-click on the picture. Select **Data Operations>Make Current Value Default**

5. Save the Target VI

6. Run the Target VI (assuming the ***LabSocket-E Start.vi*** and other components necessary for a practical application are already present on the VI block diagram).

A simple Target VI using a picture element and its representation in a browser is shown in Figure 8.4.



a) Front Panel *b) Browser Representation*

Figure 8.4. Rendering of Picture Elements in Browser

Controls and indicators may be superimposed in front of the picture element. Note that elements in the browser are displayed from back to front in order of increasing tabbing order value (ie tabbing order value 0 is at the back of the browser representation, with other elements displayed in front of it). Similarly, for elements in tab pages, elements are displayed from front to back in order of increasing tab "Control Order".

8.4 Advanced VIs

LabSocket-E provides several VIs for optional customization and monitoring of the system performance. These VIs is available on the tools palette under **Addons > LabSocket-E > Advanced.**

LabSocket-E Init.vi

The Primary Processing Loop of LabSocket-E includes a wait function that is used to reduce the CPU consumption of the loop. By default the wait period is set to 20 ms.

On real-time targets that have limited processing power, it may be beneficial to reduce the overall CPU consumption of an application by using "LabSocket-E Init.vi" (Figure 8.5) to increase the wait time in the LabSocket-E Primary Processing Loop.

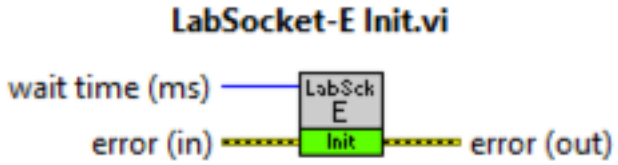


Figure 8.5. LabSocket-E Init.vi

This VI must be called before "LabSocket-E Start.vi" and, if present, "LabSocket-E Diagnostics.vi".

LabSocket-E Diagnostics.vi

The VI "LabSocket-E Diagnostics.vI" (Figure 8.6) generates a detailed log file that provides insight into the behavior of key components of the LabSocket-E system. When the VI is placed on the block diagram of a Target VI, a diagnostic data file will appear in the directory specified by the Path input. It is recommended that this VI be called before LabSocket-E Start.

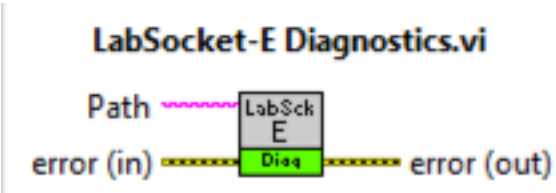


Figure 8.6. LabSocket-E Diagnostics.vi

The output data file name is of the form Diagnostics_YYYMMDD_HHMMSS.txt where YYYMMDD_HHMMSS represents the timestamp of the creation of the diagnostics file.

Recommended path inputs are OS-dependent:

- Real-Time Linux: /home/lvuser/natinst/bin
- VxWorks: /c/ni-rt/startup
- PharLap: C:\ni-rt\startup

This VI outputs to the data file a small amount of log data each time a browser connects to the system. To ensure that the VI does not consume excessive amounts of disk space over an extended period of time, it is recommended that this VI be used only for development and testing purposes and not for long-term production use.

If contacting Bergmans Mechatronics with a support issue regarding LabSocket-E, please include with the support request a copy of any relevant diagnostics files that are available.

LabSocket-E Monitor.vi

The performance of the Primary Processing Loop may be optionally monitored using "LabSocket-E Monitor.vi" (Figure 8.7). The VI has two outputs:

- Primary Processing Loop State – This output can have one of two values
  running - LabSocket is operating normally
  stopped - Primary Processing Loop has stopped. The stopped state occurs when the Target VI stops execution. This state should never occur during normal system operating.

- Primary Processing Loop Period (ms) This is the time period for one iteration of the Primary Processing Loop, in milliseconds.

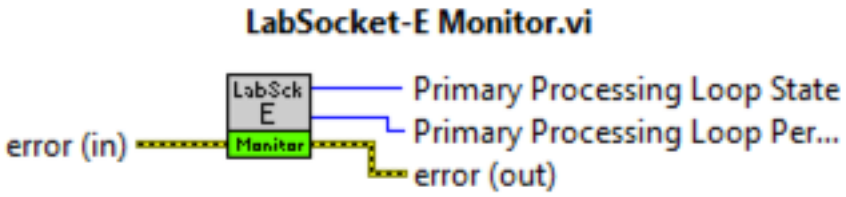


Figure 8.7. LabSocket-E Monitor.vi

For best results, the Target VI and the optional initialization VI "LabSocket Init.vi" should be configured such that the Primary Processing Loop Period is less than 500 ms.

A recommended location for "LabSocket-E Monitor.vi" is in a continuously executing while loop of the Target VI. The demo VI "LSE Demo.vi" that is included with LabSocket-E (Section 7) illustrates the use of "LabSocket-E Monitor.vi".

Appendix A. Image Synchronization for cRIO-903x

To enable dynamic synchronization of picture elements or to invoke the functionality of the #LS_image preprocessor tag (Section 8.1) on cRIO-903x targets, perform the following steps.

1. Download the beta version of the OpenG ZIP library v4.1.0 from: https://lavag.org/topic/18755-open-g-zip-tools-on-linux-rt/?do=findComment&comment=113880

2. Install the package on the development PC using VI Package Manager

3. Install the RT portion of the package to the cRIO-903x using the Add/Remove Software function in NI-MAX (Figure A.1)

4. In the LabVIEW Project, right-click on the Target platform node, select Properties, select Conditional Disable Symbols and add a Symbol "LS_image_sync" with Value "T" (no quotes for either field) (Figure A.2).

5. Enable the embedded UI of the cRIO-903x platform using NI-MAX.

6. Deploy VI to RT platform.

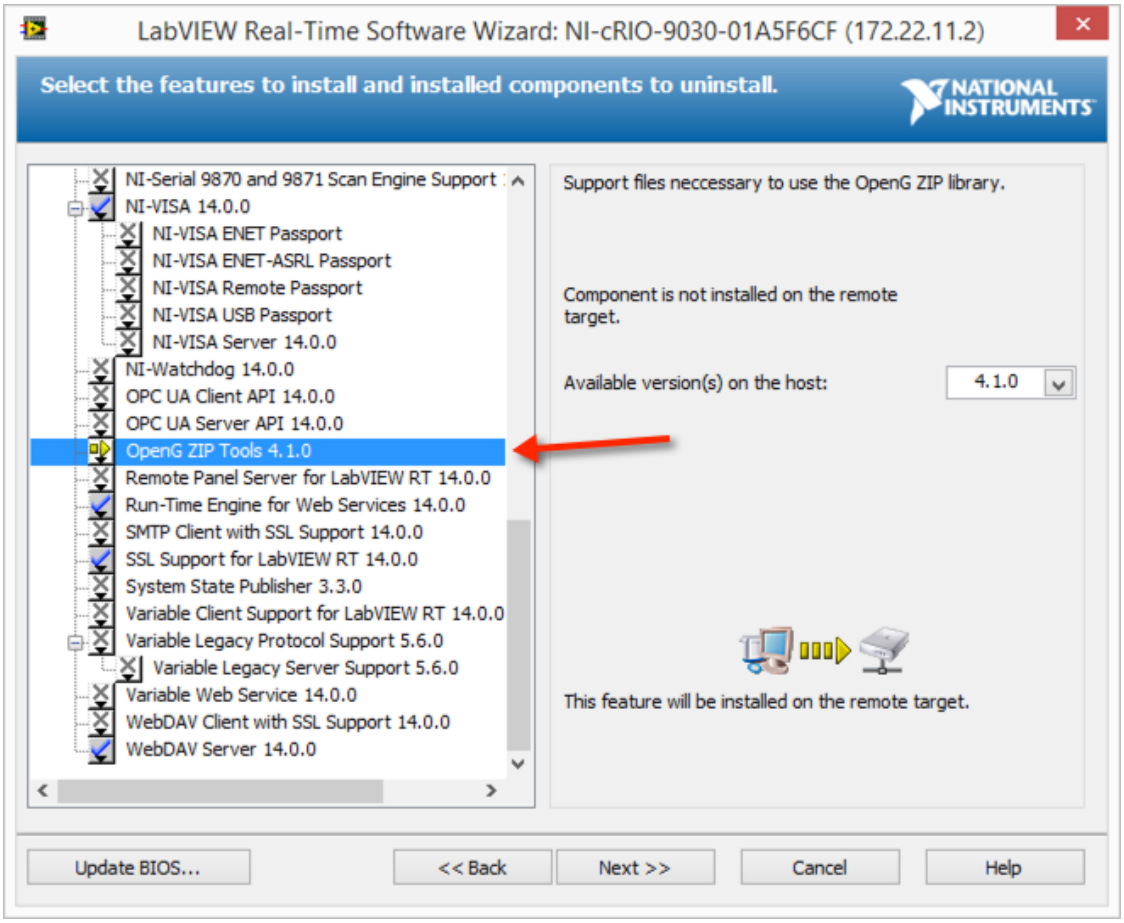


Figure A.1 Installing OpenG ZIP Tools on cRIO-903x Using NI-MAX

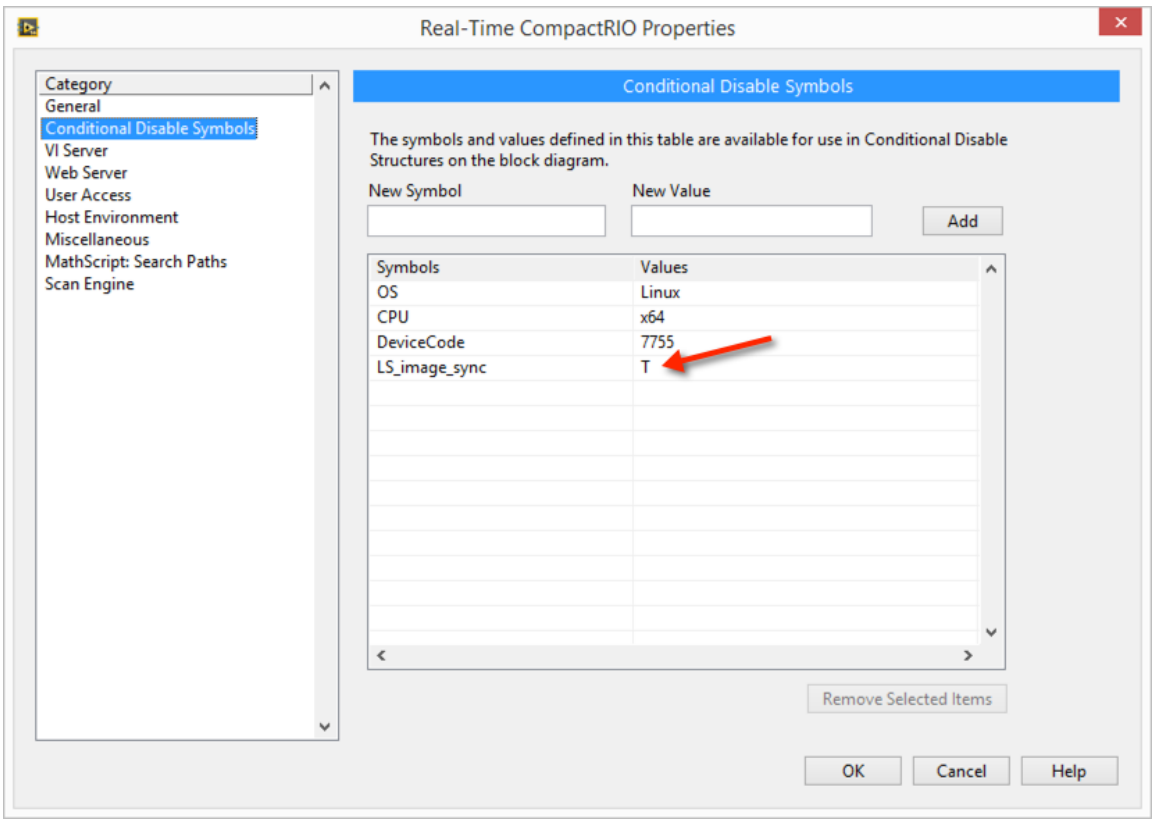


Figure A.2 Conditional Disable Symbol to notify LabSocket-E that the cRIO-903x target is configured to support image synchronization

Appendix B. Known Issues and Solutions

Working with Target VIs in Interactive Mode

1. If the Target VI is stopped and then restarted in interactive mode within less than about two minutes, the LabSocket-E Start VI will pause for about one minute and then generate Error 60 (“The specified port or network address is currently in use”). **Solution** - Wait about two minutes before re-launching a Target VI in interactive mode.

2. When a Target VI is modified, saved and then immediately run in interactive mode, LabSocket-E Start will generate Error 1055 (“Object reference is invalid”). **Solution** - Before running a Target VI in interactive mode, close the VI and re-open it whenever the VI has been modified