

Quick Start Guide for the SPECpower_ssj2008 Benchmark.

1) Identify a system to be tested (the system under test, or SUT) and a system to control the test (benchmark controller system).

2) Attach both systems to a network, and give each system's network adapter a different ip address (example: SUT uses 192.168.1.1 controller uses 192.168.1.2). Verify both systems can ping each other. If both systems have been set up with hostname resolution (DNS, /etc/hosts, etc.), this benchmark allows the use of hostnames as well as ip addresses. In this case, ensure that you are able to ping each system from the other by hostname.

3) Locate/download versions of Java appropriate for the architectures and operating systems of the SUT and controller systems.

4) Install the Java versions on both the SUT and the Benchmark Controller system, making a note as to the exact path to the java executable on each system. Example: "c:\java-1.5.0_10\bin\java.exe"

5a) Using the Installation CD, install the Installshield benchmark package named "setup.jar" onto both the SUT and the benchmark controller with the following command:

```
java -jar setup.jar <- will install using a GUI assuming you are in a graphical environment.
```

```
java -jar setup.jar -i console <- will install using a text-based method.
```

If one of the above commands doesn't work, try prepending the path of the java executable you installed in step 4 at the beginning of the command. Default java versions may not support the full command set to install this benchmark correctly, thus necessitating that we use the appropriate Java version installed in step 4 of this Guide.

Please note that you are given the option to install the benchmark anywhere you would like on your systems, and are not confined to the default pathname.

5b) As an alternate methodology, you can copy the benchmark directories (ccs, ptd, ssj, images, and redistributable sources) from the installation CD to both the SUT and the benchmark controller, preferably all placed under a directory you create called "SPECpower_ssj2008".

6) On the SUT, look inside the SSJ subdirectory. Edit the SPECpower_ssj_config.props file to reflect the configuration details of the system you plan to benchmark. Editing this props file is not required unless you plan to submit your results to SPEC, but you may find the usefulness of the benchmark reports enhanced when using the correct configuration information, especially after you have run the benchmark several times and have made configuration changes.

7) Edit the runssj.bat or runssj.sh (depending on your OS type) on the SUT. If using Windows, change the "set JAVA=" line near the top of the batch file to reflect the exact path to the java executable that you installed in step 5. If using Linux/Unix, either make sure the java executable is in your PATH, or edit the "JAVA=" to reflect the fully qualified path of the java executable.

8) On the controller system, change to the PTD subdirectory. We will test basic functionality by starting the power/temperature daemon in dummy mode. Examine the run scripts appropriate for your Operating System, which might be easier for those who aren't comfortable with command line flags. There are two sets, "runpower" and "runtemp", and they are both set up to use dummy mode initially. We will start ptd for measuring power in dummy mode first, so execute the "runpower" script in one command window after you have changed into the ptd directory. Once ptd has initialized and is awaiting connections, open a second command window on your controller system, change directories to the ptd directory, and execute the appropriate "runtemp" script for your OS. Once the dummy ptd running in temperature mode has started and is awaiting connections, we can proceed to the next step.

*An alternate methodology is to run ptd from the command line. Execute the ptd version appropriate to your OS with the following arguments: "<name of ptd executable> 0 <phony serial device>". This should start a dummy power daemon on your controller system. For a Windows-based system, use COM1 for the <serial device>. For Linux, try /dev/ttyS0. (An actual serial port is not needed for this step, so any arbitrary string can be used for the serial device name). Now we need to start a dummy temperature daemon as well, and we will do so by opening a second command line, changing to the ptd directory, and running ptd with the following arguments: "<name of ptd executable> -p 8889 -t 1000 <phony serial device>". This should start a dummy temperature daemon.

9) Still on the controller system, change to the CCS directory and edit the runCCS.bat or runCCS.sh file. If using Windows, change the "set JAVA=" line to reflect the path to the java executable you installed in step 3 and 4. Ensure that the "set SSJHOME=" line is set to reflect the actual path where SSJ is installed on your controller system, as the benchmark report won't complete properly at the end of the test without being able to locate certain graphing libraries in the SSJ directory. If using Linux/Unix, use the above procedure but modify the "JAVA=" and "SSJHOME=" lines."

10) Edit the ccs.props file on the controller system, and change the line "ccs.wkld.ssj_dir.IP =" line to the IP address you set for your SUT in step 2 (or, if using hostname resolution mentioned in step 2, specify the hostname of the SUT here). At this time, you can also edit the ccs.config.hw.*, ptd.pwr1.config.*, and ptd.temp1.config.* lines to match the configuration of the controller system, as well as the power analyzer and temperature sensor you plan to use. Doing this part is not required unless you plan to submit the results to SPEC.

11) On the SUT, start SSJ by executing the runssj script appropriate for your Operating System. When it is finished loading, you should see the message "SSJ instances ready for CCS connection".

12) On the controller machine, execute the runccs script appropriate for your Operating System. If you have set everything correctly in the previous steps, you should see the benchmark start and run for over an hour. Once the run is finished, check the benchmark report in the results subdirectory under the CCS directory on the controller system. Even though it will be marked invalid due to the dummy meters, you should still see a sample report with the actual performance data and dummy wattage readings.

13) Once you are satisfied that you can get a full run out of your system, it is time to attach your real Power Analyzer and Temperature Sensor to the controller system, and edit the "runpower" (for your power analyzer) and "runtemp" (for

your temperature sensor) scripts appropriate for your operating system. You will need two separate command windows open, one for the "runpower" script, and the other for the "runtemp" script. Make sure to change the DEVICE line to reflect the power analyzer or temperature sensor you have attached (run the ptd executable without any arguments to show the supported meter list). Make sure that the meter types (both power analyzer and temperature sensor) you are using are in the approved list for this benchmark (check the SPECpower website for more details), otherwise you will not be able to submit your result to SPEC for review. Also make sure to edit the DEVICE_PORT line to reflect the serial port you have your power analyzer and temperature sensor connected to, depending on which script you are editing. If PTD fails to start, try modifying the DEVICE_PORT argument to reflect the actual serial port the meter is attached to. Example: try COM2 instead of COM1 for Windows, /dev/ttyS1 instead of /dev/ttyS0 for Linux. Also, if your power analyzer uses a GPIB interface instead of serial, make sure to add the -g flag into the runpower script at the bottom of the file where the command arguments are. (Example: %PTD% -g -p %NETWORK_PORT% %DEVICE% %DEVICE_PORT%)

*As an alternate methodology, you can run ptd on the command line: one instance for your power analyzer and the other for your temperature sensor. Execute the ptd version appropriate to your OS with the following arguments: "<name of ptd executable> <meter type> <serial device>". This should start a power daemon on your controller system, listening on port 8888 by default. For a Windows-based system, use COM1 for the <serial device>. For Linux, try /dev/ttyS0 or /dev/ttyS1. If your power analyzer uses a GPIB interface, make sure to utilize the -g flag. Now we need to start ptd in temperature mode as well, and we will do so by opening a second command line, changing to the ptd directory, and running ptd with the following arguments: "<name of ptd executable> -p 8889 -t <temperature sensor number> <serial device>". Again, if you are not sure what temperature sensor number you need to use, run ptd on the command line without any arguments to see a list of devices. Also, pay special attention to the fact that the -p <port number> argument for ptd in temperature mode needs to be different than the port number that the other instance of ptd running in power mode is using (default 8888).

For optimized benchmark results:

14) Locate a version of Java and appropriate performance tuning flags for your architecture by referencing previous SPECpower_ssj2008 submissions that match your OS and architecture, or failing that, check previous SPECjbb2005 submissions (<http://www.spec.org/jbb2005/results>) that match your configuration. Search through the results for a similar system configuration as the SUT you plan to use for SPECpower, and note the "JVM Version" and the "JVM Command Line", as well as the number of JVMs used. The "JVM Command Line" contains the tuning flags that you will be using for your SPECpower_ssj2008 run. Make note of everything after "java" on the JVM command line. Be aware that if you are using a SPECjbb2005 submission to model your JVM tuning flags from, they may not be optimal for SPECpower_ssj2008, but will likely be better than the defaults.

Based on the previously submitted results you referenced at the beginning of this step, change the "set JVMS=" line in the runssj.bat (if using Windows) to reflect the number of JVMs you plan to run on your SUT. Edit the "set JAVA_OPTIONS_Ssj" line further down in the file, and change the "-Xms256m -Xmx256m" portion of the line with the tuning flags from the "JVM Command Line" that you found from a similar JBB2005 submission at the beginning of this step.

If you are using Linux/Unix on the SUT, you will be editing the runssj.sh file. Edit the "JVMS=" line to reflect the number of JVMs you plan to run on your SUT. Edit the "JAVAOPTIONS_SSJ" line you find in the script, and change the "-Xms256m -Xmx256m" portion of the line with the tuning flags from the "JVM Command Line" that you found from a similar SPECpower_ssj2008 or SPECjbb2005 submission referenced at the beginning of this step.

Finally, ensure that you are using the appropriate Java version by comparing the "JVM Version" line with the version of java you may have previously installed on your SUT. Running "java -version" should give you a verbose version string to check against the "JVM Version" line, to determine if you should consider installing a different Java version on your SUT. If you get unexpected results from "java -version", your PATH could be incorrect, so try changing directories into your Java installation directory where the actual java binary resides, and re-running the command.

15) For results intended to be reviewed to SPEC, you are required to collect the list of running services on your SUT's Operating System to keep for the entire review period, and for possible disclosure if requested. In order to get this list, here are some examples:

```
Windows: net start > services.txt
Linux (Red Hat): /sbin/runlevel > services.txt; /sbin/chkconfig -list >>
services.txt
Solaris 10: svcs -a > services.txt
```

For more detailed help, please refer to the Users Guide for additional instruction, and feel free to check the FAQ maintained on the SPECpower website for answers to common questions.