# Build and run on Windows

I’m using the library with Teledyne ADQ8 digitisers, and have built is using VS2017.

There is dependency on POSIX libraries:

* dlfcn, which I fulfilled with [dlfcn-win32](https://github.com/dlfcn-win32/dlfcn-win32).
* Pthreads, which I fulfilled with [pthreads4w](https://github.com/fwbuilder/pthreads4w).

Functions like template void PVBaseIn::read() need to be declared with DLL build semantics, i.e. having the NDS3\_API attribute, in order to DLL build successfully on Windows.

This also applies to functions like allocateDevice().

## void EpicsInterfaceImpl::registrationTerminated()

There’s an interesting issue with temporary file names, which are emitted from the tmpnam function with Windows backslash characters for the path. These are inappropriately interpreted by the EPICS ioc shell as command switches. This issue resolved by replacing the Windows-style slashes with Unix-style forward slashes. Windows is happy to open the file with those semantics.

## void EpicsFactoryImpl::loadNdsNamingRules(const iocshArgBuf \* arguments)

This method loads the database naming rules that are used to form a completed EPICS database.

If the file name is incorrectly specified, it was previously a silent fail.

This caused the database to be incorrectly loaded.

I’ve added an exception throw, if the file is invalid.

# NDS commands issued from st.cmd

The EPICS start-up script issues a command “nds setLogLevelInfo <PORT>”. This is the only such command issued.

This causes the application to fall over with debug assertions indicating corrupted memory after:
 parameters\_t response = findNode->second(parameters);

Where the command is invoked.

I have not determined the reason for this.

Wy workaround is to comment out the command in the start-up script.

This is consequently not having an impact on me, as it is only setting the log level to what is the default value anyhow.