



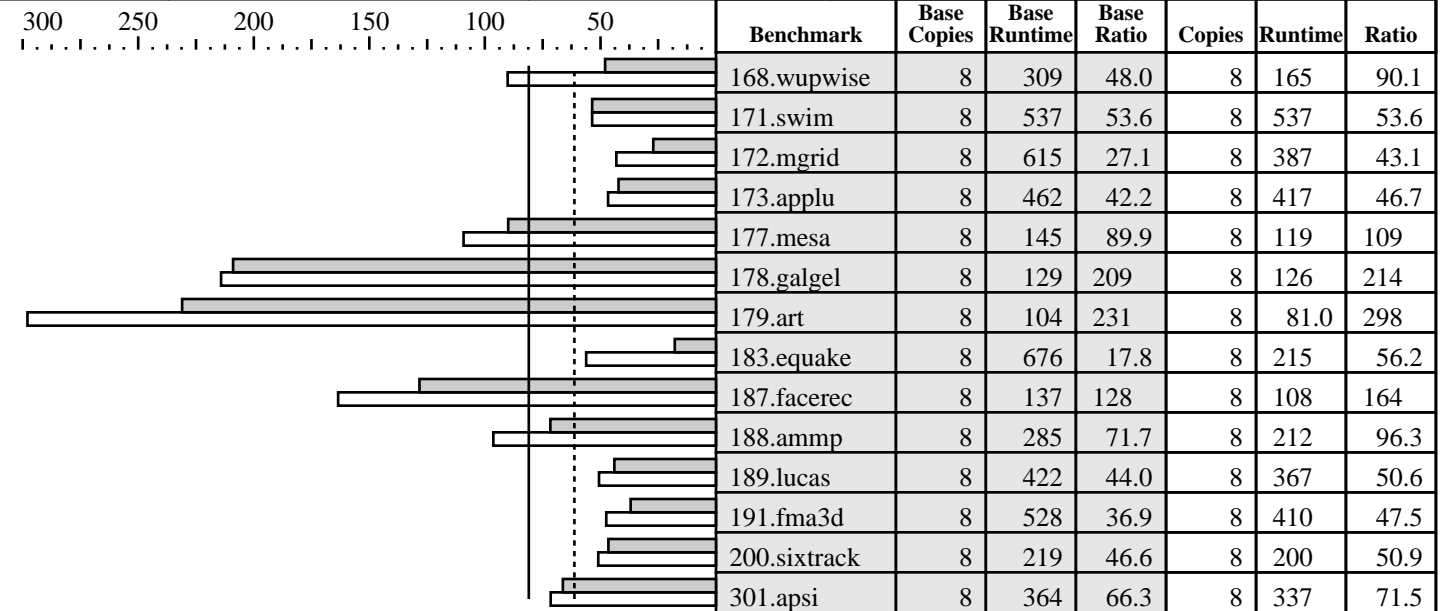
CFP2000 Result

Copyright ©1999-2004, Standard Performance Evaluation Corporation

Hewlett-Packard Company
hp AlphaServer GS80 68/1224

SPECfp_rate2000 = 81.0
SPECfp_rate_base2000 = 61.3

SPEC license #: 2 | Tested by: HP | Test date: Sep-2002 | Hardware Avail: Aug-2002 | Software Avail: Dec-2002



Hardware

CPU: Alpha 21264C
 CPU MHz: 1224
 FPU: Integrated
 CPU(s) enabled: 8 cores, 8 chips, 1 core/chip
 CPU(s) orderable: 1 to 8
 Parallel: No
 Primary Cache: 64KB(I)+64KB(D) on chip
 Secondary Cache: 16MB off chip per CPU
 L3 Cache: None
 Other Cache: None
 Memory: 16GB
 Disk Subsystem: mfs (Memory File System)
 Other Hardware: None

Software

Operating System: Tru64 UNIX T5.1B
 Compiler: Compaq C V6.5-011-48C5K
 Spike V5.2 (506 48C5K)
 Compaq Fortran V5.5-1877-48BBF
 Compaq Fortran 77 V5.5-1877-48BBF
 KAP Fortran V4.4 k340504 20010517
 KAP Fortran 77 V4.1 k310440 980926
 KAP C V4.2 k010737S 010515
 File System: mfs
 System State: Multi-user

Notes/Tuning Information

Baseline C: cc -arch ev6 -fast -O4 ONESTEP
 Fortran: f90 -arch ev6 -fast -O5 ONESTEP

Peak:

All use -arch ev6 -non_shared ONESTEP (except applu and ammp)
 Individual benchmark tuning:
 168.wupwise: kf77 -call_shared -inline all -tune ev67
 -unroll 12 -automatic -align commons -arch ev67
 -fkapargs=' -aggressive=c -fuse
 -fuselevel=1 -so=2 -r=1 -o=1 -interleave
 -ur=6 -ur2=060 ' +PFB
 171.swim: same as base
 172.mgrid: kf90 -call_shared -arch generic -O5 -inline
 manual -nopipeline -unroll 9 -automatic -transform_loops
 -fkapargs=' -aggressive=a -fuse -interleave
 -ur=2 -ur3=5 -cachesize=128,16000 ' +PFB



CFP2000 Result

Copyright ©1999-2004, Standard Performance Evaluation Corporation

Hewlett-Packard Company
hp AlphaServer GS80 68/1224

SPECfp_rate2000 = 81.0
SPECfp_rate_base2000 = 61.3

SPEC license #: 2 | Tested by: HP | Test date: Sep-2002 | Hardware Avail: Aug-2002 | Software Avail: Dec-2002

Notes/Tuning Information (Continued)

```

173.applu: kf90 -O5 -transform_loops
          -fkapargs=' -o=0 -nointerleave -ur=14
          -ur2=260 -ur3=18' +PFB
177.mesa: kcc -fast -O4 +CFB +IFB
178.galgel: f90 -O5 -fast -unroll 5 -automatic
179.art: kcc -assume whole_program -ldensemalloc
          -call_shared -assume restricted_pointers
          -unroll 16 -inline none -ckapargs='
          -fuse -fuselevel=1 -ur=3' +PFB
183.quake: cc -call_shared -arch generic -fast -O4
          -ldensemalloc -assume restricted_pointers
          -inline speed -unroll 13 -xtaso_short +PFB
187.facerec: f90 -O4 -nopipeline -inline all
          -non_shared -speculate all -unroll 7
          -automatic -assume accuracy_sensitive
          -math_library fast +IFB
188.ammp: cc -arch host -O4 -ifo -assume nomath_errno
          -assume trusted_short_alignment -fp_reorder
          -readonly_strings -ldensemalloc -xtaso_short
          -assume restricted_pointers -unroll 9
          -inline speed +CFB +IFB +PFB
189.lucas: kf90 -O5 -fkapargs='-ur=1' +PFB
191.fma3d: kf90 -O4 -transform_loops -fkapargs='-cachesize=128,16000' +PFB
200.sixtrack: f90 -fast -O5 -assume accuracy_sensitive
          -notransform_loops +PFB
301.apsi: kf90 -O5 -inline none -call_shared -speculate all
          -align commons -fkapargs=' -aggressive=ab
          -tune=ev5 -fuse -ur=1 -ur2=60 -ur3=20
          -cachesize=128,16000'

```

Most benchmarks are built using one or more types of profile-driven feedback. The types used are designated by abbreviations in the notes:

+CFB: Code generation is optimized by the compiler, using feedback from a training run. These commands are done before the first compile (in phase "fdo_pre0"):

```

mkdir /tmp/pp
rm -f /tmp/pp/${baseexe}*

```

and these flags are added to the first and second compiles:

```

PASS1_CFLAGS = -prof_gen_noopt -prof_dir /tmp/pp
PASS2_CFLAGS = -prof_use -prof_dir /tmp/pp

```

(Peak builds use /tmp/pp above; base builds use /tmp/pb.)

+IFB: Icache usage is improved by the post-link-time optimizer Spike, using feedback from a training run. These commands are used (in phase "fdo_postN"):

```

mv ${baseexe} oldexe
spike oldexe -feedback oldexe -o ${baseexe}

```

+PFB: Prefetches are improved by the post-link-time optimizer



CFP2000 Result

Copyright ©1999-2004, Standard Performance Evaluation Corporation

Hewlett-Packard Company
hp AlphaServer GS80 68/1224

SPECfp_rate2000 = 81.0
SPECfp_rate_base2000 = 61.3

SPEC license #: 2 | Tested by: HP | Test date: Sep-2002 | Hardware Avail: Aug-2002 | Software Avail: Dec-2002

Notes/Tuning Information (Continued)

Spike, using feedback from a training run. These commands are used (in phase "fdo_post_makeN"):

```
rm -f *Counts*
mv ${baseexe} oldexe
pixie -stats dstride oldexe 1>pixie.out 2>pixie.err
mv oldexe.pixie ${baseexe}
```

A training run is carried out (in phase "fdo_runN"), and then this command (in phase "fdo_postN"):

```
spike oldexe -fb oldexe -stride_prefetch -o ${baseexe}
```

When Spike is used for both Icache and Prefetch improvements, only one spike command is actually issued, with the Icache options followed by the Prefetch options.

vm:

```
vm_bigpg_enabled = 1
vm_bigpg_thresh=16
vm_swap_eager = 0
```

proc:

```
max_per_proc_address_space = 0x4000000000
max_per_proc_data_size = 0x4000000000
max_per_proc_stack_size = 0x4000000000
max_proc_per_user = 2048
max_threads_per_user = 0
maxusers = 16384
per_proc_address_space = 0x4000000000
per_proc_data_size = 0x4000000000
per_proc_stack_size = 0x4000000000
```

Portability: galgel: -fixed
submit = runon cpu