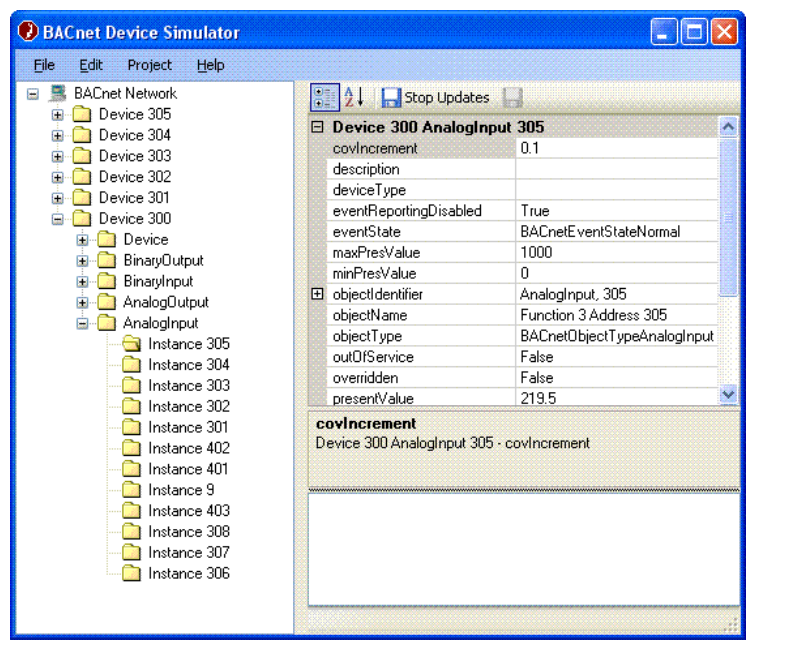
Tyler Schulenberg

3-12-15

**System and Unit Test Report**

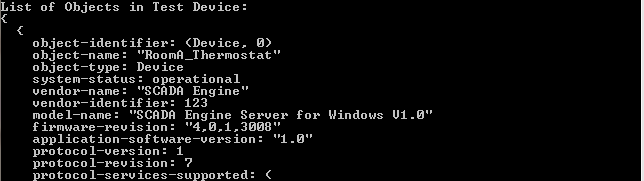
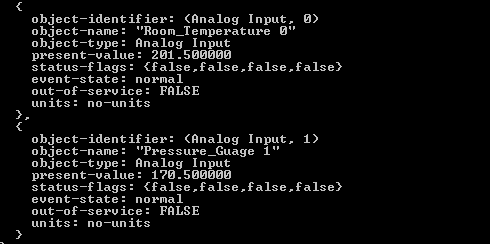
**BACnet Device Simulation:**

Initially the building itself was going to be the main networking test site but we were only able to work on Tuesdays, unfortunately the entire team had classes until the late afternoon. A second problem arose from how the current building devices are installed. The devices rely on an Orca networking controller that is currently installed to allow the building staff to access them. The device adds on more networking layers which block normal BACnet networking calls the ability to talk to the devices. With testing on site out of the question we turned to the BACnet Device Simulator for windows operating systems by SCADA Engine. It emulates a local BACnet network that supports every device of the protocol for testing purposes.



**Win 32 0.8.2 Tool Executable:**

The tools executable files can be found on the BACnet Sourceforge site. These tools allowed us to access BACnet devices on the simulated network using sample functions from the BACnet stack. Using these tools we could see the object values outputted to stdout and test functions such as read (single)/multiple properties, who-has, and who-is.



**Wireshark:**

Wireshark is useful for watching communication between the application and BACnet devices on the network. Using this we were able to realize that the devices at the Monterrey building were using an additional protocol to communicate so we had to focus mainly on simulated testing.

**Win 32 Development Ports found in BACnet stack 0.8.2:**

With the 0.8.2 stack a development environment using Microsoft Visual Studio is supplied for users to use as a place to start. Bugs exist in the code and although I was able to compile the sample projects, they would cash on any run and I was unable to figure out the error since it relates to a missing executable that should be part of the project.

**Linux version of BACnet stack:**

The main part of the BACnet Stack 0.8.2 library code is designed to be compiled on a Linux environment and was extremely easy to set up and use. We were unable to discover devices on a simulated network using the lab computers but were able to test chunks of code for functionality which helped us learn the code. Since we were unable to simulate a network properly on Linux environment we attempted to build the code using minGW and Borland compilers that the stack was designed to use but to no success.