IMPROVING $PORT PERFORMANCE ON $ARCH

PLATFORM-BASED PERFORMANCE TUNING OF WEBKIT
(PORT=QT ARCH=MIPS74KF)

Adrián Pérez de Castro
Embedded Linux Conference
April 29 — May 1, 2014
THE CHALLENGE

MAKE A QTWEBKIT-BASED BROWSER USEABLE ON LIMITED HARDWARE

MIPS 74Kf @500 MHz
RAM: 256 MB
No GPU
MIPS74KF

“Classic” MIPS32
+ FPU
+ MMU
+ DSP
DSP?

No. Not really a DSP.

Instructions suitable for signal processing.
THE PLAN

PROFILE → OPTIMIZE → VALIDATE
WHAT TO OPTIMIZE

Video/audio decoding.
Image operations.
WHERE TO OPTIMIZE

Can we improve the platform overall, not just WebKit?

Yes!

QtWebKit uses the Qt drawing functions. A/V decoding uses GStreamer, which uses Orc.

Good candidates for SIMD code.
LIMITATIONS

No Valgrind.
No GDB.
No perf.
No performance counters.

↓

qemu + gdbserver.
gperftools.
CLOCK_PROCESS_CPUTIME_ID
ROLL YOUR OWN TOOLS

(WITH HELP FROM EXISTING ONES)
# Use full path to avoid using the shell's time builtin
# One line per run with user/system time and page faults
/usr/bin/time -a -o timings.txt \
  -f '%%U %S %F %x %C' $COMMAND

# For example, measuring the qtdemux GStreamer component
/usr/bin/time -a -o timings.txt \
  -f '%%U %S %F %x %C' gst-launch -q \
  filesrc=file.mp4 ! qtdemux ! video/x-h264 ! fakesink
Beware of CLOCK_PROCESS_CPUTIME_ID's resolution!

```c
#define CLOCK_MAX_RESOLUTION_DELTA  (10000.0 * 1e-9)

bool usePosixClock() {
    static bool checked = false;
    static bool useposix;
    if (!checked) {
        if (posixClockAvailable()) {
            double res_theoretical = posixClockTheoricalResolution();
            double res_empirical = posixClockEmpiricalResolution();
            useposix = fabs(res_theoretical - res_empirical) <= CLOCK_MAX_RESOLUTION_DELTA;
        } else {
            useposix = false;
        }
        checked = true;
    }
    return useposix;
}
```

clock.cc
% g++ -DMAIN -o clock clock.cc
% ./clock
CLOCK_PROCESS_CPUTIME_ID is supported
Resolution (advertised/empirical): 0.0000000010/0.0000002460s
Sampled resolution: 0.00000005470s
Printing the lines above took 0.0000483550s

% LD_PRELOAD=/usr/lib/libprofiler.so \
  ./websnap http://igalia.com 1000 pprof
Loading 100% Layout completed
Load successful
libprofile.so detected (0x7f77468e8f90, 0x7f77468e8fd0), output 'pprof'
Profiling started, code: 0x1, timeout: 0
PROFILE: interrupts/evictions/bytes = 634/537/22168
http://igalia.com 1000 6.2709987870s

% mkdir out && ./runtests 1000 < urls.txt

github.com/aperezdc/websnap
...AND BEYOND

Ad-hoc Python/Bash scripts:

- Fix library paths in profiler output.
- Data munging.
- Measurements comparison.
- Generate CSV files.
- Report generation.

...
SOME RESULTS
(DETAILED)
ALPHA BLENDING
RESULTS

Speedup histogram
UP TO 30% FASTER RENDERING

Thanks to:

Orc backend using MIPS DSP instructions
QImage composition operations
Color conversion (RGB16/888 → ARGB32)
Alpha premultiplication and blending
String conversions and comparisons
UPSTREAM STATUS

Orc backend complete upstream
Initial work based on Qt 4.8
Most of the code is already in Qt 5.2
Rest in the next release
No backport to Qt 4.8
THANK YOU
FOR YOUR ATTENTION

perezdecastro.org
+AdrianPerezDeCastro
@aperezdc