```
/*************************************************************************************
** Notes on SPEC CPU2006
**
** Intended audience: Those who would like to learn more about
** measuring performance of modern computer systems using standardized benchmarks.
**
** Used: CPE 631 Advanced Computer Systems and Architectures
**         CPE 619 Modeling and Analysis of Computer and Communication Systems
**
** ver 0.1, Spring 2012 SPEC CPUv1.2
**
** @Aleksandar Milenkovic, milenkovic@computer.org
*************************************************************************************/
```

Using SPEC CPU2006 in LaCASA Lab

1 About SPEC CPU2006

CPU2006 is SPEC's industry-standardized, CPU-intensive benchmark suite,
stressing a system's processor, memory subsystem and compiler.
SPEC designed CPU2006 to provide a comparative measure of compute-intensive performance
across the widest practical range of hardware using workloads developed from real user applications.
These benchmarks are provided as source code and require the user
to be comfortable using compiler commands as well as other commands
via a command interpreter using a console or command prompt window in order to generate
executable binaries.
The current version of the benchmark suite is V1.2,
released in September 2011; submissions using V1.1 are only accepted through December 19, 2011.

2 SPEC CPU2006 on nsf02 machine

SPEC installation directory is located at /opt/cpu2006v1.2.
It includes subdirectories with documentation (Doc), Spec Utilities (bin),
and benchmarks (benchspec).

Please note that only superusers can modify this directory.
However, all users can run spec utilities, namely runspec,
that allows for building and running benchmarks
(providing configuration file specifies the SPEC output directory
be the one where user can write).

To do so, a user should go to /opt/cpu2006v1.2
and perform the following command:

```
<<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
[milenka@eb136i-nsf02 cpu2006v1.2]$ source shrc

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>>
```

Benchmarks are in the directory /opt/cpu2006v1.2/benchspec/CPU2006.

```
<<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
[milenka@eb136i-nsf02 CPU2006]$ pwd
/opt/cpu2006v1.2/benchspec/CPU2006
[milenka@eb136i-nsf02 CPU2006]$ ls
400.perlbench  433.milc       445.gobmk     458.sjeng       471.omnetpp    999.specrand
all_mixed.bset
401.bzip2      434.zeusmp     447.dealII    459.GemsFDTD    473.astar      CFP2006.bset
403.gcc        435.gromacs    450.soplex    462.libquantum  481.wrf        CINT2006.bset
410.bwaves     436.cactusADM  453.povray    464.h264ref     482.sphinx3    all_c.bset
416.gamess     437.leslie3d   454.calculix  465.tonto       483.xalancbmk  all_cpp.bset
429.mcf        444.namd       456.hmmer     470.lbm         998.specrand   all_fortran.bset

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>>
```

3 Running SPEC benchmarks using runspec

For reporting performance benchmarks should be compiled and run
using SPEC runspec utility. More details about runspec can be found at
http://www.spec.org/cpu2006/Docs/runspec.html.
If you want to have reportable results of CPU2006,
you need to understand the runspec utility, configuration files that control compilation and execution,
and redirection of output files generated in runs.

4 Running SPEC benchmarks (outside SPEC utilities)

We often want to run individual benchmarks from a command line,
or to run a spec benchmark on top of an architectural simulator.
To do this, you may locate executables and input files and copy them in your working directory.
You also need to know how to launch a particular benchmark.

We will use 401.bzip2 benchmark as an example.
Note: all benchmarks in SPEC CPU2006 share identical structure.

Go to a directory for 401.bzip2.

```
<<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
[milenka@eb136i-nsf02 CPU2006]$ cd 401.bzip2/
```

```
[milenka@eb136i-nsf02 401.bzip2]$ pwd
/opt/cpu2006v1.2/benchspec/CPU2006/401.bzip2
[milenka@eb136i-nsf02 401.bzip2]$ ls
Docs  Spec  build  data  exe  run  src  version.txt

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>>
```

You see doc, data, run, exe, and src directories.
For example, src contains source files.

```
<<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
[milenka@eb136i-nsf02 401.bzip2]$ ls src
blocksort.c  bzlib.c  bzlib_private.h  crctable.c    huffman.c  randtable.c  spec.h
bzip2.c      bzlib.h  compress.c       decompress.c  Makefile   spec.c

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>>
```

4.1 Executables

Executables can be located in the exe directory.
They are prepared by running runspec utility with a configuration file
that specifies conditions of compilation (compiler, optimization level, etc).

```
<<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
[milenka@eb136i-nsf02 401.bzip2]$ cd exe/
[milenka@eb136i-nsf02 exe]$ ls
bzip2_base.gccO3.exe64

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>>
```

Copy the executable into your working directory.

4.2 Inputs

The directory data contains input files.
SPEC supports three classes of inputs test (smallest), train (a bit larger),
and reference input sets that are used in reporting performance.

```
<<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
[milenka@eb136i-nsf02 401.bzip2]$ ls data
all  ref  test  train

[milenka@eb136i-nsf02 401.bzip2]$ ls data/ref/input/
chicken.jpg  control  input.source  liberty.jpg  text.html

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>>
```

Copy selected input files to your working directory.

The next section describes how to run individual benchmarks from the command line
(some benchmarks have multiple input data sets).

5 SPEC CPU2006 command lines

The following command lines for each of the SPEC CPU2006 benchmarks were obtained using 'specinvoke -n' in the needed run directories.

NOTE: the command lines below should be modified to match executables (bzip2_base.gccO3.exe64 instead of bzip2).

```
400.perlbench (3 inputs)
------------------------

    reference inputs:

    perlbench -I./lib checkspam.pl 2500 5 25 11 150 1 1 1 1 >
perlbench.ref.checkspam.out 2> perlbench.ref.checkspam.err

    perlbench -I./lib diffmail.pl 4 800 10 17 19 300 > perlbench.ref.diffmail.out 2>
perlbench.ref.diffmail.err

    perlbench -I./lib splitmail.pl 1600 12 26 16 4500 > perlbench.ref.splitmail.out 2>
perlbench.ref.splitmail.err


401.bzip2 (6 inputs)
---------------------

    reference inputs:

    bzip2 input.source 280 > bzip2.ref.source.out 2> bzip2.ref.source.err

    bzip2 chicken.jpg 30 > bzip2.ref.chicken.out 2> bzip2.ref.chicken.err

    bzip2 liberty.jpg 30 > bzip2.ref.liberty.out 2> bzip2.ref.liberty.err

    bzip2 input.program 280 > bzip2.ref.program.out 2> bzip2.ref.program.err

    bzip2 text.html 280 > bzip2.ref.text.out 2> bzip2.ref.text.err

    bzip2 input.combined 200 > bzip2.ref.combined.out 2> bzip2.ref.combined.err

403.gcc (9 inputs)
-------------------
    reference inputs:

    gcc 166.i -o 166.s > gcc.ref.166.out 2> gcc.ref.166.err

    gcc 200.i -o 200.s > gcc.ref.200.out 2> gcc.ref.200.err

    gcc c-typeck.i -o c-typeck.s > gcc.ref.c-typeck.out 2> gcc.ref.c-typeck.err

    gcc cp-decl.i -o cp-decl.s > gcc.ref.cp-decl.out 2> gcc.ref.cp-decl.err

    gcc expr.i -o expr.s > gcc.ref.expr.out 2> gcc.ref.expr.err

    gcc expr2.i -o expr2.s > gcc.ref.expr2.out 2> gcc.ref.expr2.err
    gcc g23.i -o g23.s > gcc.ref.g23.out 2> gcc.ref.g23.err

    gcc s04.i -o s04.s > gcc.ref.s04.out 2> gcc.ref.s04.err

    gcc scilab.i -o scilab.s > gcc.ref.scilab.out 2> gcc.ref.scilab.err

410.bwaves (1 input)
--------------------
    reference inputs:
    bwaves > bwaves.ref.out 2> bwaves.ref.err

416.gamess (3 inputs)
---------------------

    reference inputs:

    gamess < cytosine.2.config > gamess.ref.cytosine.out 2> gamess.ref.cytosine.err

    gamess < h2ocu2+.gradient.config > gamess.ref.h2ocu2+.out 2>
gamess.ref.h2ocu2+.err

    gamess < triazolium.config > gamess.ref.triazolium.out 2>
gamess.ref.triazolium.err

429.mcf (1 input)
-----------------

    reference inputs:
    mcf inp.in > mcf.ref.out 2> mcf.ref.err

433.milc (1 input)
------------------

    reference inputs:
    milc < su3imp.in > milc.ref.out 2> milc.ref.err

434.zeusmp (1 input)
--------------------
    reference inputs:
    zeusmp > zeusmp.ref.out 2> zeusmp.ref.err

435.gromacs (1 input)
---------------------
    reference inputs:
    gromacs -silent -deffnm gromacs -nice 0 > gromacs.ref.out 2> gromacs.ref.err


436.cactusADM (1 input)
-----------------------
    reference inputs:
    cactusADM benchADM.par > cactusADM.ref.out 2> cactusADM.ref.err


437.leslie3d (1 input)
----------------------

    reference inputs:
    leslie3d < leslie3d.in > leslie3d.ref.out 2> leslie3d.ref.err

444.namd (1 input)
------------------
    reference inputs:
```

```
    namd --input namd.input --iterations 38 --output namd.out > namd.ref.out 2>
namd.ref.err

445.gobmk (5 inputs)
--------------------
    reference inputs:

    gobmk --quiet --mode gtp < 13x13.tst > gobmk.ref.13x13.out 2> gobmk.ref.13x13.err

    gobmk --quiet --mode gtp < nngs.tst > gobmk.reff.nngs.out 2> gobmk.ref.nngs.err

    gobmk --quiet --mode gtp < score2.tst > gobmk.ref.score2.out 2>
gobmk.ref.score2.err

    gobmk --quiet --mode gtp < trevorc.tst > gobmk.ref.trevorc.out 2>
gobmk.ref.trevorc.err

    gobmk --quiet --mode gtp < trevord.tst > gobmk.ref.trevord.out 2>
gobmk.ref.trevord.err

447.dealII (1 input)
--------------------
    reference inputs:
    dealII 23 > dealII.ref.out 2> dealII.ref.err

450.soplex (2 inputs)
---------------------
    reference inputs:

    soplex -s1 -e -m45000 pds-50.mps > soplex.ref.pds-50.out 2> soplex.ref.pds-50.err

    soplex -m3500 ref.mps > soplex.ref.ref.out 2> soplex.ref.ref.err

453.povray (1 input)
--------------------
    reference inputs:
    povray SPEC-benchmark-ref.ini > povray.ref.out 2> povray.ref.err


454.calculix (1 input)
----------------------
    reference inputs:
    calculix -i hyperviscoplastic > calculix.ref.out 2> calculix.ref.err


456.hmmer (2 inputs)
--------------------
    reference inputs:

    hmmer nph3.hmm swiss41 > hmmer.ref.nph3.out 2> hmmer.ref.nph3.err

    hmmer --fixed 0 --mean 500 --num 500000 --sd 350 --seed 0 retro.hmm >
hmmer.ref.retro.out 2> hmmer.ref.retro.err

458.sjeng (1 input)
-------------------
    reference inputs:
    sjeng ref.txt > sjeng.ref.out 2> sjeng.ref.err

459.GemsFDTD (1 input)
----------------------
    reference inputs:
    GemsFDTD > GemsFDTD.ref.out 2> GemsFDTD.ref.err
```

```
462.libquantum (1 input)
-------------------------
    reference inputs:
    libquantum 1397 8 > libquantum.ref.out 2> libquantum.ref.err

464.h264ref (3 inputs)
----------------------
    reference inputs:

    h264ref -d foreman_ref_encoder_baseline.cfg > h264ref.ref.foreman_baseline.out 2>
h264ref.ref.foreman_baseline.err

    h264ref -d foreman_ref_encoder_main.cfg > h264ref.ref.foreman_main.out 2>
h264ref.ref.foreman_main.err

    h264ref -d sss_encoder_main.cfg > h264ref.ref.sss.out 2> h264ref.ref.sss.err


465.tonto (1 input)
-------------------
    reference inputs:
    tonto > tonto.ref.out 2> tonto.ref.err

470.lbm (1 input)
-----------------
    reference inputs:
    lbm 3000 reference.dat 0 0 100_100_130_ldc.of > lbm.ref.out 2> lbm.ref.err

471.omnetpp (1 input)
---------------------
    reference inputs:
    omnetpp omnetpp.ini > omnetpp.ref.log 2> omnetpp.ref.err

473.astar (2 inputs)
--------------------
    reference inputs:

    astar BigLakes2048.cfg > astar.ref.BigLakes2048.out 2> astar.ref.BigLakes2048.err

    astar rivers.cfg > astar.ref.rivers.out 2> astar.ref.rivers.err

481.wrf (1 input)
-----------------
    reference inputs:
    wrf > wrf.ref.out 2> wrf.ref.err

482.sphinx3 (1 input)
---------------------
    reference inputs:
    sphinx_livepretend ctlfile . args.an4 > sphinx3.ref.out 2> sphinx3.ref.err

483.xalancbmk (1 input)
-----------------------
    reference inputs:
    Xalan -v t5.xml xalanc.xsl > xalancbmk.ref.out 2> xalancbmk.ref.err
```