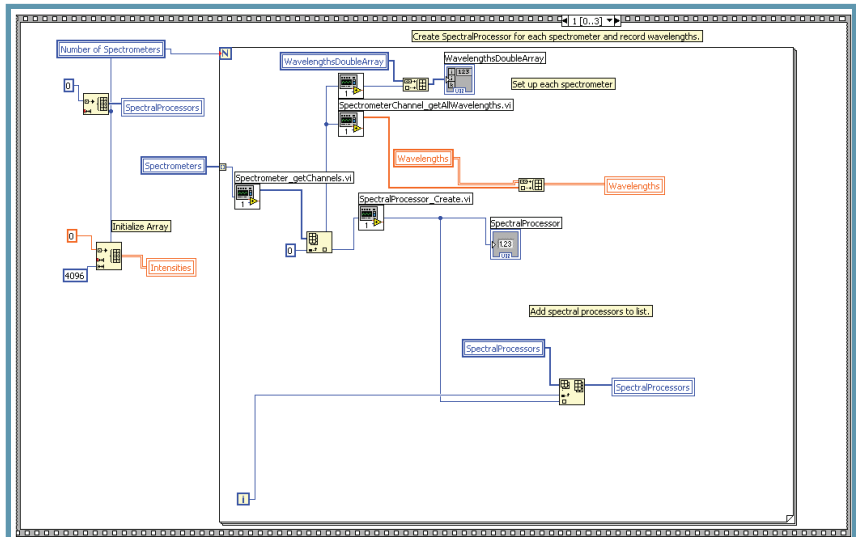


OmniDriver and SPAM

More Than a Device Driver

OmniDriver is a software driver package for Windows, Mac OSX and Linux operating systems that allows you to easily write custom software solutions for your Ocean Optics USB spectrometers and direct-attach devices. OmniDriver takes the best of our software driver packages and allows you to harness the power of high-speed data acquisition in a single cross-platform driver. Integrate OmniDriver into your own software application for complete control over USB spectrometers and devices in virtually any OS environment.

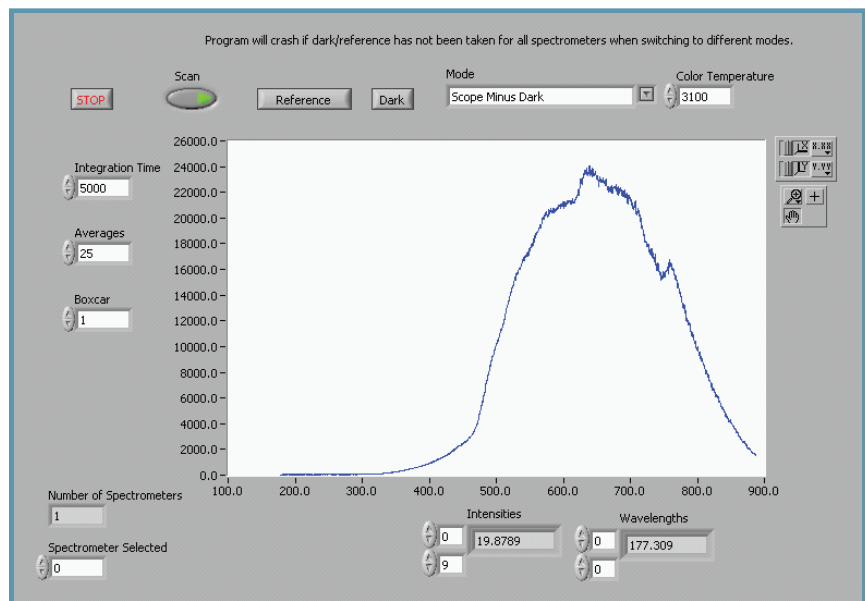
What's more, our Spectral Processing and Manipulation (SPAM) module performs all spectral processing math from subtracting dark to radiometric color analysis. SPAM provides you with the ability to harness spectral processing commands for your own applications but does not require you to use Ocean Optics spectrometers or hardware. SPAM is available as a stand-alone module or as part of the OmniDriver package (OMNI+SPAM).



Example OmniDriver Code for LabVIEW

Developed in Java™

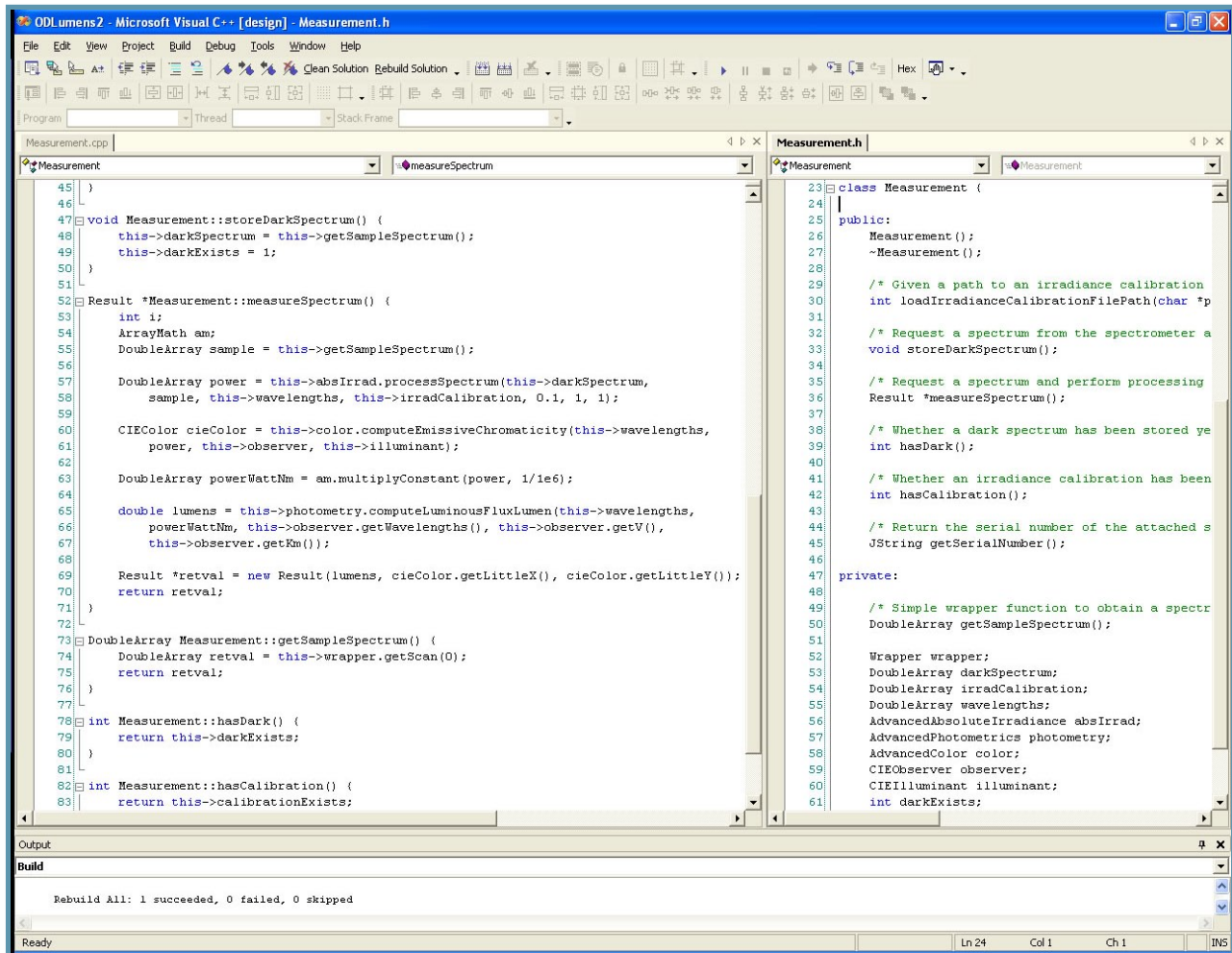
OmniDriver was created in the Java environment and includes native libraries for select Windows, Mac OSX and Linux operating systems. Using OmniDriver, you can develop robust applications to control multiple Ocean Optics USB spectrometers and direct-attach devices across these different operating systems. Ocean Optics is the first and only provider in the optical sensing industry to offer this level of cross-platform compatibility.



Resulting Spectrum of USB4000 with LS-1 Light Source

Complete Platform & Device Independence

Applications written in Java are environment-independent; they can work across all operating systems. This is a very advantageous feature for product developers and OEMs wanting to expand their product offering with systems that work on multiple operating platforms. In developing OmniDriver, we took the Java philosophy a step further to create a device driver that is not only platform-independent, but also spectrometer-independent; the same Java code works with all Ocean Optics USB spectrometers and direct-attach devices.



```
Measurement.h
class Measurement {
public:
    Measurement();
    ~Measurement();

    /* Given a path to an irradiance calibration
    int loadIrradianceCalibrationFilePath(char *p
    /* Request a spectrum from the spectrometer a
    void storeDarkSpectrum();
    /* Request a spectrum and perform processing
    Result *measureSpectrum();
    /* Whether a dark spectrum has been stored ye
    int hasDark();
    /* Whether an irradiance calibration has been
    int hasCalibration();
    /* Return the serial number of the attached s
    JString getSerialNumber();

private:
    /* Simple wrapper function to obtain a spectr
    DoubleArray getSampleSpectrum();

    Wrapper wrapper;
    DoubleArray darkSpectrum;
    DoubleArray irradCalibration;
    DoubleArray wavelengths;
    AdvancedAbsoluteIrradiance absIrrad;
    AdvancedPhotometrics photometry;
    AdvancedColor color;
    CIEObserver observer;
    CIEIlluminant illuminant;
    int darkExists;
};

Measurement.cpp
void Measurement::storeDarkSpectrum() {
    this->darkSpectrum = this->getSampleSpectrum();
    this->darkExists = 1;
}

Result *Measurement::measureSpectrum() {
    int i;
    ArrayMath am;
    DoubleArray sample = this->getSampleSpectrum();

    DoubleArray power = this->absIrrad.processSpectrum(this->darkSpectrum,
        sample, this->wavelengths, this->irradCalibration, 0.1, 1, 1);

    CIEColor cieColor = this->color.computeEmissiveChromaticity(this->wavelengths,
        power, this->observer, this->illuminant);

    DoubleArray powerWattNm = am.multiplyConstant(power, 1/1e6);

    double lumens = this->photometry.computeLuminousFluxLumen(this->wavelengths,
        powerWattNm, this->observer.getWavelengths(), this->observer.getV(),
        this->observer.getKm());

    Result *retval = new Result(lumens, cieColor.getLittleX(), cieColor.getLittleY());
    return retval;
}

DoubleArray Measurement::getSampleSpectrum() {
    DoubleArray retval = this->wrapper.getScan(0);
    return retval;
}

int Measurement::hasDark() {
    return this->darkExists;
}

int Measurement::hasCalibration() {
    return this->calibrationExists;
}
```

Example of C++ code in Microsoft Visual Studio

“I don’t know how to program in Java”

No problem! If you know how to program in C, C++, Visual Basic, or one of many Microsoft Office applications, you can use OmniDriver. Our wrapper libraries take care of the Java code; we provide Framework (Mac OSX), Dynamic Link Library (Windows), Shared Object (Linux) and a .NET object (Windows). Get a taste of the power of OmniDriver, and see examples of our code at:

www.oceanoptics.com/products/omnidriver.asp



OmniDriver Features

What does it take to make such a robust driver?

High-res Timing: Time stamping that is accurate to sub-microsecond performance; great for chemical kinetics and other applications that require complex time accountability.

LabView Support: OmniDriver version 7.1 and above provides drivers for LabVIEW to enable you to configure Ocean Optics spectrometers as real-time virtual instruments in National Instruments' LabVIEW graphical programming environment.

Support for Ocean Optics USB Spectrometers and Direct-attach Devices:

- USB2000*
- USB2000-FLG*
- USB2000+
- USB4000
- HR2000*
- HR2000+
- HR4000
- QE65000
- NIR256
- NIR512
- Red Tide (USB650)
- ADC1000-USB (includes Deep Well)
- MMS-Raman*
- Curie*

* *Discontinued Products*

Support is Coming Soon for Our Latest Spectrometers:

- Jaz
- Maya2000
- Maya2000 Pro

Measurement Corrections:

- Detector nonlinearity
- Electrical dark (automatic baseline removal)
- Stray light
- Boxcar smoothing (averaging across pixels)
- Averaging multiple scans

Control of the Following Extended Functions (Depending on Spectrometer Model):

- Strobe-enabled
- Thermo-electric cooling (TEC), and reading detector and PCB temperatures
- Analog input for the supported spectrometers (voltage only)
- Gated fluorescence mode
- Analog output (4-20 mA current output for LS-450 and soon the AOUT; voltage out for HR4000)
- Digital (TTL) input/output (control of GPIO pins)
- Setting external trigger modes
- Reading out wavelength calibration
- Setting single and continuous strobe delays

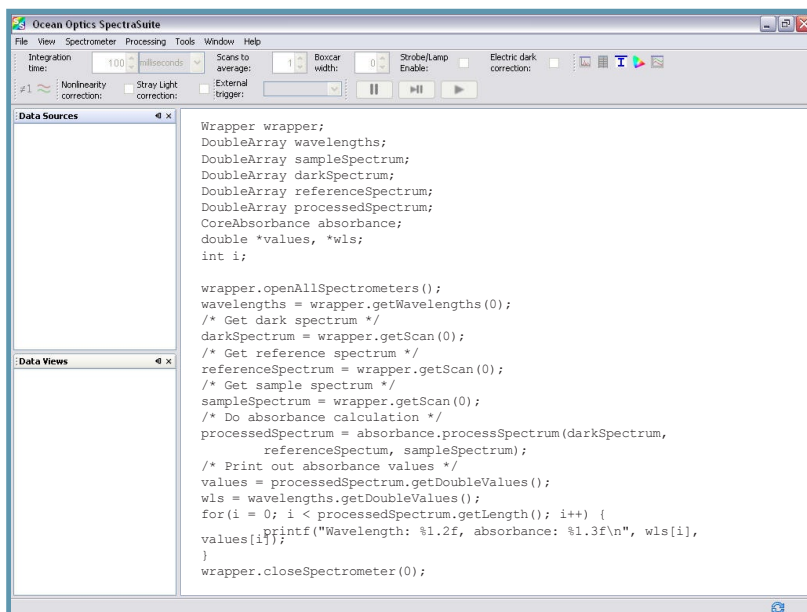
Operating System Support:

Windows: Windows XP, NT, and 2000

(Vista and 64-bit coming soon)

Mac: OSX 10.3 or later

Linux: Many x86 distributions are supported
Kernel 2.4.20 and libusb 0.1.10 or later are required



```
Wrapper wrapper;
DoubleArray wavelengths;
DoubleArray sampleSpectrum;
DoubleArray darkSpectrum;
DoubleArray referenceSpectrum;
DoubleArray processedSpectrum;
CoreAbsorbance absorbance;
double *values, *wls;
int i;

wrapper.openAllSpectrometers();
wavelengths = wrapper.getWavelengths(0);
/* Get dark spectrum */
darkSpectrum = wrapper.getScan(0);
/* Get reference spectrum */
referenceSpectrum = wrapper.getScan(0);
/* Get sample spectrum */
sampleSpectrum = wrapper.getScan(0);
/* Do absorbance calculation */
processedSpectrum = absorbance.processSpectrum(darkSpectrum,
referenceSpectrum, sampleSpectrum);
/* Print out absorbance values */
values = processedSpectrum.getDoubleValues();
wls = wavelengths.getDoubleValues();
for(i = 0; i < processedSpectrum.getLength(); i++) {
printf("Wavelength: %1.2f, absorbance: %1.3f\n", wls[i],
values[i]);
}
wrapper.closeSpectrometer(0);
```

Example of absorbance measurement in C++

NOTE: OmniDriver does NOT support PCI or ISA products.

SPAM Features

Powerful tools for processing your spectral data

A Robust Set of Functions:

- Scope mode
- Scope minus dark
- Absorbance
- Transmission
- Reflection
- Relative irradiance (with user-specified color temperature)
- Raman (with user-specified wavelength)
- Blackbody and CIE Relative daylight spectrum generators with user-defined color temperature
- Peak finding and metrics (centroid, full width at half max in units of pixels and wavelengths, center wavelength, integral, pixel number, 90% enclosing width)

Choice of Measurement Units:

- Nanometers
- Microns
- Pixel number
- Gigahertz
- Wave numbers
- Raman shifts

SPAM Corrections:

- Nonunity reference
- Reference monitoring

Coming soon to SPAM:

- Linear regressions
- Matrix math
- Linear and cubic splines

Spectral Calculations:

- Absolute irradiance
- New calibration from lamp files
- Photometry
- Lumens
- Lux
- Candela
- Luminance
- μ Joule
- μ Watt
- μ Joule/cm²
- μ Watt/cm²
- dBm
- Photons/cm²
- Total photons
- Moles of photons
- Electron volts
- Color
- CIE 1931 & CIE 1965 observers
- Energy, power, photons (integrated over a range, or as a spectrum)
- CIE Illuminants A, B, C, D50, D55, D65, D75, E (unity), F1-F12
- Reflective and emissive color
- Emissive color can use relative or absolute irradiance
- Photosynthetic Active Response (PAR)
- CIE XYZ
- x, y, z
- Color spaces
- Color Rendering Index (CRI)
- General CRI Ra
- Special CRI R1- R14
- Correlated Color Temperature (CCT)
- Dominant wavelength and purity
- u'v'w', u,v hue angle, u,v saturation
- CIE Whiteness and Tint
- CIELAB (L*a*b*, hue angle, chroma)
- CIE 1960 u,v
- Hunter Lab

