

1 How powerful is you Nexus 7?

Geekbench 2, Androbench, AnTuTU... all those benchmarks are available to figure out the power of your Nexus 7. What is the reality when it comes to programming your Nexus 7? What can you expect of this 7 inch tablet?

For developers, Java Eclipse (SDK) is the obvious choice for Android devices. Unlashing the power of your Android tablet through C/C++, native code, should be possible through the Eclipse(NDK) with some complexity(Java plus C++/C plus the interface...). Renderscript could be a solution too, but it lacks documentation and examples for serious programming. Java AIDE allows you to write, compile and run programs directly on your Android tablet. Looking for simplicity and efficiency, RFO Basic gives access to many features of your Android devices such as graphics and multi-touch, device sensors... Programs may be compiled into applications. RFO Basic relies on Java. X-11 basic, with as syntax similar to the old GFA-Basic ATARI-ST, is still in a beta version. Python is available via QPython or Kivy. QPython implements Python language but is still in beta stage. Kivy Launcher enables you to run Kivy applications, an open source Python library for rapid development. Kivy applications make use of innovative user interfaces, such as multi-touch apps. C/C++ are available through C4DROID and CCTOOLS, an implemenattion of GNU GCC 4.8.1 compiler. C4DROID's author also wrote a Pascal compiler, PascalGUI. Fortran and LUA are available for download from CCTOOLS menu. Two versions of Processing language are available.

Java, Basic, C/C++, Python and Fortran seem good candidates to evaluate the power of a Nexus 7 with a test case.

2 code

For this task, a short CFD code, Computational Fluid Dynamics, was written. The geometry used is 2D1/2, cylindrical coordinates(z, r). The physics associated with this software includes the Navier-Stokes equations(fluid) and the Maxwell equations(electromagnetism), Magnetohydrodynamics, MHD in short. It solves the permanent conducting flow in a circular tube in the presence of a converging-diverging current. The current distribution is obtained by applying potential differences across a small electrode and two large screen electrodes covering the ends of the nonconducting tube, an MHD pump. The original Fortran code was published in *An Introduction to Computational Fluid Mechanics* by Chuen-Yen-Chow, in 1983. A revised edition, *An Introduction to Computational Fluid Mechanics by example* was released in 2011 by Biringen and Chow. The MHD stationary flow calculation is not included any more in the update but the details pertaining to the equations discretization, stability analysis... can still be found in their Benard and Taylor instabilities example, the instationary solution of Navier-Stokes equations coupled with the temperature equation.

For simplicity, a stream-vorticity formulation is used. Details of the equations solved are given below in nondimensional form. Standard boundary condi-

tions, or even simplified ones, are used (value or derivative given). Discretization of the nonlinear terms in the Navier-Stokes, the one involving the velocity components, was, historically, a source of problems. The numerical scheme has to properly capture the flow direction.

Upwind differencing form solves this problem. The spatial difference is on the upwind side of the point indexed (i,j). This numerical scheme is only first order by reference to Taylor serie expansion. The second order upwind schemes introduces non physical behaviour, such as oscillations. TVD, Total Variation Diminishing schemes, are a cure to this problem. They introduced stable, non-oscillatory, high order scheme, preserving monotonicity, no overshoot or undershoot for the solution. It is the result of more than 30 years of research in CFD...

Only upwind scheme was present in the original Fortran code. It was rewritten using a general TVD formulation. CTU, Corner Transport Upwind, was also added, experimental, not so much tested. Details can be in good CFD books such as *An Introduction to Computational Fluid Dynamics-The Finite Volume Method* by Versteeg and Malalasekera or *Finite Volume Methods for Hyperbolic Problems* by Leveque.

The solution procedure is straightforward. The current flow, RH variable, is solved via the Laplace equation solver. Then the electromagnetic force, EM variable, is computed. Time stepping is used to find the solution of the flow until convergence criteria are matched, error or maximum step. A Poisson solver is used. Comments are given in the Fortran source code.

Results are presented for a Reynolds number of 50, a magnetic pressure number C of 0.3, using the upwind scheme.

3 PC results

The PC cases were run on a Windows 7 X-64 I3 2.1 GHz laptop. The results are :

- 62,39 ms for GNU Fortran
- 78 ms for GNU GCC C99
- 150 ms for JAVA JDK 7U45
- 1,02 s for Pypy 2.0
- 6,78 s for Python 3.3.2

For this particular run, Fortran is the best for computations. C is very closed. Java JDK is a good surprise. All the sources for those languages were compiled. For interpreted languages, such as Python 3.3.2, the results are very disappointing as expected. Pypy, a fast, compliant implementation of Python, is faster than the original Python. Some more general remarks.

Some scientists are moving their codes to Python. They wanted to benefit from the scripting capabilities of interpreted languages. Users do not need to edit the source code to change the boundary equations or add a subroutine to solve a particular equation... Fipy, a finite volume code from NIST, uses this approach(see <http://www.ctcms.nist.gov/fipy/>) Most of the critical parts(solvers...) of such codes are written in C or in Fortran... Traceability of the R&D work is the major advantage. Short execution times may not be at the rendez-vous...

Another approach is to use a dedicated language such the one implemented in FreeFem++ software(see <http://www.freefem.org/ff++/>). FreeFem++ is a finite elements partial differential equation solver. It can solve multiphysics non linear systems in 2D and 3D. A problem with one billion of unknowns was in 2 minutes on the Curie Thin Node, a CEA machine. CEA is the French Atomic Energy and Alternative Energies Commission.

What does the Nexus 7 has to offer?

4 Nexus 7 results

The Nexus 7 cases were run on a 1.2 GHz 2012 Tablet. The Nexus 7 2013, with its Qualcomm Snapdragon S4 Pro at 1.5 GHz, may boost those results a step further.

The execution times are :

- 70 ms Fortran CCTOOLS with with -march=armv7-a -mfloat-abi=softfp -mfpu=vfpv3 -O3
- 79 ms Fortran CCTOOLS with -march=armv7-a -mfloat-abi=softfp -mfpu=vfpv3 -O2
- 120 ms C99 C4DROID with -march=armv7-a -mfloat-abi=softfp -mfpu=vfpv3 -O2
- 380 ms C99 C4DROID with -march=armv7-a -mfloat-abi=softfp -mfpu=vfpv3 (float instead of double)
- 394 ms C99 C4DROID with mfloat-abi=softfp (float instead of double)
- 420 ms C99 C4DROID with -march=armv7-a -mfloat-abi=softfp -mfpu=vfpv3
- 450 ms C99 C4DROID with mfloat-abi=softfp
- 1163 ms C4DROID with -msoft-float (float instead of double)
- 1500 ms C4DROID with -msoft-float
- 2100 ms JAVA AIDE Dexer optimizations
- 1563 ms Eclipse Java

- 3030 ms JAVA AIDE
- 24,7022 s QPython

Preliminary remarks. Some dispersions were observed between runs with C4DROID. Only best execution times are given. C99 CCTOOLS is more stable on this particular point. On the average, CCTOOLS C99 is equivalent to C4DROID, results not published here. For performances, the same ranking emerges: Fortran, C, Java, Python. With the proper flags set for compilation, CCTOOLS Fortran is able to cope with the PC results, a very good surprise. C99 is lagging behind the C99 PC. On the contrary, Java scores are a bad surprise. Do we have to blame the Dalvik virtual machine? Python, with its 24 s, is definitely to be forgotten for serious scientific computations. Tests were not conducted on RFO Basic and Kivy according to the Java/Python performances...

Nexus 7 2012 seems to be very powerful on this particular example. Can the results be extrapolated for more demanding programs, software that needs more time to run?

5 conclusions

Nexus 7 tablets are very high quality products. Google Android is smart and fun. Some dedicated programming language may be needed to easily unleash the power sleeping of those processors.