



# WinDriver PLX 9050 Sample

The source code for this project is provided with Jungo WinDriver. To compile this application, you will need a compiler and CMake installed.

## Files

- **p9050\_diag.c**  
The main file which demonstrates access to the PLX 9050, using `plx_lib.c`
- **../lib/plx\_lib.c**  
A library for accessing PLX devices using the WinDriver High Level APIs
- **../diag\_lib/plx\_diag\_lib.c**  
A library for accessing PLX devices using the WinDriver High Level APIs
- **CMakeLists.txt**  
An input file for the CMake build system.
- **readme.pdf**  
Describes the sample files.

We provide several methods of compiling this code:

## Compiling this project using Microsoft Visual Studio/Visual Studio Code

- If you are using Microsoft Visual Studio 2017 and higher or Visual Studio Code, make sure to have installed CMake support for it.
- Open the `CMakeLists.txt` file and Visual Studio will process it and allow to access the relevant target using the **CMake Targets View**.
- This will allow you to build the project.

## Compiling using a different IDE/Compiler:

- From the terminal, run the following command from the working directory of this project:  

```
$ cmake . -b build
```

This will create a Unix Makefile for the project in a new sub-directory named `build`. To build it, change directory to that sub-directory and run

```
$ make
```
- You can use CMake to generate projects for various other platforms and IDEs. Consult CMake's documentation for more info.

## Creating your own project

- Create a new project using your IDE.
- Choose console mode project.
- Include the following files in the project:  
`p9050_diag.c`  
`../lib/plx_lib.c`  
`../diag_lib/plx_diag_lib.c`

- Include the WinDriver Diagnostics samples shared files:  
`(WD_BASEDIR)/samples/c/shared/wdc_diag_lib.c`  
`(WD_BASEDIR)/samples/c/diag_lib.c`  
`$(WD_BASEDIR)` is the directory where WinDriver is installed at.
- Link your project with `$(WD_BASEDIR)/lib/wdapi<version>.lib` (Windows)  
or `$(WD_BASEDIR)/lib/libwdapi<version>.so` (Linux)  
or `$(WD_BASEDIR)/lib/libwdapi<version>.dylib` (MacOS)  
In order to access WinDriver's High-Level API.  
`$(WD_BASEDIR)` is the directory where WinDriver is installed at.
- Make sure to add the relevant flags to your system:  
`-DKERNEL_64BIT` if using a 64-bit operating system.  
`-DWD_DRIVER_NAME_CHANGE` if using a renamed driver.

## Converting to a GUI application:

This sample was written as a console mode application (rather than a GUI application) that uses standard input and standard output. This was done in order to simplify the source code. You may change it into a GUI application by removing all calls to `printf()` and `scanf()` functions, and calling `MessageBox()` instead (on Windows). On other operating systems - you can use the relevant libraries such as GTK or Qt.