FPGA Interface Python Example for NI Linux Real-Time

This example demonstrates how you can use the FPGA Interface Python API to perform host-to-target and target-to-host communication with a compiled LabVIEW FPGA bitfile (.LVBITX). The idea for this example, as well as the LabVIEW project and bitfile are pulled directly from the existing FPGA Interface C API Example by ColdenR: <https://forums.ni.com/t5/NI-Linux-Real-Time-Documents/FPGA-Interface-C-API-Example-for-NI-Linux-Real-Time-and-Eclipse/ta-p/3512138>

This example runs from the command line of the NI Linux Real-Time OS (this example specifically uses the CompactRIO-9068), and showcases the following:

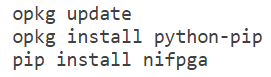
* Opening and closing FPGA Interface sessions
* Reading to and writing from indicators and controls
* Setting up, reading to, and writing from DMA FIFOs
* Using IRQs for FPGA/Host synchronization
* Integration with basic Python data logging techniques

**Steps to Run this example (consult the attached Readme for condensed steps):**

1. Follow the installation procedure for NI Linux RT on the Getting Started page: <http://nifpga-python.readthedocs.io/en/latest/installation.html#ni-linux-rt>
   1. Install the appropriate drivers onto your Windows machine: <http://www.ni.com/downloads/>
      1. This example uses the cRIO-9068, so we would install the CompactRIO drivers
   2. Enable SSH OR the serial console through NI MAX: https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z000000P8bQSAS
   3. Connect via SSH OR the serial console and log in as an administrator:

<https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z000000P8bQSAS> (same link as above)

* 1. Run these commands to install the FPGA Interface Python API



1. Leaving your SSH or serial console open, transfer the example Python script (‘main.py’) and bitfile (‘MyBitfile.lvbitx’) to the Real-Time target (\*\*note: if your target is not a cRIO-9068, you will have to recompile the FPGA VI for your specific target):
   1. Set up either WebDAV (recommended; default installation): <https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z0000019PlESAU>

or FTP: <https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z000000P8ahSAC>

* 1. Move the files to any location on the Real-Time target (ensuring that they’re both in the same directory). On my RIO, I placed these in the ~/home/lvuser directory **🡨screenshots here**
  2. In your SSH or serial session, navigate to the directory that you placed the files in and run the Python script. Here’s what it looked like for me:

**Screenshots here… cd .. cd home/lvuser → python main.py**

**Attach readme here, main.py, LV project/VIs and MyBitfile.lvbitx**