Boston Micromachines DM-SDK
Getting Started with MATLAB
Version 3.1.1

Introduction

The Boston Micromachines Deformable Mirror (**DM**) Software Development Kit (**SDK**) provides a common interface to all BMC products. It allows users to write one code base that can be used with any product. A MATLAB function interface is provided for rapid application and algorithm development.

Software Requirements

- Mathworks MATLAB 2008a or later
- Windows 7 64-bit or later OR Linux 64-bit (recommended Ubuntu 17.04 or later)

Usage

The DM-SDK MATLAB functions are installed in:

Windows:	C:\Program Files\Boston Micromachines\Bin64\Matlab
Linux:	/opt/Boston Micromachines/lib/Matlab

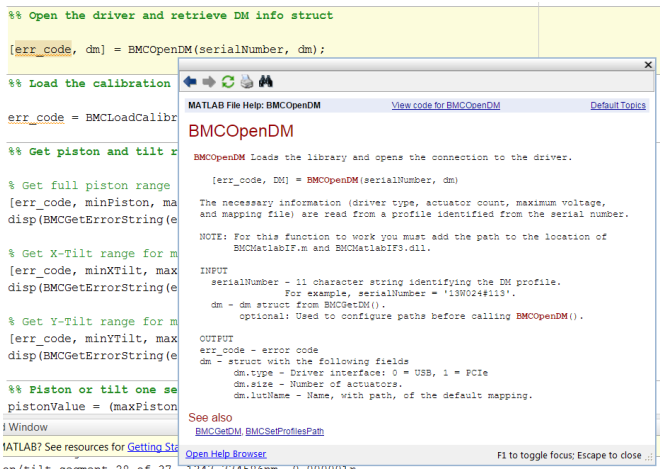
Add the following line at the top of your MATLAB script to use the DM-SDK:

```
if ispc
    addpath('C:\Program Files\Boston Micromachines\Bin64\Matlab')
else
    addpath(fullfile('/opt','Boston Micromachines','lib','Matlab'))
end
```

Documentation

This document is intended as a primer. See the documentation provided with the functions for detailed usage. Click on a function in the script and hit F1, or use the help command, for example:

```
help BMCOpenDM
```



Examples

Several MATLAB script examples are provided with the DM-SDK. They can be found in:

Windows:	C:\Program Files\Boston Micromachines\Examples\Matlab
Linux:	/opt/Boston Micromachines/Examples/Matlab

ExamplePokeAll.m

Opens the driver and gets the DM information struct. Then individually sets all actuators to a value in order. Closes the driver when finished.

ExampleAllFunctions.m

Calls more of the available functions to demonstrate their use.

ExampleTiltSegments.m

Demonstrates open loop control of a segmented Hex DM. Individually pistons and tilts all segments in order.

ExamplePCIe.m

Demonstrates the features of the PCIe card. Configures the PCIe card to automatically output a sequence of actuator poke values to the DM. Then configures it to a pattern and dither waveform.

Functions

Below is a list of functions for controlling the mirror and drive electronics. These functions and the necessary dynamic link library (DLL) are installed with the Boston Micromachines SDK in the folder listed above.
See the documentation provided with the functions for detailed usage.

General Operation	
BMCOpenDM	Load the library, open the connection to the driver, and get DM struct needed for some functions and some useful settings.
BMCGetDM	Get DM struct needed for some functions and some useful settings.
BMCGetDefaultMapping	Retrieve the default driver mapping set in the DM profile.
BMCGetErrorString	Returns a string describing code for input error.
BMCCloseDM	Close the connection to the driver.
BMCCloseDriver	Deprecated. Unload the library and close all connections to the driver.

Actuator Control	
BMCPokeDM	Set a single actuator to a value leaving all others unchanged.
BMCSendData	Send data to the mirror using the default mapping.
BMCSendDataCustomMapping	Send data to the mirror using the given custom driver mapping.
BMCGetActuatorData	Get a copy of the full array of the last command values.

Open Loop Segment Control	
BMCGetSegmentPistonRange	Get the Piston range for a given X-Tilt, Y-Tilt.
BMCGetSegmentXTiltRange	Get the X-Tilt range for a given Piston, Y-Tilt.
BMCGetSegmentYTiltRange	Get the Y-Tilt range for a given Piston, X-Tilt.
BMCLoadCalibrationFile	Read calibration file for user unit to DAC value conversion.
BMCSetSegment	Set the Piston, X-Tilt, Y-Tilt, of a single segment of an SLM.

PCIe Driver Specific	
BMCConfigureDither	Configure the driver hardware for dithering functionality.
BMCConfigureSequence	Configure the driver hardware for sequencing functionality.
BMCEnableDither	Start the driver hardware's dithering functionality.
BMCEnableSequence	Start the driver hardware's sequencing functionality.

These functions will return ERR_NOT_IMPLEMENTED if used with a USB driver.

SDK Configuration	
BMCVersionString	Returns the DM-SDK version number.
BMCConfigureLog	Override the default log file with your own.
BMCSetCalibrationsPath	Override the default calibration search path with your own.
BMCSetMapsPath	Override the default mapping install directory with your own.
BMCSetProfilesPath	Override the default profile install with your own.

These functions take a DM struct from BMCGetDM which should then be passed to BMCOpenDM for changes to take effect.

Troubleshooting

Linux

Error using loadlibrary

The following error comes up when opening a DM in MATLAB 2017a on Ubuntu 17.04:

```
Error using loadlibrary
There was an error loading the library "/opt/Boston
Micromachines/lib/Matlab/libBMCMatlabIF3.so"
'/opt/Boston Micromachines/lib/Matlab/libBMCMatlabIF3.so.3.0.0' is not a valid
shared library.
```

To confirm the problem, run the following command, and see the following errors:

```
>> !ldd '/opt/Boston Micromachines/lib/Matlab/libBMCMatlabIF3.so'
/opt/Boston Micromachines/lib/Matlab/libBMCMatlabIF3.so:
/opt/MATLAB/R2017a/sys/os/glnxa64/libstdc++.so.6: version `CXXABI_1.3.9' not
found (required by /opt/Boston Micromachines/lib/Matlab/libBMCMatlabIF3.so)
/opt/Boston Micromachines/lib/Matlab/libBMCMatlabIF3.so:
/opt/MATLAB/R2017a/sys/os/glnxa64/libstdc++.so.6: version `CXXABI_1.3.9' not
found (required by /opt/Boston Micromachines/lib/libBMC.so.3)
/opt/Boston Micromachines/lib/Matlab/libBMCMatlabIF3.so:
/opt/MATLAB/R2017a/sys/os/glnxa64/libstdc++.so.6: version `GLIBCXX_3.4.21' not
found (required by /opt/Boston Micromachines/lib/libBMC.so.3)
…
```

To work around the problem, run MATLAB with the system C++ library preloaded as follows:

```
LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libstdc++.so.6 matlab
```