НИИСИ РАН

Запуск ОС Linux как этап функционального тестирования микропроцессоров

Чибисов Петр Александрович

Типы тестов микропроцессоров:

- Тесты разработчика;
- Программы аттестации архитектуры;
- Псевдослучайные тесты;
- Переборные тесты;
- Загрузка одной или нескольких ОС;
- **Программы и приложения под ОС;**
- **Тесты производительности.**

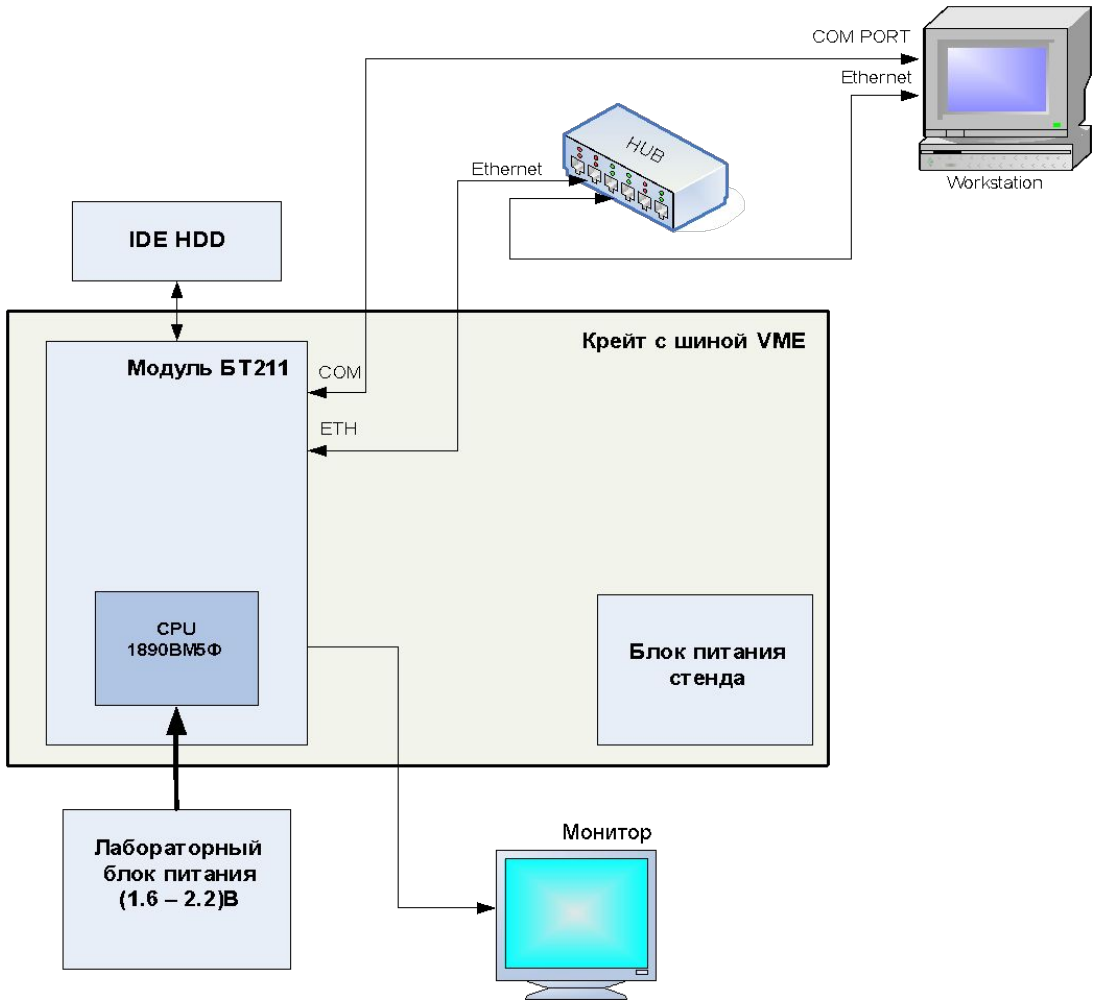
Тесты под ОС Linux:

- Зачем их запускать?
- Какие ОС и тесты запускать?
- Типичные сценарии запуска тестов.
- Что показывают тесты производительности?
- Какие ошибки были найдены?
- Что в планах на будущее?

Зачем запускать тесты под ОС?

- большой архитектурный тест;
- множество самопроверяющихся testcase'ов;
- идеи для шаблонов псевдослучайных тестов;
- огромное количество тестов системы под ОС (LTP, сборка RPM-пакетов, SPEC, X...);
- изучение производительности процессора (анализ трасс + результаты тестов);
- решение задач на native-платформе (gcc, gdb, тесты C+asm), программирование cp2;
- измерение основных электрических параметров потребления ядер микропроцессоров;
- академический интерес.

Схема тестовой установки



Какие ОС и тесты запускаются?

ОС: **ОСРВ 2000/3000**, **Linux** Red Hat / Debian

Тесты:

- baget-2.4.37 и baget-2.6, компиляция ядер ОС Linux;
- LTP, пакет тестов Linux Test Project, версии 20070531-5;
- memtester-4.2.0, тест памяти;
- тесты производительности процессора CPU SPEC2000, CPU SPEC2006;
- CP_NDEV, тест копирования файлов;

Тесты, запускаемые под ОС Linux

- mpfr-3.0.0, mpc-0.8.2, mpfrcx-0.3.1, mpir-2.2.1, gappa-0.14.0, gmp-5.0.1, математические библиотеки точных вычислений, содержат встроенные тесты;
- glucas-2.9.2, пакет вычислений простых чисел, хорошо нагружает FPU;
- ruby-1.9.2, Python-2.5, perl-5.8.8, php-5.3.8, языки программирования, содержат встроенные тесты;
- icarus verilog-0.9.3, моделирование VerilogHDL;
- lame-3.97, flac-1.2.1, ffmpeg-0.5, кодеры/декодеры mp3/flac/видео;
- kdegames-3.5.10, графические приложения – игры для KDE;
- koffice-1.6.3, полный пакет офисных программ KDE;

Тесты, запускаемые под ОС Linux

- mozilla-firefox - 3.6.13, Интернет-браузер
  (gtk+-2.10.14, cairo-1.2.6, pango-1.14.0, pkgconfig-0.15.0, neon-0.28.6, bison-2.4, atk-1.9.1, libIDL-0.8.8, libnotify-0.4.4, libsigc++-2.2.4, libxml2-2.7.3, m4-1.4.15, numactl-2.0.3, dbus-0.60, sqlite-2.8.17, curl-7.21.3);
- wormux-0.9.2.1, графическая игра;
- gcc–4.5.2 (C,C++,F77,F90,Java,...) selftests;

Тесты, запускаемые под ОС Linux

- Lmbench, тест производительности системы;
- paranoia (разные оптимизации);
- X : KDE/Gnome;
- тесты gcc (кросс-компиляция);
- тесты производительности: dhrystone, whetstone, coremark, iobench, ...;
- тесты cp2 (dsplib);
- HPL (MPI + ATLAS/GotoBLAS);
- тесты posix под ос3000;

Типичные сценарии запуска тестов

Вариант 1:

```
for i in `find …`; do … ; done
```

Вариант 2:

```
tar zxf   … . tgz
./configure CC="gcc -march=7k …"
make
make check
```

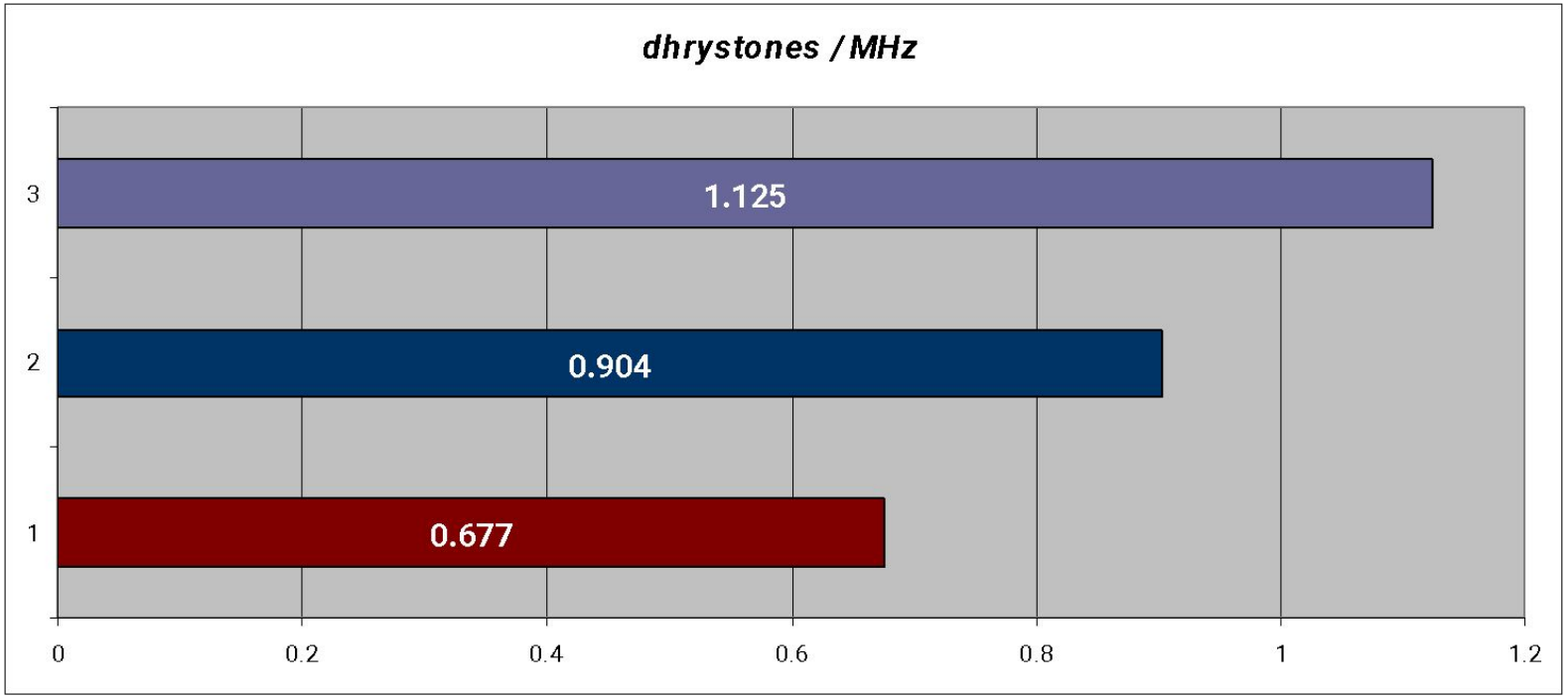
Тесты производительности

- dhrystone;
- whetstone;
- coremark;
- lmbench-3.0.9;
- SPEC2000 (INT + FP);
- SPEC2006 (INT + FP);
- read / write speed в ОС3000;
- Switch Context / Thread response time в ОС3000.

Сравнение производительности

1. Тест dhrystone под ОС3000:

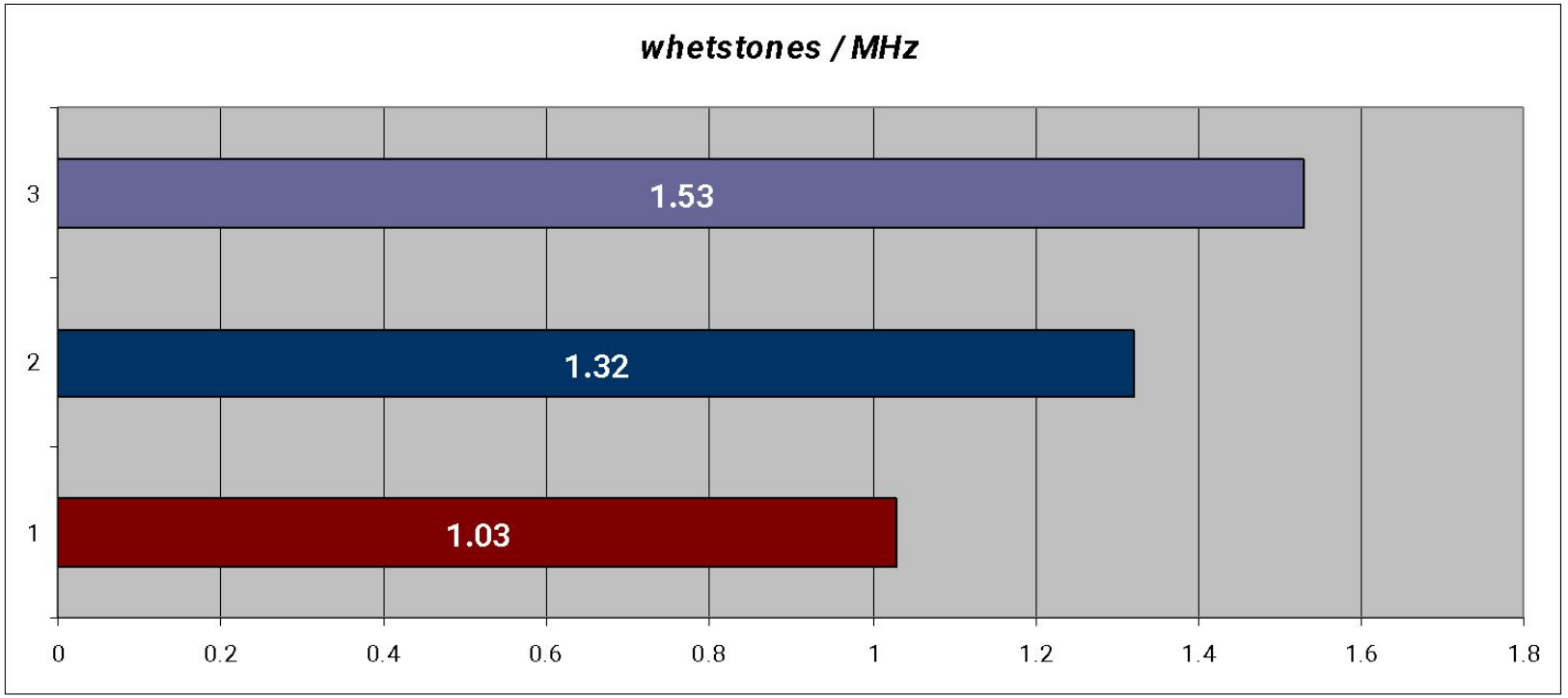
   1 – ВМ5Ф, 2 – ВМ6Я, 3 – разрабатываемый 65 нм



Сравнение производительности

2. Тест whetsone (fpu) под ОС3000:

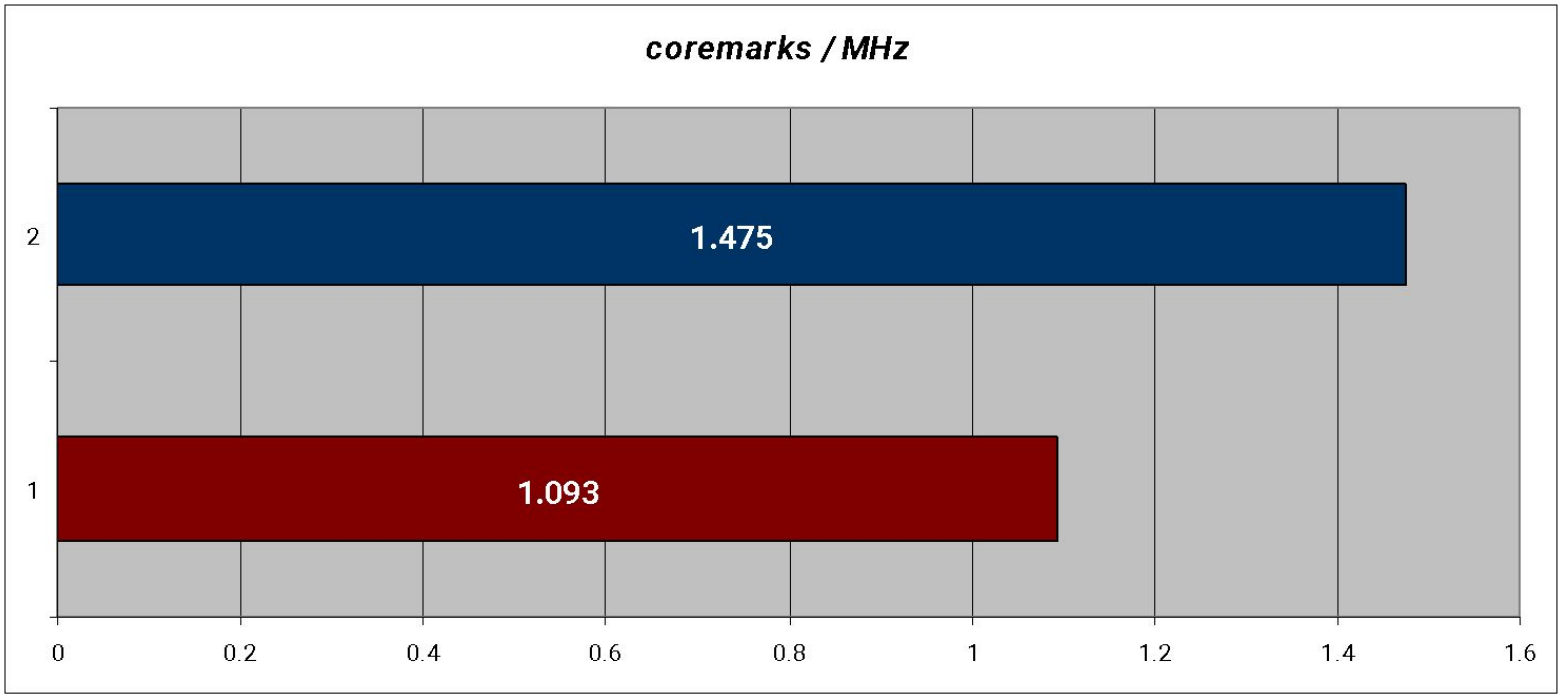
1 – ВМ5Ф, 2 – ВМ6Я, 3 – разрабатываемый 65 нм



Сравнение производительности

3. Тест coremark под ОС Linux:

1 – ВМ5Ф, 2 – ВМ6Я



Сравнение производительности

4.1 Тесты **lmbench** под ОС Linux:

1 – ВМ5Ф, 2 – ВМ6Я [F_{core}=260, F_{mem}=130 MHz]

```
Processor, Processes - times in microseconds - smaller is better
------------------------------------------------------------------------------
Host                 OS  Mhz null null      open slct sig  sig  fork exec sh
                             call  I/O stat clos TCP  inst hndl proc proc proc
-------- ------------ ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
1890VM5   Linux 2.6.37+  260 1.45  4.1  74.4 136  112 4.48 46.6 3688 13K  45K
1890VM6   Linux 2.6.37+  260 1.25  3.1  54.6  98   90 3.23 29.3 2339 9020 31K

Basic integer operations - times in nanoseconds - smaller is better
----------------------------------------
Host        intgr intgr  intgr  intgr  intgr
             bit   add    mul    div    mod
-------------- ------ ------ ------ ------
1890VM5     3.89   5.44   10.5   54.5   32.2
1890VM6     3.88   4.07    7.1   54.3   28.2
```

Сравнение производительности

4.2 Тесты **lmbench** под ОС Linux:

1 – ВМ5Ф, 2 – ВМ6Я

```
Basic float/double operations - times in nanoseconds - smaller is better
-------------------------------------  --------- -----------------
Host      float  float  float  float    double double double double
          add    mul    div    bogo      add    mul    div    bogo
------- ------ ------ ------ ------  ------  ------ ------ ------
1890VM5  17.9   18.6   65.8  158.1      21.8   22.5    79.4   181.6
1890VM6  11.0   11.3   58.0  105.5      14.9   15.2    71.7   126.2
```

Сравнение производительности

4.3 Тесты **lmbench** под ОС Linux:

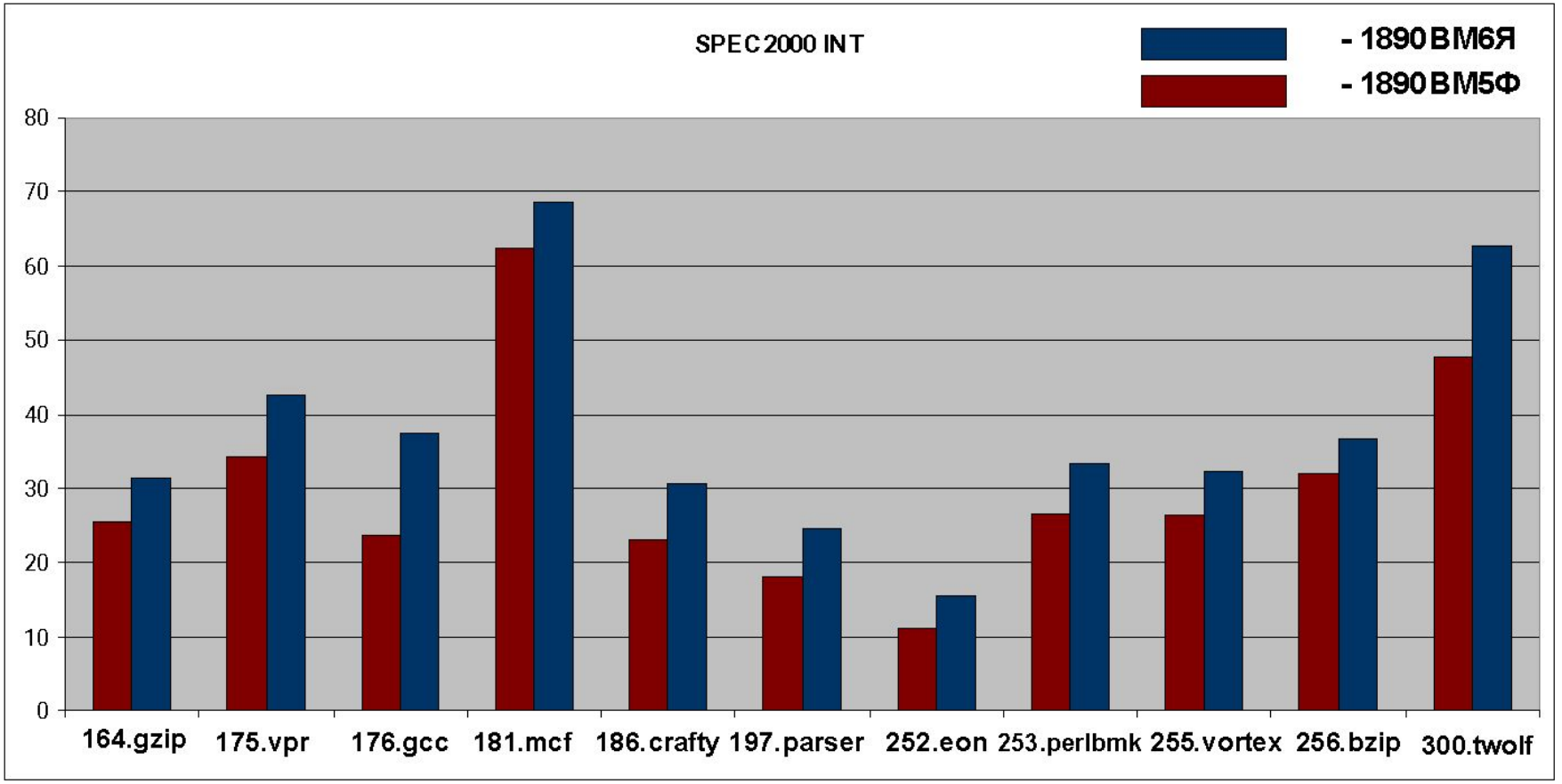
1 – ВМ5Ф, 2 – ВМ6Я

```
Memory latencies in nanoseconds - smaller is better
-------- -------- ----------------------------
Host       L1 $   L2 $    Main mem    Rand mem
-------- ------   ----   ----------   ---------
1890VM5   8.002   111.8       180.8      540.4
1890VM6   8.009   103.8       184.9      544.4

File & VM system latencies in microseconds - smaller is better
-------------------------------------- -------  -----  ------ ------
Host          0K File          10K File     Mmap    Prot   Page   100fd
           Create   Delete   Create  Delete Latency  Fault  Fault  selct
--------- ------- -------- ------ ------ -------  -----  ------ ------
1890VM5    429.0    360.8   1385.0  595.2    82.4K  1.407    57.8  61.4
1890VM6    300.1    265.4   1000.0  437.3    56.0K  1.779    36.5  49.0
```

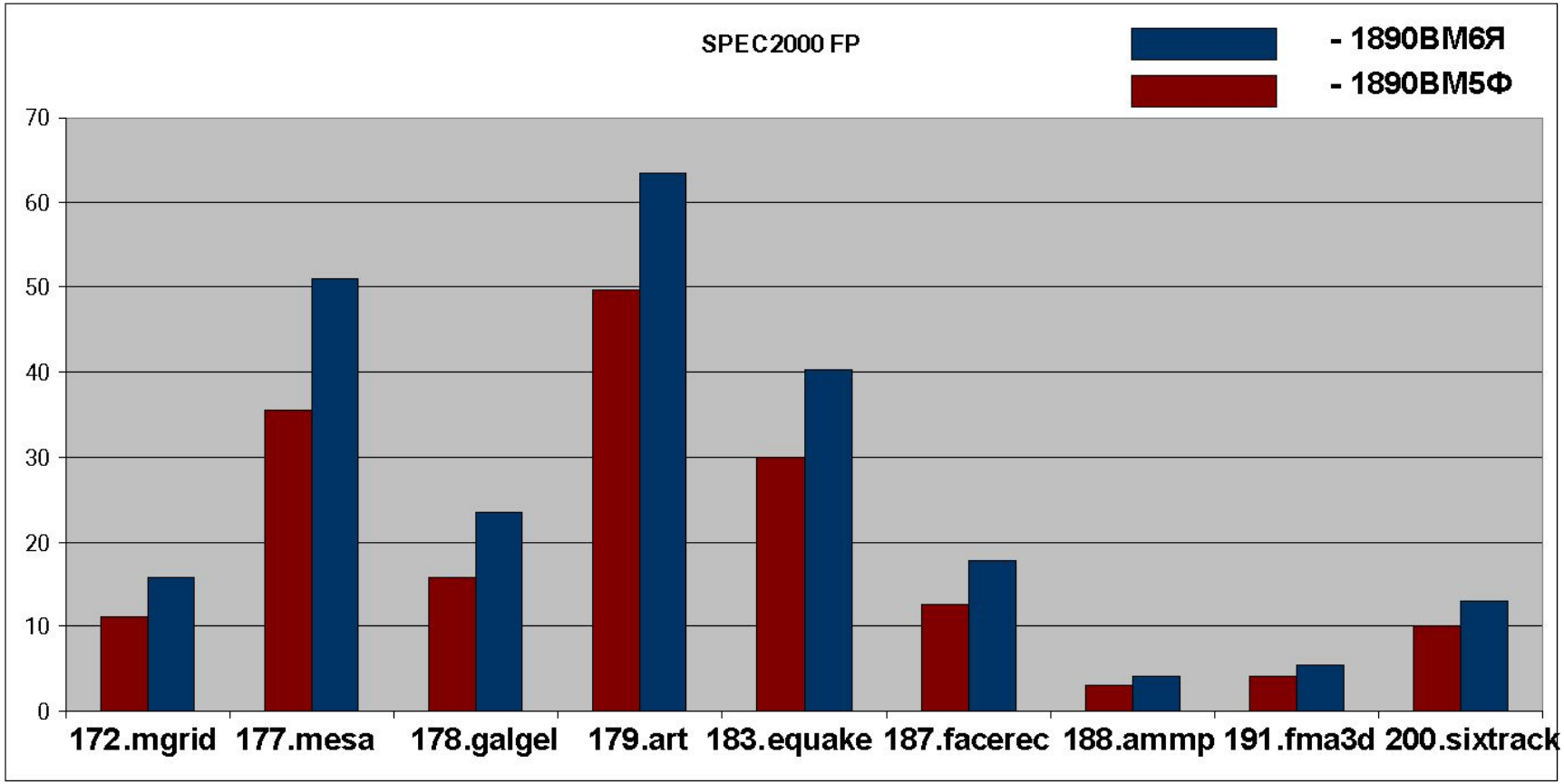
Сравнение производительности

CPU SPEC2000 INT (F_{core} = 192 MHz, F_{mem} = 96 MHz)



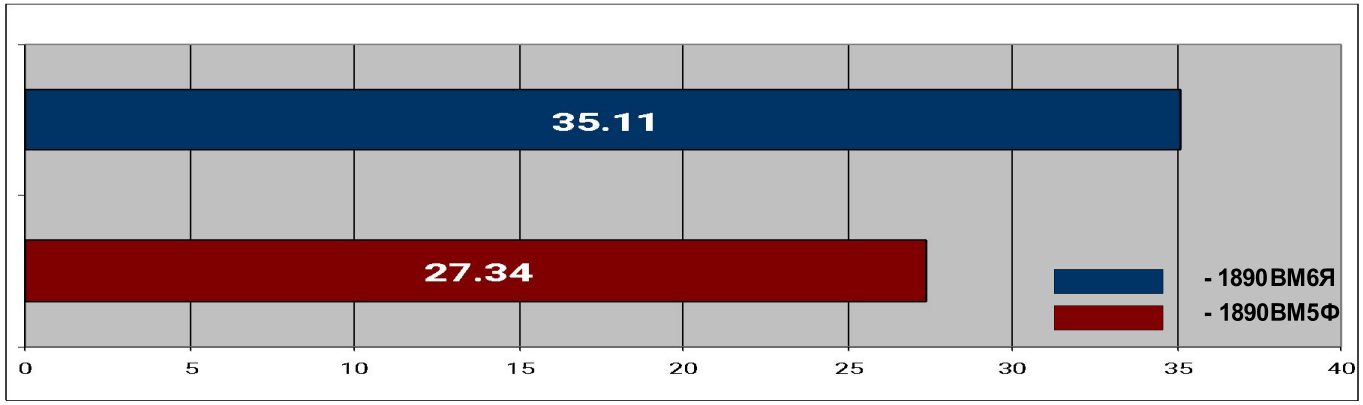
Сравнение производительности

CPU SPEC2000 FP (F_{core} = 192 MHz, F_{mem} = 96 MHz)

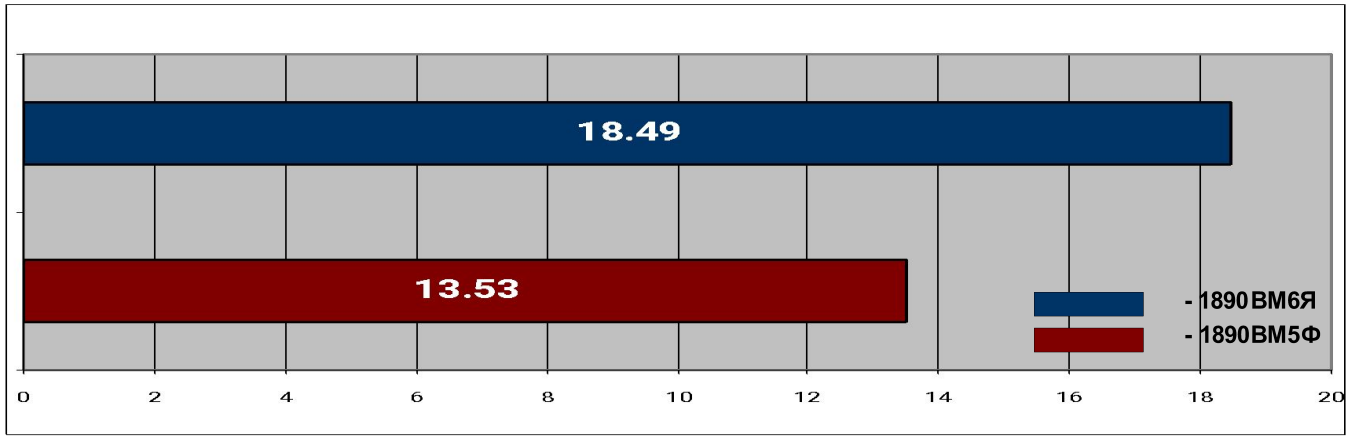


Сравнение производительности

CPU SPEC2000 INT: +28.4%



CPU SPEC2000 FP: +36.6%



Локализация ошибок в микропроцессоре

1. Однократный сбой?
2. Программная ошибка?
3. Какая именно shell-команда вызывает сбой? (локализация testcase)
4. Как влияют F_{core} / F_{mem}, SS, BP, Sp, L2?
5. Как ведет себя testcase на ПЛИС и на vmips (golden model emulator)?
6. Есть ли ошибка в RTL-коде?

Примеры найденных ошибок

1. процессор 1890ВМ5Ф, 18 марта 2011:
make check для perl-5.8.8

Зависание процессора в ситуации:

```
2624 PC=0x4cf0ec [38560ec] 8e420000 lw $v0,0x0($s2)
2625 PC=0x4cf0f0 [38560f0] 8f838024 lw $v1,0x8024($gp)
2626 PC=0x4cf0f4 [38560f4] c4420014 lwc1 f2,0x14($v0)

Exception CpUnusable, cause=11 at PC=0x4cf0f4 triggered,
instr=c4420014 Priority is 10; delay state is NORMAL;
```

Примеры найденных ошибок

2. процессор 1890ВМ6Я, ноябрь 2010:

компиляция теста ATLAS под ОС Linux.
(компилятор иногда останавливается с сообщением о неизвестной ошибке - падает программа `cc1`).
Ошибка в цикле:

```
72d62c: ac400014    sw zero,20(v0)
72d630: 8c420004    lw v0,4(v0)
72d634: 00000000    nop             [□ INT]
72d638: 1440fffc    bnez v0,72d62c
72d63c: 00000000    nop
```

в случае прихода прерывания в один из тактов выполнения инструкции перехода (***bnez***). При этом, переход ошибочно происходил, несмотря на v0==0.

Ошибка исчезает при отключении суперскалярности.

В новой версии 1890ВМ6Я (сентябрь 2011) этой ошибки нет.

Примеры найденных ошибок

3. процессор 1890ВМ6Я, 2 сентября 2011:
запуск инсталлятора ОС Linux Debian 6.0.2.

Процессор не вызвал *Reserved Instruction Exception* по инструкции **`rdhwr 3,29`** (opcode=**`0x7c03e83b`**), тогда как ядро ОС ждёт исключения.

1). Замена в исходниках ядра (balo) инструкции с опкодом **`0x7c03e83b [rdhwr v1,$29]`** на инструкцию **`0x7c03e833`** (всегда вызывает RI - см. II-й pdf описание поля special3).

2). Замена **`0x7c03e83b`** на **`0x7c03e833`** во всех библиотечных файлах файловой системы.

Примеры найденных ошибок

4. процессор 1890ВМ6Я, 16 сентября 2011:
запуск поправленной версии инсталлятора ОС Linux Debian.

Процессор неправильно отработал инструкцию eret в обработчике RI Exception, вызывав Coprocessor Unusable Exception.
Пример кода:

```
24017 PC=0x80008480 [8480] df630018 ld $v1,0x18($k1)
24018 PC=0x80008484 [8484] 42000018 eret
```

Ошибка только в случае, если **ld** вызывает **dcache miss + dTLB hit**.

Ошибка исправляется добавлением двух **ssnop** между **ld** и **eret**.
(Файл ядра arch/mips/kernel/genex.S)

Примеры найденных ошибок

5. процессор 1890ВМ5Ф, ревизия 2 (2008г.),
тест SPEC2000 252.eon:
неверные данные у **mfc1** в ситуации:

```
madd.D $fp0,...
lw …
addiu …
jr
mfc1 ...,$fp1
```

в режиме 32-х разрядной совместимости FPU.

Изучение кода ошибок

- Трассы кода, набор инструкций;
- Сегментация памяти;
- Режимы работы (K,S,U; 32/64);
- Исключительные ситуации;
- Прерывания;
- Кэш-память;
- Сопроцессоры.

Выводы

+ огромное количество готовых тестов;
+ относительно простой запуск;
+ тестирование с большим уровнем асинхронных прерываний/событий;
+ большая уверенность в проекте, чем после прогона базы тестов;
- избыточность тестов;
- иногда трудно локализовать ошибку.

Планы по развитию методики

- MPI и параллельные вычисления;
- изучение QEMU, OVP - эмуляция многотредовых многоядерных процессоров;
- тесты на F77, F90;
- test profiling и test coverage;
- Использование системы buildroot;
- улучшение шаблонов для псевдослучайного тестирования.

Спасибо за внимание!

Вопросы?