EXCELITAS TECHNOLOGIES®

Optem® FUSION Controller Software Development Manual

Part No. MAN-350014A

APPLIED MICROSCOPY

Part No: MAN-350014A
Status: Released

Excelitas Technologies Corp.
200 West Street, 4th Floor East
Waltham, MA 02451
United States
www.excelitas.com

Phone Europe
+49 (0) 551 6935-0
Phone North America
+1 (800) 429 0257
Phone Asia/Pacific
+65 64 99 7777
Technical support: *Inspection@excelitas.com*

Contents

1 Introduction

Optem® FUSION Overview 12
Optem® FUSION Controller Software Development Manual Installer Components 13
License Information 14
Changes to Excelitas Products 15
Technical Support 15

2 Installing Optem® FUSION Software

Installing the Software 18
Performing the Optem Fusion Setup 18
Performing an Optem Fusion Console Installation 22
Performing an Optem Fusion JSON Server Installation 27
Performing a Software Development Kit Installation 34

3 Optem® FUSION SDK Architecture

Main System Components 40
Server to Optem® FUSION Hardware Communication 41

4 Optem® FUSION Interface Functions

Communication Protocol 44
Single Operation Transaction 44
Devices 48
Methods 48
Parameters and Result Values 49
Batch Operations 49
SDK Functions 50
Devices, Operations, Parameters and Values 51

5 Demo Scripts and Test Examples

Overview 62
Demo Collections 62
Sample Scripts 62
Initialize Controller 62
Enable Illumination 63
Enable Motor 63
Home Motor 63
Check Position After Performing Homing 63
Get PWM Settings 63
Set Light Level to 50% Power 63
Running Demo Scripts 64
Optem Fusion Demo - Postman Script 64
XML Demo Script Tool 69
C# Test Example 72
Running a C# Test Example 72
Locating the C# Test Example Folder 74

List of Figures

Figure 1 Optem Fusion Select Components Window – Setup 19
Figure 2 Ready to Install Window – Setup 19
Figure 3 Installation Progress Window – Setup 20
Figure 4 Installation Complete Window – Setup 21
Figure 5 Select Components Window – Optem Fusion Console 22
Figure 6 Ready to Install Window – Optem Fusion Console 23
Figure 7 Installation progress Window – Optem Fusion Console 23
Figure 8 Select Destination Location Window – Optem Fusion Console 24
Figure 9 Select Additional Tasks Window – Optem Fusion Console 24
Figure 10 Ready to Install Window – Optem Fusion Console 25
Figure 11 Installation Progress Window – Optem Fusion Console 25
Figure 12 Installation Complete Window – Optem Fusion Console 26
Figure 13 Select Components Window – Optem Fusion JSON Server 27
Figure 14 Ready to Install Window – Optem Fusion JSON Server 28
Figure 15 Installation Progress Window – Optem Fusion JSON Server 28
Figure 16 Select Destination Location Window – Optem Fusion JSON Server 29
Figure 17 Select Components Window – Optem Fusion JSON Server 30
Figure 18 Share Installation Directory Window – Optem Fusion JSON Server 31
Figure 19 Optem Fusion JSON Server (Optional) Window – Optem Fusion JSON Server 32
Figure 20 Ready to Install Window – Optem Fusion JSON Server 32
Figure 21 Installation Progress Window – Optem Fusion JSON Server 33
Figure 22 Installation Complete Window – Optem Fusion JSON Server 33
Figure 23 Optem® FUSION SDK Select Components Window – SDK 34
Figure 24 Ready to Install Window – SDK 35
Figure 25 Installation Progress Window – SDK 35
Figure 26 Select Destination Location Window – SDK 36
Figure 27 Optem Fusion JSON Server (Optional) Window – SDK 37
Figure 28 Ready to Install Window – SDK 37
Figure 29 Installation Progress Window – SDK 38
Figure 30 Installation Complete Window – SDK 38
Figure 31 Client to Server Communication Interface 40
Figure 32 Method Examples 45
Figure 33 Parameter Examples 45
Figure 34 Start menu – Optem Fusion Demo - Postman Script 65
Figure 35 Optem Fusion Demo - Postman Script Terminal Window – Step 1 66
Figure 36 Optem Fusion Demo - Postman Script Terminal Window – Step 2 66

Figure 37 Demoscript Fail Prompt 67
Figure 38 Newman Install Prompt Window 67
Figure 39 Optem® FUSION Demo Script Terminal Window – Running 68
Figure 40 Optem® FUSION Demo Script Terminal Window – Finished 68
Figure 41 Start menu – Optem Fusion Demo - XML Script 70
Figure 42 XML Script Demo Window 71
Figure 43 XML script file folder 71
Figure 44 Start menu – Optem® FUSION C# Test Examples 72
Figure 45 C# Test Example Executable File Folders 73
Figure 46 C# Test Example Folder Location 74
Figure 47 C# Test Example Folder Contents 74

List of Tables

Table 1 Web Server Error Codes 46
Table 2 SDK Result Error Codes 47
Table 3 Protocol Methods 48
Table 4 JSON Devices and Supported Interfaces 51
Table 5 IMotion Parameters 52
Table 6 IMotion Operations 54
Table 7 IIlluminate Parameters 55
Table 8 ITunableLens Parameters 56
Table 9 IIO Parameters 57
Table 10 IController Parameters 58
Table 11 IController Operations 59
Table 12 IController Configuration 59

Revision History

Revision Date	Rev#	Issued By	Change Details
26-September-2023	A	Carmelo Scaffidi	First release.

List of Acronyms

CCW	Counter-Clock Wise (for stepper motor)
CW	Clock Wise (for stepper motor)
CW	Continuous Wave (for LED)
CRC	Cyclic Redundancy Check
DI	Digital Input
IIS	Internet Information Services
LED	Light Emitting Diode
OEM	Original Equipment Manufacturer
PFABUS	Precision Focus Automation Bus
PFABUS MFC	PFABUS Multi-Functional Controller
PWM	Power Modulation
SDK	Software Development Kit

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER

1

Introduction

This chapter describes all the features and functions available in the Optem® FUSION Controller Software Development Manual.

The following topics are covered:

- Optem® FUSION Overview, pg. 12
- *Optem® FUSION Controller Software Development Manual Installer Components, pg. 13*
  - License Information, pg. 14
- Changes to Excelitas Products, pg. 15
- Technical Support, pg. 15

Optem® FUSION Overview

The Optem® FUSION hardware is designed to be integrated with the customer's hardware.

In order to avoid dependency on customer's platform, the Excelitas software is designed as a service running on a computer with a Windows operating system, Windows 10 or newer.

The client communicates with the server using the JSON-RPC over HTTP protocol and is compliant to the standard described in "Communication Protocol" on page 44.

The Optem® FUSION Controller Software Development Manual is used to configure and control the Optem® FUSION hardware. This reference guide describes all the features and functions available in the Optem® FUSION Controller Software Development Manual. It does not describe how to operate the Optem® FUSION hardware. For detailed Optem® FUSION hardware configuration and operation procedures, refer to the *MAN-350013 Optem® FUSION Controller User Manual*.

Using the Optem® FUSION Controller Software Development Manual, you can perform the following main functions:

- Control Optem® FUSION System Controller

NOTE: *This can also be performed using the Optem® FUSION Console, which is installed in the Optem® FUSION setup, as part of the complete installation or as a stand alone installation.*

The Optem® FUSION Controller Software Development Manual is intended to be used by software engineers who have a good working knowledge of the hardware.

WARNING! *When operating the hardware, be sure to observe all safety precautions provided in the MAN-350013 Optem® FUSION Controller User Manual.*

Optem® FUSION Controller Software Development Manual Installer Components

The Optem® FUSION Controller Software Development Manual installer includes the following Excelitas components:

- Optem Fusion Console
- Optem Fusion SDK
  - SDK Documentation
  - SDK Demo Samples
- Optem Fusion Server

The SDK installer also includes the following:

- Microsoft Visual C++ 2015 Runtime Library 32-bit
- Microsoft Visual C++ 2015 Runtime Library 64-bit
- Microsoft Visual C++ 2019 Runtime Library 64-bit
- Node.js 10.15.3 32-bit
- Node.js 10.15.3 64-bit
- newman 5.3.2
- newman-reporter-htmlextra 1.7.2

The SDK was written using the following dependencies:

- InnoSetup 5.5.4
- TinyXML 2.6.2
- jsonrpc-lean commit 534a9cd
- rapidjson 1.1.0
- curl 7.65.1
- NModBus4 2.1.0.0
- libmodbus 3.16

C# example dependencies:

- Newtonsoft.Json.12.0.3
- RestSharp.106.11.4

License Information

Redistributable license information:

- Microsoft Visual C++ 2015 Runtime Library:
  - *https://docs.microsoft.com/en-us/visualstudio/productinfo/2015-redistribution-vs*
  - *https://visualstudio.microsoft.com/license-terms/mt171576/*
- Microsoft Visual C++ 2019 Runtime Library:
  - *https://visualstudio.microsoft.com/license-terms/*
- Node.js 10.15.3:
  - *https://raw.githubusercontent.com/nodejs/node/master/LICENSE*
- newman 5.3.2:
  - *https://github.com/postmanlabs/newman/blob/HEAD/LICENSE.md*
- newman-reporter-htmlextra 1.7.2:
  - *https://github.com/DannyDainton/newman-reporter-htmlextra/blob/HEAD/LICENSE.md*

C# example dependency license information:

- Newtonsoft.Json.12.0.3:
  - *https://github.com/JamesNK/Newtonsoft.Json/blob/master/LICENSE.md*
- RestSharp.106.11.4:
  - *https://github.com/restsharp/RestSharp/blob/dev/LICENSE.txt*

Source code dependency license information:

- TinyXML 2.6.2:
  - *http://www.grinninglizard.com/tinyxmldocs/*
  - *https://opensource.org/licenses/Zlib*
- InnoSetup 5.5.4:
  - *http://www.jrsoftware.org/ishelp/*
- jsonrpc-lean commit 534a9cd:
  - *https://github.com/uskr/jsonrpc-lean/blob/master/LICENSE.txt*
- rapidjson 1.1.0:
  - *https://github.com/Tencent/rapidjson/blob/master/license.txt*
- curl 7.65.1:
  - *https://curl.haxx.se/docs/copyright.html*
- NModBus4 2.1.0.0:
  - *https://github.com/NModbus4/NModbus4/blob/2.1.0/LICENSE.txt*
- libmodbus 3.1.6:
  - *https://github.com/stephane/libmodbus/blob/v3.1.6/COPYING.LESSER*

Changes to Excelitas Products

Excelitas reserves the right to improve, change, or modify products without incurring any obligations to make changes to previous Excelitas equipment.

Technical Support

For technical support, please contact our Excelitas customer support team at *Inspection@excelitas.com*.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER

2

Installing Optem® FUSION Software

This chapter describes how to install the Optem® FUSION software.

The following topics are covered:

- Installing the Software, pg. 18
  - Performing the Optem Fusion Setup, pg. 18
  - Performing an Optem Fusion Console Installation, pg. 22
  - Performing an Optem Fusion JSON Server Installation, pg. 27
  - Performing a Software Development Kit Installation, pg. 34

Installing the Software

The Optem® FUSION SDK is installed on the workstation. If you need to reinstall or update the software for some reason, you can follow the instructions provided in this section.

NOTE: *The installation auto creates a new folder for console and gives the option to uninstall or exit the software if a preexisting install is already there.*

By default, the software is installed in the **C:\Program Files\WdiDevice\Optem Fusion** folder. In the Optem Fusion folder, each component is installed in their respective folder, for example:

- Optem Fusion Console.x.x.x.xxx
- Optem Fusion SDK.x.x.x.xxxx
- Optem Fusion Server.x.x.x.xxxx

NOTE: *The third party components (i.e. run-time libraries, web server components) may affect different directories.*

The Optem® FUSION Software package is distributed as a .zip archive file. The installer features a standard Microsoft Windows setup interface, which is simple and easy to use.

You have the option of doing a full installation (see "Performing the Optem Fusion Setup" on page 18) or a console only installation (see "Performing an Optem Fusion Console Installation" on page 22) or a server only installation (see "Performing an Optem Fusion JSON Server Installation" on page 27) or a SDK only installation (see "Performing a Software Development Kit Installation" on page 34).

Performing the Optem Fusion Setup

To perform the Optem Fusion setup:

1. Extract the files from the .zip file into any folder on the workstation's PC.
2. Double-click **Optem Fusion Setup #.#.#.###.exe**.

3. On the **Select Components** window, select all three check boxes or any component you wish to install, and click **Next** to proceed with the installation.

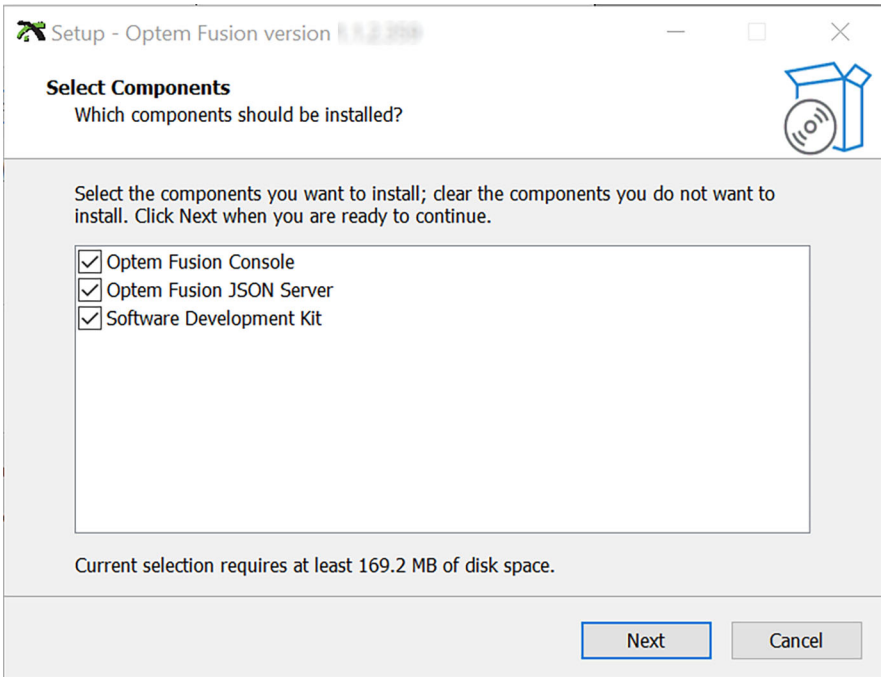


Figure 1 *Optem Fusion Select Components Window – Setup*

4. In the **Ready to Install** window, click **Install**. The installation begins.

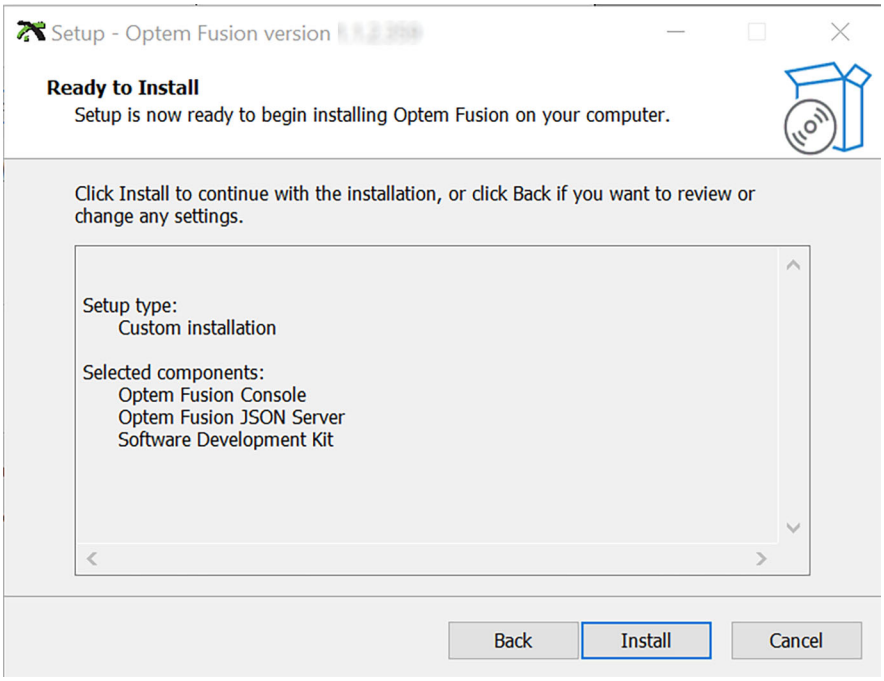


Figure 2 *Ready to Install Window – Setup*

5. Wait until the **Installing** window has finished.

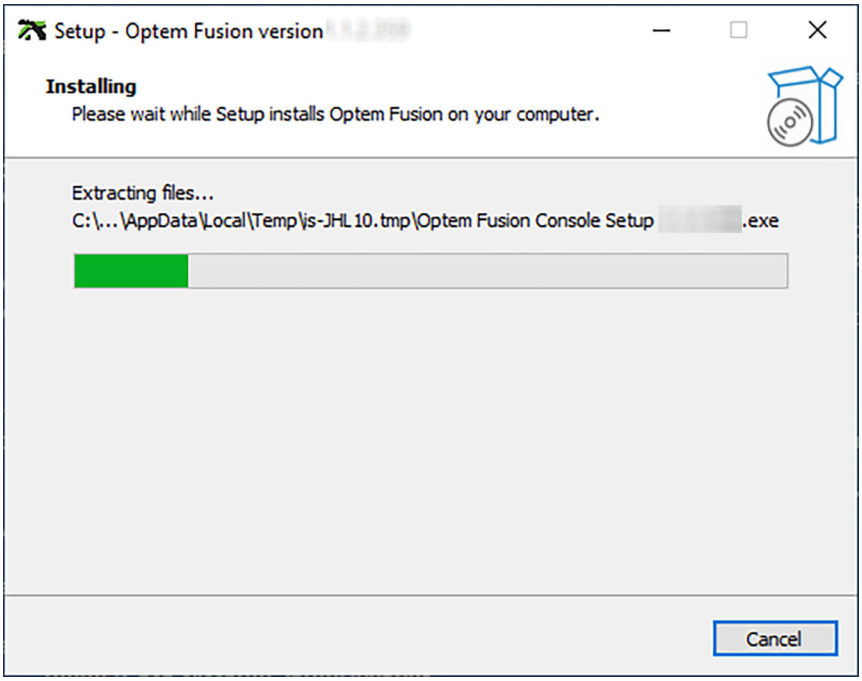


Figure 3 *Installation Progress Window – Setup*

6. If you want to continue with the full setup, perform step 5 to step 7 of "Performing an Optem Fusion Console Installation" on page 22, then step 5 to step 11 of "Performing an Optem Fusion JSON Server Installation" on page 27, and then step 5 to step 8 of "Performing a Software Development Kit Installation" on page 34, and then continue to step 7 of this procedure.

   If you want to install only the console, perform "Performing an Optem Fusion Console Installation" on page 22, and then continue to step 7 of this procedure.

   If you want to install only the server, perform "Performing an Optem Fusion JSON Server Installation" on page 27, and then continue to step 7 of this procedure.

   If you want to install only the SDK, perform "Performing a Software Development Kit Installation" on page 34, and then continue to step 7 of this procedure.

7. When the installation is complete, click **Finish**.

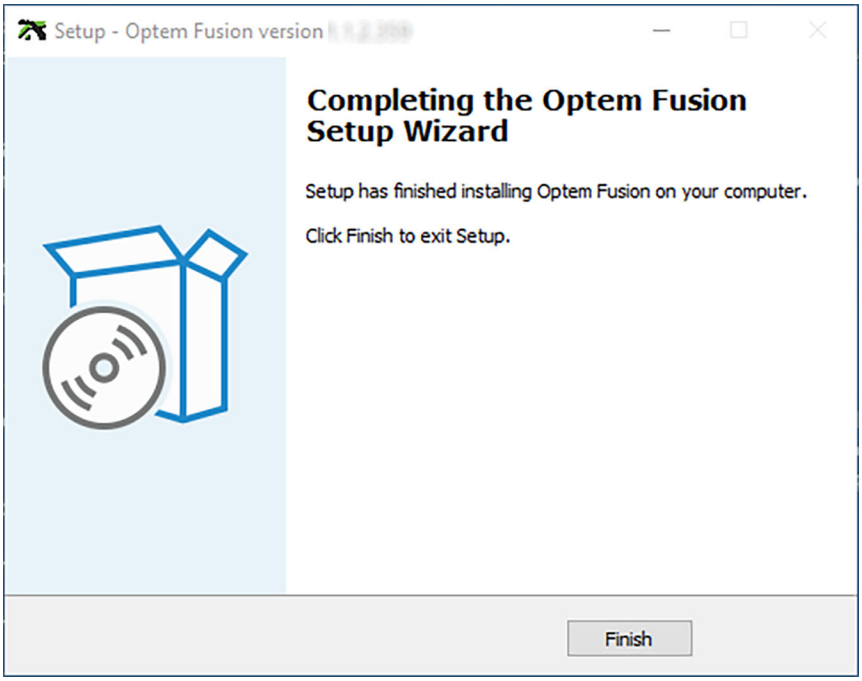


Figure 4 *Installation Complete Window – Setup*

Performing an Optem Fusion Console Installation

To perform an Optem Fusion Console installation:

1. Perform step 1 to step 2 of "Performing the Optem Fusion Setup" on page 18.
2. On the **Select Components** window, select **Optem Fusion Console** check box and click **Next** to proceed with the installation.

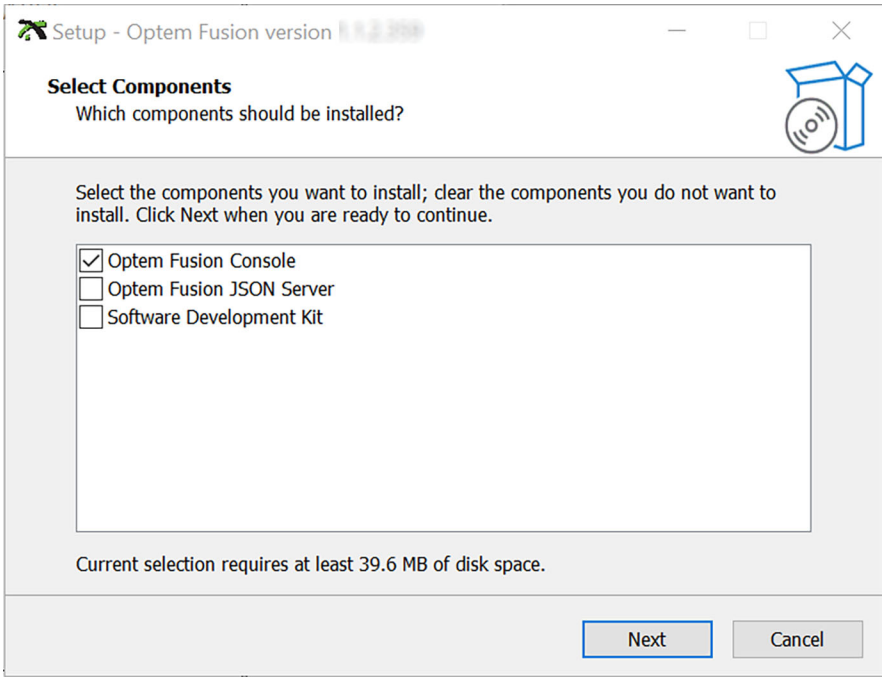


Figure 5 *Select Components Window – Optem Fusion Console*

3. In the **Ready to Install** window, click **Install**. The installation begins.

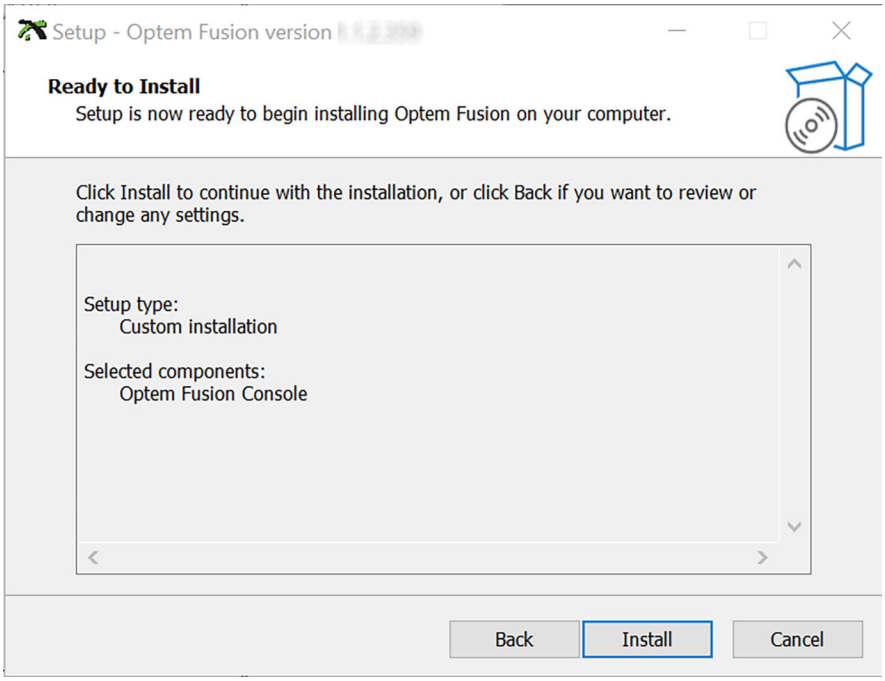


Figure 6 *Ready to Install Window – Optem Fusion Console*

4. Wait until the **Installing** window has finished.

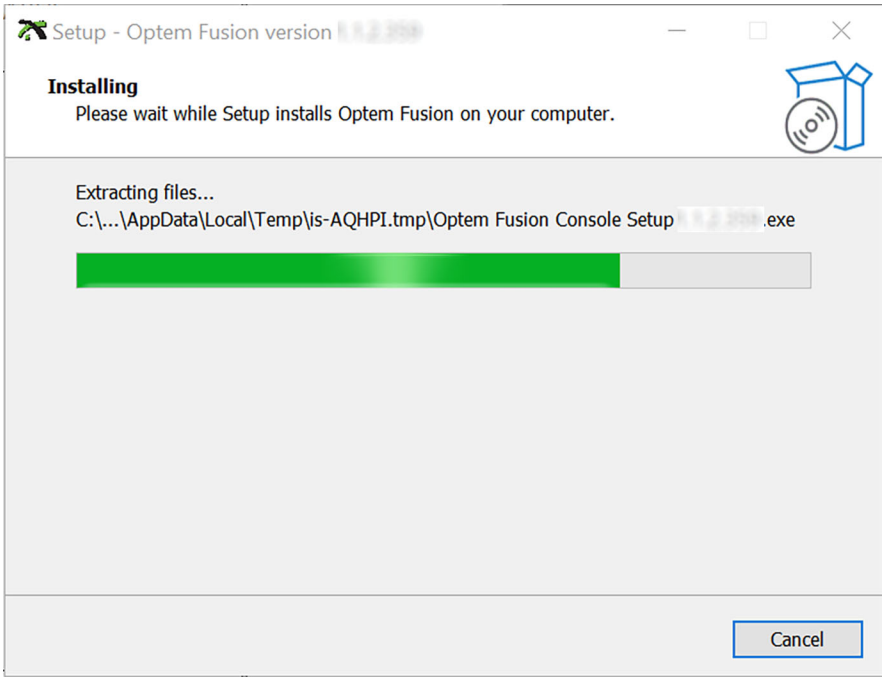


Figure 7 *Installation progress Window – Optem Fusion Console*

5. In the Console **Select Destination Location** window, browse to select an installation folder that is different from the current location of the software, and then click **Next**.

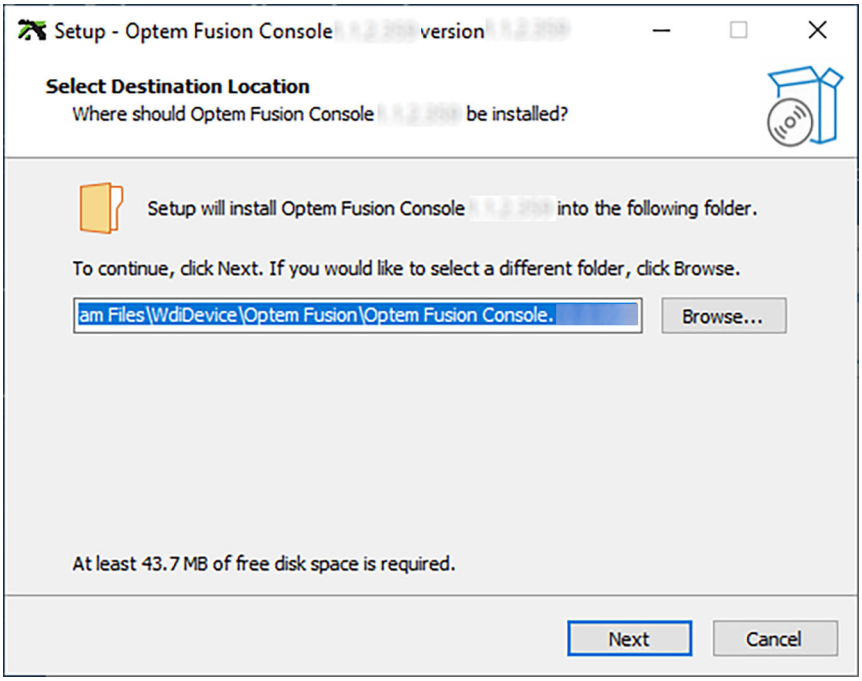


Figure 8 *Select Destination Location Window – Optem Fusion Console*

6. In the Console **Select Additional Tasks** window, if you want the setup program to install an icon on the desktop, select the **Create a desktop icon** check box, and then click **Next**.

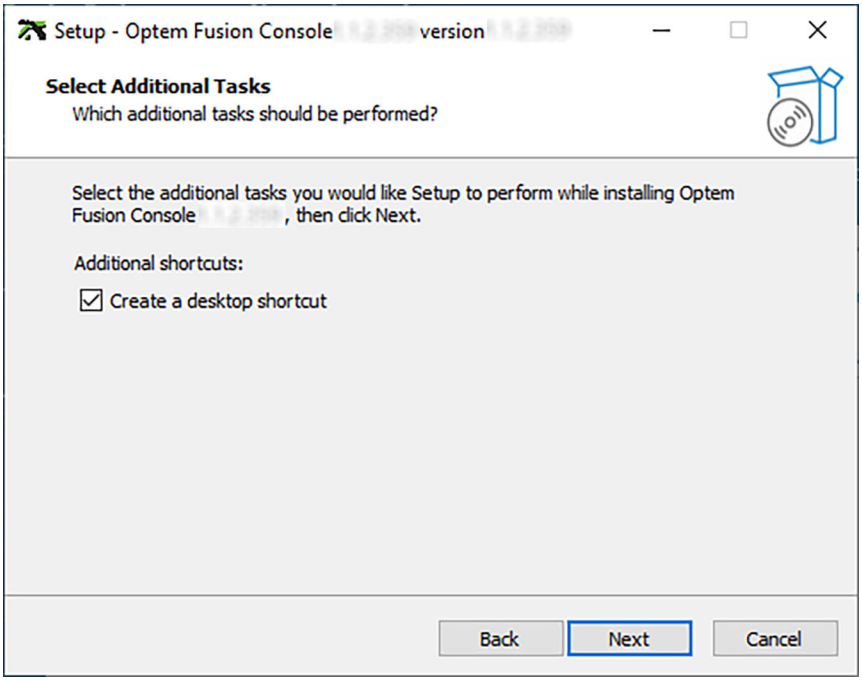


Figure 9 *Select Additional Tasks Window – Optem Fusion Console*

7. In the Console **Ready to Install** window, click **Install**. The installation begins.

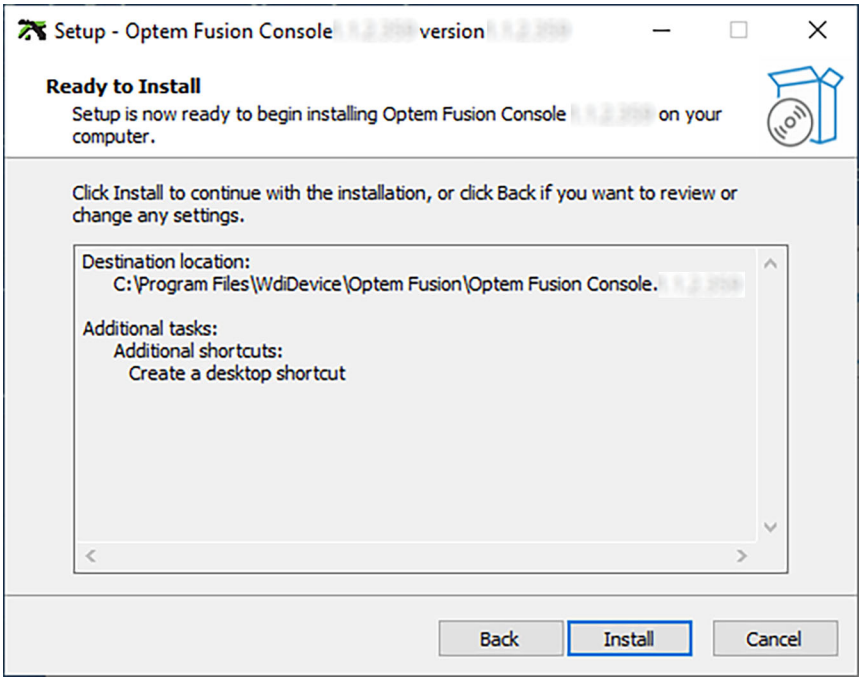


Figure 10 *Ready to Install Window – Optem Fusion Console*

8. Wait until the Console **Installing** window has finished.

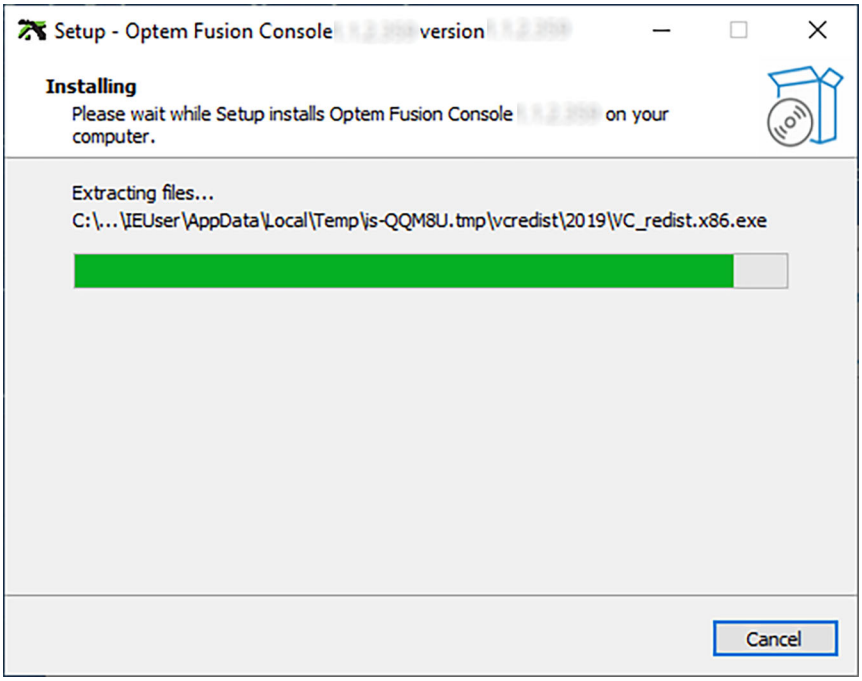


Figure 11 *Installation Progress Window – Optem Fusion Console*

9. When the Console installation is complete, click **Finish**.

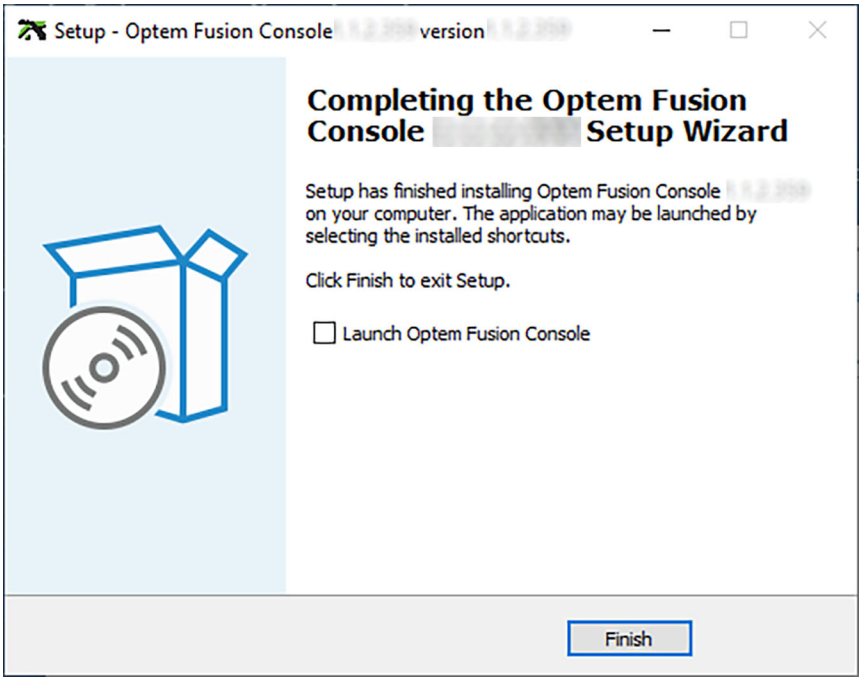


Figure 12 *Installation Complete Window – Optem Fusion Console*

Performing an Optem Fusion JSON Server Installation

To perform an Optem Fusion JSON Server installation:

1. Perform step 1 to step 2 of "Performing the Optem Fusion Setup" on page 18.
2. On the **Select Components** window, select **Optem Fusion JSON Server** check box and click **Next** to proceed with the installation.

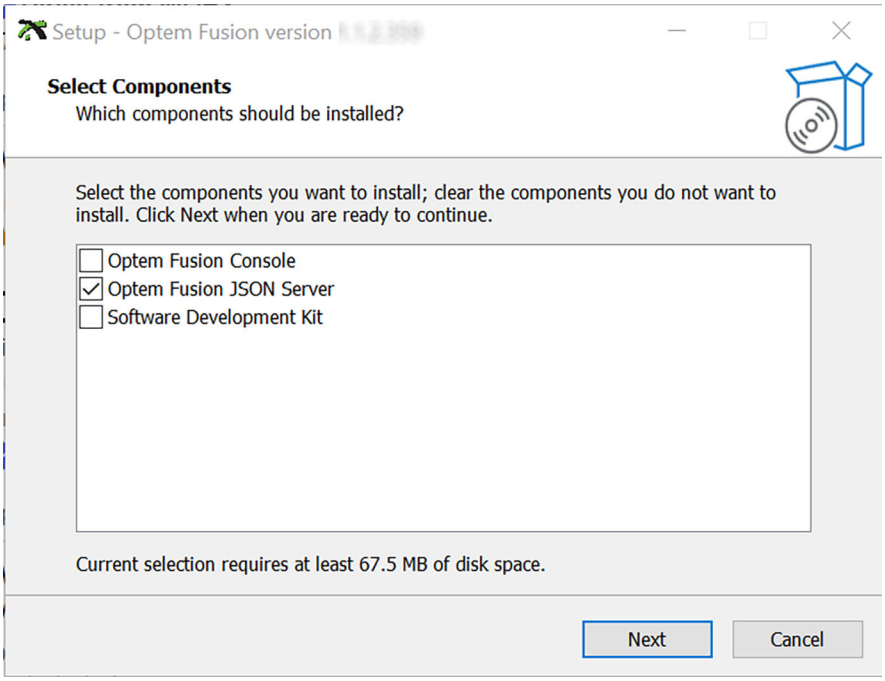


Figure 13 *Select Components Window – Optem Fusion JSON Server*

3. In the **Ready to Install** window, click **Install**. The installation begins.

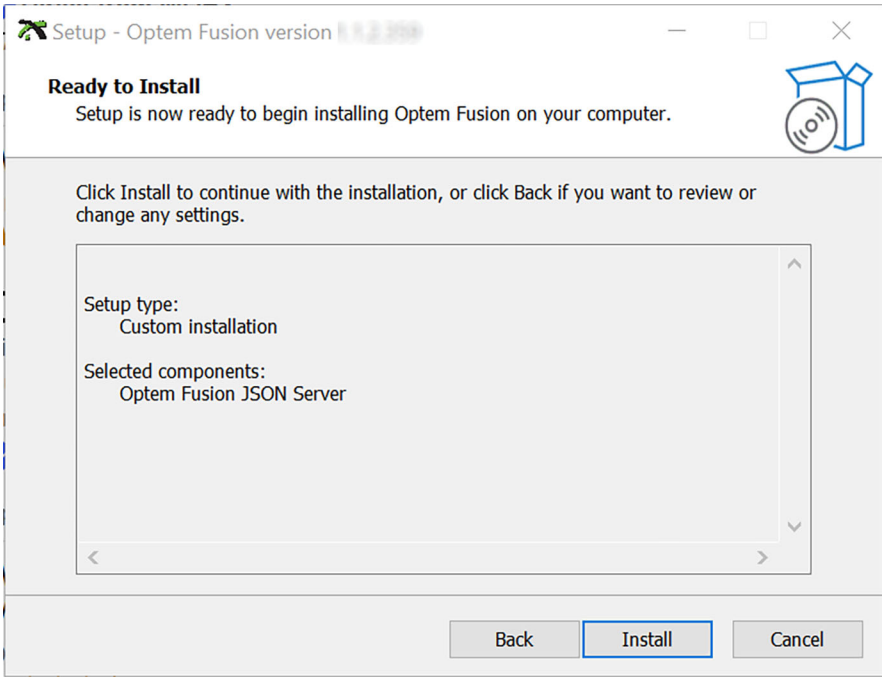


Figure 14 *Ready to Install Window – Optem Fusion JSON Server*

4. Wait until the **Installing** window has finished.

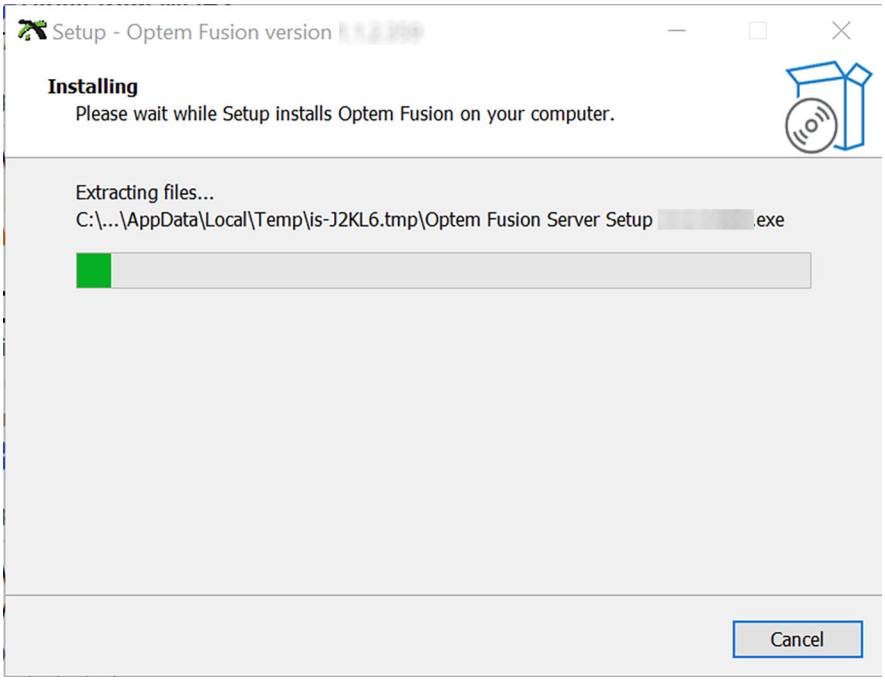


Figure 15 *Installation Progress Window – Optem Fusion JSON Server*

5. In the Server **Select Destination Location** window, browse to select an installation folder that is different from the current location of the software, and then click **Next**.

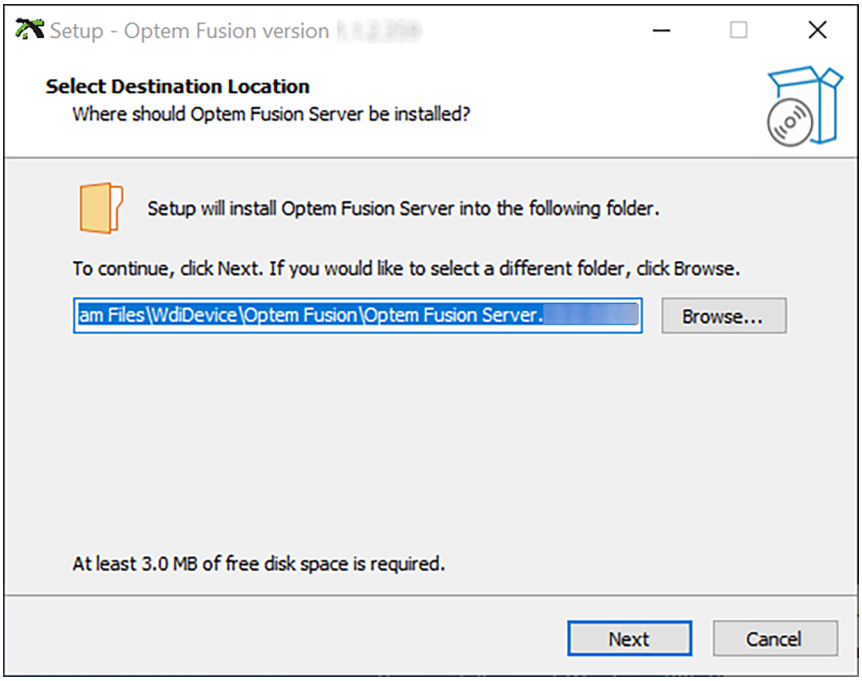


Figure 16 *Select Destination Location Window – Optem Fusion JSON Server*

6. In the Server **Select Components** window, do one of the following:
   - If you want to share the installation directory, click the **Share installation folder** check box, and then click **Next**..
   - If you do not want to share the installation directory, do not click the **Share installation folder** check box, and then click **Next**.

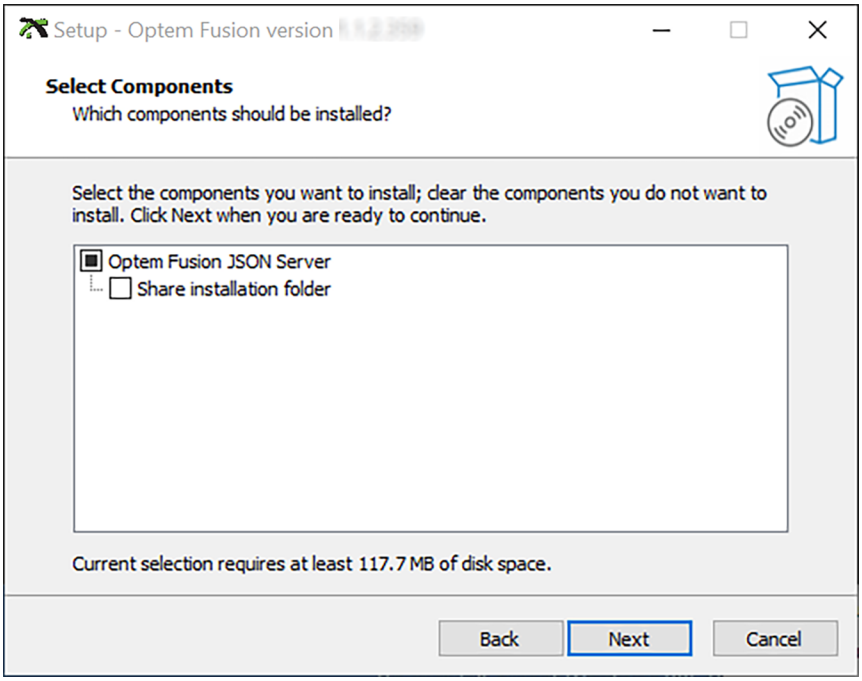


Figure 17 *Select Components Window – Optem Fusion JSON Server*

7. In the Server **Share Installation directory** window, do one of the following:
   - If you want to share the installation directory, type a directory path in the **Share Name** text box, and then click **Next**.
   - If you do not want to share the installation directory, leave the **Share Name** text box empty, and then click **Next**.

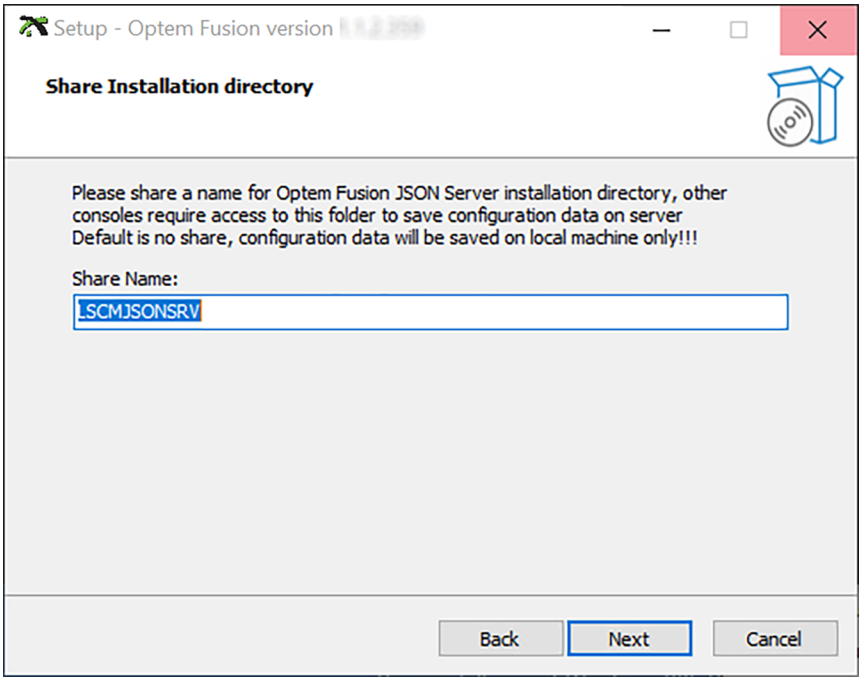


Figure 18 *Share Installation Directory Window – Optem Fusion JSON Server*

8. In the Server **Optem Fusion JSON Server (Optional)** window, type the port number in the **IIS Port Number** text box, and then click **Next**.

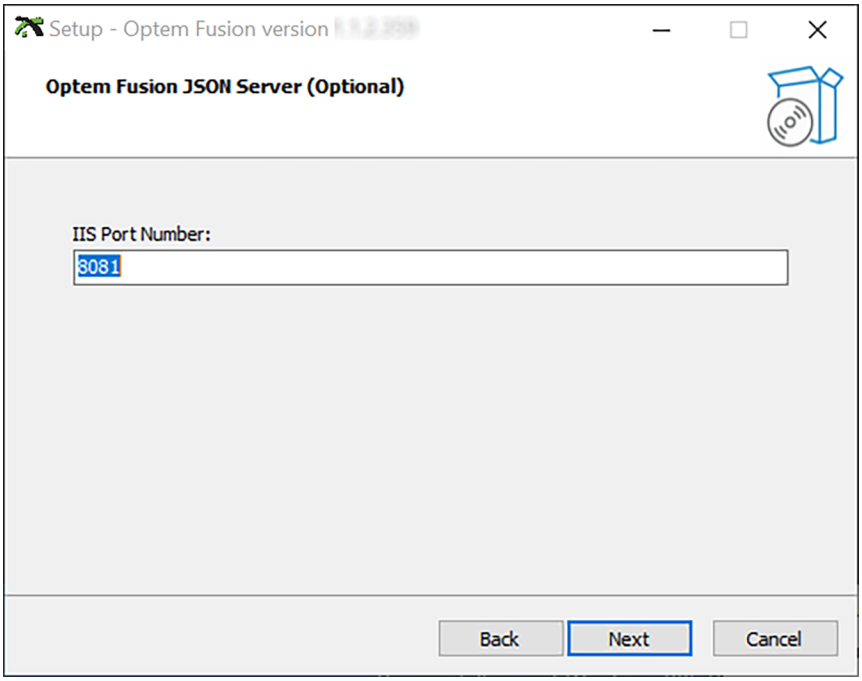


Figure 19 *Optem Fusion JSON Server (Optional) Window – Optem Fusion JSON Server*

9. In the Server **Ready to Install** window, click **Install**. The installation begins.

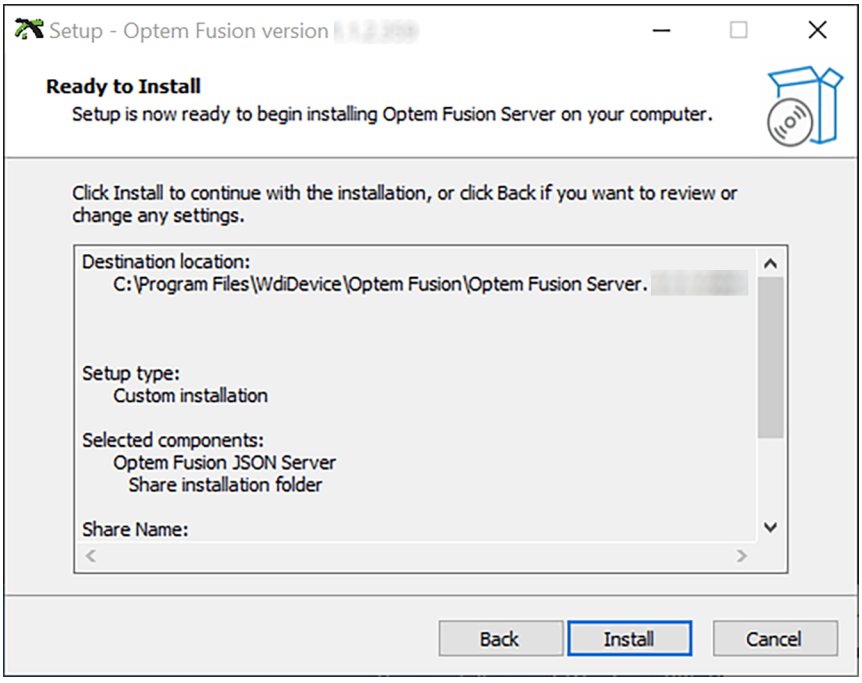


Figure 20 *Ready to Install Window – Optem Fusion JSON Server*

10. Wait until the Server **Installing** window has finished.

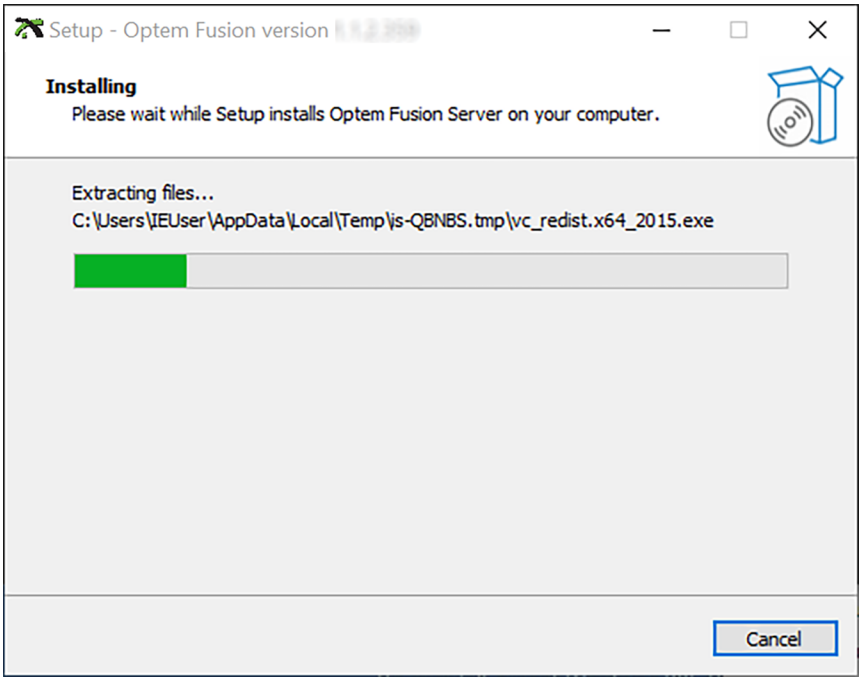


Figure 21 *Installation Progress Window – Optem Fusion JSON Server*

11. When the Server installation is complete, click **Finish**.

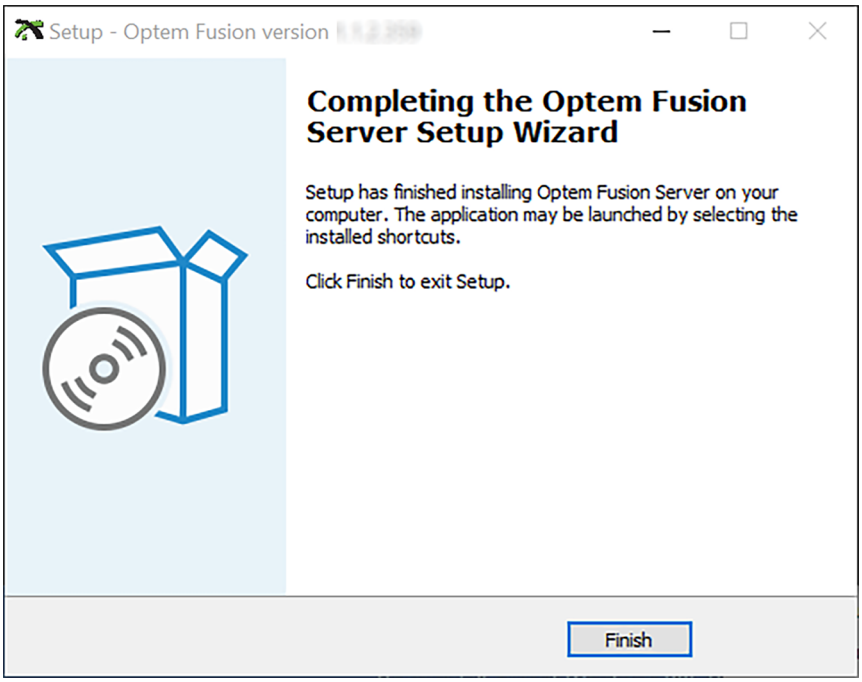


Figure 22 *Installation Complete Window – Optem Fusion JSON Server*

Performing a Software Development Kit Installation

To perform a Software Development Kit installation:

1. Perform step 1 to step 2 of "Performing the Optem Fusion Setup" on page 18.
2. On the **Select Components** window, select **Software Development Kit** check box and click **Next** to proceed with the installation.

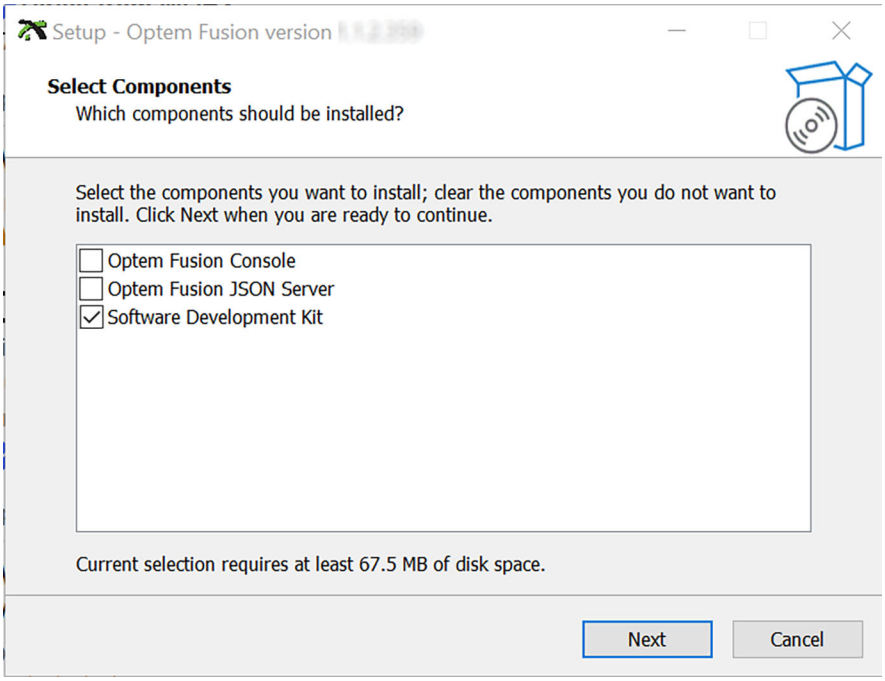


Figure 23 *Optem® FUSION SDK Select Components Window – SDK*

3. In the **Ready to Install** window, click **Install**. The installation begins.

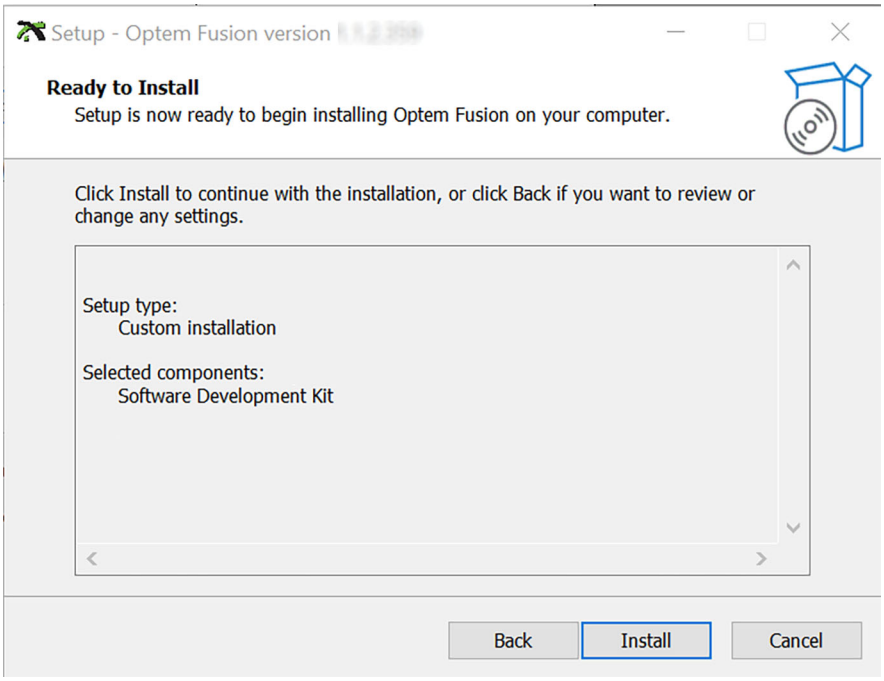


Figure 24 *Ready to Install Window – SDK*

4. Wait until the **Installing** window has finished.

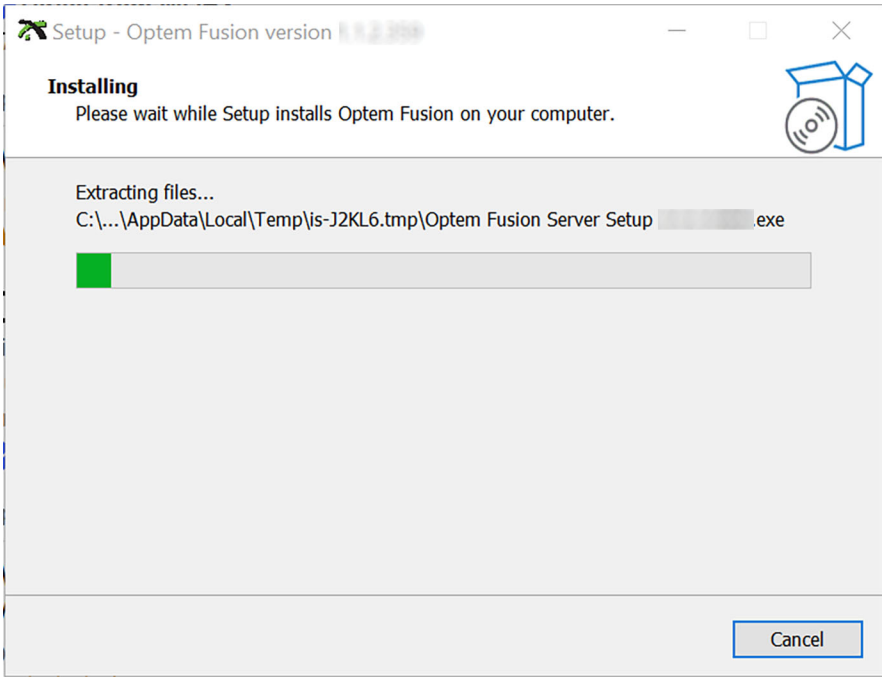


Figure 25 *Installation Progress Window – SDK*

5. In the SDK **Select Destination Location** window, browse to select an installation folder that is different from the current location of the software, and then click **Next**.

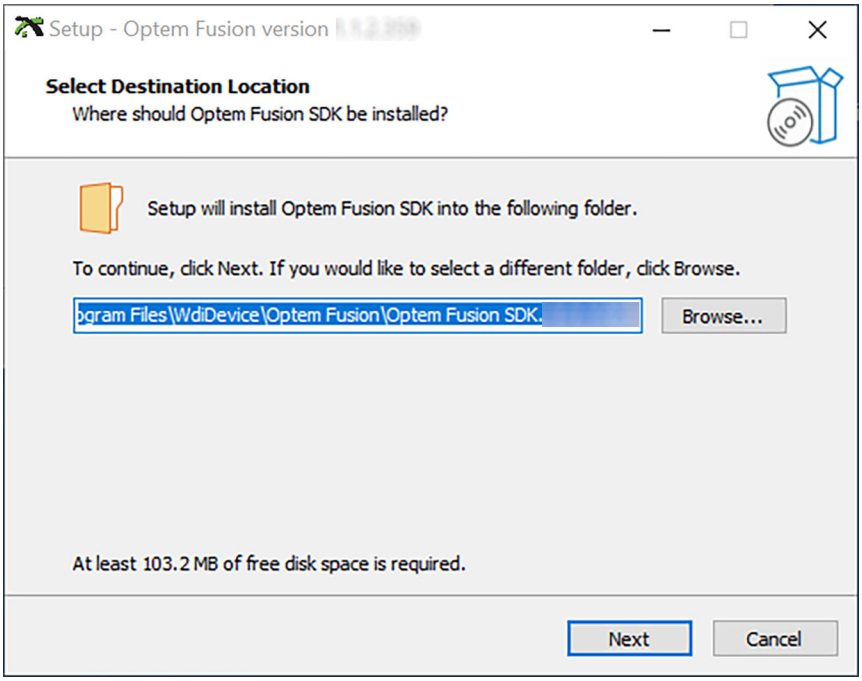


Figure 26 *Select Destination Location Window – SDK*

6. In the SDK **Optem Fusion JSON Server (Optional)** window, type the port number in the **IIS Port Number** text box, and then click **Next**.

NOTE: *The port must be the same one that was used for the server installation.*

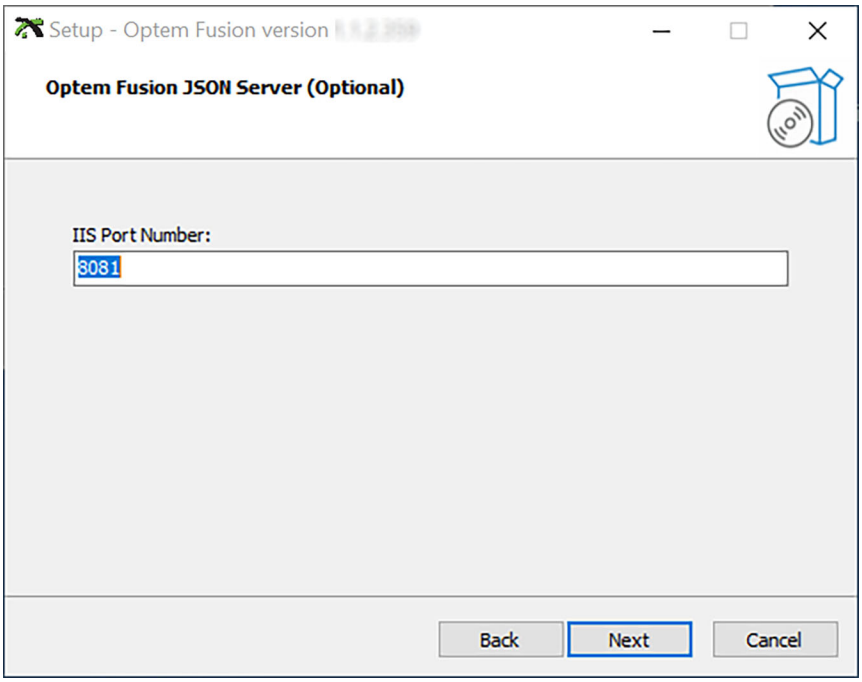


Figure 27 *Optem Fusion JSON Server (Optional) Window – SDK*

7. In the SDK **Ready to Install** window, click **Install**. The installation begins.

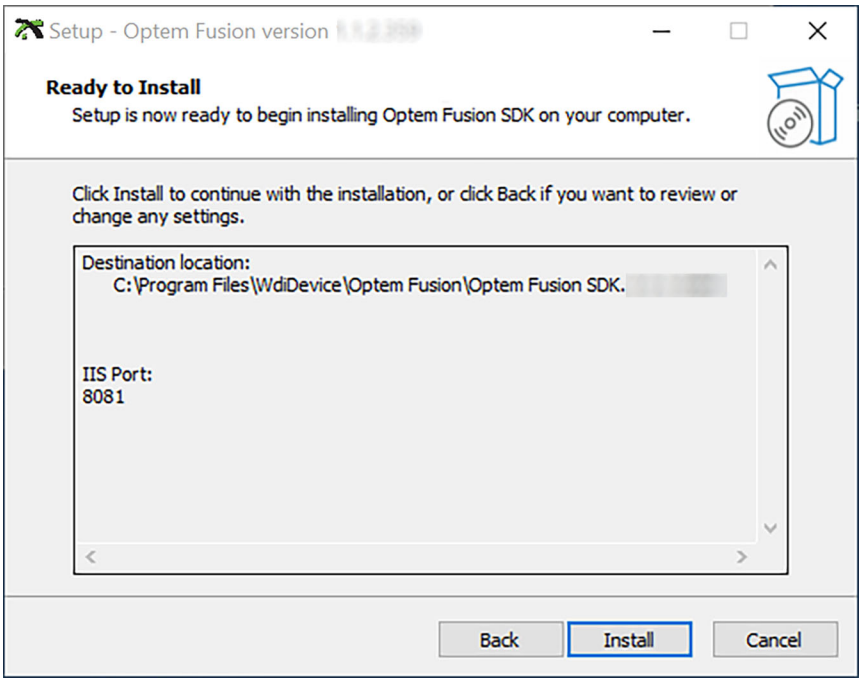


Figure 28 *Ready to Install Window – SDK*

8. Wait until the SDK **Installing** window has finished.

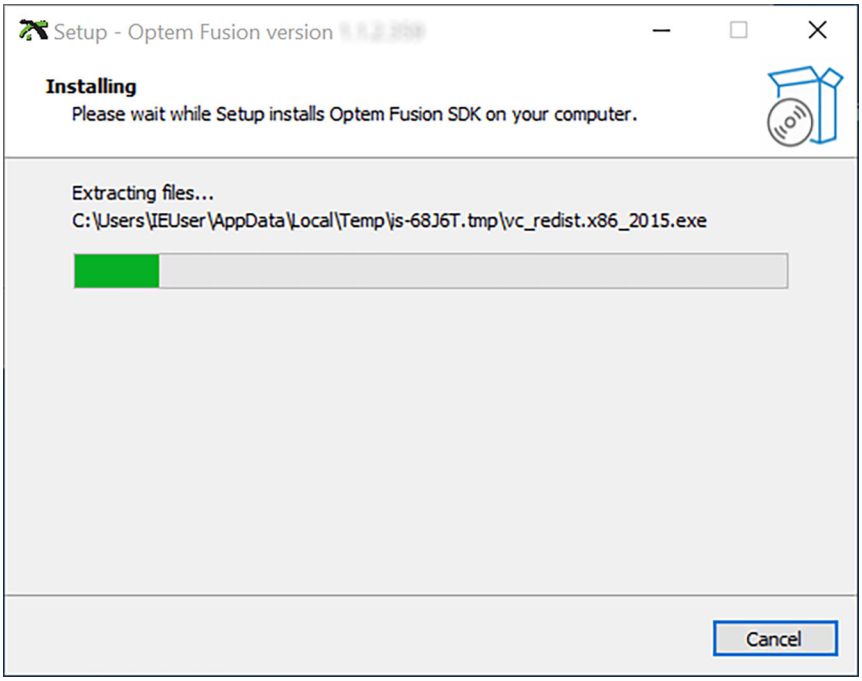


Figure 29 *Installation Progress Window – SDK*

9. When the SDK installation is complete, click **Finish**.

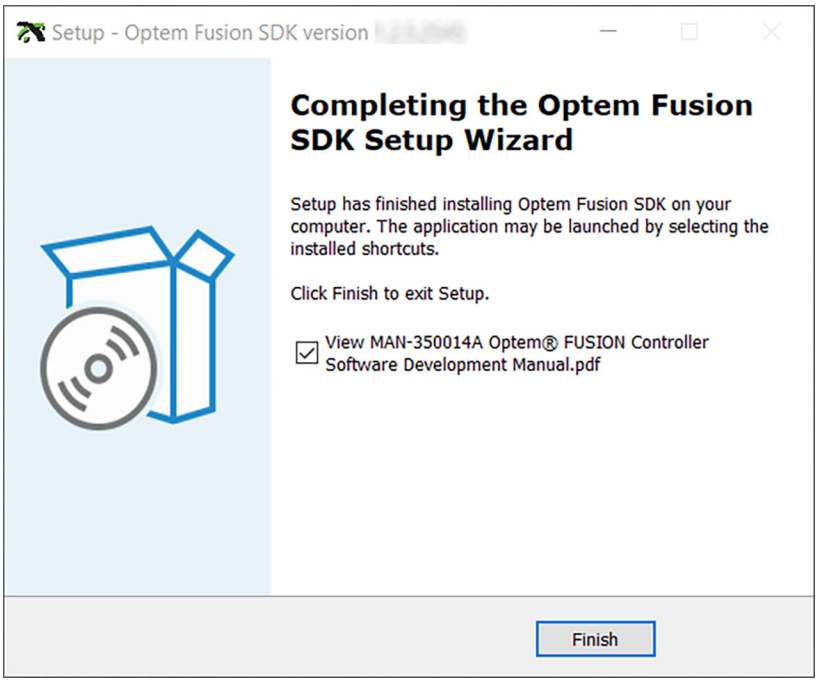


Figure 30 *Installation Complete Window – SDK*

CHAPTER

3

Optem® FUSION SDK Architecture

The chapter provides a general description of the Optem® FUSION SDK architecture.

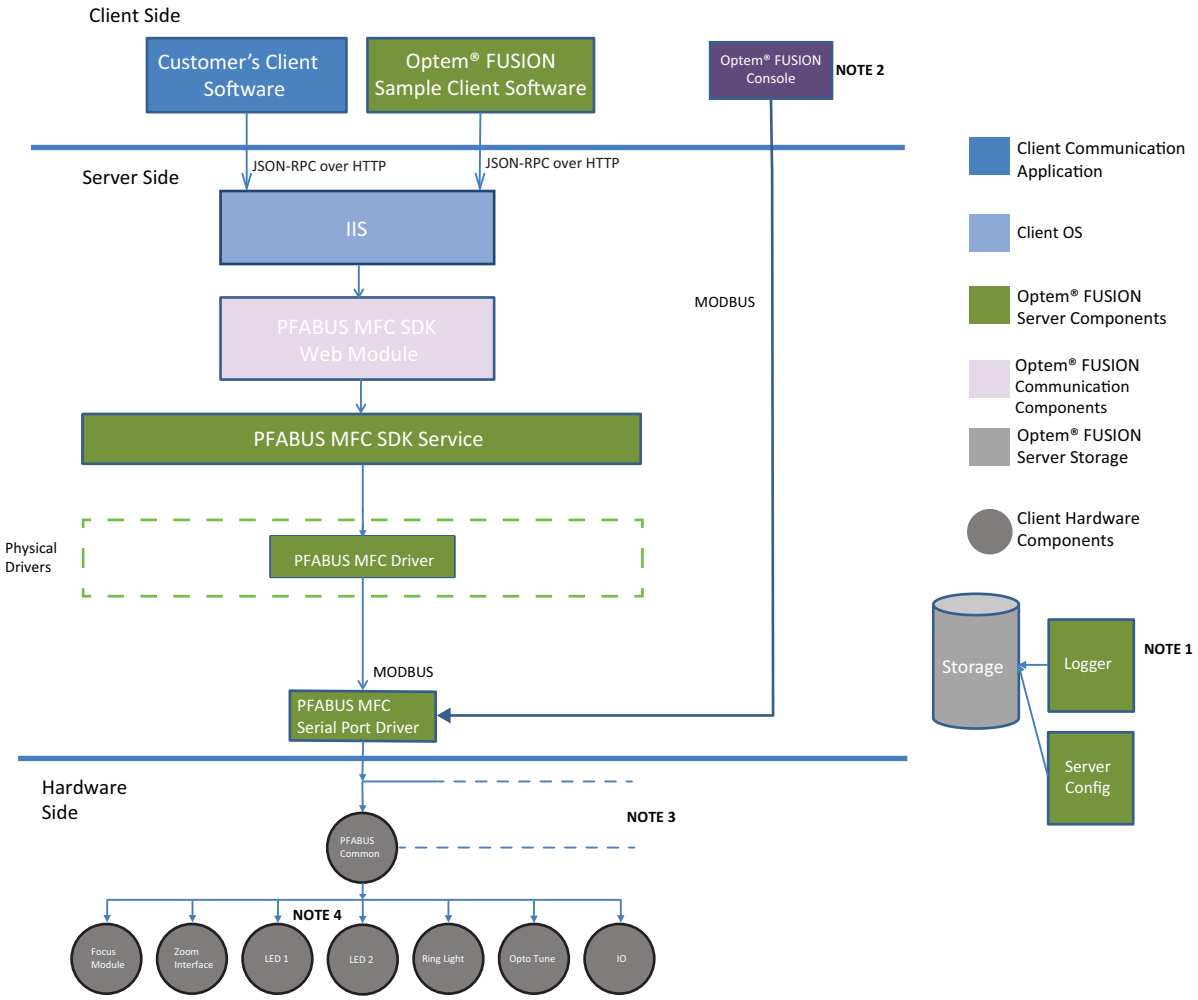
The following topics are covered:

- Main System Components, pg. 40
- Server to Optem® FUSION Hardware Communication, pg. 41

Main System Components

The Optem® FUSION hardware has to be controlled by the software satisfying requirements listed in "Optem® FUSION Interface Functions" on page 43. This software is called Optem® FUSION SDK and it is a part of the Optem® FUSION delivery.

In order to eliminate dependency on the customer's development platform and tools, the Optem® FUSION SDK is built using a client/server approach where the customer software constitutes the client side and Excelitas software constitutes the server side (see Figure 31).



NOTE 1: The Logger and the Server Config are used by all Optem® FUSION software components.
NOTE 2: Server configuration applications.
NOTE 3: The PFABUS MFC SDK supports up to 10 devices which can be identified using the mandatory method parameter i32ControllerIndex.
NOTE 4: Secondary LED channel is not supported if parallel mode is enabled.

Figure 31 *Client to Server Communication Interface*

The following is a list of the main components of the Optem® FUSION SDK:

- Excelitas Sample scripts – consists of a simple GUI and communication protocol implementation (see "Communication Protocol" on page 44). Excelitas Sample Client is delivered to customers in source code form. Its sources can be used by customers as reference design for their client software.
- IIS - OS component – supports communication with a few clients simultaneously.
- PFABUS™ MFC SDK Web Module – interprets client's request and serializes access to the hardware.
- PFABUS™ MFC SDK Service – performs operation requested by clients. Some operations can involve more than one device at a time.
- PFABUS™ MFC Driver – controls up to ten PFABUS™ common devices.
- Logger – maintains log files.
- Storage – keeps configuration information and server log files.
- Serial Port Driver – controls communication between drivers and configuration applications from one side and serial port hardware from another side.

WARNING! *Do not run the Server Configuration Applications while the Optem Fusion SDK Server is running and vice versa.*

Server to Optem® FUSION Hardware Communication

The Optem® FUSION controller uses RS-485 serial communication port to connect to the server. SDK supports multiple Optem® FUSION devices connected to multiple serial ports and/or multiple Optem® FUSION devices with different MODBUS ID connected to the same serial port.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER

4

Optem® FUSION Interface Functions

This chapter describes all the features, functions and communication protocol available in the Optem® FUSION SDK interface.

The following topics are covered:

- Communication Protocol, pg. 44
  - Single Operation Transaction, pg. 44
  - Devices, pg. 48
  - Methods, pg. 48
  - Parameters and Result Values, pg. 49
  - Batch Operations, pg. 49
- SDK Functions, pg. 50
  - Devices, Operations, Parameters and Values, pg. 51

Communication Protocol

Communication protocol is a text-based protocol which uses JSON RPC notation. This format is chosen to simplify analysis of the log of the host. Protocol complies with JSON (RFC 4627) as data format.

JSON-RPC specification can be found at *http://www.jsonrpc.org/specification*.

There are two types of the transactions: single operation and batch command set. These types are described in the following sections.

NOTE: *To access the web service remotely, i.e. from another computer, ensure that the configured web service port (default = 8081) is open for TCP connections. This can be checked using the installation computer's firewall application.*

NOTE: *It is important that all port numbers are properly set in the ServerConfig file to avoid unexpected behaviour. The configuration from this file can also be modified using the Optem FUSION® Console.*

Single Operation Transaction

Single operation request should be executed and responded immediately. If an operation takes a long time to be completed then a response with a result value of 1 should be sent right after the operation has been started.

Request

In accordance with JSON-RPC specification, format every request as described below:

```
{"jsonrpc": "2.0", "MethodName":"Name", "ParameterName":value, "id":number}
```

- Method Name format is described in "Methods" on page 48. It is a mandatory field.
- Parameters should be specified in accordance with JSON-RPC specification. This field is optional. Different options for parameters formatting are described in "Parameters and Result Values" on page 49.
- The ID is an arbitrary identifier to trace calls and responses. ID should be unique for each transaction. The system will not respond to a request without an ID.

Request Syntax Example

{"jsonrpc":"2.0", "method":"Component Name.Interface Name.Method Name",
"params":{"Name":Value,"i32ControllerIndex":0},"id":N}

In the above example, the components indicated in black are fixed and required for all requests. The components indicated in red are user-inputted values to accomplish the action desired by the user. "i32ControllerIndex":0 is a required parameter in all requests to indicate the controller ID instance index. See the descriptions of the user-inputted values below.

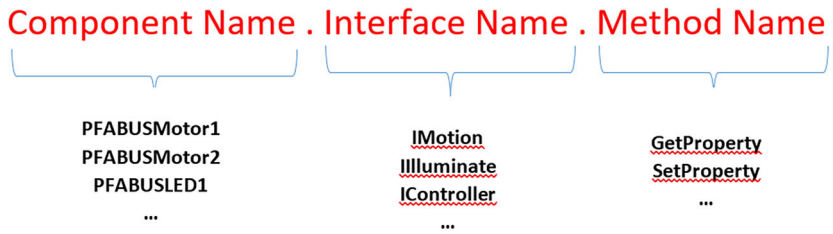


Figure 32 *Method Examples*

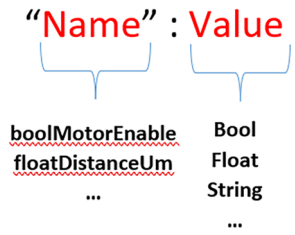


Figure 33 *Parameter Examples*

N – An identifier established by the Client that **must** contain a String, Number, or Null value if included. If it is not included it is assumed to be a notification. The value **should** normally not be Null and Numbers **should not** contain fractional parts.

Successful Response

In accordance with JSON-RPC Specification successful responses are formatted as following:

- Operation result is operation specific and is described in the document defining the operation.
- For the data query requests the field result of the response contains requested data.
- For the parameter change (set) requests the result has to be 0.
- For the operation request the result can be 0 if the operation was completed or 1 if the operation was started but not completed yet. It is the host responsibility to identify operation status later using subsequent query requests.

  For example: --> {"jsonrpc": "2.0", "method": "subtract", "params": [42, 23], "id": 1}
  <-- {"jsonrpc": "2.0", "result": 19, "id": 1}

Error Response

Error response is generated when an operation cannot be performed for any reason: invalid message format, invalid method name, invalid device state, etc.

An error object can contain the following fields:

- **code** – this can be either operation specific or one of the predefined values according to Table 1.
- **message** – this is the text representation of the error code.
- **source** – this describes the part of the software which detected the error.
- **data** – this contains operation specific and error specific information provided by the software.

  For example: {"jsonrpc": "2.0", "error": {"code": -32600, "message": "Invalid Request", “source”: “error source”, “data”: 33}, "id": 1}

Table 1 Web Server Error Codes

Code	Message	Meaning	Data
-32700	Parse error	Invalid JSON was received by the server. An error occurred on the server while parsing the JSON text.	Request text.
-32600	Invalid Request	The JSON sent is not a valid Request object.	Part of the request text starting with the unexpected symbol.
-32601	Method not found	The method does not exist / is not available.	Method name starting with unknown part.
-32602	Invalid parameters	Invalid method parameter(s).	Part of the request text starting with the unexpected/invalid parameter.
-32603	Internal error		
-32001	Operation Canceled	Operation was canceled due to previous error.	This is triggered by the previous error code.
-32002	Execution Denied	Operation cannot be performed due to invalid device state (i.e. device is busy or not initialized).	Device state.
-32003	Operation Timeout		

Table 2 SDK Result Error Codes

Code	Name	Description
0	Error_Success	
-1	Error_IncorrectParamValue	Incorrect parameter value passed to operation function.
-2	Error_OperationFailed	Device failed to process request.
-3	Error_IncorrectDeviceStatus	Device cannot process the request in current state.
-4	Error_OperationTimeout	Device operation timeout.
-5	Error_LimitSwitchTriggered	Limit switch triggered.
-6	Error_UnknownParameter	Unknown parameter type.
-7	Error_IncorrectDevice	Command to wrong device.
-8	Error_DeviceNotInitialized	Device not initialized.
-9	Error_UnknownOperation	Command not recognized.
-10	Error_InvalidOperation	Command not found in requested group (Operation, Set or Get).
-11	Error_IncorrectDataType	Incorrect data type (INT or FLOAT).
-12	Error_DeviceBusy	Operation cannot be performed – requested device is currently busy.
-13,	Error_DeviceNotHomed	Device (LLC, OOA or ZAA) must be homed before requesting any operation.
-14	Error_InitDeviceFailed	Device initialization failed.
-15	Error_CloseDeviceFailed	Device closing failed.
-16	Error_OtherDeviceParam	Current device failed to get needed data from other device.
-17	Error_OperationNotAllowedWhenAf	Operation not allowed when autofocus is on.
-18	Error_WriteNotAllowed	Write access not allowed, e.g. read-only, conditional write protected.
-19	Error_EEPROMReadFailed	EEPROM read failed.
-20	Error_DeviceNotConnected	Device is not connected. Device index is set to 255 in EEPROM.
-21	Error_IncorrectEEPROMRevision	The EEPROM map device interface revision is incorrect.

Devices

- A device can be physical or virtual (implemented in FW only).
- A device can be an integrated part of the system.
- A device name represents a particular device and has to be unique.
- The mapping between physical and virtual devices and device names is a part of the system configuration.

System Device

There is one pre-allocated device name which belongs to itself and cannot be used for another device: **System**.

This device name is used to perform system-wide and group operations and to configure the system software.

Methods

In general there are three groups of methods that are supported by the protocol as described in Table 3. Each method that is supported by the SDK is classified as either a Parameter, Operation or Configuration. In this document each method class is broken out independently and described in full detail in the tables below.

Table 3 Protocol Methods

	Group	Description
1	Parameter	• These methods are always executed immediately. • Up to three parameters can be requested by a single request.
2	Operations	• These methods typically take extended time to execute but may complete immediately. • Operation can have up to three arguments.
3	Configuration	• These methods can be performed only at certain hardware states. • Execution of these methods can take a long time. • These methods are not restricted in the number of parameters.

Method name consists of three parts separated by a period:

- Component Name.Interface Name.Method Name

Examples:

PFABUSMotor1.IMotion.GetProperty
PFABUSLED1.IIlluminate.GetProperty
PFABUSCommon.IController.GetProperty

Parameters and Result Values

Result values are formatted using the same rules as parameters.

- Parameters and results can be comma-separated values in their native order, for example: {3, 15, “String data”}
- Parameters and results can be provided by name, for example: {“fVelocityMs”: 20, “fAccelerationMss”: 0.3}
- Parameters and results can contain arrays. Arrays are formatted using square brackets, for example:
  [1,2,3,4,5].
- Parameter name usually consists of the following three parts:
  - Storage type (i.e. float, int, etc.)
  - Description name (i.e. Distance)
  - Units when applicable (i.e UM)
- When a parameter value is requested then the parameter of interest should be specified as “name”:null (applicable to requests only).

Batch Operations

Batch operation consists of one or more Single Operation(s).

Syntax

The batch operation is specified using square brackets:

[{Operation1},{Operation2},…]

Each single operation in the batch can have its own ID. The batch itself does not have an ID.

Operation Response

In contrary to single operation requests the response to each operation in the batch is sent when the operation is completed and not when the operation is started.

Error Processing

If an operation in the batch fails for any reason all subsequent operations in the batch get canceled. Appropriate error response should be generated for each canceled operation in the batch except notifications.

SDK Functions

This section describes all the features and functions available in the Optem® FUSION SDK interface.

NOTE: *All methods are expected to have the **i32ControllerIndex** parameter to indicate controller instance index, for example:*

- *{"jsonrpc":"2.0", "method":"PFABUSMotor1.IMotion.Home", "params":{"i32ControllerIndex":0} "id":4}*
- {"jsonrpc":"2.0", "method":"PFABUS*Motor1*.IMotion.GetProperty", "params":{"floatCurrentAbsPositionUm":null, "u32MotionStatus":null, "i32ControllerIndex":0} "id":5}

Devices, Operations, Parameters and Values

This section describes the Optem® FUSION system devices, function parameters and values and function operations.

Table 4 describes the device names in reference to Optem® FUSION or its components.

Table 4 JSON Devices and Supported Interfaces

Component Name	Interface Name	Comments
PFABUSMotor1	IMotion	Focus Module, 7x / 12.5x Zoom Modules, and WDI OOA interface utilizing RJ45 connector.
PFABUSMotor2	IMotion	Same as PFABUSMotor1. PFABUSMotor1 and PFABUSSMotor2 have the same functionality.
PFABUSLED1	IIlluminate	Primary channel for 1.5A/3A illuminators.
PFABUSLED2	IIlluminate	Secondary channel for 1.5A/3A illuminators.
PFABUSRingLight	IIlluminate	Interface for Ring Light illuminator.
PFABUSTunable	ITunableLens	Optotune Lens Interface. Currently not supported.
PFABUSIO	IIO	Input/Ouput Interface.
PFABUSCommon	IController	Common controller.

IMotion

NOTE: *Clock wise (CW) usually refers to positive or top direction. Counter-clock wise (CCW) usually refers to negative or bottom direction.*

Table 5 IMotion Parameters

Parameter Name	Description	Method Name	Type
u32MotionStatus	Motion Status • Bits 12 to 31 - unused. • Bit 11 Soft CW limit switch - active when floatMaxSoftLimitRelPositionUm is reached by motor. • Bit 10 Soft CCW limit switch - active when floatMinSoftLimitRelPositionUm is reached by motor. • Bit 9 Hard CW limit switch - active when tripped. • Bit 8 Hard CCW limit switch - active when tripped. • Bit 7 - high in speed tracking mode. See MoveWithSpeed method for more details. • Bit 6 - Motor busy. • Bits 4 to 5 - reserved. • Bit 3 - Motion stopped due to a software inhibit request. Clears when motor stops, even if the inhibit condition was removed while the motor was still moving. • Bits 0 to 2 - reserved.	IMotion.GetProperty	Uint32
boolMotorEnable	Device enable/disable state. To set this parameter, please use the MotorEnable method in "IMotion Operations" on page 54. • True= Enabled • False= Disabled	IMotion.GetProperty	Bool
floatCurrentRelPositionUm	Current relative motor position in micrometres. • Relative position set using ClearRelPosition method.	IMotion.GetProperty	Float
floatTargetAccelerationmm_SS	Target motor acceleration in mm/s^2. Used by MoveAbsUm and MoveRelUm if not overwritten by optional parameter.	IMotion.GetProperty IMotion.SetProperty	Float

Table 5 IMotion Parameters (continued)

Parameter Name	Description	Method Name	Type
floatTargetSpeedmm_S	Motor set point velocity in mm/s. Used by MoveAbsUm and MoveRelUm if not overwritten by optional parameter.	IMotion.GetProperty IMotion.SetProperty	Float
u32Microstepping	Number of motor pulses per step.	IMotion.GetProperty IMotion.SetProperty	Uint32
u32StepsPerMillimetre	Steps per millimetre (Steps/mm).	IMotion.GetProperty	Uint32
strFriendlyName	Interface friendly name. Maximum length is 20 characters.	IMotion.GetProperty	String
boolEnableSoftLimits	Enable soft limits.	IMotion.GetProperty IMotion.SetProperty	Bool
floatSoftLimitMinPositionUm	Minimum soft limit position. Prevents CCW motion beyond this value if SoftLimits are enabled.	IMotion.GetProperty IMotion.SetProperty	Float
floatSoftLimitMaxPositionUm	Maximum soft limit position. Prevents CW motion beyond this value if SoftLimits are enabled.	IMotion.GetProperty IMotion.SetProperty	Float
floatHomeOffsetUm	Home offset in µm	IMotion.GetProperty	Float
floatTravelRangeUm	Expected travel range in µm.	IMotion.GetProperty	Float
u8InterfaceType	Interface type. Must be 1 for Stepper Motor Motion Interface.	IMotion.GetProperty	Uint8

Table 6 IMotion Operations

Method Name	Description	Parameter Name	Type
MotorEnable	Enable motor and apply position holding current.	boolMotorEnable (mandatory) True – Enable motor False – Disable motor	Bool
MoveToPosition	Move stage position relative to 0 position.	floatPositionUm (mandatory) Position in micrometers.	Float
	Optional velocity (speed) parameter for this operation only. If not provided, target speed will be used. NOTE: This parameter does not overwrite the target speed parameter.	floatSpeedmm_S (optional) Speed in micrometers per second.	Float
	Optional acceleration (speed) parameter for this operation only. If not provided, target acceleration will be used. NOTE: This parameter does not overwrite the target acceleration parameter.	floatAccelerationmm_SS (optional) Acceleration in milliliters per seconds squared.	Float
MoveDistanceUm	Move the stage relative to the current position, and step movement.	floatDistanceUm (mandatory) Step distance in micrometers.	Float
	Optional velocity (speed) parameter for this operation only. If not provided, target speed will be used. NOTE: This parameter does not overwrite the target speed parameter.	floatSpeedmm_S (optional) Speed in micrometers per second.	Float
	Optional acceleration (speed) parameter for this operation only. If not provided, target acceleration will be used. NOTE: This parameter does not overwrite the target acceleration parameter.	floatAccelerationmm_SS (optional) Acceleration in milliliters per seconds squared.	Float

Table 6 IMotion Operations (continued)

Method Name	Description	Parameter Name	Type
MoveWithSpeed	Start moving the stage with set speed until MoveStop is issued or limit switch is activated. The speed is a mandatory parameter for this operation only, and does not overwrite the target speed parameter.	floatSpeedmm_S (mandatory) Speed in micrometers per second.	Float
MoveStop	Stop motor motion	No extra parameters	No extra parameters
Home	Execute homing procedure and reset position to zero.	No extra parameters	No extra parameters
Park	Move stage to zero relative position.	No extra parameters	No extra parameters
ClearRelPosition	Set current position to 0 relative position.	No extra parameters	No extra parameters

IIlluminate

Table 7 IIlluminate Parameters

Parameter Name	Description	Method Name[a]	Type
strFriendlyName	Interface friendly name. Maximum length is 20 characters.	Get: LED 1, LED 2, Ring Light	String
u8InterfaceType	Interface type. Must be 3 for LED interface and 4 for Ring Light interface.	Get: LED 1, LED 2, Ring Light	Uint8
i32LEDEnable	LED or Ring Light Enable • 0 = disable • 1 = enable	Get: LED 1, LED 2, Ring Light Set: LED 1, LED 2, Ring Light	Integer
i32Mode	Mode for LEDs: • 0 = Continuous • 1 = PWM • 2 = Pulse Trigger • 3 = Pulse Follow Mode for Ring Light: • 0 = Variable Voltage • 1 = PWM	Get: LED 1, LED 2, Ring Light Set: LED 1, LED 2	Integer
floatCurrentDimmingPrcnt	Current dimming (%).	Get: LED 1, LED 2 Set: LED 1, LED 2	Float

Table 7 IIlluminate Parameters (continued)

Parameter Name	Description	Method Name[a]	Type
u32MaxCurrentmA	Absolute maximum current in mA.	Get: LED 1, LED 2	Uint32
floatDutyCyclePrcnt	Duty Cycle (%).	Get: LED 1, LED 2, Ring Light Set: LED 1, LED 2, Ring Light	Float
floatFrequencyHz	Frequency (Hz).	Get: LED 1, LED 2, Ring Light Set: LED 1	Float
i32PulseLengthUs	Pulse Length (µs) for Pulse Trigger mode.	Get: LED 1, LED 2 Set: LED 1, LED 2	Integer
i32DelayUs	Delay (µs) for Pulse Trigger mode.	Get: LED 1, LED 2 Set: LED 1, LED 2	Integer
floatVoltageDimmingPrcnt	Voltage dimming (%).	Get: Ring Light Set: Ring Light	Float
u32MaxVoltagemV	Maximum Voltage 0 to 10V in mV.	Get: Ring Light	Uint32
floatLEDTemperatureC	Not supported	Not supported	Float
floatControllerTemperatureC	Not supported	Not supported	Float

a. ***Get: ...*** *denotes the following methods; PFABUSLED1.IIlluminate.GetProperty, PFABUSLED2.IIlluminate.GetProperty, and PFABUSRingLight.IIlluminate.GetProperty.* ***Set:...*** *denotes the following methods; PFABUSLED1.IIlluminate.SetProperty, PFABUSLED2.IIlluminate.SetProperty, and PFABUSRingLight.IIlluminate.SetProperty.*

ITunableLens

NOTE: *ITunableLens is currently not supported in the controller*

Table 8 ITunableLens Parameters

Parameter Name	Description	Method Name	Type
strFriendlyName	Interface friendly name. Maximum length is 20 characters.	ITunableLens.GetProperty	String

IIO

Table 9 IIO Parameters

Parameter Name	Description	Method Name	Type
strFriendlyName	Interface friendly name. Maximum length is 20 characters.	IIO.GetProperty	String
strModelName	Interface model name, currently equivalent to u8DeviceIndex.	IIO.GetProperty	String
u8DeviceIndex	Connected device from predefined table. Value of 0 means custom device, value of 255 means no device connected.	IIO.GetProperty	Uint8
u8InterfaceRevision	Interface Revision. Must be 6 for current version. The version changes only if EEPROM Interface Mapping is changing.	IIO.GetProperty	Uint8
u8InterfaceType	Interface type. Must be 6 for I/O Interface.	IIO.GetProperty	Uint8
boolOutput1Enabled	Enable/disable state of IO 1. If set to True, IO 1 is set as an output.	IIO.GetProperty IIO.SetProperty	Bool
boolOutput1DefSetting	Default settings for Output 1 (Low or High).	IIO.GetProperty	Bool
boolOutput2Enabled	Enable/disable state of IO 2. If set to True, IO 2 is set as an output.	IIO.GetProperty IIO.SetProperty	Bool
boolOutput2DefSetting	Default settings for Output 2 (Low or High).	IIO.GetProperty	Bool
boolOutput3Enabled	Enable/disable state of IO 3. If set to True, IO 3 is set as an output.	IIO.GetProperty IIO.SetProperty	Bool
boolOutput3DefSetting	Default settings for Output 3 (Low or High)	IIO.GetProperty	Bool
boolOutput4Enabled	Must be False, I/O 4 is always input.	IIO.GetProperty	Bool
boolEStopMotionEnabled	Enable/disable state of E-STOP for motion device. If set to TRUE Input 4 acts as E-STOP for motion device.	IIO.GetProperty IIO.SetProperty	Bool
boolEStopLEDEnabled	Enable/disable state of E-STOP for Illumination devices. If set to True Input 4 acts as E-STOP for Illumination devices.	IIO.GetProperty IIO.SetProperty	Bool
boolOutput4DefSetting	Must be False as I/O 4 is always input.	IIO.GetProperty	Bool
boolOutput5Enabled	Enable/disable state of IO 5. If set to True, IO 5 is set as an output.	IIO.GetProperty IIO.SetProperty	Bool
boolOutput5DefSetting	Default settings for Output 5 (Low or High).	IIO.GetProperty	Bool
boolInput1Status	Current status for Input 1 (Low or High). False is Low and True is High. Relates to boolOutput1Setting.	IIO.GetProperty	Bool
boolInput2Status	Current status for Input 2 (Low or High). False is Low and True is High. Relates to boolOutput2Setting.	IIO.GetProperty	Bool
boolInput3Status	Current status for Input 3 (Low or High). False is Low and True is High. Relates to boolOutput3Setting.	IIO.GetProperty	Bool
boolInput4Status	Current status for Input 4 (Low or High). False is Low and True is High.	IIO.GetProperty	Bool

Table 9 IIO Parameters (continued)

Parameter Name	Description	Method Name	Type
boolInput5Status	Current status for Input 5 (Low or High). False is Low and True is High. Relates to boolOutput5Setting.	IIO.GetProperty	Bool
boolOutput1Setting	Controller setting for Output 1 (Low or High). Relates to boolInput1Status.	IIO.SetProperty	Bool
boolOutput2Setting	Controller setting for Output 2 (Low or High). Relates to boolInput2Status.	IIO.SetProperty	Bool
boolOutput3Setting	Controller setting for Output 3 (Low or High). Relates to boolInput3Status.	IIO.SetProperty	Bool
boolOutput5Setting	Controller setting for Output 5 (Low or High). Relates to boolInput5Status.	IIO.SetProperty	Bool
u8LEDCh1TriggerSource	Trigger Source for LED1: 0 - DI1 1 - DI2 2 - DI3 3 - DI5	IIO.GetProperty	Uint8
boolLEDCh1TriggerInverted	Trigger Inverted for LED1: False - Active High Trigger True - Active Low Trigger	IIO.GetProperty	Bool
u8LEDCh2TriggerSource	Trigger Source for LED2: 0 - DI1 1 - DI2 2 - DI3 3 - DI5	IIO.GetProperty	Uint8
boolLEDCh2TriggerInverted	Trigger Inverted for LED2: False - Active High Trigger True - Active Low Trigger	IIO.GetProperty	Bool

IController

Table 10 IController Parameters

Parameter Name	Description	Method Name	Type
u32CRC	CRC The value is generated by the controller firmware at boot up.	IController.GetProperty	Uint32
u32DescriptorCount	Number of descriptors in use (max 7)	IController.GetProperty	Uint32
u32SerialNumber	Controller serial number. Set by production.	IController.GetProperty	Uint32
u32BoardNumber	Main (PFA) board serial number. Set by production.	IController.GetProperty	Uint32

Table 10 IController Parameters (continued)

Parameter Name	Description	Method Name	Type
u32ModelNumber	Controller Serial Number	IController.GetProperty	Uint32
u32ManufacuringDate	Manufacturing date. Set by production.	IController.GetProperty	Uint32
u32DaughterBoardSerialNumber	Daughter board serial number. Set by production.	IController.GetProperty	Uint32
strFriendlyName	Interface friendly name. Maximum length is 20 characters.	IController.GetProperty	String
boolCanbusTermResistor	CANBUS termination resistor enable/ disable.	IController.GetProperty	Bool
boolStandalone	Standalone controller.	IController.GetProperty	Bool
boolManufacturingMode	Manufacturing mode enable.	IController.GetProperty	Bool
boolModbusTermResistor	Modbus termination resistor enable/ disable.	IController.GetProperty	Bool
u8ModbusID	Modbus node address.	IController.GetProperty	Uint8
u8CanbusID	CANBUS node ID.	IController.GetProperty	Uint8
u8MapRevision	EEPROM Map Revision. Must be 1 for current version. Values of 0 and 15 indicate mapping before this document is accepted. The version changes only if EEPROM Mapping is changing. Interfaces has its own revision.	IController.GetProperty	Uint8
floatFPGATemperatureC	FPGA Temperature (Celsius).	IController.GetProperty	Float

Table 11 IController Operations

Method Name	Description	Parameter Name	Type
InitializeDevice	Initializes the controller values from EEPROM. Also called by System.ISystem.Init method. **Do not call this more than once during the SDK service run time.**	No extra parameters	No extra parameters

Table 12 IController Configuration

Parameter Name[a]	Description	Type
i32EncComport	Encoder COM Port.	Integer
i32BaudRate	COM Port Baud rate.	Integer
i32ModbusNodeAddress	Modbus Slave/Node Address.	Integer

a. These parameter names can also be viewed and modified using the Optem FUSION® Console.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER

5

Demo Scripts and Test Examples

The chapter provides a general description on how to use the Optem® FUSION demo scripts and test examples.

The following topics are covered:

- Overview, pg. 62
  - Demo Collections, pg. 62
- Sample Scripts, pg. 62
  - Initialize Controller, pg. 62
  - Enable Illumination, pg. 63
  - Enable Motor, pg. 63
  - Home Motor, pg. 63
  - Check Position After Performing Homing, pg. 63
  - Get PWM Settings, pg. 63
  - Set Light Level to 50% Power, pg. 63
- Running Demo Scripts, pg. 64
  - Optem Fusion Demo - Postman Script, pg. 64
- XML Demo Script Tool, pg. 69
- C# Test Example, pg. 72
  - Running a C# Test Example, pg. 72
  - Locating the C# Test Example Folder, pg. 74

Overview

To communicate with the Optem® FUSION SDK System, POST requests, compliant to the standard described in "Communication Protocol" on page 44, are used.

Default port is 8081, configurable on installation. IIS module is in the same IP address as the Optem® FUSION SDK server. Sample usage can be viewed from the Optem® FUSION SDK demo Postman Collection files.

Use null as key values for parameters with 'N' value types, e.g. “MakeZero”:null

Use an array as key value for parameters with ordinal sub element keys, e.g. ""floatMagnificationCnt":[5.0,10.0,20.0,50.0,100.0]

Demo Collections

The demo collections can be modified using Postman which can be downloaded from:

https://www.getpostman.com/downloads/

The demo scripts run the collections through newman which is a command-line interpreter for Postman.

Sample Scripts

Initialize Controller

--> {"jsonrpc":"2.0", "method":"System.ISystem.Init", "id":1}

<-- {"jsonrpc":"2.0","result":{"PFABUSIO (i32ControllerIndex:0)":0,"PFABUSLED1 (i32ControllerIndex:0)":0,"PFABUSLED2 (i32ControllerIndex:0)":0},"PFABUSMotor1 (i32ControllerIndex:0)":0,"PFABUSMotor2 (i32ControllerIndex:0)":0},"PFABUSRingLight (i32ControllerIndex:0)":0}},"id":1}

Enable Illumination

--> {"jsonrpc":"2.0", "method":"PFABUSLED1.IIlluminate.SetProperty", "params":{"i32LEDEnable":1,
"i32ControllerIndex":0}, "id":1}

<-- {"jsonrpc": "2.0","id": 1,"result": {"i32LEDEnable": 1}}

Enable Motor

--> {"jsonrpc":"2.0",
"method":"PFABUSMotor1.IMotion.MotorEnable","params":{"boolMotorEnable":true,
"i32ControllerIndex":0}, "id":1}

<-- {"jsonrpc": "2.0","result": 1,"id": 1}

Home Motor

--> {"jsonrpc":"2.0", "method":"PFABUSMotor1.IMotion.Home","params":{"i32ControllerIndex":0}, "id":1}

<-- {"jsonrpc": "2.0","result": 1,"id": 1}

Check Position After Performing Homing

--> {"jsonrpc":"2.0", "method":"PFABUSMotor1.IMotion.GetProperty", "params":{"u32MotionStatus":null,
"floatCurrentRelPositionUm":null, "i32ControllerIndex":0}, "id":5}

<-- {"jsonrpc":"2.0","result":{"u32MotionStatus":0,"floatCurrentRelPositionUm":0.0},"id":5}

Get PWM Settings

--> {"jsonrpc":"2.0", "method":"PFABUSLED1.IIlluminate.GetProperty", "params":{"i32Mode":null,
"floatFrequencyHz":null, "i32ControllerIndex":0}, "id":1}

<-- {"jsonrpc":"2.0","result":{"i32Mode":1,"floatFrequencyHz":20000.0},"id":1}

Set Light Level to 50% Power

--> {"jsonrpc":"2.0", "method":"PFABUSLED1.IIlluminate.SetProperty",
"params":{"floatDutyCyclePrcnt":50.0, "i32ControllerIndex":0}, "id":1}

<-- {"jsonrpc":"2.0","result":{"floatDutyCyclePrcnt":50.0},"id":1}

Running Demo Scripts

This section provides information for using the Optem Fusion Demo - Postman Script and the XML Demo Script Tool which are included in the full SDK installation. The scripts are Postman collections which are run using newman.

The demo files are in the demo sub-folder of the installation location.

Optem Fusion Demo - Postman Script

NOTE: *An alternate way to test JSON methods is through an XML Script described in "XML Demo Script Tool" on page 69.*

To run the Optem Fusion Demo - Postman Script, perform the following:

NOTE: *Run the demo on the Windows account that was used to install the SDK, otherwise see step 8 of this procedure to install the command line script interpreter.*

1. In the Windows Start Menu select “Optem Fusion Demo - Postman Script”.

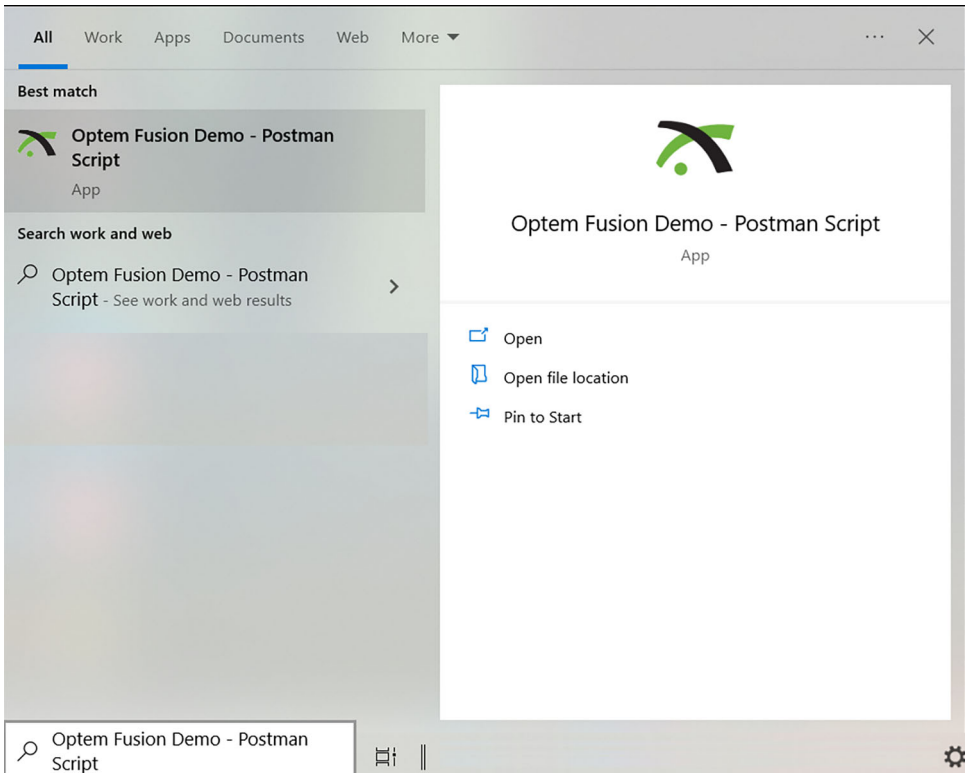


Figure 34 *Start menu – Optem Fusion Demo - Postman Script*

NOTE: *Optem Fusion Console and any other application that uses the controller serial COM port should be disconnected before starting the Optem Fusion SDK service and vice versa.*

NOTE: *Refer to the MAN-350013 Optem® FUSION Controller User Manual for further information.*

2. Turn off screen-savers, device sleep, hibernation, screen blanking, and similar events to avoid interrupting the demo process.

3. When prompted by the script, press **Enter** to continue.

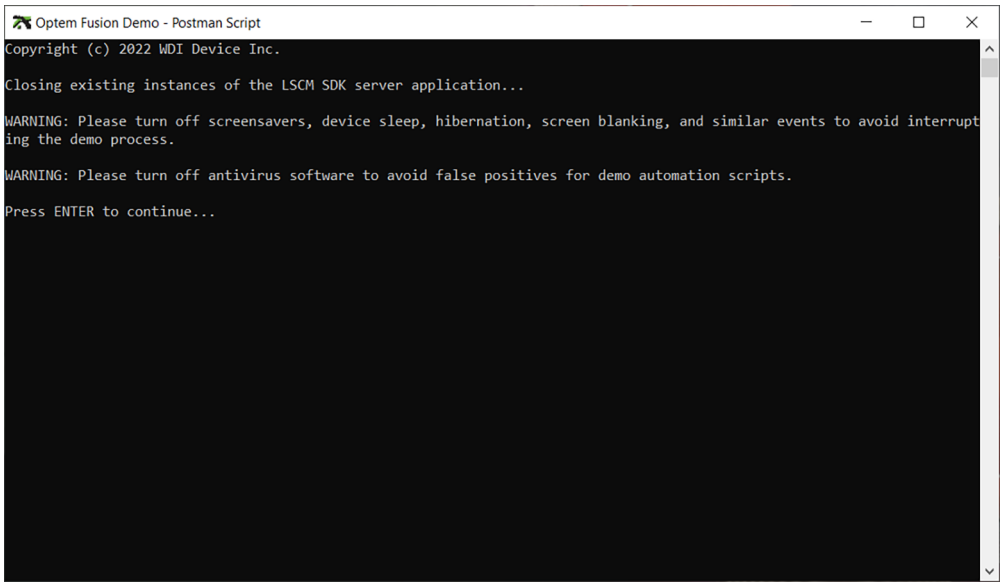


Figure 35 *Optem Fusion Demo - Postman Script Terminal Window – Step 1*

4. Reload correct values, appropriate to your controller, to EEPROM using the Optem FUSION® Console.
5. Power cycle the controller.
6. Restart the SDK service.
7. When prompted by the script, press **Enter** to continue.

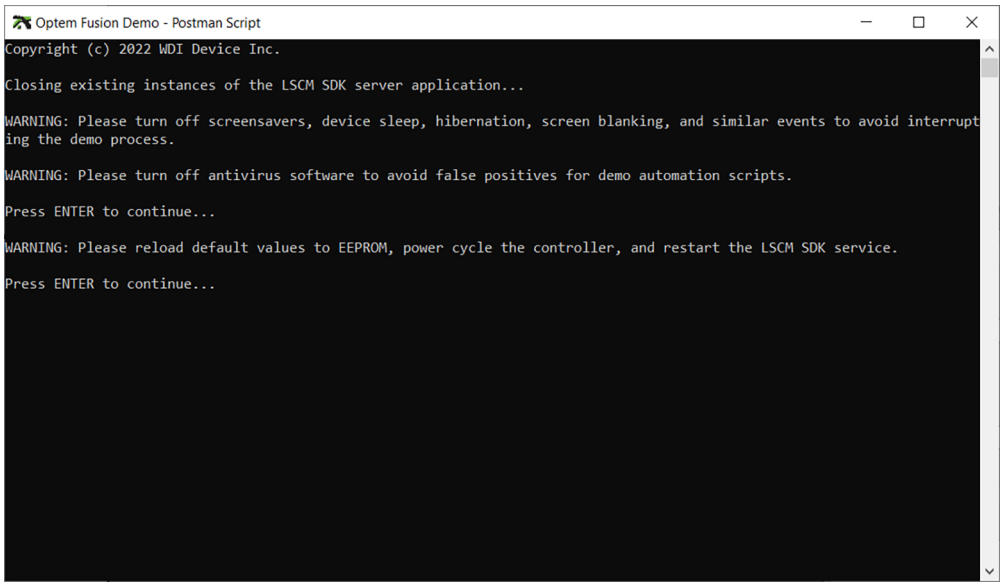


Figure 36 *Optem Fusion Demo - Postman Script Terminal Window – Step 2*

8. If this prompt appears (see Figure 37), install the command line script interpreter by typing **npm install -g newman** in your command prompt window.

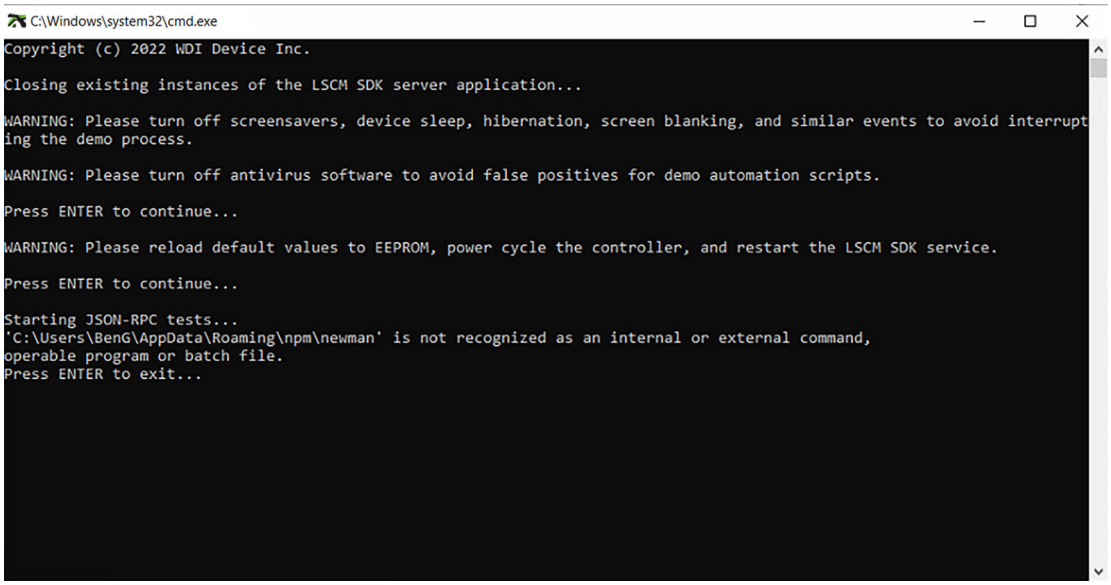


Figure 37 *Demoscript Fail Prompt*

NOTE: *You should have newman version 5.3.2 or higher after installing this.*

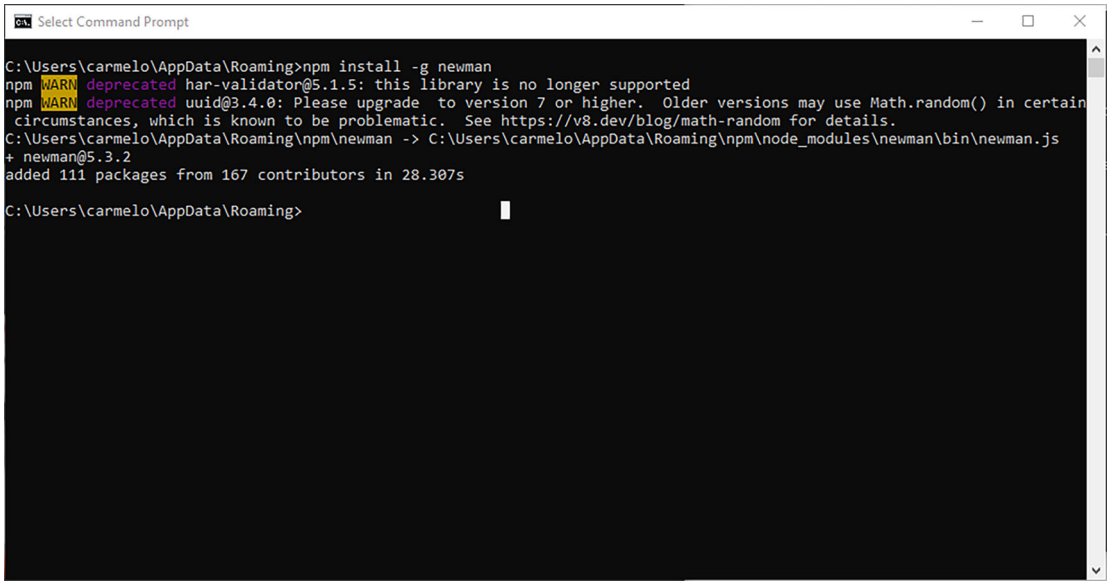


Figure 38 *Newman Install Prompt Window*

9. Observe the progress of the script in the open terminal window.

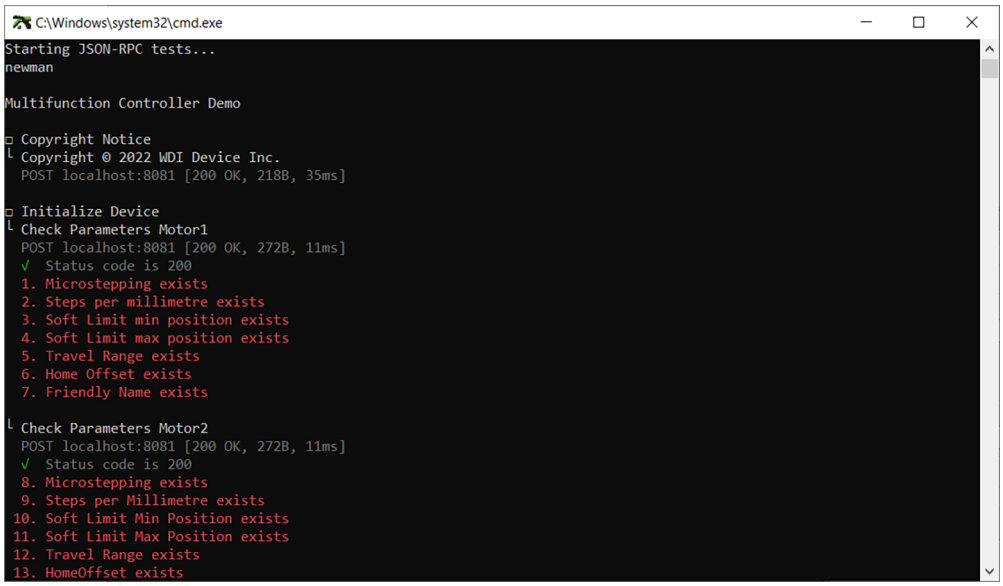


Figure 39 *Optem® FUSION Demo Script Terminal Window – Running*

10. Review the results of the tests at completion and note any errors and issues (see Figure 40).

11. When prompted by the script, press **Enter** to exit.

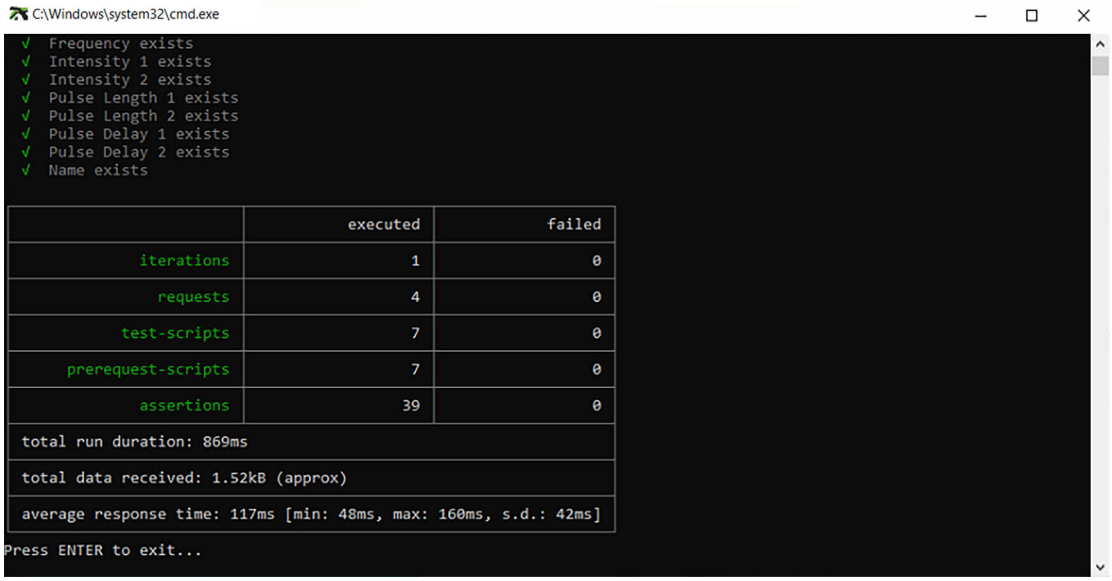


Figure 40 *Optem® FUSION Demo Script Terminal Window – Finished*

12. See "Optem® FUSION Interface Functions" on page 43 for further information.

XML Demo Script Tool

The XML Demo tool is an application that takes a XML script that describes a series of operations on the motor and LED and executes it. This is an alternate way to test JSON methods in contrast to the Postman Script described in "Optem Fusion Demo - Postman Script" on page 64.

The tool is able to execute the following commands:

- Motors: Focus/Zoom, Value in µm
  - MoveUp
  - MoveDown
  - MoveAbs
  - Home
  - SetVelocity
- LED: Value in %, Delay in msec
  - Illumination
  - LedEnable, Value is true or false

To run the Optem Fusion Demo - XML Script, perform the following:

NOTE: *Run the demo on the Windows account that was used to install the SDK, otherwise see step 8 of this procedure to install the command line script interpreter.*

1. In the Windows Start Menu select “Optem Fusion Demo - XML Script”.

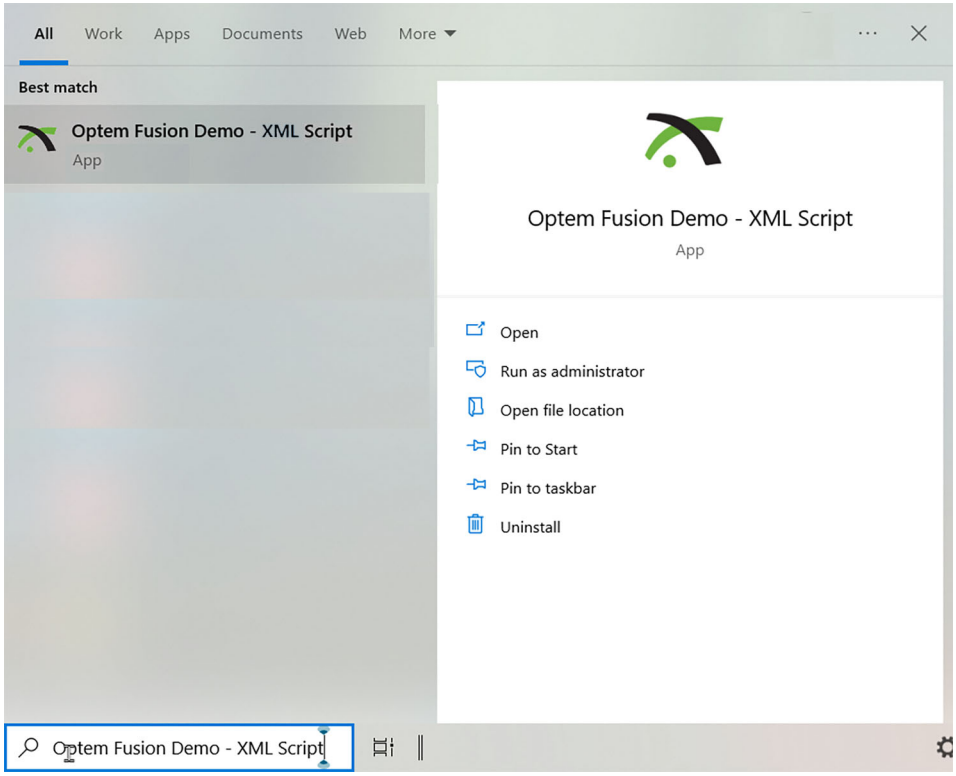


Figure 41 *Start menu – Optem Fusion Demo - XML Script*

NOTE: *Optem Fusion Console and any other application that uses the controller serial COM port should be disconnected before starting the Optem Fusion SDK service and vice versa.*

NOTE: *Refer to the MAN-350013 Optem® FUSION Controller User Manual for further information.*

2. Turn off screen-savers, device sleep, hibernation, screen blanking, and similar events to avoid interrupting the demo process.

3. Once the tool is open, input the name of the host machine in the Host Name text box (outlined in red in Figure 42)

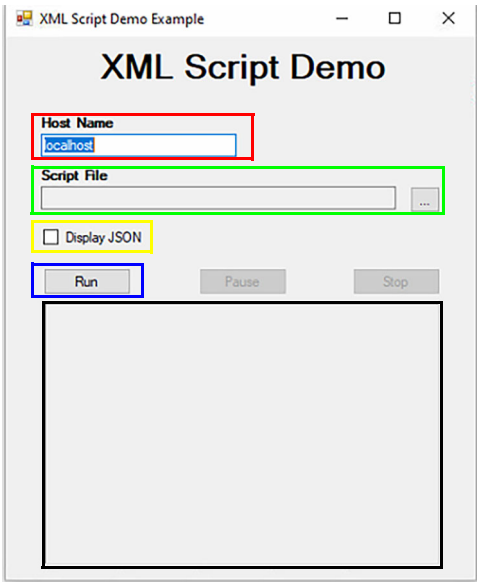


Figure 42 *XML Script Demo Window*

4. Click the selection button to the right of the Script File text box (outlined in green in Figure 44), to select the XML script file to load (outlined in red in Figure 43).

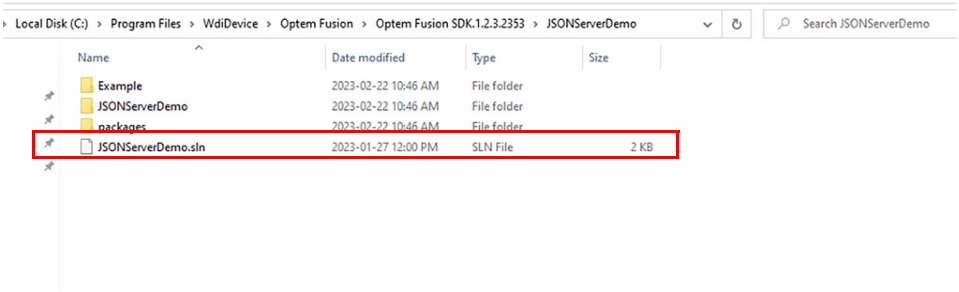


Figure 43 *XML script file folder*

5. Select the **Display JSON** checkbox (outlined in yellow in Figure 42) if you want to show the JSON request that is sent to the server.

6. Click **Run** (outlined in blue in Figure 42), to start the XML script. The operations will be output to the log window (outlined in black in Figure 42) as they happen.

7. You can click **Pause** (pauses the operations) or **Stop** (stops the operations) at any time.

C# Test Example

This section provides information for using the C# test example which is included in the SDK installation. This example demonstrates how to integrate the Optem Fusion controller with C#.

Running a C# Test Example

To run a C# test example, perform the following:

1. In the Windows Start Menu, locate the Optem Fusion folder and click either:
   – Optem Fusion Application Example (outlined in red in Figure 44), or
   – Optem Fusion Console Example (outlined in green in Figure 44)

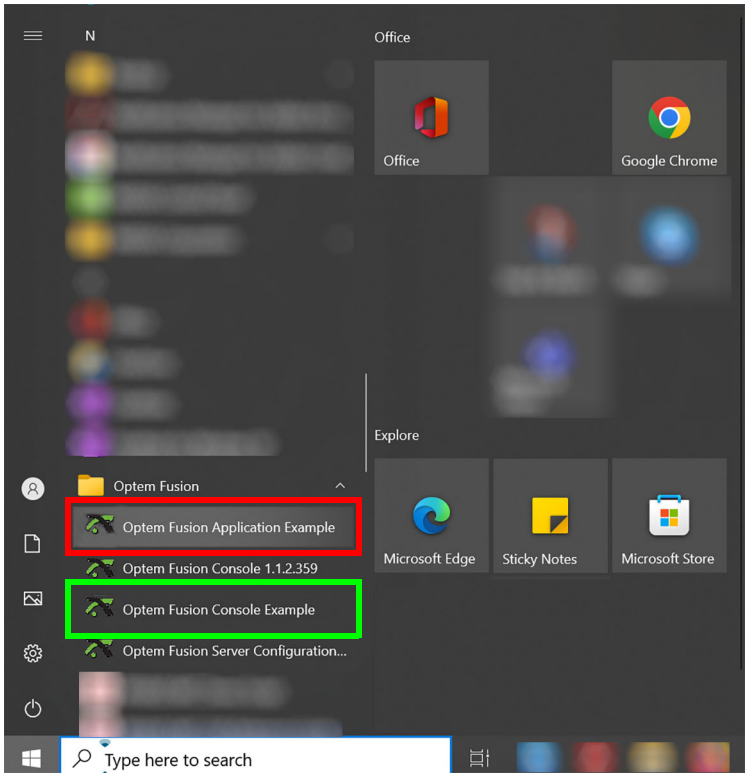


Figure 44 *Start menu – Optem® FUSION C# Test Examples*

OR

2. Locate the executable file from the following folder (see Figure 45):
   - C:\Program Files\WdiDevice\Optem Fusion\Optem Fusion SDK.x.x.x.xxxx\Optem Fusion SDK C# Test Example\optemfusion_app_test_example\bin\Release\optemfusion_app_test_example.exe

   OR

   - C:\Program Files\WdiDevice\Optem Fusion\Optem Fusion SDK.x.x.x.xxxx\Optem Fusion SDK C# Test Example\optemfusion_console_test_example\bin\Release\optemfusion_console_test_example.exe

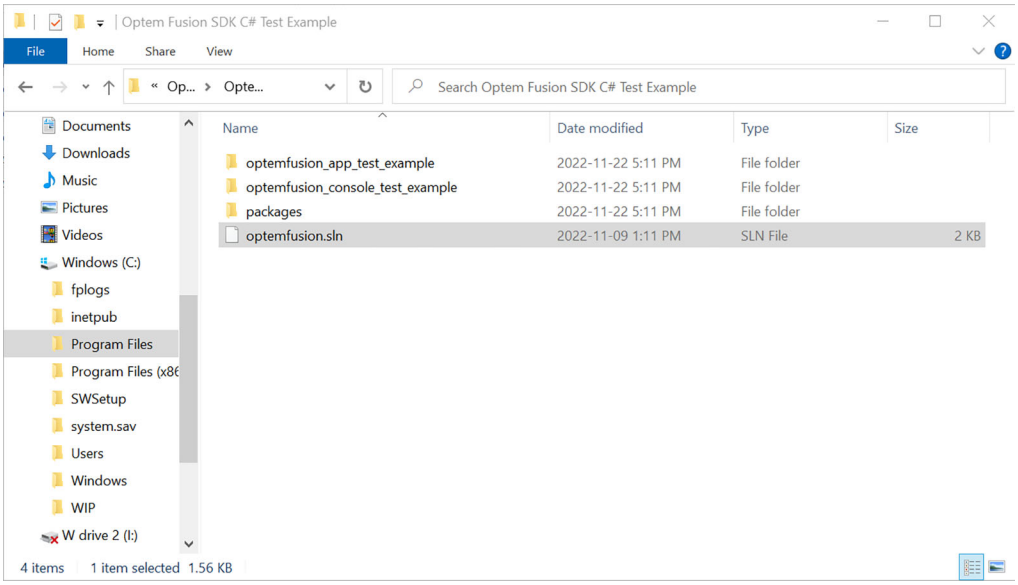


Figure 45 *C# Test Example Executable File Folders*

Locating the C# Test Example Folder

To locate the C# test example folder, perform the following:

1. Locate the C# test example folder (outlined in red in Figure 46), in the **C:\Program Files\WdiDevice\Optem Fusion\Optem Fusion SDK.x.x.x.xxxx** folder.

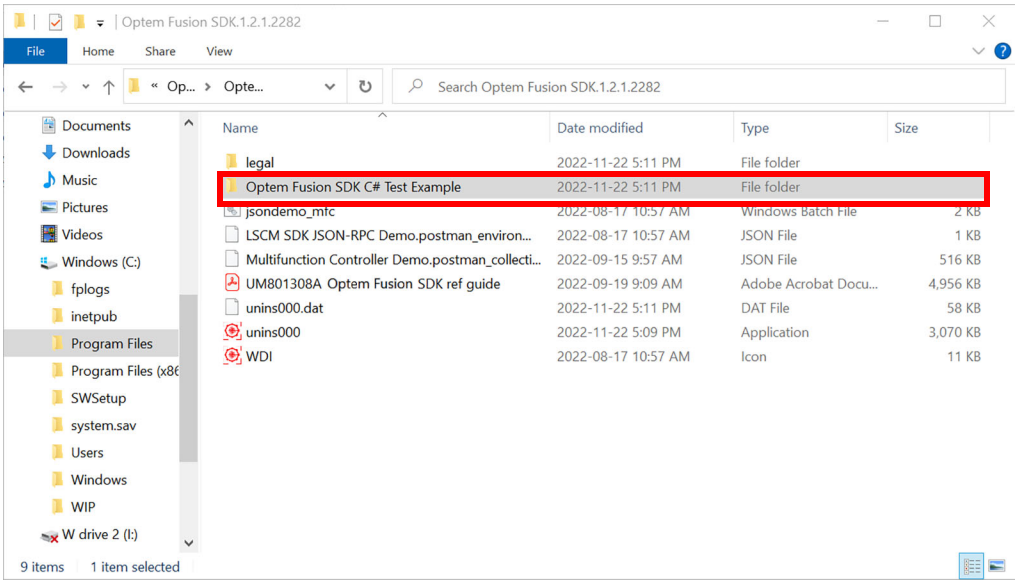


Figure 46 *C# Test Example Folder Location*

2. Open the folder for contents.

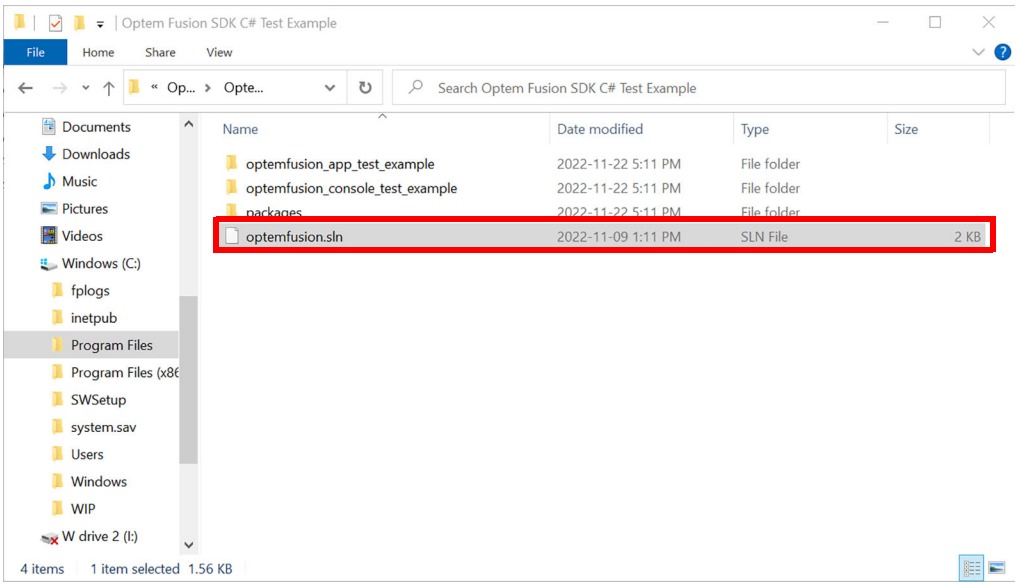


Figure 47 *C# Test Example Folder Contents*

3. Double click **optemfusion.sln** (outlined in red in Figure 47), to open the solution file.

NOTE: *In order to open the solution file, you need to have Visual Studio installed on the computer where the Optem Fusion SDK is installed.*

NOTE: *The test example can be referenced from the **optemfusion_app_test_example.cs** (a GUI test example) or the **optemfusion_console_test_example.cs** (a command line test example), under the Optem Fusion project.*

4. See "Optem® FUSION Controller Software Development Manual Installer Components" on page 13 for project dependencies and license information.

THIS PAGE INTENTIONALLY LEFT BLANK

Index

C

communication protocol 43

D

device, system 48

H

hardware 12, 40
hardware configuration 12
HTTP protocol 12

J

JSON-RPC 12

O

Optem Fusion SDK main components 41

S

safety precautions 12
software package 18
system device 48

T

technical support, contacting 15

W

workstation installation 18

THIS PAGE INTENTIONALLY LEFT BLANK