OmniDriver User Guide - How to Use the Timeout Feature

Installation

Follow the normal procedure for installing OmniDriver on your system. Then perform the steps in the following section ***if*** you are developing on a 32-bit Windows system.

IMPORTANT: Converting over to the WinUSB driver (32-bit systems only)

After installing OmiDriver, some additional installation steps are necessary if you are using any of the **32-bit** Windows platforms (XP, Vista, or Windows 7).

These steps are necessary because the new timeout feature is only supported by the WinUSB driver, not by the ezusb.sys driver. By default, 32-bit Windows platforms use the older ezusb.sys driver.

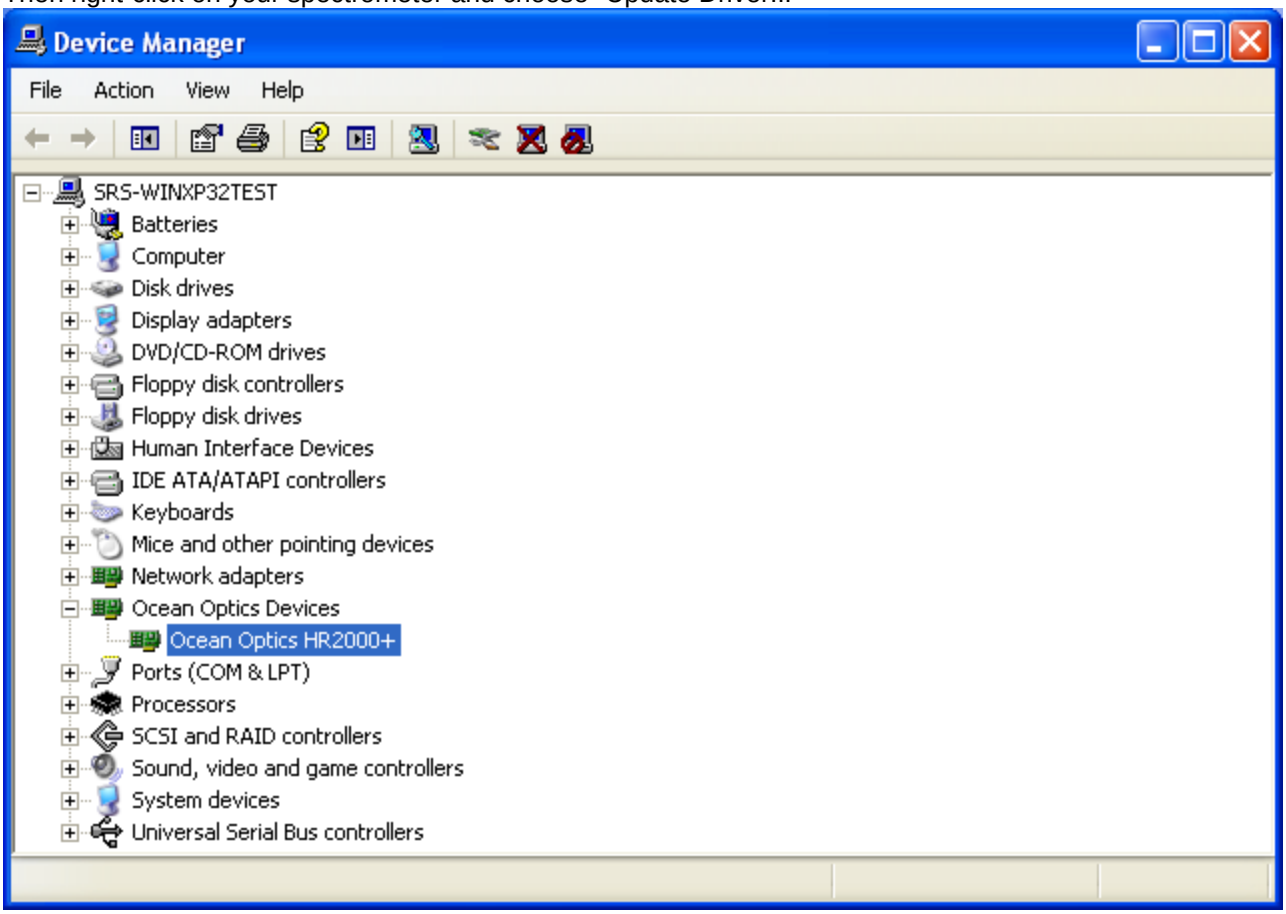
1. In C:\Program Files\Ocean Optics\OmniDriverSPAM\OOI_HOME,
    a. Delete NatUSB.dll
    b. Copy or rename "NatUSB_32_winusb.dll" to "NatUSB.dll"
2. If you are using SpectraSuite on the same 32-bit PC, you must manually replace SpectraSuite's NatUSB.dll file with the NatUSB_32_winusb.dll mentioned in step 1. Otherwise SpectraSuite will no longer "see" any of your spectrometers. This step is ***only*** necessary on 32-bit Windows systems. To do this:
    a. cd C:\Program Files\Ocean Optics\OmniDriverSPAM\OOI_HOME
    b. copy NatUSB_32_winusb.dll to C:\Program Files\Ocean Optics\SpectraSuite\spectrasuite\lib, renaming it to NatUSB.dll
3. Switch your spectrometer over to the WinUSB driver (follow the detailed steps below)

How to Manually Switch Your Spectrometer Over to the WinUSB Driver

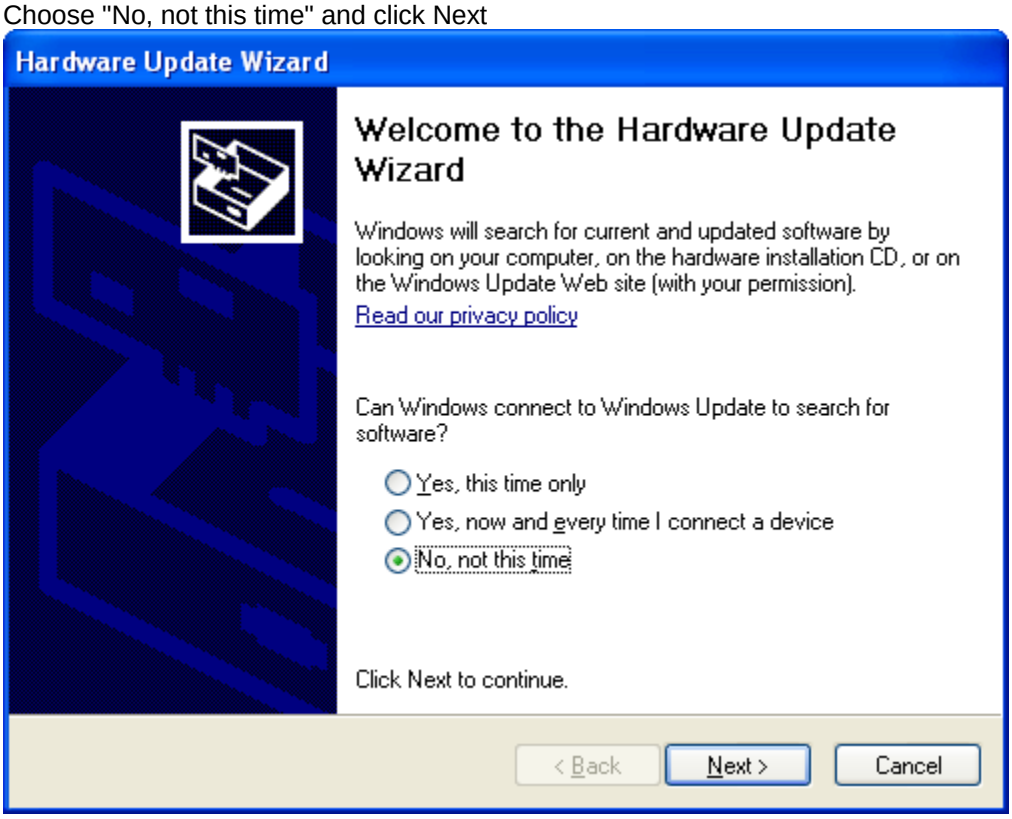
You must perform the following procedure if you are developing on **32-bit** Windows platforms (XP, Vista, and Windows 7) because the new timeout feature is only supported by the WinUSB driver, not by the ezusb.sys driver. By default, 32-bit Windows platforms use the older ezusb.sys driver, which does not support the new timeout feature.

The steps shown below are for Windows XP 32-bit. The exact procedure will be slightly different for Windows Vista 32-bit and Windows 7 32-bit.

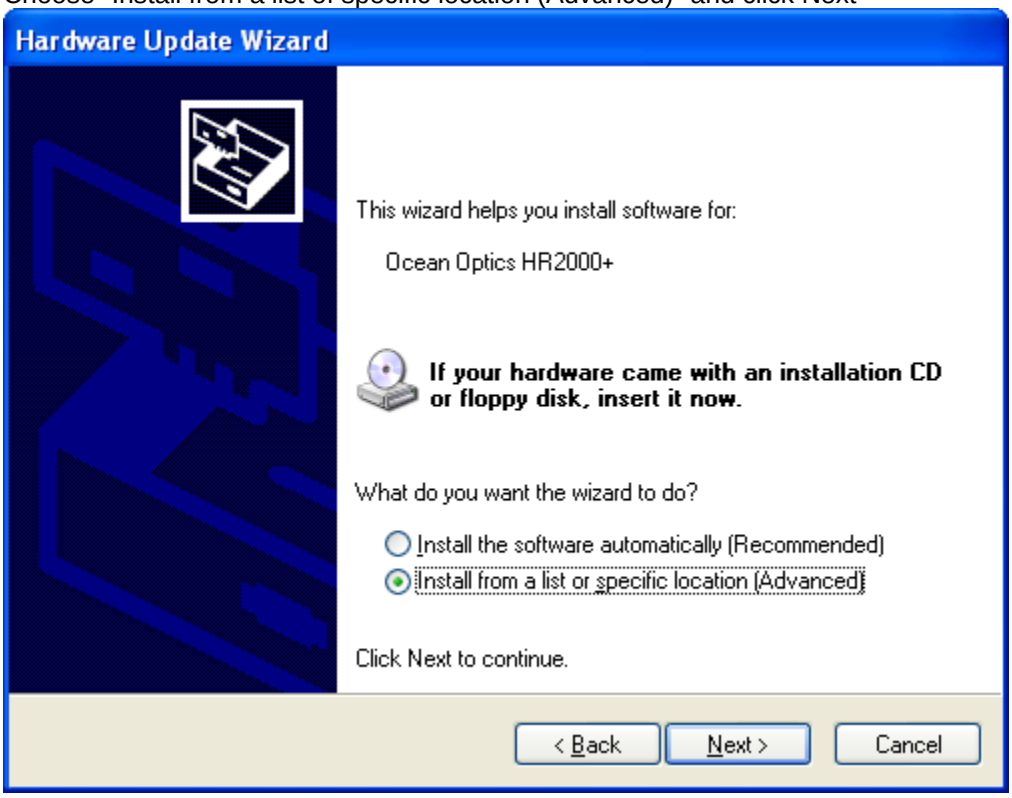
Open the Device Manager window
(Click on Control Panel → System → Hardware tab → Device Manager button)
Then right-click on your spectrometer and choose "Update Driver..."



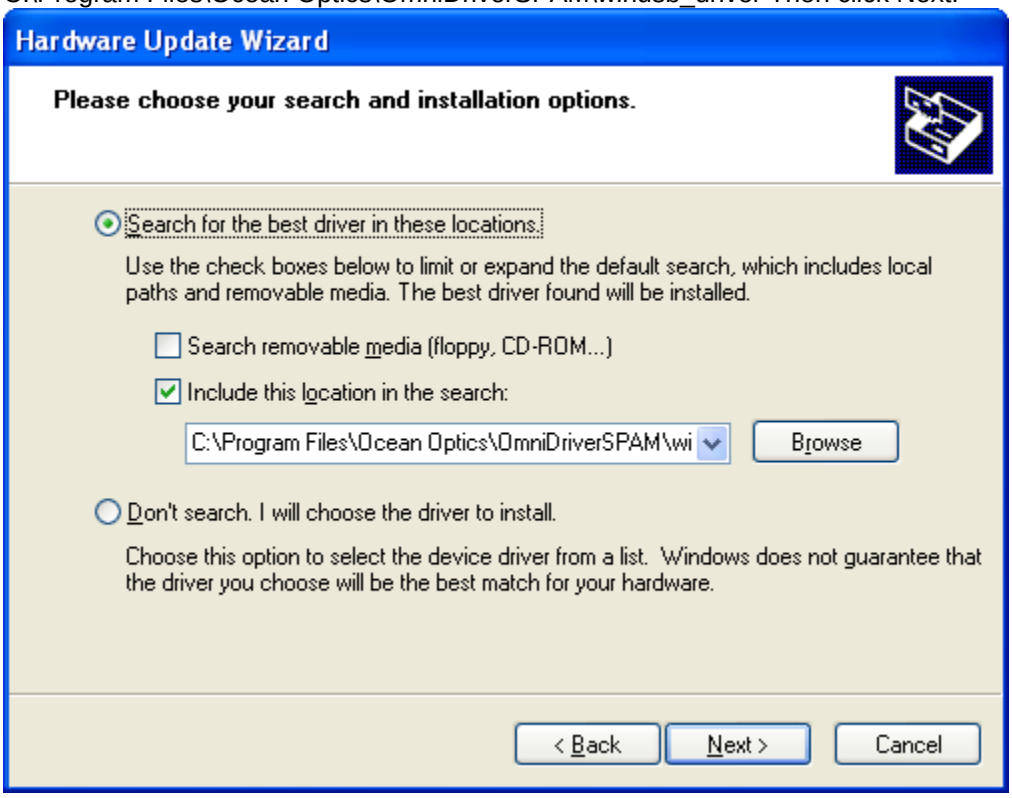
Choose "No, not this time" and click Next



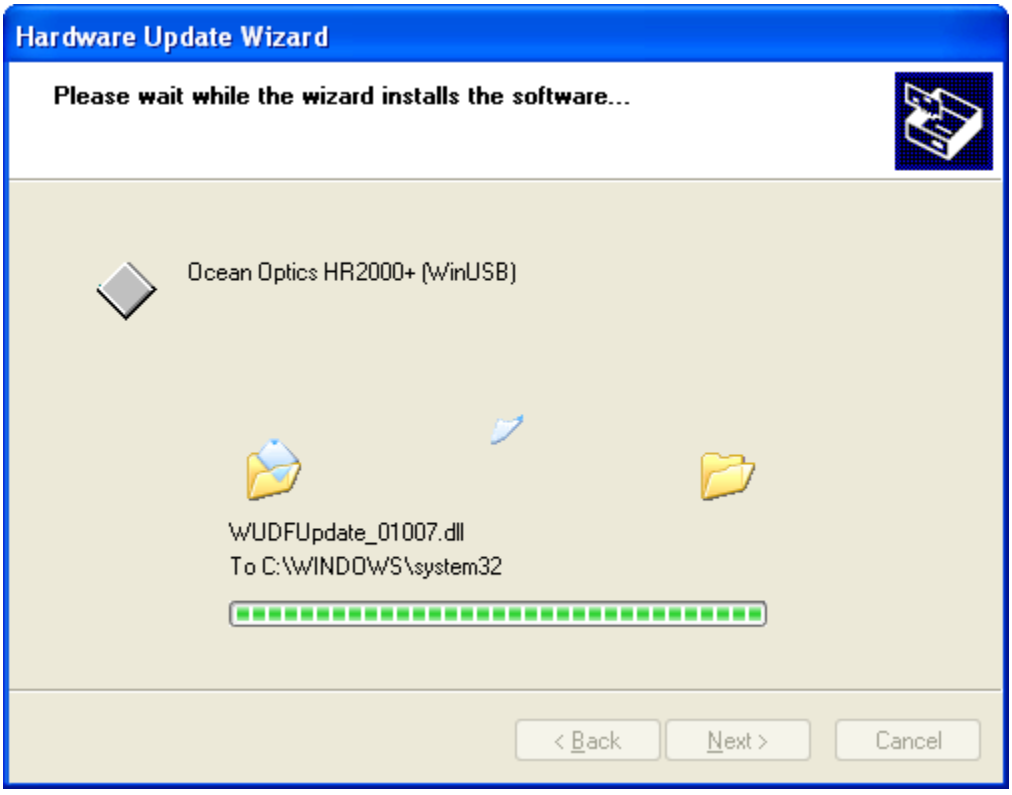
Choose "Install from a list of specific location (Advanced)" and click Next



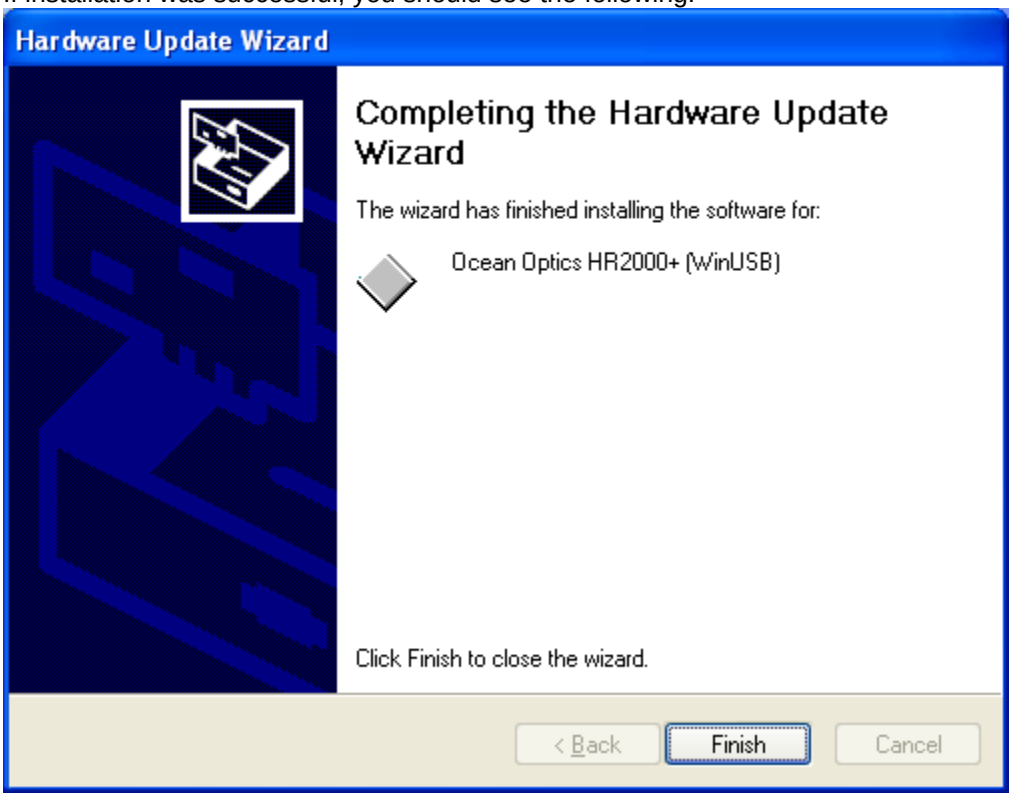
Choose "Include this location in the search" and Browse to C:\Program Files\Ocean Optics\OmniDriverSPAM\winusb_driver Then click Next.



You should see:



If installation was successful, you should see the following.



Caveats

- This solution only applies to Windows (32-bit and 64-bit architectures), not Linux or MacOSX.
- This solution only works with the WinUSB driver, not with the ezusb driver. Once you have installed WinUSB, your Windows system cannot revert to ezusb. If you are on a 32-bit system, and you are using SpectraSuite as well as OmniDriver, you will need to manually copy OmniDriver's NatUSB.dll into the SpectraSuite install directory. Otherwise SpectraSuite will no longer be able to detect any spectrometers. The installation instructions explain how to perform this step.
- This solution has only been tested and verified for the HR2000+ and HR4000 spectrometers. It may possibly work with other spectrometers, but these have not been tested yet.
- This solution only works for USB devices. It does not apply to network-attached spectrometers such as the "Jaz Network" spectrometer.
- This solution does not work for external trigger mode "3". It does work for all other trigger modes.
- The USBProgrammer utility does not work with the WinUSB driver. You will not be able to use this utility on systems using WinUSB. USBProgrammer only works on systems using the older ezusb driver.

How to Program the Timeout Feature

To set the timeout, call wrapper.setTimeout(int timeoutMilliseconds)

To determine if a timeout has occurred, you must call wrapper.isTimeout(int spectrometerIndex, int channelIndex) after the wrapper.getSpectrum() method returns control to your application. If a timeout has occurred, the returned spectrum will be set to all zeroes, and isTimeout() will return TRUE.

Beware: If your timeout setting is shorter than the integration time, your call to getSpectrum() may time-out.

Example of Typical Usage of the Timeout Feature

When calling getSpectrum(), if a timeout occurs before your external trigger event occurs, you can simply call getSpectrum() again and continue to wait for the trigger event. Typically your logic will look something like the following:

```
wrapper.setIntegrationTime()
wrapper.setExternalTriggerMode() // choose some external hardware trigger mode
wrapper.setTimeout()
while (keepWaitingForTrigger() == true)  // you implement the keepWaitingForTrigger() method
{
        wrapper.getSpectrum()
        if (wrapper.isTimeout() == true)
                continue; // timeout occurred, go to top of loop and try again

        // Don't forget to check for a valid spectrum result !!
        If  (wrapper.getWrapperExtensions().isSpectrumValid() == false)
        {
                // Your error handler
                …
        }
        // Process valid spectrum
        …
}
```

Additional Programming Considerations

- The timeout setting applies to all channels of the spectrometer. It is not possible to set different timeout values for each channel.
- If your application uses multi-threading, you must be careful to restrict all references to a given spectrometer to a single thread. You may not acquire spectra from the same spectrometer from multiple threads. This restriction has always been the case for OmniDriver. This means that all channels of multi-channel spectrometers must be referenced from a *single* thread. Refer to the chapter on multi-threading in the "OmniDriver Programming Manual" for more details.
- When switching your spectrometer from normal mode to one of the external trigger modes, you may need to call getSpectrum() once to "flush" out the spectrum acquisition that occurred

immediately prior to switching modes. This can happen because in "normal" mode, the spectrometer is continuously acquiring spectra, so when you switch modes, the spectrometer will probably be in the middle of acquiring a spectrum. This spectrum will be returned by the next call you make to getSpectrum(), but this spectrum was not acquired as a result of an actual external trigger signal.

FAQ

QUESTION: What happens if the timeout period is shorter than the integration time?
ANSWER: some of the calls to getSpectrum() will timeout, and some will return valid spectra. The ratio of valid spectra to timeouts will depend on the ratio of your timeout period to the integration time. However, this situation should not cause you to miss spectral acquisitions, unless the integration time is so short your program cannot call getSpectrum() soon enough to capture the next spectrum.

QUESTION: What happens in the following scenario?

1. Set spectrometer to one of the external trigger modes
2. Call getSpectrum()
3. Timeout occurs before the trigger occurs
4. Trigger event happens before you call getSpectrum() a second time
5. Call getSpectrum()

Result: the second call to getSpectrum() will return immediately with the spectrum that was acquired when the trigger event occurred.

If *multiple* trigger events happen after the first getSpectrum() has timed-out and before you call getSpectrum() a second time, when you *do* call getSpectrum() the second time, it will immediately return with the spectrum from the *FIRST* trigger event. Data from any additional trigger events during this window will be lost.

If a timeout occurs before the spectrum can be acquired, the spectral array returned by getSpectrum() will contain all zeroes.

By default, the timeout period is set to "infinity". If you set a non-zero timeout and later wish to turn-off the timeout feature, you can do this by calling wrapper.setTimeout() with a value of zero. This causes getSpectrum() to wait "forever".

Trouble-Shooting Runtime Errors

Symptom: your application cannot detect any spectrometers

This can be caused by failing to copy NatUSB_32_winusb.dll to NatUSB.dll in the OOI_HOME directory. Look at the Installation section above for details on how to do this.

Symptom: UnsatisfiedLinkError … NatUSBSetTimeout()

If you are running your application on a 32-bit Windows system, you may have forgotten to manually switch your spectrometer over to the newer WinUSB driver. Look at the Installation section above for details on how to do this.