
# BioRadio Software Development Kit
# MATLAB® Driver Guide

**Telephone:** toll-free: 1-877-CleveMed (1-877-253-8363)
or (216) 791-6720
9:00 a.m. - 5:00 p.m. EST
Monday – Friday

**Fax:** (216) 791-6739

**E-Mail:** Customer Support: **support@CleveMed.com**
Sales: **sales@CleveMed.com**

**Web:** **http://www.CleveMed.com**

**Mailing Address:** Cleveland Medical Devices Inc.
4415 Euclid Avenue, Fourth Floor
Cleveland, Ohio 44103

## Introduction

The software DLL (Dynamic Link Library) interface to the BioRadio allows for programmatic interaction in Windows applications. Such interaction has been designed for The MathWorks MATLAB® software, allowing BioRadio communications and control from within MATLAB. This document describes the MATLAB M-files provided for such utility, and their appropriate usage.

Please consult the BioRadio 150 User's Guide for more in-depth information on the operation of your BioRadio device.

### MATLAB Version Compatibility

This version of the BioRadio driver is designed to work with MATLAB versions 6.5 (R13) and higher, but ***if you are using MATLAB version 6.5***, you may need to download MATLAB's **DLL Interface Library**. See The MathWorks article on *Calling Generic DLL Functions From MATLAB*:
http://www.mathworks.com/company/newsletters/news_notes/win03/patterns.html

### Included Files

As well as the documentation you're reading, the BioRadio SDK MATLAB Driver consists of the following Win32 DLL software interface to the BioRadio 150,

```
BioRadio150DLL.dll
```

along with the following M-files, (enclosing MATLAB functions of the same names,) which make calls to functions within the DLLs:

```
BioRadio150_Load.m
BioRadio150_Find.m
BioRadio150_Start.m
BioRadio150_Program.m
BioRadio150_Ping.m
BioRadio150_Read.m
BioRadio150_Stop.m
BioRadio150_Unload.m
BioRadio150_Test.m
Contents.m
```

and, additionally,

```
BioRadio150DLL.h
BioRadioConfig.h
```

### Multiple Devices and the Object Handle

The BioRadio SDK allows for operation of and acquisition from multiple BioRadios simultaneously. Each time a device object is created, (`calllib('BioRadio150DLL', 'CreateBioRadio')` in `BioRadio150_Load.m`,) a reference handle to the object is returned. Subsequent library function calls to operate

upon the device are provided its corresponding object's handle. However, in order to use multiple BioRadios simultaneously with the MATLAB driver, the provided M-files should be modified to avoid use of global variables.

### Data Collection Interval

When customizing `BioRadio150_Test` or writing original applications using the BioRadio, a timing structure is suggested to control the rate at which data is collected from the PC's communication port, where the Computer Unit has delivered it. The port has a finite buffer; only so much data can accumulate there between collections (when cleared) before the buffer fills and is incapable of holding more. Therefore, if the data collection interval is set too high, the buffer will overfill and data will be lost. The maximum time to which to set the data collection interval and avoid dropping packets is dependent upon: buffer size, PC speed, and what else is taking up processing time in the computer. 80ms (milliseconds) between reads is a typical value, and is the default value used in `BioRadio150_Test` (in the `pause` statement).

### Online Help

Documentation is also included in comments in the M-Files, preceding and in-line with the code. This documentation supports MATLAB's `help` function, so as to be accessible from the MATLAB command line by typing, for example, `help BioRadio150_Load`.

### Going Further

The MATLAB functions included provide fairly basic interaction with the device. Some additional functionality for device control is provided in the BioRadio 150 DLL. The MATLAB `calllib` function can be used to access all BioRadio 150 DLL functions, not only those included in these M-files. Please refer to the BioRadio SDK 150 DLL documentation for more.

## Function Descriptions

### BioRadio150_Load

Calling `BioRadio150_Load` is the first step in communicating with the BioRadio. `BioRadio150_Load` loads the BioRadio software interface, and prepares a few necessary data structures.

**Inputs:**

- `pathToDllDirectory`: The path string of the directory containing BioRadio150.DLL and BioRadio150.h

**Outputs:**

- `deviceHandle`: Reference handle to software device object

### BioRadio150_Find

Find the BioRadio 150 Computer Units connected the PC, returning their (COM) port names. Only devices which are not in use (Communication Started) will be listed.

**Prerequisite calls:**

`BioRadio150_Load`

**Outputs:**

- `Devices`: Names, (in a *cell array of strings*,) of the ports to which BioRadio 150 Computer Units are connected and not in use.

### BioRadio150_Start

Once loading has been accomplished, calling `BioRadio150_Start` will initiate data acquisition.

**Prerequisite calls:**

`BioRadio150_Load`

**Inputs:**

- `deviceHandle`: Reference handle to the software device object (returned in `BioRadio150_Load`)
- `portName`: String name of COM port to which the BioRadio is currently connected (*ex.,* '`COM4`' -- use `BioRadio150_Find` or the Windows Device Manager)
- `programDevice`: Boolean (`0`: false, `1`: true) whether the device configuration should be programmed to the file whose path string is the next parameter. If false, the device's current configuration will be pinged (acquired) instead.

- `pathToConfigFile`: If the previous parameter (`programDevice`) is true, this parameter must be provided; a full-path string to a valid BioRadio 150 configuration file (*ex.,* '`C:\CleveMed\CleveLabs\ConfigFilesLabECGI.ini`')

## BioRadio150_Ping

`BioRadio150_Ping` retrieves from the User Unit -- and populates the software object with -- the BioRadio's current device configuration.
This function is implicitly run in `BioRadio150_Start` if the `programDevice` parameter is set to `false`.

**Prerequisite calls:**

```
BioRadio150_Load
BioRadio150_Start
```

**Inputs:**

- `deviceHandle`: Reference handle to the software device object (returned in `BioRadio150_Load`)

## BioRadio150_Program

`BioRadio150_Program` programs the User Unit to, and populates the software object with, the device configuration specified in the file at the path provided. See the BioRadio 150 User's Guide for more information on configuration files.
This function is called explicitly in `BioRadio150_Start` if the `programDevice` parameter is set to `true`.

**Prerequisite calls:**

```
BioRadio150_Load
BioRadio150_Start
```

**Inputs:**

- `deviceHandle`: Reference handle to the software device object (returned in `BioRadio150_Load`)
- `pathToConfigFile`: The path string to a valid device configuration file.

## BioRadio150_Read

While data is being acquired, `BioRadio150_Read` retrieves, and returns as output, BioRadio data waiting at the computer's serial port.

**Prerequisite calls:**

```
BioRadio150_Load
BioRadio150_Start
```

**Inputs:**

- `deviceHandle`: Reference handle to the software device object (returned in `BioRadio150_Load`)

**Outputs:**

- FastInputsData: a two-dimensional array, with columns corresponding to enabled fast inputs, and each row a collected data point.
- SlowInputsData: a two-dimensional array, with columns corresponding to enabled slow inputs, and each row a collected data point. Slow inputs are sampled at 1/10$^{th}$ the rate of Fast Inputs

## BioRadio150_Stop

`BioRadio150_Stop` terminates acquisition and communication with the BioRadio.

**Prerequisite calls:**

```
BioRadio150_Load
BioRadio150_Start
```

**Inputs:**

- `deviceHandle`: Reference handle to the software device object (returned in `BioRadio150_Load`)

## BioRadio150_Unload

`BioRadio150_Unload` removes BioRadio configuration and execution information from MATLAB's memory space.

**Prerequisite calls:**

```
BioRadio150_Load
```

**Inputs:**

- `deviceHandle`: Reference handle to the software device object (returned in `BioRadio150_Load`)

## BioRadio150_Test

`BioRadio150_Test` has been provided as an example MATLAB program illustrating usage of the previously described functions.

`BioRadio150_Test` initiates communication with the BioRadio, programs the device to a specified configuration, then continuously acquires data, plotting selected inputs' data (in real-time,) and writing to output files in (ASCII Comma-Separated Values) format. Data from each *enabled* Fast Input, (Ch 1-8, Airflow, DC Aux,) is printed to one "Fast Data" output file, and data from each enabled Slow Input, (Accelerometer X, Accelerometer Y, Body Position, Heart Rate, and $SpO_2$) to another "Slow Data" file. A caller-specified number of enabled fast inputs are plotted to a chart. After completing 50 iterations, communication is stopped and memory used for the software device object is released.

**Inputs:**

- `portName`: String name of COM port to which the BioRadio is currently connected (*ex.,* '`COM4`')
- `pathToDllDirectory`: Path string to the directory that contains `BioRadio150DLL.dll` and `BioRadio150DLL.h` (*ex.,* '`C:\BioRadio SDK Matlab`')
- `pathToConfigFile`: Path string to a valid BioRadio 150 configuration file (*ex.,* '`C:\CleveMed\CleveLabs\ConfigFilesLabECGI.ini`')
- `pathToFastDataOutputFile`: Path string to a file (one will be created if nonexistent) to which data from the BioRadio 150's Fast Inputs will be output, in Comma-Separated Value format.
- `pathToSlowDataOutputFile`: Path string to a file (one will be created if nonexistent) to which data from the BioRadio 150's Slow Inputs will be output, in Comma-Separated Value format.
- `NumChannelsToPlot`: Number of enabled Fast Input channels (beginning from the first enabled) to plot to the real-time graph.