Running and Debugging MPJ Express with Eclipse

User Guide

7th February, 2014

Document Revision Track

Version	Updates	By
1.0	Initial version document	Rizwan Hanif, Amjad Aziz, Aamir Shafi
1.1	Complete step details, Steps for running and debugging in hybdev configuration	Aleem Akhtar, Aamir Shafi, Mohsan Jameel

Table of Contents

List of Figures

1. MPJ Express Debugger – Eclipse Plugin

The MPJ Express Debugger is based on Eclipse debugger. Client/Server design of Eclipse debugger allows user to debug programs running locally on your workstation as well as programs running remotely on other systems in the network. In case of local debugging, program will be launched on your workstation or in case of remote debugging; program will be launched on a system that is accessible through a network where you want to launch your job. Current version of MPJ Express Debugger supports both local and remote debugging modes.

1.1 Local Debugging

In local debugging, launched program and debugger client runs on the same workstation which makes local debugging simplest and most common kind of debugging. You can then use breakpoints, step into, step out, or step return features to debug your program. In local debugger both MPJ Express process and MPJ Express Debugger should be on same machine.

1.2 Remote Debugging

In remote debugging, you launch MPJ Express program on your workstation and MPJ Express debugger client runs on some other system on the same network. Remote debugging comes handy when you are developing program for a device that cannot host the development platform. It is also useful when debugging MPJ Express programs running on cluster. For Multicore mode of MPJ Express remote debugging user specify host name and port of running application while for Cluster mode user specify `mpjdev.conf` file path.

Note: To use remote debugging, you must be using a Java VM that supports this feature i.e. JRE 1.6 or higher.

2. Getting Started with MPJ Express

This section shows how MPJ Express programs can be executed in the multicore and cluster configuration using MPJ Express Debugger. Current version (1.0.0) does not support debugging of applications using native device.

Complete video guide can be found at http://vimeo.com/86049819

2.1 Pre-requisites

- Java 1.6 (stable) or higher
- MPJ Express (version 0.40 or higher)
- Eclipse IDE (version 3.6 or higher)
- Apache ant 1.6.2 or higher (For those who are interested in compiling source code)
- Perl (Optional): MPJ Express needs Perl for compiling source code because some of the Java code is generated from Perl templates. The build file will generate Java files from Perl templates if it detects Perl on the machine. It is a good idea to install Perl if you want to do some development with MPJ Express.

2.2 Installation of MPJ Express Debugger

1. Download MPJ Express Debugger from MPJ Express official website and unpack it
2. Copy MPJ_Express_Debugger.jar file and paste in Eclipse plugins directory
3. Run eclipse as an Administrator
4. Go to Run→Run Configurations menu and check if there is MPJ Express Application option added or not

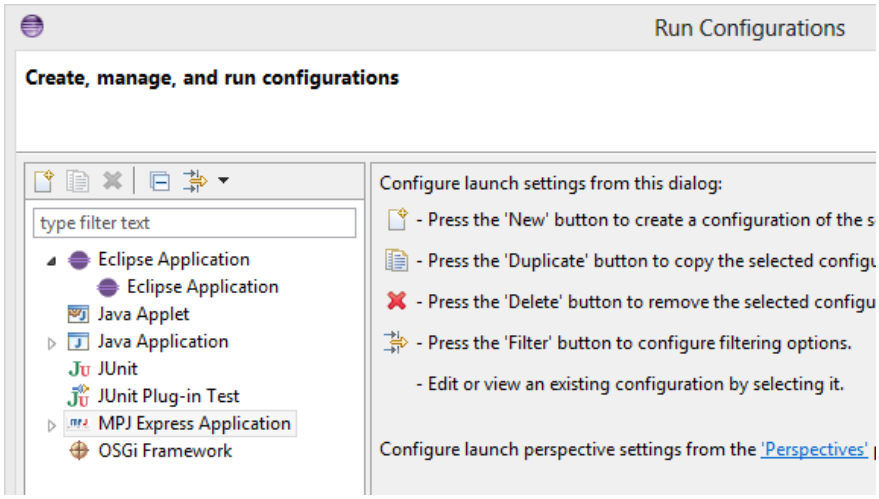

Figure 1: MPJ Express Application option in Run Configurations

2.3 MPJ Express New Application

MPJ Express can be run in Eclipse using three configurations namely multicore, cluster and hybrid configuration. Steps for installing MPJ Express in all configurations are same but differ where we provide device field for all the configurations.

1. Download MPJ Express and unpack it.
2. Set Environment Variables (MPJ_HOME, PATH). You can follow **Windows User Guide** for Windows and **Linux User Guide** for Linux for detail steps on setting Environment Variables.
3. Create a new Java Project. Name it as "MPJEclipseTest" and click on Next (Do not click on finish at this point).

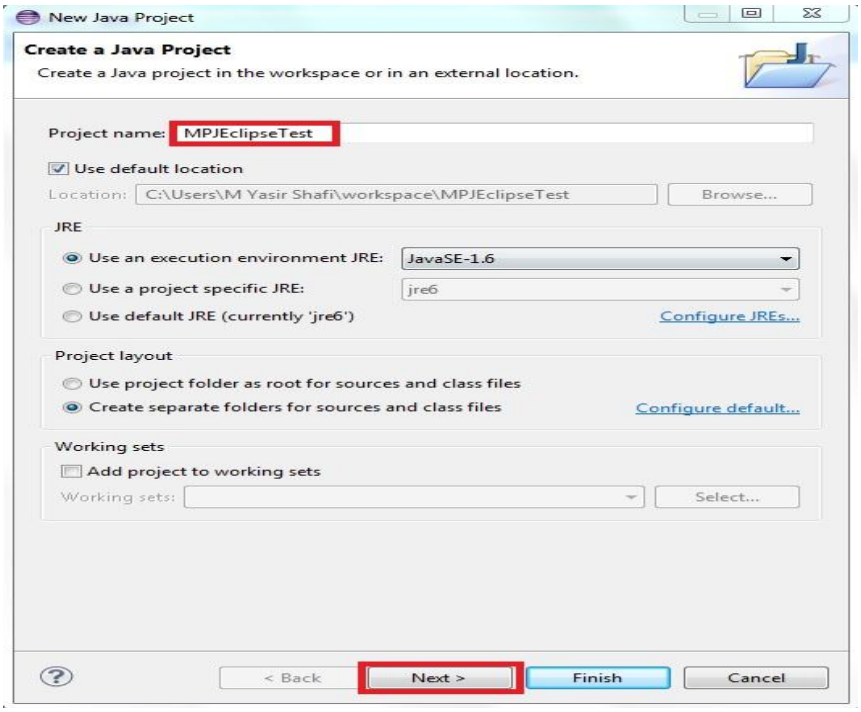

Figure 2: Create a new java project

4. On Java setting page for this project, click on Libraries tab and select "Add External JARs". Look for mpj.jar in lib folder of "MPJ Express" root directory. Click finish

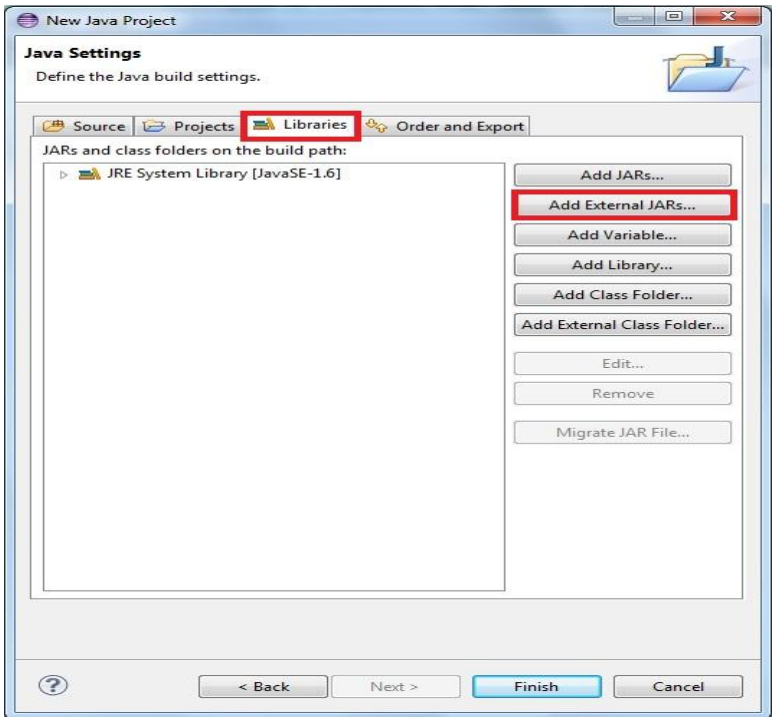

Figure 3: Add External Libraries

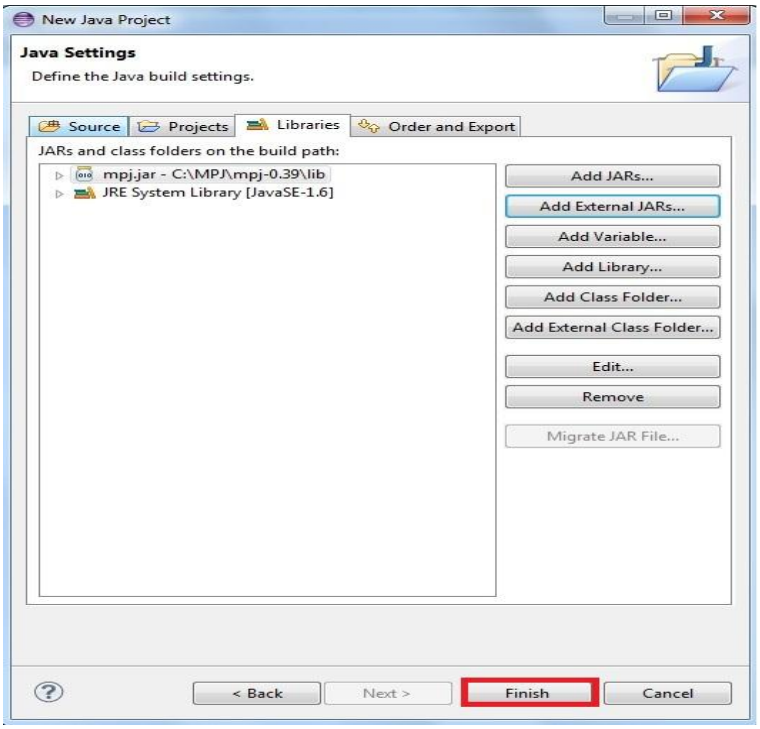

Figure 4: Click finish to add project

5. Add a new Java class and write HelloWorld MPJ Express program in it

```
import mpi.*;

public class HelloWorld {

	public static void main(String args[]) throws Exception {
		MPI.Init(args);
		int me = MPI.COMM_WORLD.Rank();
		int size = MPI.COMM_WORLD.Size();
	System.out.println("Hi from <"+me+">");
MPI.Finalize();
	}
}
```

6. To compile this program in Eclipse, Right click on Java Class and go to Run as → Run Configurations

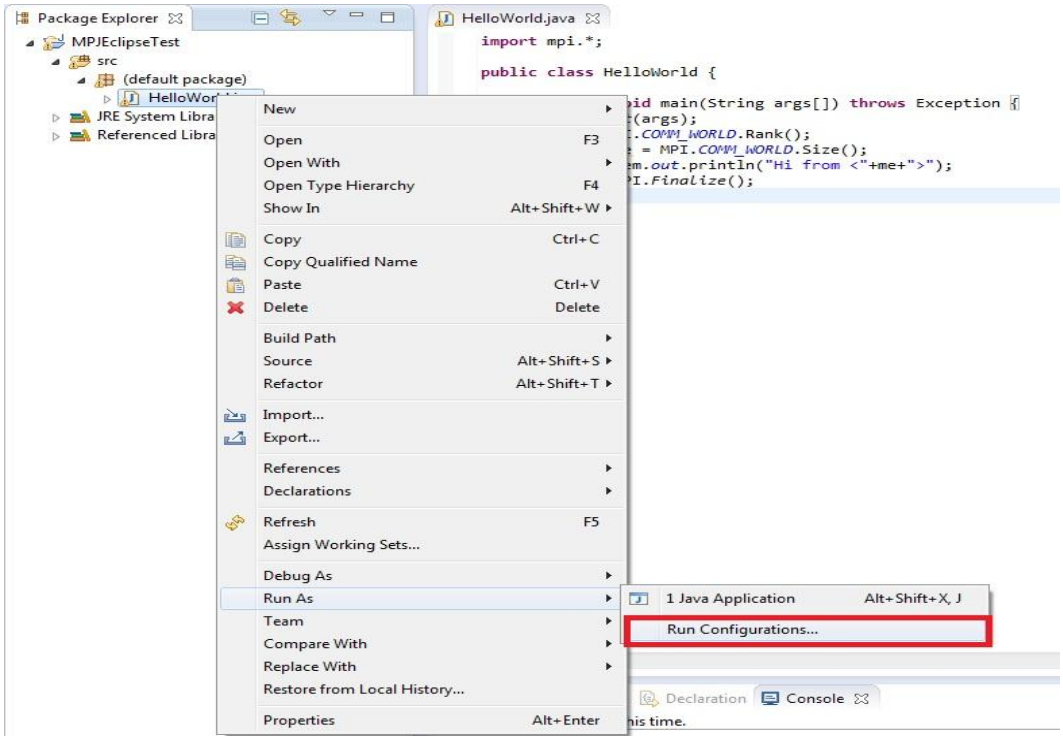

Figure 5: Run Configurations

7. Create, manage and run configurations will be opened. Double click on **MPJ Express Application**. A new run configuration named "HelloWorld" will be added with different tabs.

8. Select MPJ Parameters tab. Here you can enter number of processes you want to want run (np), device you want to use (device [multicore, niodev, hybdev]) and additional arguments if required.

9. Browse for MPJ Express root directory and set it for MPJ_HOME field

10. Enter number of processes (np). See figure-9

11. Click Apply

2.4 Running MPJ Express in the Multicore Configuration

This section outlines steps to execute parallel Java programs in the multicore configuration.

1. Follow steps from 1 to 11 of above section 4.3
2. Enter multicore in device field of MPJ Parameters tab in Run Configurations
3. Click apply and run
4. MPJ Express Application will start running in multicore mode

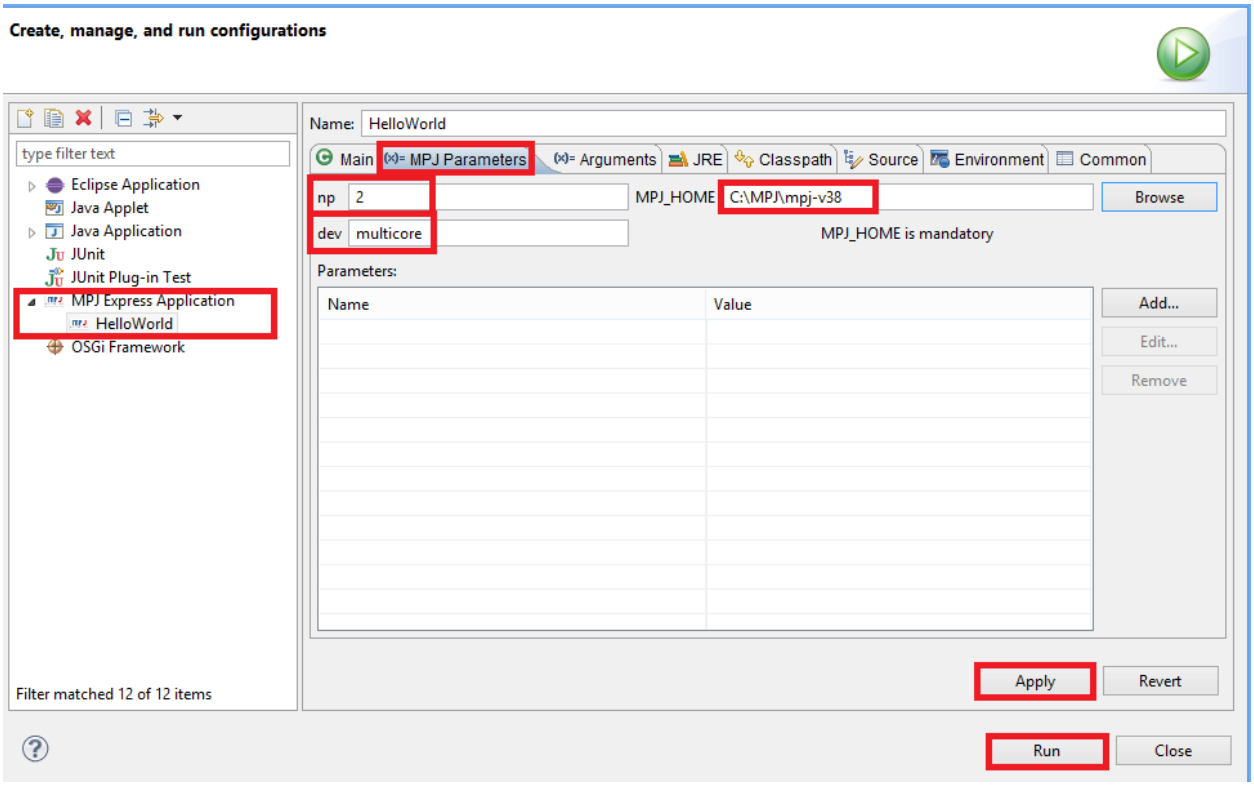

Figure 6: MPJ Parameters for multicore Configuration

2.5 Running MPJ Express in the Cluster Configuration

This section outlines steps to execute parallel Java programs in the cluster configuration with niodev and hybdev communication device driver.

1. Write a machines file stating machine name, IP addresses, or aliases of the nodes where you wish to execute MPJ Express processes. Save this file as 'machines' in mpj-user directory. This file is used by scripts like mpjboot, mpjhalt, mpjrun.bat and mpjrun.sh to find out which machines to contact. For detail configuration steps you can follow section-2.3 of Linux/Windows guide of MPJ Express.

   Suppose you want to run a process each on 'machine1' and 'machine2', then your machines file would be as follows

   ```
   machine1
   machine2
   ```

   Note that in real world, 'machine1' and 'machine2' would be fully qualified names, IP addresses or aliases of your machine

2. Start the daemons

   - Windows users

     a. Run %MPJ_HOME%/bin/installmpjd-windows.bat. Users of Windows Vista and 7 needs to "Run as Administrator". Also, for running properly, users of Windows Vista and 7 might need to turn off their firewall.

     b. Go to Control-Panel→Administrative Tools→Services→MPJ Daemon and start the service.

   - Linux users

     a. Open Terminal and move to current project directory where machines file is placed. Run `mpjboot machines` command. Daemons will start running

3. Follow steps from 1 to 11 of section 4.3
4. Enter niodev in device field of MPJ Parameters tab in Run Configuration
5. Click apply and Run

   MPJ Express Application will start running in niodev mode

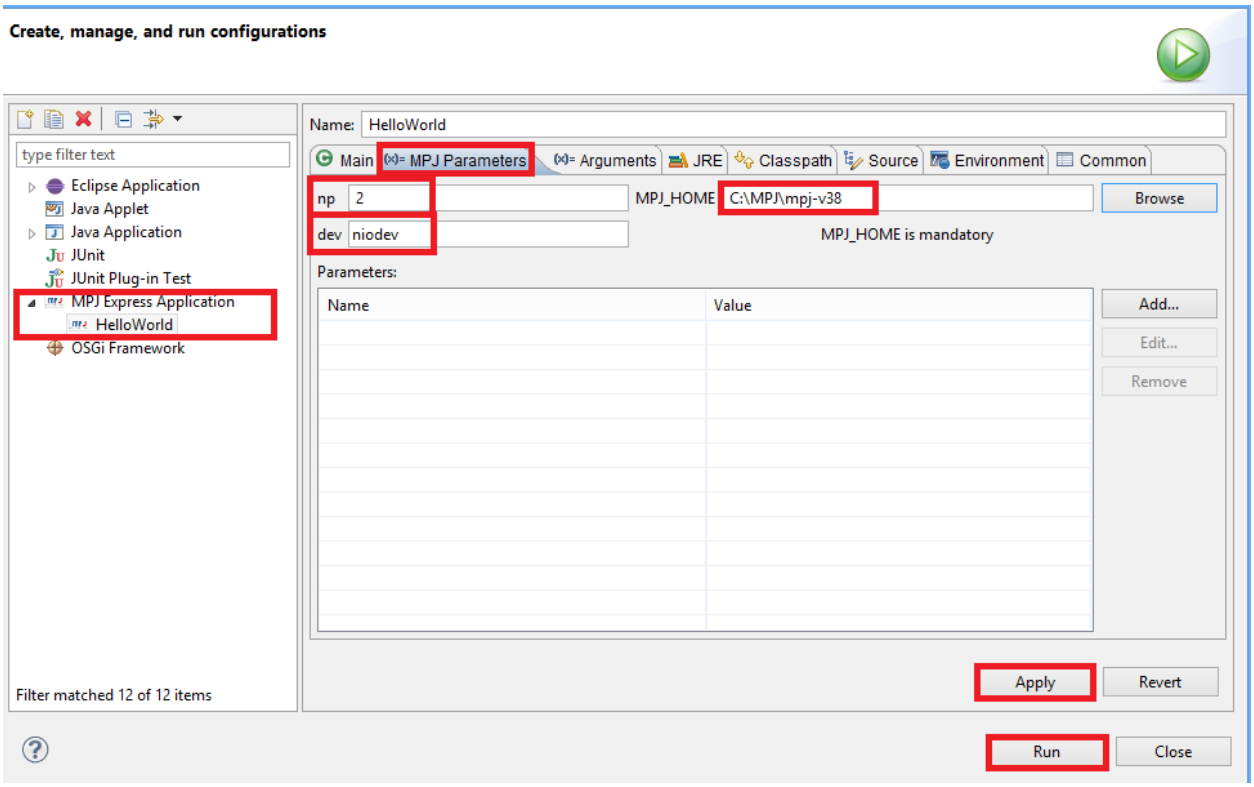

Figure 7: MPJ Parameters for niodev device

To execute parallel Java programs in the hybrid configuration with hybdev communication device driver.

1. Follow steps from 1 to 3 of section 4.5
2. Enter hybdev in device field of MPJ Parameters tab in Run Configuration
3. Click apply and Run

   MPJ Express Application will start running in hybdev mode

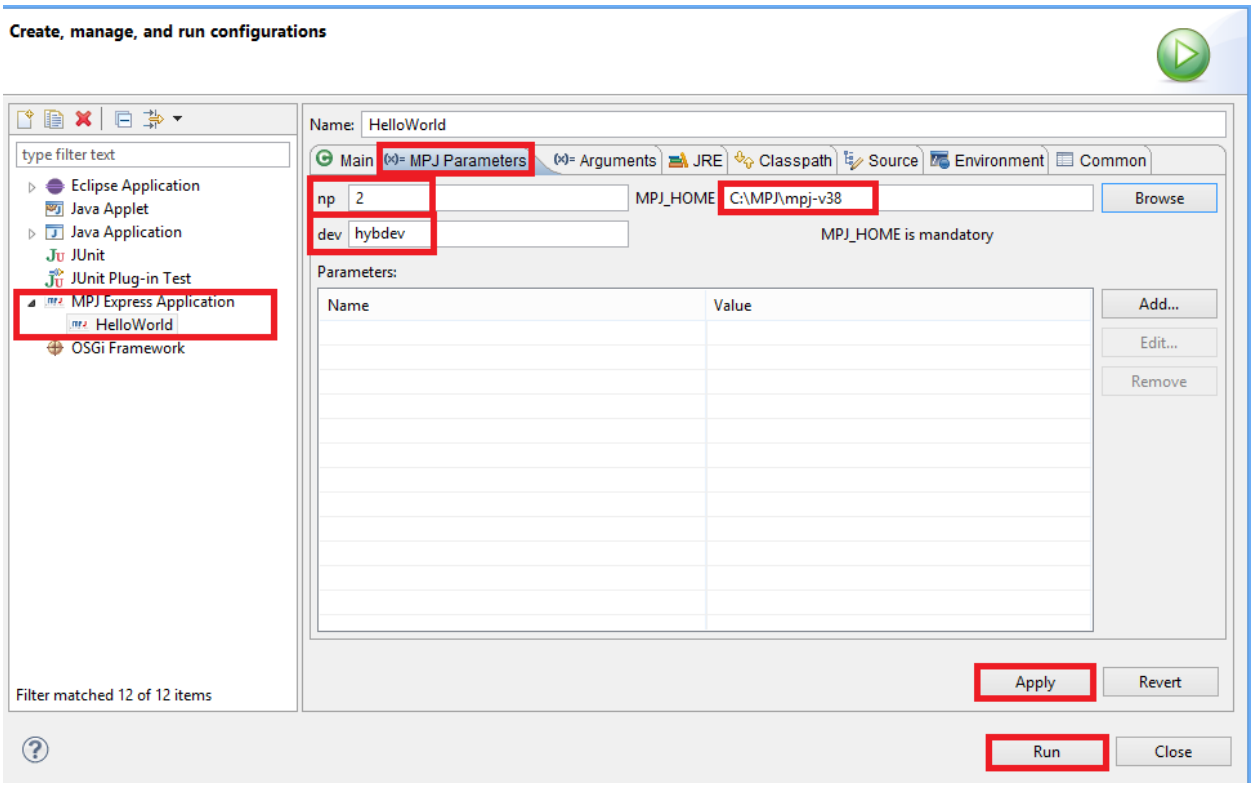

Figure 8: MPJ Parameters for hybdev device

3. MPJ Express Applications Debugging

Users can debug MPJ Express applications using MPJ Express Debugger for both configurations modes [multicore, cluster]. Steps for both modes are same. There are two types of debugging that current MPJ Express Debugger supports

- Local Debugging
- Remote Debugging

Detail steps can be followed below

3.1 Local Debugging

1. Create a new MPJ Express Application in eclipse. You can follow steps from 1 to 11 of section 4.3
2. Introduce breakpoints where you want to stop your MPJ Express Application.
3. Click on Run→ Debug Configurations. Create, manage, and run configuration window will open. Here you can see two options for MPJ Express [MPJ Express Application and Remote MPJ Express Application]. Since we are debugging in local mode so double click on MPJ Express Application option. A new configuration with "HelloWorld" will be added.
    a. Browse for MPJ Express root directory and set field MPJ_HOME
    b. Enter number of processes in np field
    c. Enter Device name [multicore, niodev, or hybdev] in device field.
    d. Add a new parameter with **Name = debug** and **value = 8000**. Note 8000 is port number, you can use whatever port you want provided the port you chose if available.
4. Click Apply and Debug

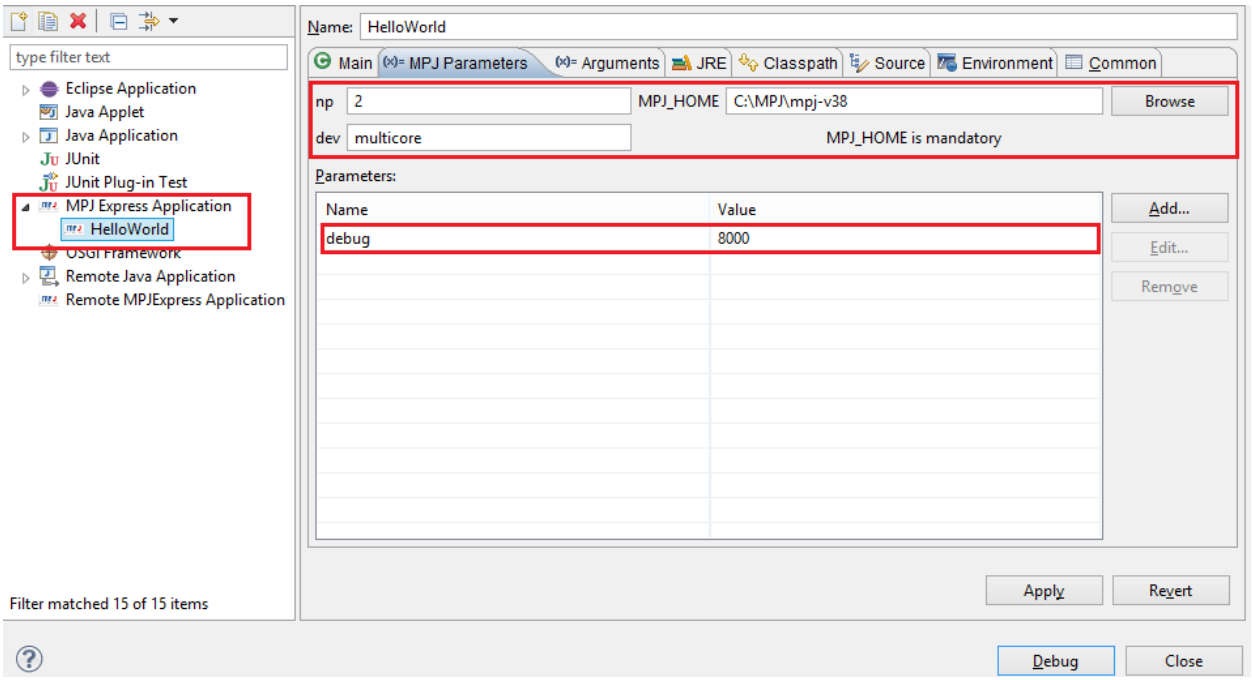

Figure 9: MPJ Parameters for Local Debugging

5. Application will open in Debug Perspective with all processes suspended on breakpoint.

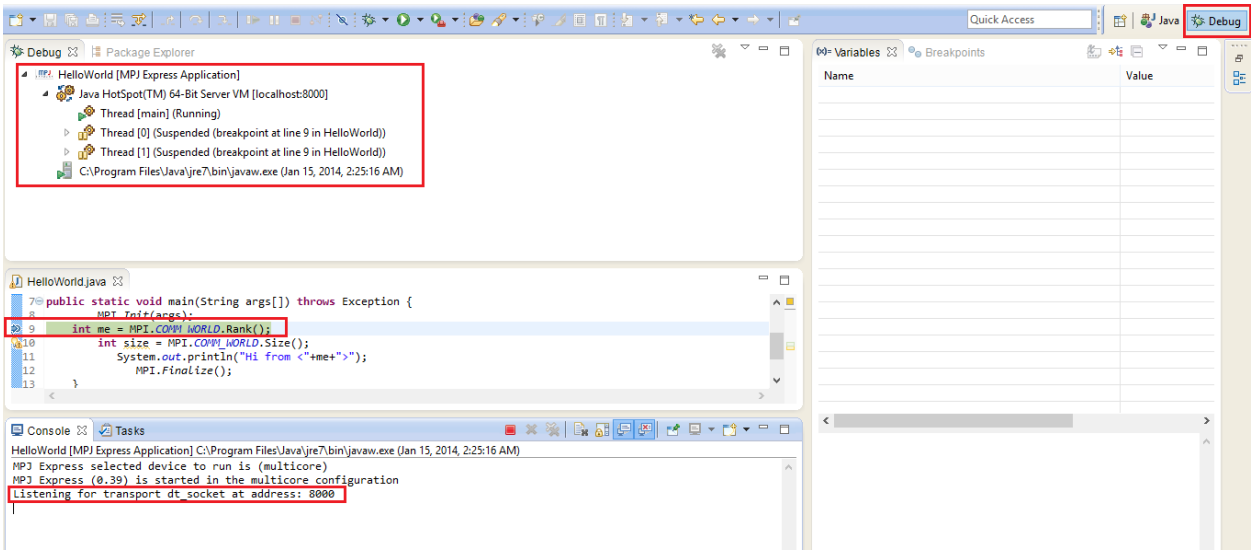

Figure 10: Debug Perspective for MPJ Express Debugging

6. You can now select each thread and use Step Into, Step Over and Step Return functionality to debug your application.

Note: For niodev and hybdev you will need to write machines files and start daemons to work it properly. To get help on how to start daemons follow step 1 and Step 2 of section 4.5

3.2 Remote Debugging

There are two types of Remote Debugging that current MPJ Express Debugger supports

- Multicore Remote Debugging
- Distributed Remote Debugging

Steps for both types are almost same.

1. Create a new MPJ Express Application in eclipse. You can follow steps from 1 to 11 of section 4.3
2. Introduce breakpoints where you want to stop your MPJ Express Application.
3. Click on Run →Run Configurations. Create, manage, and run configuration window will open. Double click on MPJ Express Application option. A new configuration with "HelloWorld" will be added.
    a. Browse for MPJ Express root directory and set field MPJ_HOME
    b. Enter number of processes in np field
    c. Enter Device name [multicore, niodev, hybdev] in device field.
    d. Add a new parameter with **Name = debug** and **value = 8000**. Note 8000 is port number, you can use whatever port you want provided port you chose if available.
4. Click Apply and Run

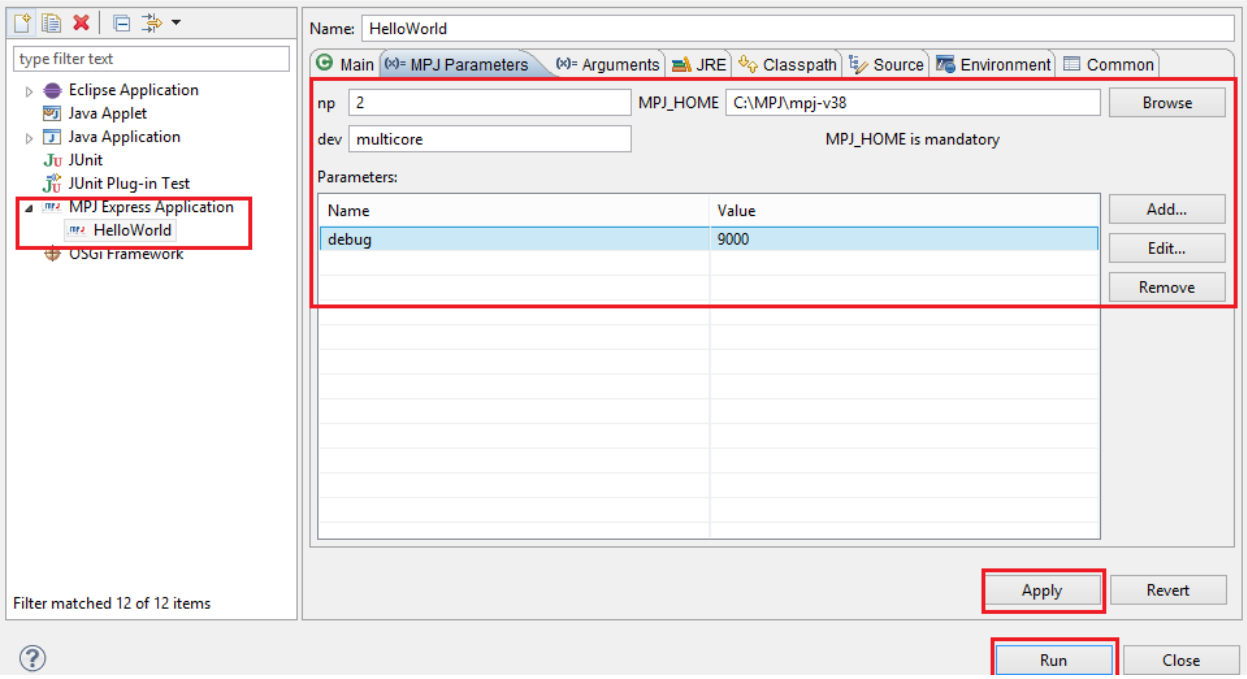

Figure 11: MPJ Parameters for Remote Debugging

5. Application will open in Debug perspective with application listening to your port defined in debug parameter.

6. Now go to Run → Debug Configuration and select Remote MPJ Express Application and double click on it. A new configuration will be added

➢ **For Multicore Remote Debugging**

   a. Open connect tab and select multicore as Connection type
   b. Enter Host value, localhost or IP address of system where MPJ Express application is running
   c. Enter Port value. This should be same what you defined in your debug parameter of MPJ Express Parameters
   d. Click Apply and Debug

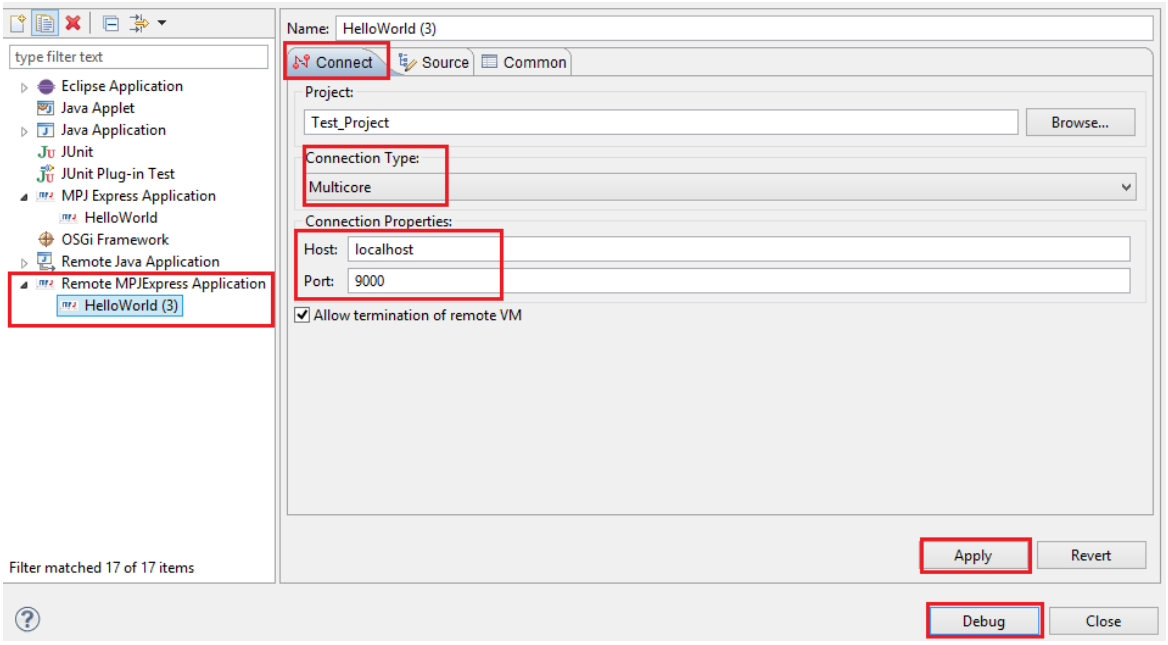

Figure 12: Connect Parameters for Remote Multicore Configuration

MPJ Express Application will start remotely debugging in multicore mode

➢ **For Distributed (niodev/hybdev) Remote Debugging**

   a. Open connect tab and select Distributed Memory as Connection type
   b. Browse for Configuration file (mpjdev.conf). By default path will be set to user directory where it is placed. mpjdev.conf must be accessible to workstation where your debug client is running.
   c. Click Apply and Debug

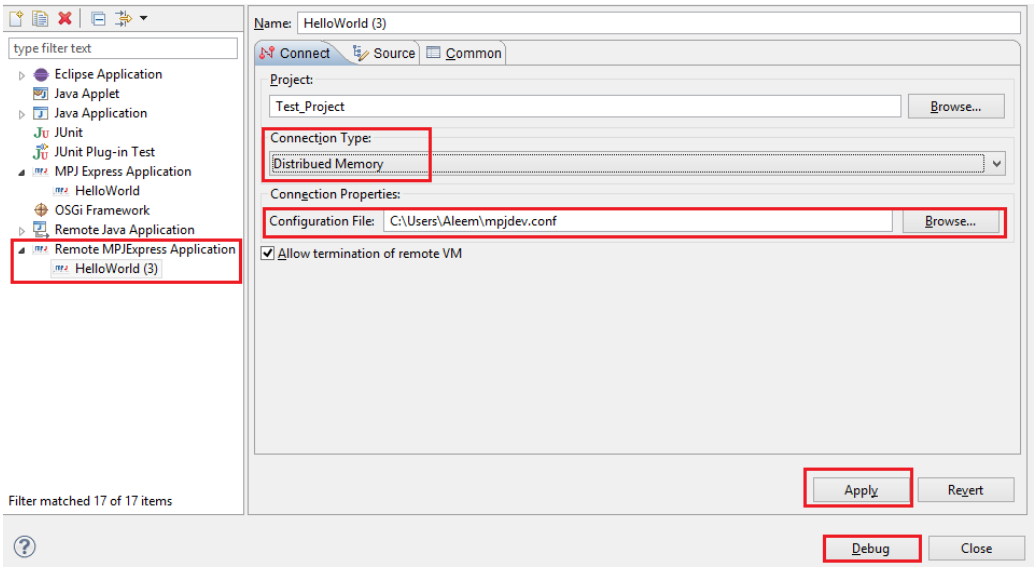

Figure 13: Connect Parameters for Remote Distributed Debugging

MPJ Express application will start running in niodev/hybdev configuration debug mode. You can debug your application using Step Into, Step out, Step Return functionalities.

Note: For niodev and hybdev you will need to write machines files and start daemons to work it properly. To get help on how to start daemons follow step 1 and Step 2 of section 4.5

4. Known Issues and Limitations

A list of known issues and limitations of the MPJ Express Debugger are listed below.

1. Users of the Windows Vista and 7 might find install mpjd-windows.bat script not working directly. In such case, try executing with "Run as Administrator". If the issue still persists, try executing the script mpjdaemon.bat directly or turning off the firewall.

2. If you face trouble with Remote VM Connection failed. Run Eclipse as Administrator. Linux user can start Eclipse as Administrator through command line.

   Failed to connect to remote VM. Connection refused.
   Connection refused

3. If MPJ Express application fail to run and get stuck at this line: MPJ Express (0.40) is started in the cluster configuration
   Then just restart daemons and your application will start running fine.
4. If you are running your applications on Rocks cluster then you might get below mentioned error. To solve this just add a new parameter in MPJ Parameters Tab with **name=wdir and value=absolute path to bin directory of your project**.

```
java.io.IOException: Cannot run program "java" (in directory "/state/partition1/home2/MPJ_Work/MPJ_Test"): error=2,
        at java.lang.ProcessBuilder.start(ProcessBuilder.java:1029)
        at runtime.daemon.HybridDaemon.startNewProcess(HybridDaemon.java:323)
        at runtime.daemon.HybridDaemon.<init>(HybridDaemon.java:124)
        at runtime.daemon.MPJDaemon.<init>(MPJDaemon.java:272)
        at runtime.daemon.MPJDaemon.main(MPJDaemon.java:1242)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:601)
        at runtime.daemon.WrapperSimpleApp.run(WrapperSimpleApp.java:187)
        at java.lang.Thread.run(Thread.java:722)
Caused by: java.io.IOException: error=2, No such file or directory
        at java.lang.UNIXProcess.forkAndExec(Native Method)
        at java.lang.UNIXProcess.<init>(UNIXProcess.java:135)
        at java.lang.ProcessImpl.start(ProcessImpl.java:130)
        at java.lang.ProcessBuilder.start(ProcessBuilder.java:1021)
        ... 10 more
```

Figure 14: State partition error

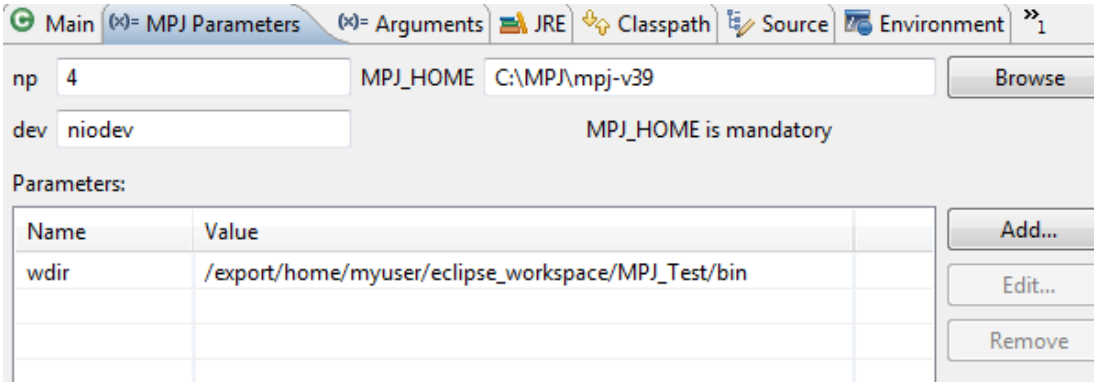

Figure 15: Solution of State partition error

5. Contact and Support

For help and support, join and post on the MPJ Express mailing list (mpjexpress-users@lists.sourceforge.net). Alternatively, you may also contact us directly:

1 Aamir Shafi (aamir.shafi@seecs.edu.pk)

2 Mohsan Jameel (mohsan.jameel@seecs.edu.pk)

3 Bryan Carpenter (bryan.carpenter@port.ac.uk)

4 Mark Baker (http://acet.rdg.ac.uk/~mab)

5 Guillermo Lopez Taboada (http://www.des.udc.es/~gltaboada)