Medtronic	NEUROMODULATION CONFIDENTIAL	DOCUMENT/RECORD
This document/record is electronically controlled, printed copies are considered uncontrolled.		
Identifier	**Version**	**Author**
NRP1088-36356	2.0	Benjamin Isaacson

Title: User Manual, Nexus-D API	**Pages:** (including this page) 29

APPROVALS

Signed By	Responsibility	Date/Time (GMT)
Johnsen, Tom	Regulatory Affairs Approver	05/22/2014 07:29:18 PM
Benjamin Isaacson	Author / SW Design Lead	05/22/2014 08:02:42 PM
Bourget, Duane	Project Lead	05/22/2014 08:39:05 PM

Nexus-D API

Instructions for Use in a Research System

CAUTION – Investigational device. Limited by Federal (or United States) law to investigational use.

Table of Contents

List of Figures

List of Tables

1. Warnings / Precautions

⚠ **Caution** – The Nexus-D API Software is only to be used with clinician supervision.

2. Scope

The Nexus-D API (Application Programming Interface) is complementary to the Nexus-D System and is intended to simplify the use of the Nexus-D System. The Nexus-D API can only be used with a Nexus-D System, but the Nexus-D System does not require the use of the Nexus-D API. A host application can still interface directly to the Nexus-D System without using the Nexus-D API.

Therefore, while this document contains pertinent information about the Nexus-D API, it will also reference the Nexus-D System Instructions for Use in a Research System (Nexus-D IFU) to explain many concepts and themes. This document focuses on the key components of the Nexus-D API and the instructions to use the Nexus-D API. A user of the Nexus-D API and this document must be familiar with the Nexus-D System.

3. Abbreviations and Definitions

Table 1: Definitions

Term	Definition
API	Application Programming Interface
GNU	GNU's Not Unix
GPL	General Public License
IDE	Integrated Development Environment
IFU	Instructions For Use
INS	Implantable Neurostimulator
JDK	Java Standard Edition Development Kit
JRE	Java Runtime Environment
JSSC	Java Simple Serial Connection
LGPL	Lesser GNU Public License
SW	Software
USB	Universal Serial Bus

4. Applicable Documents

Table 2: Applicable Documents

Number	Version	Title
NRP1088-35896	4.0	User Manual, Nexus-D System
NRP1088-35790	7.0	Interface Specification, Nexus-D System
NRP1088-36867	4.0	Interface Specification, Nexus-D API

5. Licensing

The Nexus-D API uses a library called Java Simple Serial Connection (JSSC). The JSSC library and its use are covered by the Lesser GNU Public License (LGPL). A copy of the GNU General Public License (GPL) and the LGPL are included on the same media that this document resides upon.

The terms of the LGPL require that any software using the JSSC library includes installation information on how to install and execute a modified version of the JSSC library. If you wish to install and execute a modified version of the JSSC library with the Nexus-D API, simply replace the original jssc.jar file that is included during the installation of the Nexus-D API with a modified jssc.jar file. However, the Nexus-D API cannot be guaranteed to function properly with a modified version of the JSSC library.

Refer to Appendices A and B for the full GPL and LGPL Licenses.

6. Background

The Nexus-D API is a software package that provides an interface between the Nexus-D System and a host application running on a host computer. The Nexus-D System is a data conduit (i.e., bi-directional data port) that:

1. Transmits data from the Activa PC+S to a user computer
2. Transmits stimulation update commands from a host application running on the host computer to an Activa PC+S or Activa PC.

The Nexus-D API does not make any decisions about therapy modifications or the data being passed to the host application. The Nexus-D API does not contain any physical (i.e. – mechanical, electrical) components. The Nexus-D API is purely software only. The Nexus-D API does not execute on any Medtronic supplied hardware.

6.1. System Diagram

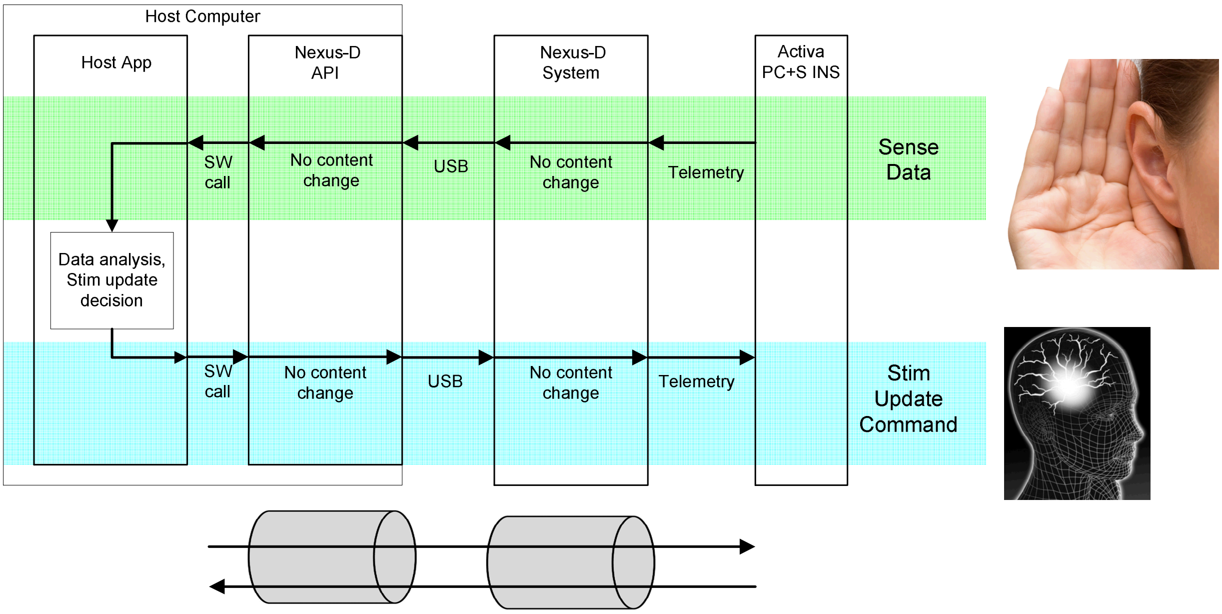

Figure 1: System Diagram

6.2. API Functionality

The Nexus-D API consists of software modules that ease the implementation of a host application by providing an abstraction layer between the host application and the Nexus-D System. This abstraction layer handles the timing between the reception of sensing data and the sending of stimulation commands. This abstraction layer also hides the details of the data and command formatting. No decisions are made by the Nexus-D API. The Nexus-D API simply passes the data and commands through to the other side.

The Nexus-D API cannot be used to configure sensing or stimulation. The 8840 Clinician Programmer must be used to configure stimulation. The Sensing Programmer must be used to configure sensing.

7. Basic Setup

7.1. Host Requirements

The Nexus-D API was built using Java version 7, which must be installed on the host computer. All other host requirements can be found in the Nexus-D IFU.

7.2. Installation

To install the Nexus-D API, copy the java archives (*.jar* files) *nexus.jar* and *jssc.jar* from the installation CD to a known location on the host computer that will be used when developing a host application.

7.3. Host Setup

7.3.1. Configuring Java

The Nexus-D API requires Java Runtime Environment (JRE) version 7 (or newer) to be installed on the computer where it will be used. You can check the version installed on your Windows based computer by going to the “Control Panel” -> “Programs” -> “Java (32-bit)” and pressing the “About” button on the “General” tab

Figure 2: About Java Display

An alternate method is to go to the Java web site located at http://java.com and click the "Do I Have Java?" link (see Figure 3: Java Download Page). Run the web applet when requested to view the results. If you do not have Java installed or "**Your Java version:**" is less than Version 7, then use the "Download Java Now" button to begin the installation process. Follow the installation instruction on the web site to install the JRE on your computer.

Figure 3: Java Download Page

7.3.2. Use of Nexus-D API in a Matlab Environment

To use the Nexus-D API within Matlab, first install Matlab on your computer per the manufacturer's instructions.

To access the Nexus-D API from MATLAB, add the java archives (*.jar* files) *nexus.jar* and *jssc.jar* to the MATLAB Java class path. Java classes available to the MATLAB environment are declared in either the static or the dynamic Java class paths. The dynamic Java class path is cleared every time MATLAB is closed so the *.jar* files will need to be added each time after starting MATLAB. Table 3 shows how to add or remove *.jar* files to the dynamic Java class path. The commands may be executed at the MATLAB command prompt or from a script.

Table 3: MATLAB Java Class Path

Task	Example Syntax	Notes
Add .jar file to dynamic java class path	>>javaaddpath('..\nexus.jar');	Adds files to dynamic java class path. Does not persist after MATLAB restart.
View classes added to java class path	>>javaclasspath ;	Displays all the classes on the static and dynamic paths.
Remove files from java class path	>>javarmpath('..\nexus.jar');	Removes specified *.jar* files from dynamic path

Detailed documentation on importing java classes to MATLAB and the various syntax variations of the above syntax examples may be found here: http://www.mathworks.com/help/matlab/matlab_external/bringing-java-classes-and-methods-into-matlab-workspace.html#f111171
Note: static Java class path is mentioned above, but not described here.

7.3.2.1. Hardware Setup

1. Connect the USB connector of the Nexus-D System to a USB port of your computer with MATLAB installed.
2. Place the antenna over the Activa PC+S INS. Make sure that the antenna is held close to the INS. This antenna allows the Nexus-D System to communicate with the INS.
3. Make sure that the Nexus-D System battery indicator shows healthy battery status. If not, install two fresh AAA batteries into the Nexus-D System.

7.3.3. Accessing Nexus-D API from MATLAB

This section describes the most basic workflow to connect the Nexus-D API to MATLAB and to communicate with the Activa PC or PC+S INS. As an example, follow the steps below in order to import a data packet (sensed from the Activa PC+S) from the Nexus-D API to MATLAB. The commands in Table 4 may be entered at the MATLAB command prompt or run from a script.

7.3.3.1. Basic Workflow

Sequentially execute the commands in Table 4 to import a data packet as sensed by the Activa PC+S INS into MATLAB. The Activa PC+S INS must first be setup for sensing using the Sensing Programmer before data packets may be received from the INS. At least one sense channel must be active to get a non-zero length data packet. Note that this is the most basic workflow to get data packets from Nexus into MATLAB and does not cover all the possible use cases that a real world application that uses the Nexus-D API might encounter. Please refer to the Nexus-D API Interface Specification for a full list of commands that may be executed using the API.

Table 4: Importing Data Packets into MATLAB

Step #	Command	Action
1.	inst = mdt.neuro.nexus.NexusInstrument.getInstance;	Creates a Nexus Instrument Object. Nexus-D API methods may now be called on this object.
2.	s = mdt.neuro.nexus.SerialConnection('COM4');	Setup a serial port connection to the Nexus-D System plugged into a USB port on the host computer running MATLAB. Make sure to include the correct COM port number in the syntax (COM4 in example).
3.	inst.connect(s);	Connect the instrument object to the serial port. Return values are: • 0 indicates that the connection was successful. • 1 indicates that the serial port was not found. • 2 indicates that the serial port is busy or was not closed correctly the last time it was opened. Try restarting MATLAB if a non-zero return value is returned.
4.	status = inst.getNexusStatus;	Returns an object *status* that has methods returning the status of the system. For example, executing *status.getState* tells whether the INS is connected or not. *status.getBatteryPercent* gives Nexus battery percentage.

5.	inst.setNexusConfiguration(30,15);	Sets the timeout periods for the Nexus-D System maintenance (sec) and supervisory (min) sessions. Refer to the Nexus-D IFU for a detailed explanation of each session type.
6.	inst.startSensing;	Initiates sensing in the Activa PC+S INS.
7.	inst.startDataSession;	Gets the Nexus-D System ready to start receiving sensed data from the INS.
8.	D = inst.getDataPacket;	Returns a Data packet object *D* that contains the sensed data packet as well as other information like whether or not stimulation was ON; what stimulation group was active; etc.
9.	Data = D.getData;	Returns *Data* that is a cell array containing data from all four sensed channels (depending on the channel setup).
10.	PlottableData = Data{1,1};	Returns data packet from the first sensed channel into *PlottableData*. If channel 1 was enabled for sensing on the INS, then *PlottableData* will be an int16 array of numbers that represent sensed data in LSBs. The number of array elements depends on the sampling frequency at which the data was sensed on the INS. The sample rate can be determined from the *D* object getSampleRates method. Each data packet is 400ms long, so at a 422 Hz sampling rate there will be 168 elements in *PlottableData.* You may type-cast the data into a double data type by executing: *double(PlottableData);*

7.3.4. Accessing Nexus-D API from Simulink—Example Model

This section outlines the process of streaming sensed data from the Activa PC+S using the Nexus-D System into a Simulink model with an example model.

Please note that the example given is for reference only and is not to be considered as tested code. The example uses commands from Table 4 as well as some that were needed for the example Simulink model. API commands, Simulink blocks, and MATLAB functions are application specific and the Simulink interface to the Activa PC+S INS should be designed based on the requirements of a specific application.

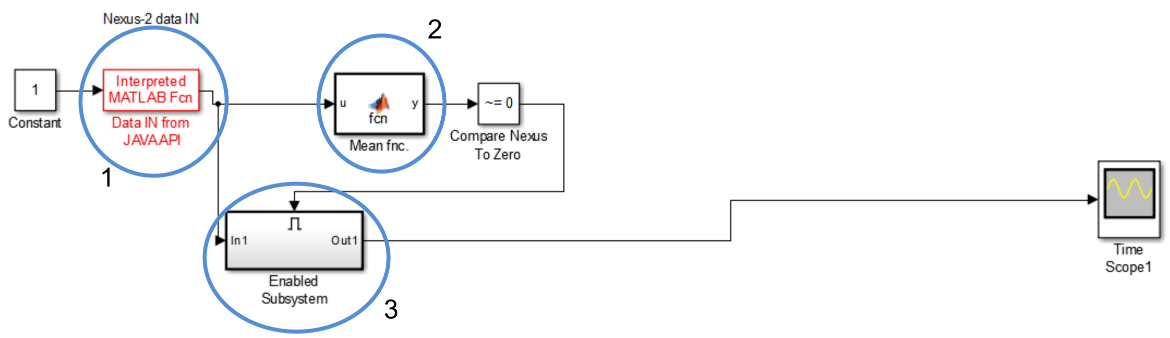

Figure 4: Example Simulink Model using Nexus-D API

Figure 4: Example Simulink Model using Nexus-D API shows a basic Simulink model that uses the Nexus-D API for streaming sensed data from the Activa PC+S INS into Simulink and displays this data on a time scope. In the Simulink *Model Properties*, under the *Callbacks* tab, the following code was used:

PostLoadFcn:

```
javaaddpath('C:\Work\nexus\NexusAPI\trunk\jssc.jar')
javaaddpath('C:\Work\nexus\NexusAPI\trunk\nexus.jar')
inst = mdt.neuro.nexus.NexusInstrument.getInstance;
s = mdt.neuro.nexus.SerialConnection('COM4');

provider = mdt.neuro.nexus.ThreadedNexusInstrument % using threaded calls
inst.connect(s);
inst.getNexusStatus;
inst.setNexusConfiguration(30,15); % Set to max timeouts

inst.startSensing; % this will leave sensing on when data session ends
inst.startDataSession;
```

StartFcn:

```
inst.startDataSession;
provider.getDataPacket; % retrieve data from Nexus
```

CloseFcn:

```
inst.setNexusConfiguration(10,2); % reset to defaults
inst.disconnect;
inst.dispose; % clean up properly
```

7.3.4.1. Model Elements

The key model elements have been circled in blue in Figure 4. They are as follows:

7.3.4.1.1. Interpreted MATLAB Function

This block performs the following operations:

i. Reads the Nexus Instrument object from the MATLAB base workspace into Simulink.
ii. Requests and receives data packets from the Nexus-D API.
iii. Extracts the data from the packets.

The following code is used in this block:

```
function [y]  = grab_data_from_API2(u)
%%
% Variables initialized

% packet_length is the expected length of data in the packet (400 ms),
% for 200 Hz sampling rate, packet length = 80, for 422 Hz,
% packet length = 168

packet_length = 168;


%%
%calling the Java object 'inst' from workspace. This is the Nexus Instrument
inst = evalin('base','inst');
provider = evalin('base','provider');

if(~mdt.neuro.nexus.ThreadManager.getInstance.isExecuting)
    nCode = inst.getLastNexusResponseCode;
```

```
    iCode = inst.getLastInsResponseCode;
    if (iCode == -1 || iCode == 105)

        DeviceStatus = inst.getNexusStatus();
        inst.getLastInsResponseCode

        if inst.getLastInsResponseCode == -1

            disp('Device Powered off. Or Antenna not in range. Please restart')

        elseif ~strcmp(DeviceStatus.getState,'MAINTENANCE_ENABLED')
            inst.startDataSession;
            if inst.getLastInsResponseCode ~= 0
                fprintf('%d\n',inst.getLastInsResponseCode);
            end
        end
    end

    resp = provider.getThreadSafeReturnVal; %getting in data packet

    checkvariable = whos('resp');

    if strcmp(checkvariable.class,'mdt.neuro.nexus.data.DataPacket')

        temp = resp.getData;

        y = double(temp{1});          %Extract the time channel data


    else
        y = zeros(1,168);
    end
    provider.getDataPacket;
else
    y = zeros(1,168);

end

end
```

Note that the data array output by this block is 168 elements long. This is because at a 422Hz sampling frequency (used in this example), a data packet that is 400ms in duration will contain 168 elements.

This code includes conditional statements that check the Nexus-D System status as well as the INS status, to keep the data packets flowing into Simulink uninterrupted. These conditional statement blocks will change depending on specific applications. This code is meant as a reference example only.

7.3.4.1.2. Mean Function

This function computes the mean of the data in every packet and enables the subsystem labeled “3” only when the mean is non-zero. The code for this block is as follows:

```
function y = fcn(u)
y = mean(u);
end
```

7.3.4.1.3. Enabled Subsystem

This subsystem is enabled when the output of the Mean Function (item 2) is non-zero. This ensures that the time scope displays the individual data packets as one continuous stream instead of discontinuous blocks. Had this

subsystem not existed and had data packets been sent to the scope directly, there would be no data shown for time periods between two data packets.
This subsystem is shown below:

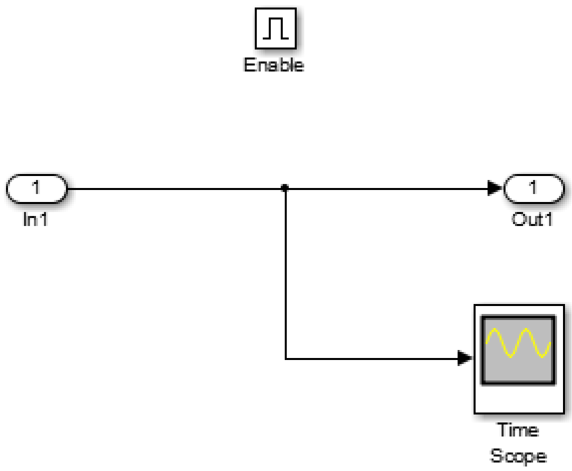

Figure 5: Simulink subsystem

7.3.5. Use of Nexus-D API in a Java Project

To create a Java application using the Nexus-D API, the Java Standard Edition Development Kit (JDK) must be installed. The JDK can be downloaded from the Oracle web site (http://www.oracle.com/java/javase). Choose the 32 bit version for your operating system and follow the installation instructions on the site. There are a number of developer tools that may be used, but an Integrated Development Environment (IDE) tool is needed at a minimum (Netbeans, Visual Studio or Eclipse are popular options). After an IDE has been installed, set the classpath to reference the Nexus-D API *.jar* files. For example, using the Eclipse IDE:

1) Create a new Java project
2) Add the Nexus-D API *.jar* files (*nexus.jar* and *jssc.jar*) to your project using the "Add External JARs.." button on the "Libraries" tab:

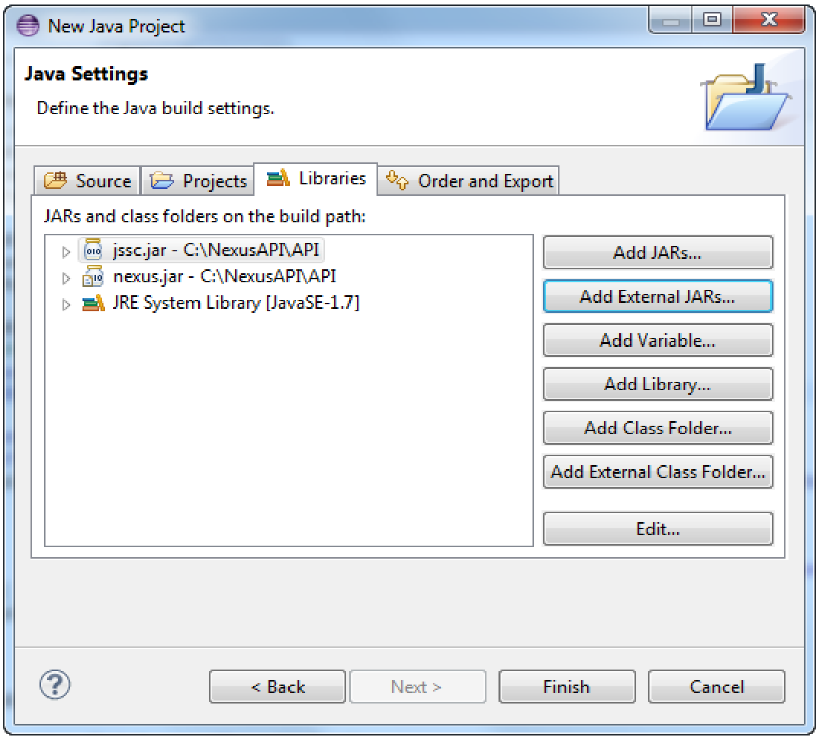

Figure 6: Adding JARs

3) If a project already exists, select project properties and add the Nexus-D jar files to the "Java Build Path":

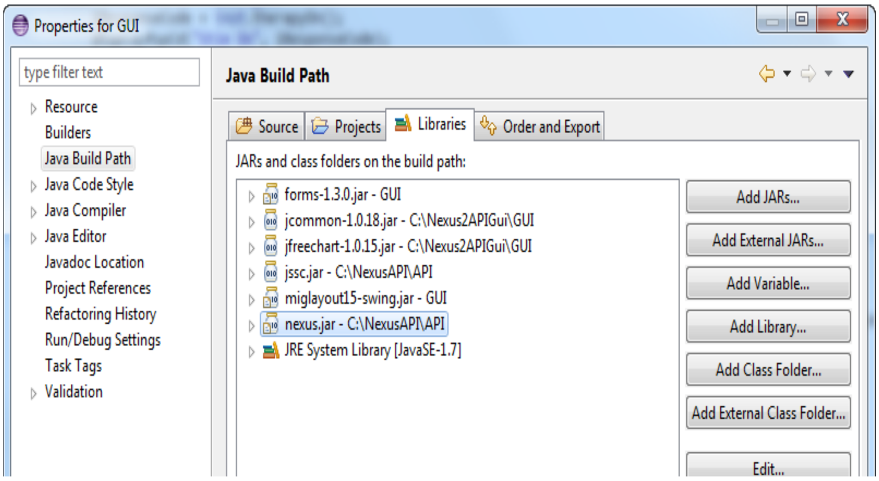

Figure 7: Java Build Path

8. General Use of the Nexus-D API

8.1. Nexus-D API Organization

The Nexus-D API is split into the packages described in the following subsections to organize the code.

8.1.1. mdt.neuro.nexus

This package contains the widely used classes central to the operation of the Nexus-D API including the communication layers, threading classes, and interface structures. Host applications will instantiate many of these classes when using the Nexus-D API.

8.1.2. mdt.neuro.nexus.commands

This package contains all of the Nexus-D command classes, which extend the ApplicationCommand class. The ApplicationCommand class encapsulates the state that is shared by all Nexus-D system commands, as explained in the Interface Specification, Nexus-D System.

A host application cannot instantiate any of these classes, as they are created by the NexusInstrument instance when an operation is requested by a host.

8.1.3. mdt.neuro.nexus.data

This package contains the data structures that represent the data returned by the Nexus-D System. This includes data packets, INS information, and Nexus-D status. A host application cannot instantiate any of these classes, as they are returned to the host by the Nexus-D API.

8.1.4. mdt.neuro.nexus.support

This package contains a logger that is used by internal API implementation classes. The user can control how much is logged by using the *setLevel* method. For example, to log all serial data sent and received by the API for debugging:

```
Logger logger = NexusLogger.getLogger();
logger.setLevel(Level.ALL); // display/log all data sent and received
logger.log(Level.INFO, "Begin API Use.");
```

Logs are located in the user home directory (e.g. C:\Users\username on Windows 7) with a file name in the form Nexus_YYYY_MM_DD_S.log (where YYYY is the year, MM is month, DD is day, and S is a sequence number).

8.2. Core Classes

8.2.1. NexusInstrument

The NexusInstrument class is the main abstraction of the Nexus-D System interface to the host application. The host will call methods of the NexusInstrument to execute a desired operation. In order to be operational, the NexusInstrument requires the steps listed in section 8.4 to be taken:

8.2.2. ThreadedNexusInstrument

The ThreadedNexusInstrument class uses the NexusInstrument class, but wraps the calls that the NexusInstrument provides. This wrapper allows the host application requests to be executed on a worker thread, so the host application's main thread does not block during the execution of the request. The class also provides access to the returned value from the most recent request and the most recent INS response code via *getThreadSafeReturnVal* Refer to section 8.8.2 Threading Considerations for more details on threading.

8.3. General Flow of Use

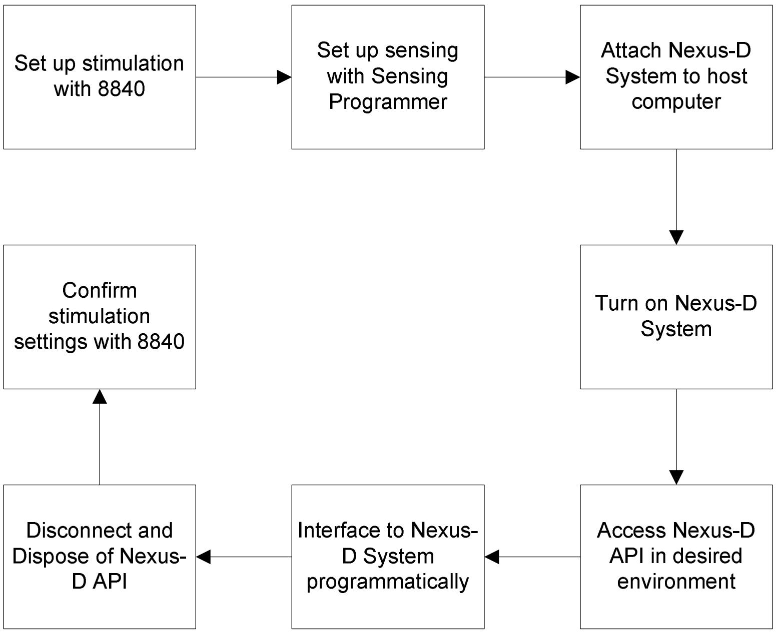

Figure 8: Nexus-D API Use Flow

8.4. Connect / Disconnect

When using the Nexus-D API, the following steps must be followed:

1. Call the *getInstance* method of NexusInstrument, which will instantiate the class if it hasn't already been instantiated
2. Create a SerialConnection object and pass it to the NexusInstrument via the *connect* method.
3. Use the NexusInstrument to interface to the Nexus-D system
4. Call the *disconnect* method of the NexusInstrument
5. Call the *dispose* method of the NexusInstrument

8.4.1. Initiate Sessions

The Nexus-D API provides the ability to initiate the various sessions that are supported by the Nexus-D System.

Table 5: Sessions

Session	Start	Stop
Supervisory	Any Nexus-D system operation (typically *getNexusStatus*)	Turn off Nexus-D system manually after performing programmatic clean up described in section 8.4.
Maintenance	*startDataSession*	*stopDataSession*

Refer to the Nexus-D IFU for a detailed explanation for each session type.

8.5. Updating Stimulation

One of the primary functions of the Nexus-D API is to update stimulation. Refer to the Nexus-D IFU for a detailed explanation of the stimulation update mechanisms available to the host application.

8.6. Querying the Nexus-D System

The Nexus-D API can be used to configure the Nexus-D System and monitor its status. Refer to the Nexus-D IFU for a detailed explanation of the Nexus-D System specific operations available to the host application.

8.7. Data Types

Some API classes represent data returned by the Nexus-D System in response to specific commands.
Most data structures contain information in the same format as it is returned by the Nexus-D System. Refer to the Nexus-D API Interface Specification for more details.

8.8. Special Considerations

8.8.1. Host Priority

A host application must choose whether the gathering of data or the updating of stimulation is of higher priority based on its specific use case. Refer to the Nexus-D IFU for a detailed explanation of this concept.

8.8.2. Threading Considerations

8.8.2.1. Blocking Calls

Operations (i.e. method calls) performed by the NexusInstrument will block and result in host application code waiting to execute until the NexusInstrument has received a response or times out and returns to the host application.

8.8.2.2. Non-Blocking Calls

Using the ThreadedNexusInstrument allows a host application to make calls to the Nexus-D API that will not block. This functionality was included in the Nexus-D API to allow the integration of multiple sensors. Therefore, if the data rate of another sensor is faster than the Activa PC+S INS, the Activa PC+S INS data rate will not limit collection of data from the other sensor. It is the responsibility of the host application to monitor the status of the ThreadManager *isExecuting* properly to know when a response has been received by the Nexus-D API.

Similarly, the ThreadManager class within the Nexus-D API allows a host computer to pass Runnable objects to be executed on a separate thread, so that the main thread is not blocked by the operation. A host could choose to implement an anonymous class that implements the Runnable interface, with the *run* method of the anonymous class containing the desired NexusInstrument call. This would allow a host application to create more complex command sequences easily.

```
ThreadManager.getInstance().doOperation(new RunnableObservable()
{

    @Override
    public void run()
    {
        // do work
        setChanged();
        notifyObservers();
    }
});
```

Figure 9: Anonymous RunnableObservable Class

8.8.3. Sending Commands

It is important that a host application synchronizes the sending of a command to the receipt of the previous command's response (i.e. – don't use a timer to control the sending of a command). Failing to wait for the previous command's response will cause communication issues between the host application and the Nexus-D System.

8.8.4. Data Conditioning

The user can optionally choose to apply a template subtraction to remove 2.5 Hz noise that is introduced into the sensed data due to the telemetry circuitry being active during data collection by the INS. The template subtraction method computes an average across a given set of data packets, called the template. This template can then be subtracted from data packets to remove artifact or noise. Perform the following steps to apply the template subtraction:

1. Collect and store 30 seconds to 3 minutes of baseline data.
2. Pass the collected data in a call to the *computeTemplate* method of NexusInstrument

3. Store the returned result of *computeTemplate*
4. Call *subtractTemplate* on a given DataPacket to remove subtract the template stored in the previous step.
5. A template should be able to be used across data collection sessions. However, if the results of the subtraction do not have the desired effect on the data, consider computing a new template. It is also possible the signal cannot be improved by using the template method.

9. Responsibilities of the Host

9.1. Make Decisions

The host application is responsible for making decisions about stimulation updates and data retrieval. The Nexus-D API assumes no knowledge of the stimulation settings of the INS; it is the responsibility of the host application to maintain knowledge of the current stimulation settings and modify them as needed.

9.2. Collect Desired Data

The Nexus-D API does not "push" data to the host application. The host application must request data via the *getDataPacket* command once a maintenance session has been started via the *startDataSession* command.

The Nexus-D API does not save or log any data requested by the host application. It is the responsibility of the host application to save any information of interest during use of the Nexus-D API.

9.3. Prevent Timeouts

The host application should query the Nexus-D System via the *getNexusStatus* command to prevent the Nexus-D System from transitioning from the active state to the sleep state.

The host application should periodically request data via the *getDataPacket* command to prevent the Nexus-D System from transitioning from a maintenance session to a supervisory session.

9.4. Handle Errors

If a command sent to the Nexus-D API is not successful, the Nexus-D API will either return the error code that resulted from the operation, an invalid value for the return parameter or a null object. In the case of the latter two situations, the Nexus-D API can be queried to determine the error code by calling *getLastNexusReponseCode* and *getLastInsResponseCode.*

Refer to the Nexus-D IFU for a discussion of the various Nexus-D System response codes that should be handled by the host.

9.4.1. Implement Retries

The host application may choose to implement retries if an error occurs during a requested operation, such as a telemetry error or INS reject. No automated retries are provided by the Nexus-D API.

> **Note:** Keep in mind that a telemetry error or lack of response from the Nexus-D API does not mean that the command did not reach the INS and modify therapy. Always perform a *getInsInfo* check to confirm the stimulation settings after an error.

10. Utility Classes

There are numerous classes within the Nexus-D API that perform special functions or are used for certain operations. The Util class contains numerous static utility methods for dealing with arrays and data types. The utilities can be used by a host application.

11. Troubleshooting

Table 6: Troubleshooting

Problem	Possible Explanation	What to do
Cannot connect to the serial connection	Another process is using the COM port specified	Eliminate any COM port conflict.
Null data returned by the Nexus-D API	Sensing is not configured properly in the INS	Use the Sensing Programmer to confirm that data compression is

		enabled
	A data session has not been started properly	Use the *startDataSession* command to start a data session
Missing data packets	Stimulation commands executing delay the retrieval of data packets	Delay stimulation updates if data retrieval is the priority
	Telemetry errors	Ensure the telemetry head is properly placed over the INS
Cannot establish communication with the Nexus-D System	The Nexus-D System is not powered on or plugged into the host computer	Ensure Nexus-D System is powered on and plugged into the USB connection of the host computer
	The Nexus-D System is not using the COM port passed to the Nexus-D API	Double check that the COM port passed to the Nexus-D API is correct
Unexpected responses from the Nexus-D System	Query the NexusInstrument *getLastInsResponseCode* and *getLastNexusResponseCode* to determine the source of the error	Refer to the Nexus-D IFU for details on what to do in response to the error
Data does not look the same as data collecting using the Sensing Programmer	Data collected using the Nexus-D System requires that data compression is enabled in the Activa PC+S. This data compression increases the noise floor when stimulation is active.	Collect data without stimulation on to optimize the sensing noise floor.

12. Contact Information

For more information about Nexus-D API, contact your Medtronic Research Site Contact.

Manufacturer
Medtronic, Inc.
710 Medtronic Parkway
Minneapolis, MN 55432-5604
USA
Internet: www.medtronic.com
Tel. 1-763-505-5000
Fax 1-763-505-1000

Manufactured at
Medtronic Inc.
7000 Central Ave
Minneapolis, MN 55432
USA
Tel. 1-763-505-5000
Fax 1-763-505-1000

Appendix A. GNU General Public License

GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require,
such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the

unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your
rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works
for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code;
keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified
it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section
7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a

volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not
used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the

Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the
patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that
country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered
work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is
in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the
combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

Appendix B. GNU Lesser General Public License

GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.
As used herein, “this License” refers to version 3 of the GNU Lesser General Public License, and the “GNU GPL” refers to version 3 of the GNU General Public License.

“The Library” refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An “Application” is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A “Combined Work” is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the “Linked Version”.

The “Minimal Corresponding Source” for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The “Corresponding Application Code” for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.
You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.
If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.
The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.
You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
d) Do one of the following:
0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.
e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.
You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

END OF DOCUMENT