Test Bed for the HMI

Last updated by | Suzana Vukovic | Jul 13, 2020 at 7:15 PM EDT

Test Bed Created via Raspberry Pi 3 & 4

Contents

- Test Bed Created via Raspberry Pi 3 & 4
  - What is OPC UA?
  - Set Up of the Raspberry Pi
  - If you have a Raspberry Pi 4 do the following additional fi...

The debugHMI tool, as it is getting built, includes one window which will have a selection pane for the user to pick which parameters they want to be plotted against time and another window that will strictly include plots for the selected variables against time. In order to be able to test this debugHMI too, a test bed that provides parameters updating by random is necessary as it provides values for the tool to plot. The test bed of choice is an OPC UA server built on a Raspberry Pi 3 and when it is up and running it is under the IP of the Raspberry Pi which is used by UAExpert on a Windows desktop to display the parameters and their values, as they update. This test bed was created using FreeOPCUA code via open62541.org and it was modified by Michael Hansen and Suzana Vukovic to meet our test needs. This code will be further updated to provide various system parameters for test purposes and they will be put into a drop down menu, selection pane, for the user to select which parameter they are interested in looking at over time for debugging purposes.

What is OPC UA?

OPC technology offers a secure, open, reliable mechanism for transferring information between Servers and Clients. In regards to OPC UA specifically, it provides more open transports, better security and a more complete information model than the original OPC DA, "OPC Classic.". OPC UA provides a very flexible and adaptable mechanism for moving data between enterprise-type systems and the kinds of controls, monitoring devices and sensors that interact with real-world data. It is a unique mechanism because it is very flexible, it has a high capability to scale, it is secure and robust and it includes plenty of services that assist in alarming and event management on-plant.

Scalability
OPC UA is scalable and platform-independent. It can be supported on high-end Servers and on low-end sensors. UA uses discoverable profiles to include tiny embedded platforms as Servers in a UA system.

Flexibility
The OPC UA Address Space is organized around the concept of an Object. Objects are entities that consist of Variables and Methods and provide a standard way for Servers to transfer information to Clients.OPC UA uses standard transports and encodings to ensure that connectivity can be easily achieved in both embedded and enterprise environments.

Secure & Robust
OPC UA implements a sophisticated Security Model that ensures the authentication of Client and Servers, the authentication of users and the integrity of their communication. OPC UA provides a full suite of services for Eventing, Alarming, Reading, Writing, Discovery and more.

Sophisticated Alarm & Event Notification
OPC UA provides a highly configurable mechanism for providing alarms and event notifications to interested

Clients. The Alarming and Event mechanisms go well beyond the standard change-in-value type alarming found in most protocols.

Set Up of the Raspberry Pi

The set up of the Raspberry Pi Test bed was as follows:

1. Make sure your Raspberry Pi is up to date with the most recent Pi OS/Raspbian firmware
    - Load the most recent firmware on your sd card (.img file extracted from the zip firmware file)
2. Connect your Raspberry Pi to the same internet network as the computer you are using
3. Once the Pi has been setup, install cmake on it using the command on your Pi command terminal

    ```
    sudo apt-install cmake
    ```

4. If you are installing the open62541 example OPC UA server (ie. simulated pump example) then install the install_opc.sh file onto your computer (that Michael Hansen can provide for you)
    - In order to load this file on your Raspberry Pi, install FileZilla on your computer
    - Next, on your computer's command terminal: ping raspberrypi.local to find out your Raspberry Pi's IP address and note this address down for later use
    - You will enter this IP address and username/password of your Raspberry Pi into the FileZilla program to connect to your Pi and to see its directories on your computer
    - You can now drag/drop any files and folders you would like, directly onto your Raspberry Pi
        - Go to your home/pi directory on your Raspberry Pi via FileZilla and drag/drop install_opc.sh into the folder
5. On your Raspberry Pi, you will now go to the command terminal and you will be running the install_opc.sh file
    - Go to the command terminal and make sure you are in the Pi directory (home/pi)

    ```
    cd ~
    ```

    - Make the file executable

    ```
    sudo chmod 777 install_opc.sh
    ```

    - You will be running the executable file and creating a folder named myopc

    ```
    sudo ./install_opc.sh myopc
    ```

    - Note: If you are unable to make the file executable and/or run the file you may have to carry out the next few steps (this installer eliminates additional spaces that get added in the code, which render the code un-readable by the Pi)
        - Install dos 2 unix file installer on the Pi

        ```
        sudo apt-get install dos2unix -y
        dos2unix install_opc.sh
        sudo u+x install_opc.sh && sudo ./install_opc.sh myopc
        ```

6. Now, the open62541 OPC UA simulation platform has been installed and the myopc folder has been created under home/pi
    - Within the myopc folder you will see a "source" folder
    - You are able to run the server by going to home/pi/myopc/source and putting the following in the command terminal:

    ```
    ./myServer
    ```

    - If you are adding a different OPC UA server, follow the remaining steps from here
7. On your computer, via FileZilla add the source folder of any other OPC UA server you are looking to add onto your Raspberry Pi

- Drag and drop the source folder into the home/pi/myopc/source folder if you are installing additional servers after already having installed the open62541 platform, otherwise simply create a new directory in home/pi for this new source folder
- Once this is loaded on your pi, via your Pi Command terminal if your source folder has a .sh file to assist in rapidly building and compiling the server then:

```
sudo chmod 777 compile.sh
./compile.sh
```

8. The new OPC UA server you have built and compiled can now be run on the Raspberry Pi using:

```
./myServer
```

   - Note: This is assuming that the file for the server executable is named "my Server" , the name of the file may vary case-by-case
   - Now the server is up and running! Congratulations!

The server's nodes can be browsed using a tool named UAExpert, which can be installed on your computer. All you need to connect your server to this tool is your Raspberry Pi's IP address, username and password. From there, you can see the structure of the nodes in your OPC UA server that is running on your Raspberry Pi and you can see plenty of information on the structure (nodeIDs, node values etc.).

If you have a Raspberry Pi 4 do the following additional firmware updates at initial startup:

- The Raspberry Pi 4 has been prone to overheating due to its over-clocking capabilities (very powerful system)
- It is highly recommended to get a cooling fan and/or heat sinks for the Raspberry Pi 4 on top of doing this firmware upgrade
- In order to carry out the firmware updates, enter the following into the Pi's command terminal:

```
sudo apt update
sudo apt upgrade
sudo apt install rpi-eeprom
```

- Once this is complete, ensure everything is up to date for the rpi-eeprom update by using:

```
sudo rpi-eeprom-update
```