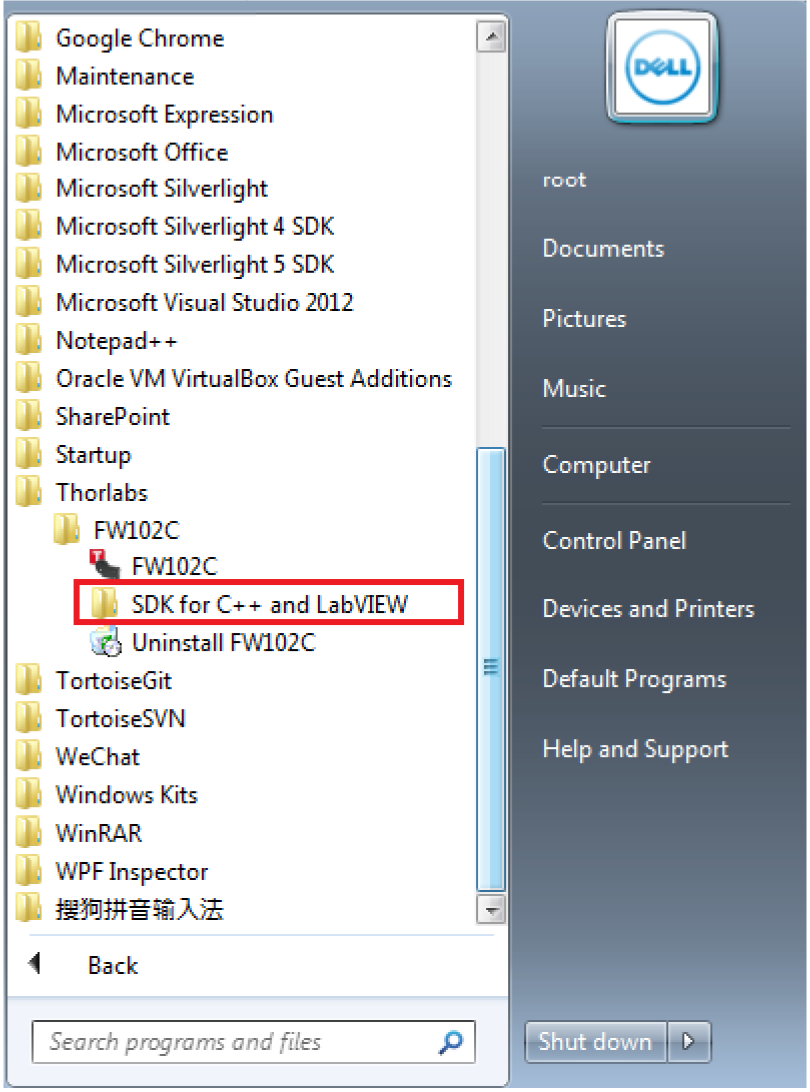
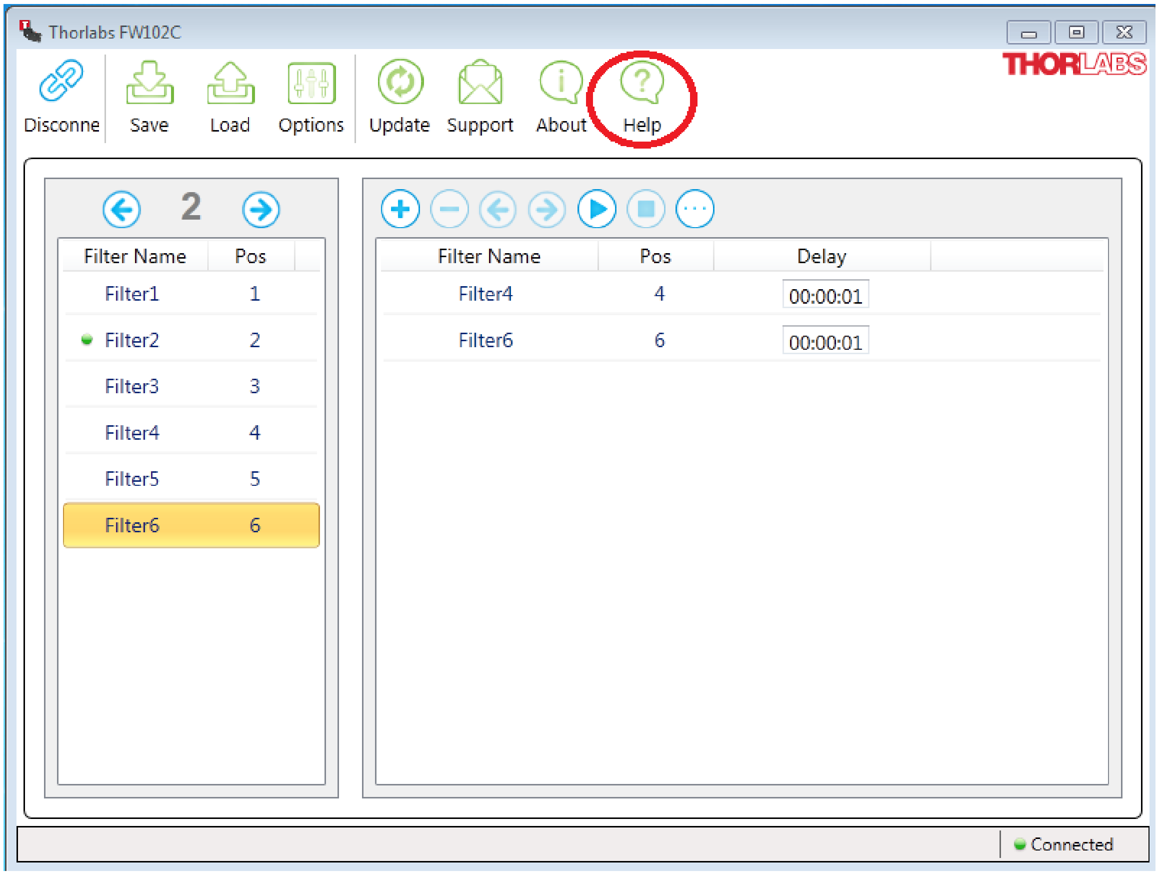
Software Development

User can start software development in C/C++ develop environment, LabVIEW etc.

The software development interface can be found in the start menu.



or by clicking *Help* in software menu.

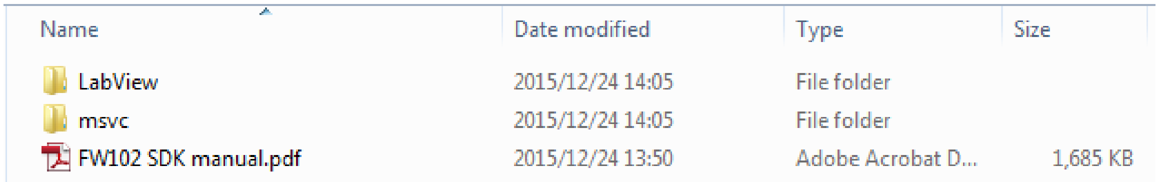


Filter Name	Pos
Filter1	1
● Filter2	2
Filter3	3
Filter4	4
Filter5	5
Filter6	6

Filter Name	Pos	Delay
Filter4	4	00:00:01
Filter6	6	00:00:01

● Connected

In this directory, you will find two folders and the support files for software development, as shown below.



Name	Date modified	Type	Size
LabView	2015/12/24 14:05	File folder	
msvc	2015/12/24 14:05	File folder	
FW102 SDK manual.pdf	2015/12/24 13:50	Adobe Acrobat D...	1,685 KB

Software Development (C/C++)

User can start software development with FilterWheel102_win32.dll in C/C++ development environment which can be found in FW102C C++ SDK.zip under \Sample directory. The corresponding header file is also in FW102C C++ SDK.zip under \Sample directory.

Copy FilterWheel102_win32.dll to your program folder, and make sure the library file and exe file are in the same folder.

Below is the description of the header file fw_cmd_library.h:

fw_cmd_library.h File Reference

Functions

- **DllExport** int **GetPorts** (char *serialNo)
- **DllExport** int **Open** (char *serialNo, int nBaud, int timeout)
- **DllExport** int **IsOpen** (char *serialNo)
- **DllExport** int **Close** (int hdl)
- **DllExport int SetTimeout(int hdl,int timeout);**
- **DllExport** int **SetPosition** (int hdl, int pos)
- **DllExport** int **SetPositionCount** (int hdl, int count)
- **DllExport** int **SetSpeed** (int hdl, int speed)
- **DllExport** int **SetTriggerMode** (int hdl, int mode)
- **DllExport** int **SetMinVelocity** (int hdl, int min)
- **DllExport** int **SetMaxVelocity** (int hdl, int max)
- **DllExport** int **SetAcceleration** (int hdl, int acceleration)
- **DllExport** int **SetSensorMode** (int hdl, int mode)
- **DllExport** int **Save** (int hdl)
- **DllExport** int **GetPosition** (int hdl, int &pos)
- **DllExport** int **GetPositionCount** (int hdl, int &poscount)
- **DllExport** int **GetSpeed** (int hdl, int &speed)
- **DllExport** int **GetTriggerMode** (int hdl, int &triggermode)
- **DllExport** int **GetMinVelocity** (int hdl, int &minvelocity)
- **DllExport** int **GetMaxVelocity** (int hdl, int &maxvelocity)
- **DllExport** int **GetAcceleration** (int hdl, int &acceleration)
- **DllExport** int **GetSensorMode** (int hdl, int &sensormode)
- **DllExport** int GetTimeToCurrentPos (int hdl, int &time)
- **DllExport** int **GetId** (int hdl, char *d)

---

fw_cmd_library.h File Reference

- ### DllExport int Close (int hdl)

close current opend port

Parameters:

hdl	*handle of port.*

Returns:

0: success; negtive number : failed.

- ### DllExport int SetTimeout(int hdl,int timeout);

set fiterwheel's timeout

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function

Parameters:

hdl	*handle of port.*
timeout	*timeout*

Returns:

0: success

- **DllExport int GetAcceleration (int hdl, int & acceleration)**

get the fiterwheel current acceleration.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
acceleration	*fiterwheel actual acceleration*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int GetId (int hdl, char * d)**

get the fiterwheel id.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
d	*output string (<255)*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int GetMaxVelocity (int hdl, int & maxvelocity)**

get the fiterwheel current max velocity.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
maxvelocity	*fiterwheel actual max velocity*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int GetMinVelocity (int hdl, int & minvelocity)**

get the fiterwheel current min velocity.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
minvelocity	*fiterwheel actual min velocity*

Returns:

0: success;

0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int GetPorts (char * serialNo)**

list all the possible port on this computer.

Parameters:

serialNo	*port list returned string include serial number and device descriptor, seperated by comma*

Returns:

non-negtive number: number of device in the list; negtive number : failed.

- **DllExport int GetPosition (int hdl, int & pos)**

get the fiterwheel current position.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
pos	*fiterwheel actual position*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int GetPositionCount (int hdl, int & poscount)**

get the fiterwheel current position count.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
poscount	*fiterwheel actual position count*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int GetSensorMode (int hdl, int & sensormode)**

get the fiterwheel current sensor mode.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
sensormode	*fiterwheel actual sensor mode*

Returns:

0: success;

0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int GetSpeed (int hdl, int & speed)**

get the fiterwheel current speed

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
speed	*fiterwheel actual speed*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int GetTimeToCurrentPos (int hdl, int & time)**

get the fiterwheel current sensor mode.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
time	*the time from last position to current position*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int GetTriggerMode (int hdl, int & triggermode)**

get the fiterwheel current trigger mode.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
triggermode	*fiterwheel actual trigger mode*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int IsOpen (char * serialNo)**

check opened status of port

Parameters:

serialNo	*serial number of the device to be checked.*

Returns:

0: port is not opened; 1 : port is opened.

- **DllExport int Open (char * serialNo, int nBaud, int timeout)**

open port function.

Parameters:

serialNo	*serial number of the device to be opened, use GetPorts function to get exist list first.*
nBaud	*bit per second of port*
timeout	*set timeout value in (s)*

Returns:

non-negtive number: hdl number returned successfully; negtive number : failed.

- **DllExport int Save (int hdl)**

save all the settings as default on power up .

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int SetAcceleration (int hdl, int acceleration)**

set fiterwheel's acceleration.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
acceleration	*fiterwheel acceleration*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int SetMaxVelocity (int hdl, int max)**

set fiterwheel's max velocity.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
max	*fiterwheel max velocity*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;

0xEB: time out;
0xED: invalid string buffer;

- **DllExport int SetMinVelocity (int hdl, int min)**

set fiterwheel's min velocity.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
min	*fiterwheel min velocity*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int SetPosition (int hdl, int pos)**

set fiterwheel's position.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
pos	*fiterwheel position*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int SetPositionCount (int hdl, int count)**

set fiterwheel's position count.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
count	*fiterwheel PositionCount*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int SetSensorMode (int hdl, int mode)**

set fiterwheel's sensor mode to mode

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*

mode	*fiterwheel sensor mode*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int SetSpeed (int hdl, int speed)**

set fiterwheel's speed.

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

Parameters:

hdl	*handle of port.*
speed	*fiterwheel speed*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

- **DllExport int SetTriggerMode (int hdl, int mode)**

set fiterwheel's trigger mode .

make sure the port was opened successful before call this function.

make sure this is the correct device by checking the ID string before call this function.

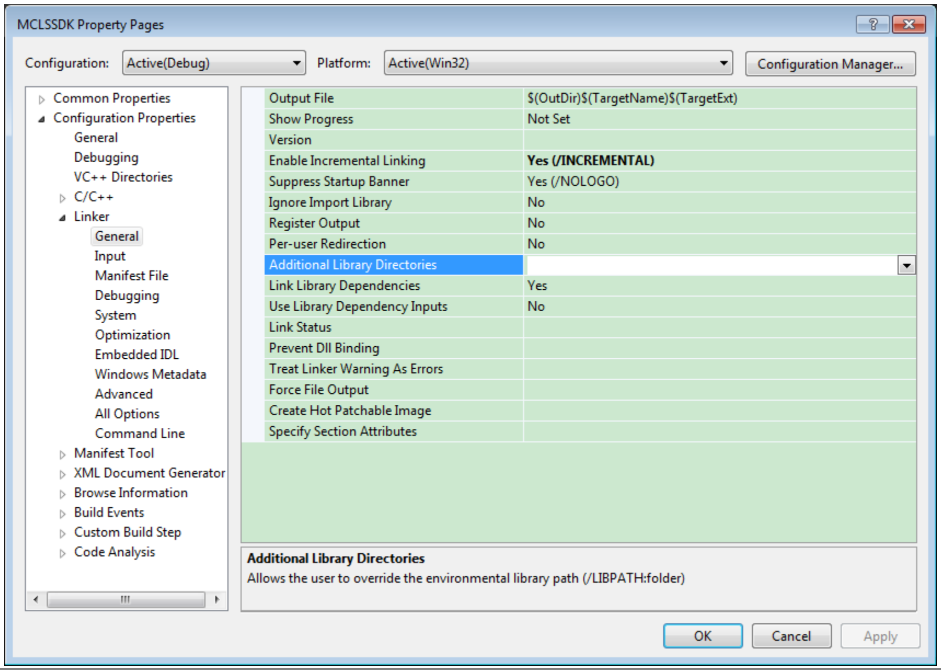
Parameters:

hdl	*handle of port.*
mode	*fiterwheel trigger mode*

Returns:

0: success;
0xEA: CMD_NOT_DEFINED;
0xEB: time out;
0xED: invalid string buffer;

The following example is a reference for configurations:
1. Copy FilterWheel102_win32.dll to your program folder, and make sure your program folder path without any blank space string.
2. Set the Additional Library Directories to your folder which contains FilterWheel102_win32.dll, as seen below.



Configuration: Active(Debug)   Platform: Active(Win32)

Output File	$(OutDir)$(TargetName)$(TargetExt)
Show Progress	Not Set
Version	
Enable Incremental Linking	**Yes (/INCREMENTAL)**
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	
Link Library Dependencies	Yes
Use Library Dependency Inputs	No
Link Status	
Prevent Dll Binding	
Treat Linker Warning As Errors	
Force File Output	
Create Hot Patchable Image	
Specify Section Attributes	

Additional Library Directories
Allows the user to override the environmental library path (/LIBPATH:folder)

Fw102C_Demo.cpp is example code files which you can also find in the sample directory. You can run it for testing and below is the description for it:

Fw102C_Demo.cpp File Reference

Functions

- void GetPos(int hdl)
- void SetPos(int hdl,int pos)
- void GetTriggerMode(int hdl)
- void SetTriMode(int hdl,int mode)
- int Init()
- int _tmain(int argc, _TCHAR* argv[])

Function Documentation

- ### int _tmain(int argc, _TCHAR* argv[])

Main function of Fw102C_Demo.cpp

Parameters:

argc	number of input arguments in command line
argv	arguments string buffer

Returns:

0: succeed;

- ### void GetPos(int hdl)

print Fw102C current position

Parameters:

hdl	handle of port.

- **void SetPos(int hdl,int pos)**

  set Fw102C to pos , print set result

  Parameters:

hdl	handle of port.
pos	*input position*

- **void GetTriggerMode(int hdl)**

  print Fw102C current trigger mode

  Parameters:

hdl	*handle of port.*

- **void SetTriMode(int hdl,int mode)**

  set Fw102C trigger mode, print set result

  Parameters:

hdl	handle of port.
mode	*input trigger mode*

- **int Init()**

  load FilterWheel102.dll

  Returns:

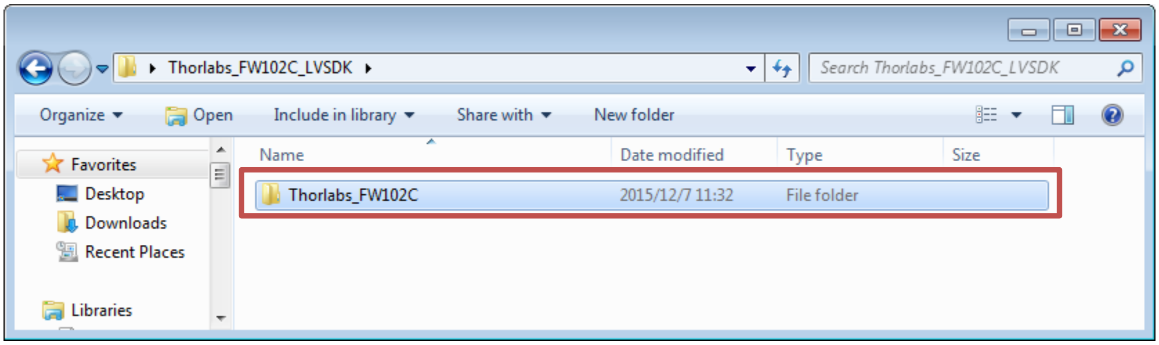
  0: success;
  -1: failed;

Software Development (LabVIEW instrument driver)

The user can start software development with LabVIEW 2011 or later versions based on LabVIEW instrument driver mechanism. The supported files are in *LabVIEW SDK* under the **Sample** directory.

How to install

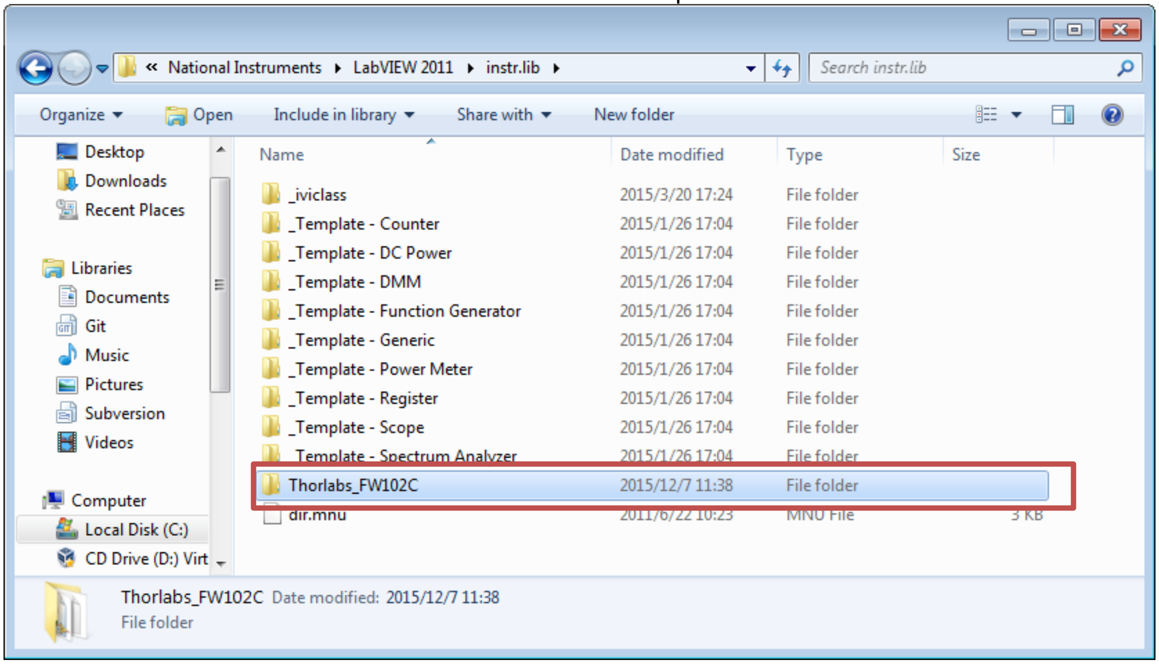
Unzip the zip file and **copy** to instr.lib folder under LabVIEW installation folder.



Name	Date modified	Type	Size
Thorlabs_FW102C	2015/12/7 11:32	File folder	

Destination folder: under %LabVIEW install path%\instr.lib
Typically, C:\Program Files (x86)\National Instruments\LabVIEW 2011\instr.lib

Note: LabVIEW 2011 or later LabVIEW versions are compatible.

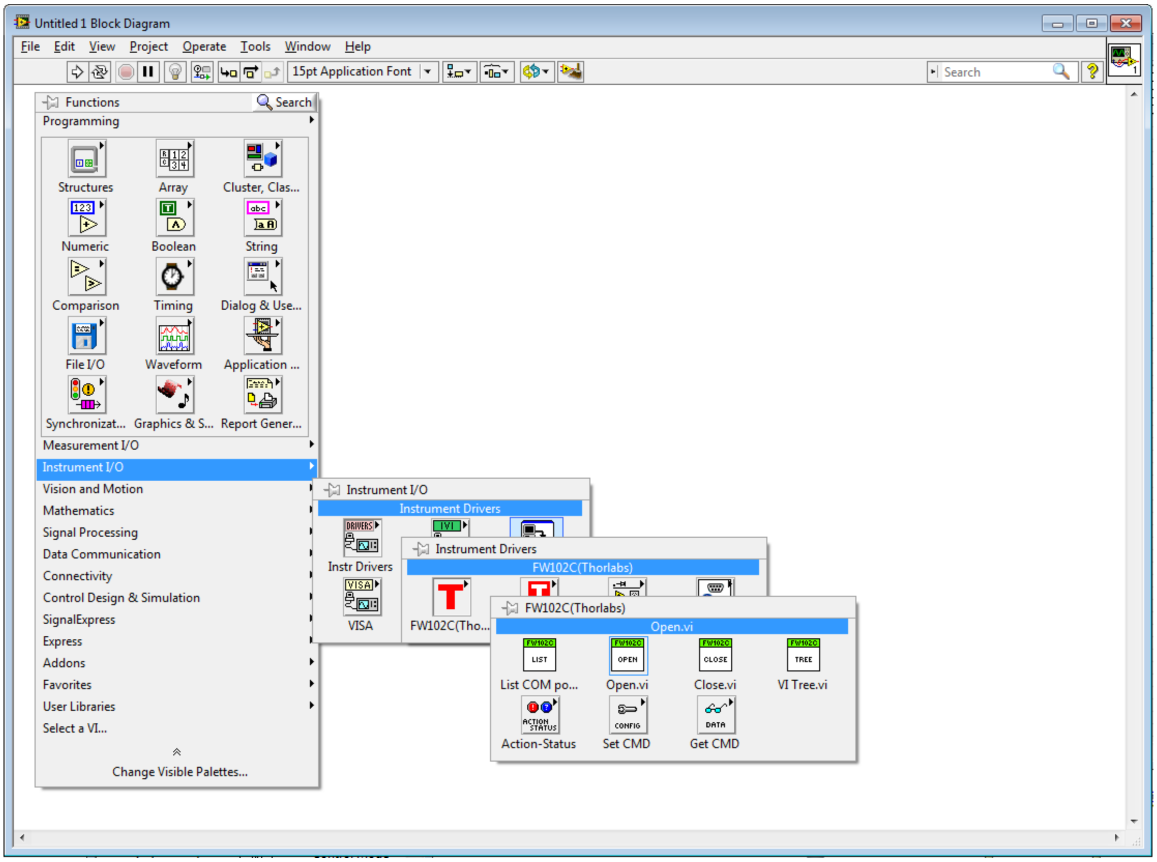


Name	Date modified	Type	Size
_iviclass	2015/3/20 17:24	File folder	
_Template - Counter	2015/1/26 17:04	File folder	
_Template - DC Power	2015/1/26 17:04	File folder	
_Template - DMM	2015/1/26 17:04	File folder	
_Template - Function Generator	2015/1/26 17:04	File folder	
_Template - Generic	2015/1/26 17:04	File folder	
_Template - Power Meter	2015/1/26 17:04	File folder	
_Template - Register	2015/1/26 17:04	File folder	
_Template - Scope	2015/1/26 17:04	File folder	
_Template - Spectrum Analyzer	2015/1/26 17:04	File folder	
Thorlabs_FW102C	2015/12/7 11:38	File folder	
dir.mnu	2011/6/22 10:23	MNU File	3 KB

Thorlabs_FW102C Date modified: 2015/12/7 11:38
File folder

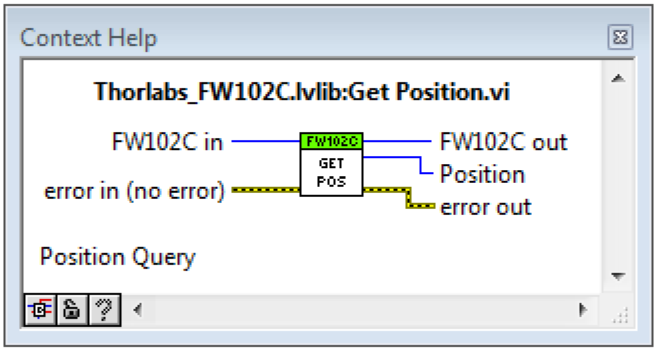
How to find VI

VI Could be found under: Functions\Instrument I/O\Instrument Drivers\



How to use

1. From VI

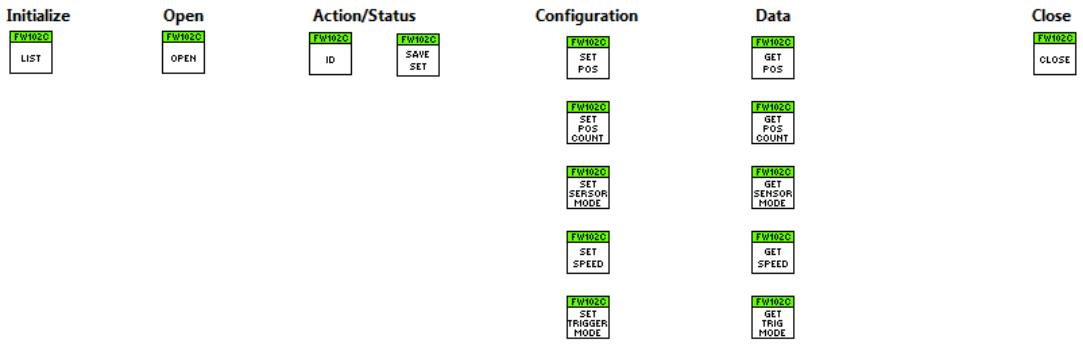


Thorlabs_FW102C.lvlib:Get Position.vi

Position Query

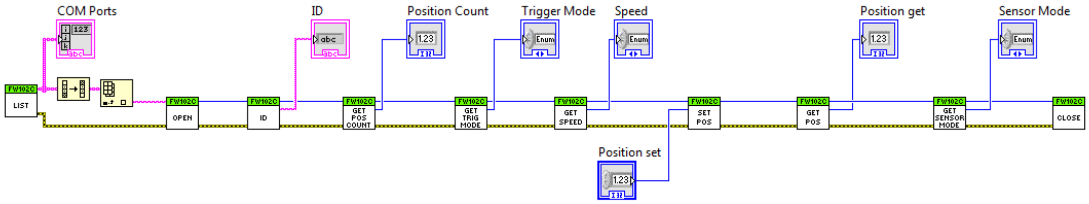
1. From VI tree

Some classic data flow in VI tree.



1. From example

An examples show the classic usage. Example path: instr.lib\Thorlabs_FW102C\Examples



Easy programming and detailed comment will help.