

## Installation of the Windows-Java-JNI IOReadWrite Interface into MATLAB 7

1. Copy the **inpout32.dll** interface and driver to c:\windows\system32  
(see [http://www.logix4u.net/inpout\\_theory.htm](http://www.logix4u.net/inpout_theory.htm))

2. Use the Windows search facility to find the location of MATLAB's Java CLASSPATH.  
Do this by searching the \$matlabroot directory for "rt.jar". On my Windows XP Pro machine with MATLAB 7.04, "rt.jar" was found at:

C:\Program Files\MATLAB704\sys\java\jre\win32\jre1.5.0\lib

Given Java's peculiar "one-up directory search" specification, this yields a CLASSPATH of:

CLASSPATH = C:\Program Files\MATLAB704\sys\java\jre\win32\jre1.5.0\

2. Copy **usdportio.dll** to CLASSPATH\bin

For example:

copy usportio.dll C:\Program Files\MATLAB704\sys\java\jre\win32\jre1.5.0\bin

3. Create a subdirectory (i.e., folder) at CLASSPATH\lib\usd.

4. Copy **IOReadWrite.jar** to CLASSPATH\lib\usd

For example:

copy IOReadWrite.jar C:\Program Files\MATLAB704\sys\java\jre\win32\jre1.5.0\lib\usd

5. Start MATLAB 7

6. From the MATLAB 7 command line:

type: *which classpath.txt*

This reveals the location of the Java ClassPath definitions file.

You should see something like:

C:\Program Files\MATLAB704\toolbox\local\classpath.txt

7. If 'which classpath.txt' returned a reasonable response, then edit the ClassPath definitions file so that Java can find the new class and native library:

- a. Use the MATLAB editor to open classpath.txt and
- b. add the following line at or near the top of the file:

*\$matlabroot/sys/java/jre/win32/jre1.5.0/lib/usd/IOReadWrite.jar*

- c. save your work and exit the editor

8. Shutdown and restart MATLAB to reinitialize the Java ClassPath definitions.

After restarting, type: *javaclasspath ---* to verify successful editing of classpath.txt

You should be able to see the *IOReadWrite.jar* file near the top of the list.

9. Load and test the **IOReadWrite** class as follows:

```
import usd.IOReadWrite;      %load the IOReadWrite class into MATLAB's workspace

ioport = IOReadWrite;      %create an IOReadWrite object (named 'ioport')

ioport.installIO32;        %Ask inpout32.dll to install kernel-level driver to allow this object
                             %to access the low-level hardware ports. Returns TRUE (1)
                             %if the request was successful, FALSE (0) if unsuccessful.

ioport.write(888,123);      %write a value of 123 to the printer's data port
                             %same as: outportb(888, 123)

X= ioport.read(888)         %read the same port that you just wrote to...Should return
                             %the same value (i.e., 123). Same as: inportb(888)

ioport.removeIO32;          %uninstall the inpout32.dll's kernel-level driver
```

10. Timing tests reveal that **IOReadWrite.write()** has a latency of approximately 65 microseconds.

11. To verify the the Java **IOReadWrite** class has been loaded:

```
[m,x,j] = inmem;
```

Returns modules currently cached in MATLAB memory.

```
m = M-files
x = MEX-files
j = Java classes
```

List the contents of the 'j' array and look for *usd.IOReadWrite*

12. Shutdown the IOReadWrite object by using *clear ioport* (or, clear all).