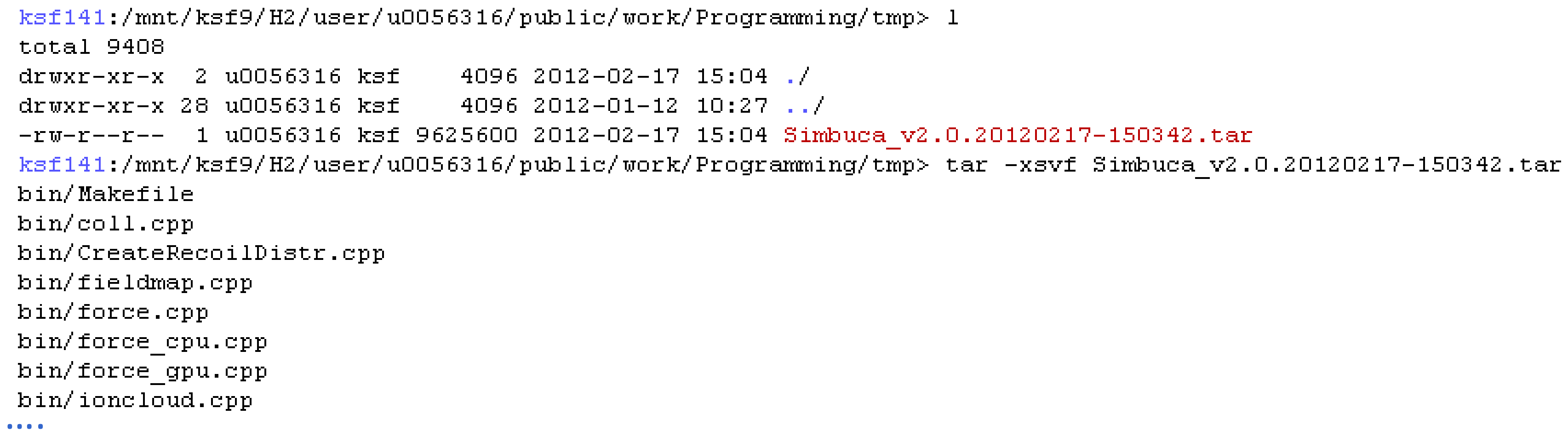
Starting up a simulation (1)

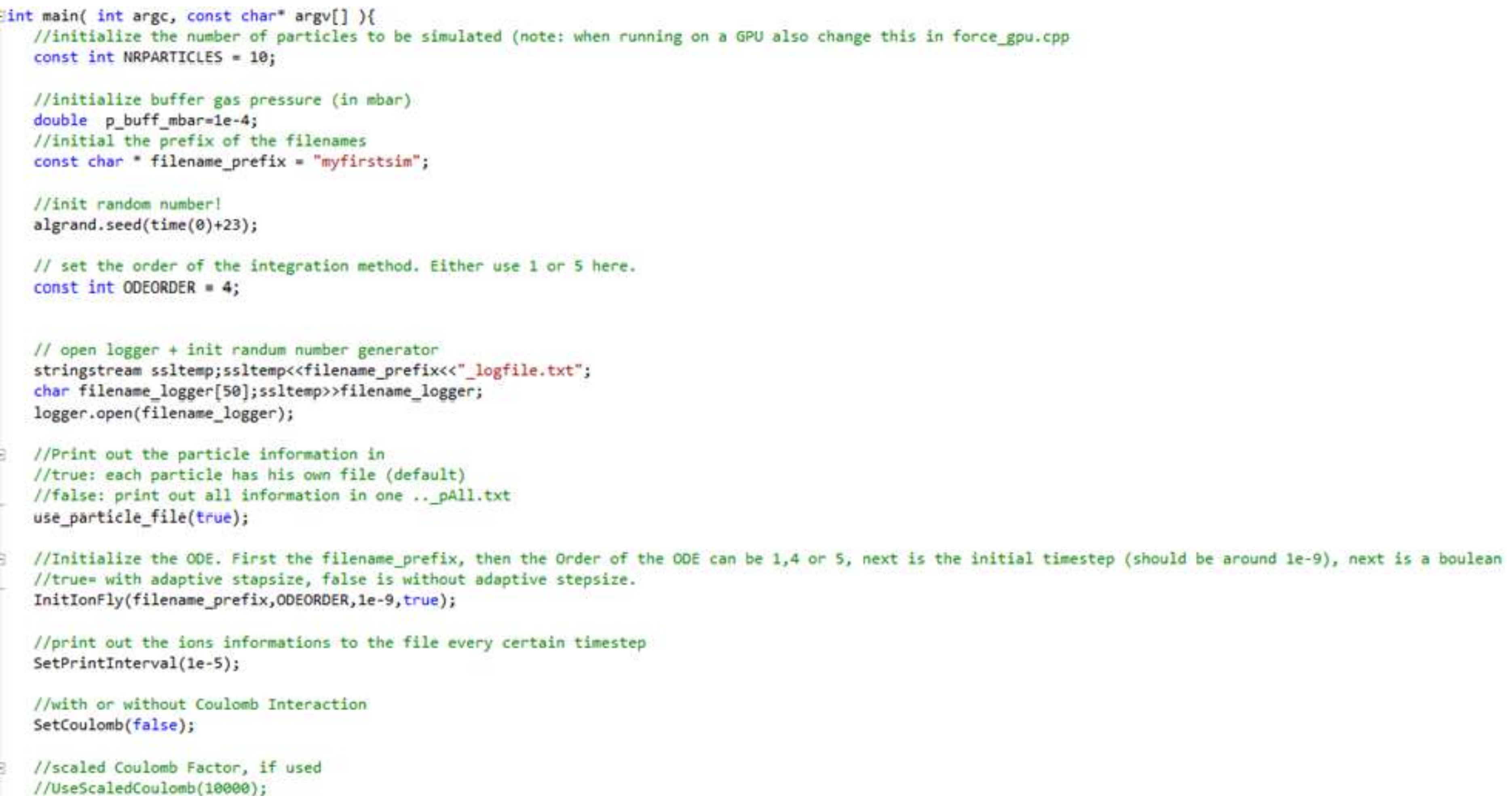
- Download program on a linux/windows PC (http://sourceforge.net/projects/simbuca)
- Compile the program



```
ksf141:/mnt/ksf9/H2/user/u0056316/public/work/Programming/tmp> l
total 9408
drwxr-xr-x  2 u0056316 ksf    4096 2012-02-17 15:04 ./
drwxr-xr-x 28 u0056316 ksf    4096 2012-01-12 10:27 ../
-rw-r--r--  1 u0056316 ksf 9625600 2012-02-17 15:04 Simbuca_v2.0.20120217-150342.tar
ksf141:/mnt/ksf9/H2/user/u0056316/public/work/Programming/tmp> tar -xsvf Simbuca_v2.0.20120217-150342.tar
bin/Makefile
bin/coll.cpp
bin/CreateRecoilDistr.cpp
bin/fieldmap.cpp
bin/force.cpp
bin/force_cpu.cpp
bin/force_gpu.cpp
bin/ioncloud.cpp
....
```

Starting up a simulation (2)

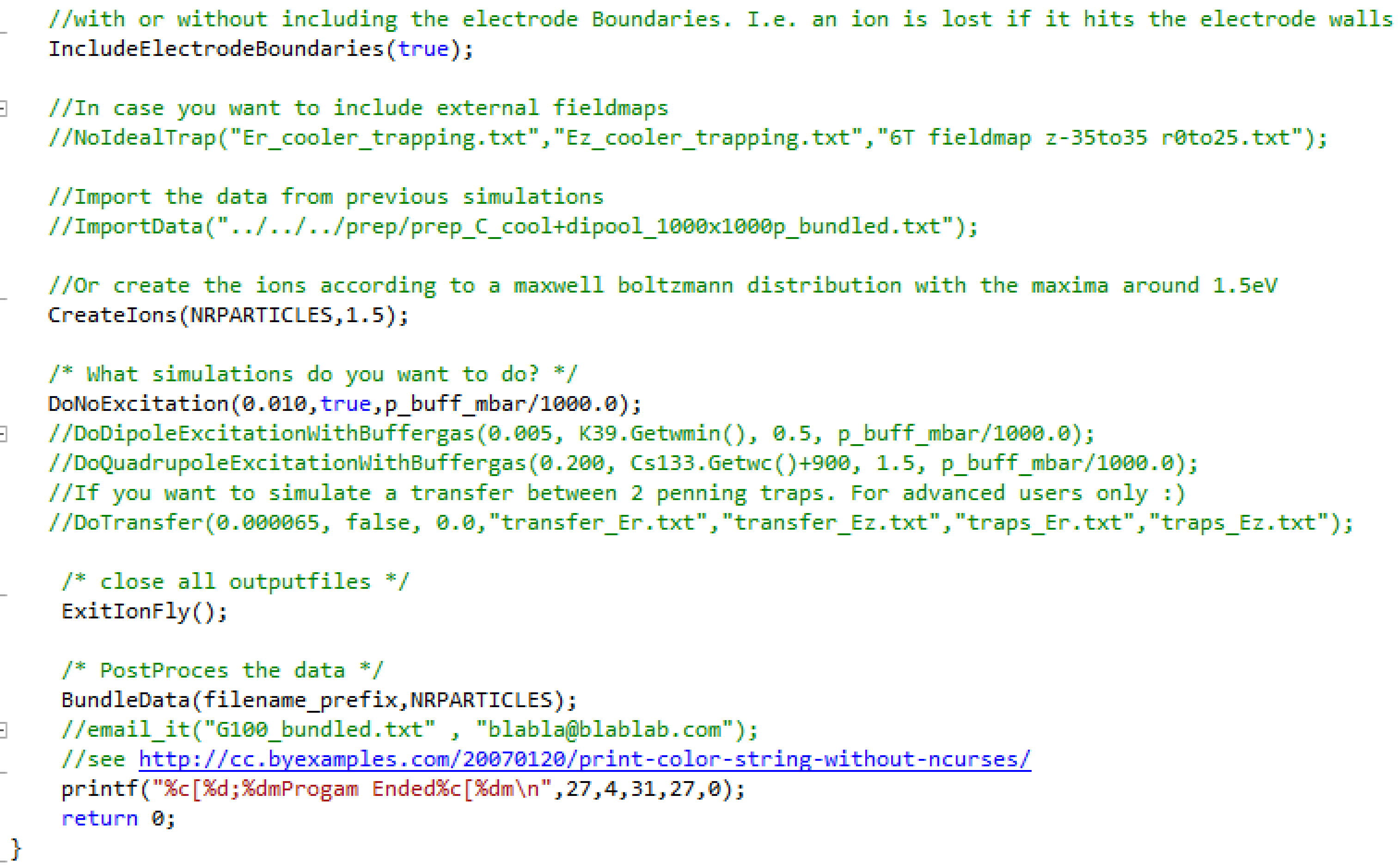
- Change variables in main.cpp according to what simulation you want to do



```
int main( int argc, const char* argv[] ){
    //initialize the number of particles to be simulated (note: when running on a GPU also change this in force_gpu.cpp
    const int NRPARTICLES = 10;

    //initialize buffer gas pressure (in mbar)
    double  p_buff_mbar=1e-4;
    //initial the prefix of the filenames
    const char * filename_prefix = "myfirstsim";

    //init random number!
    algrand.seed(time(0)+23);

    // set the order of the integration method. Either use 1 or 5 here.
    const int ODEORDER = 4;


    // open logger + init randum number generator
    stringstream ssltemp;ssltemp<<filename_prefix<<"_logfile.txt";
    char filename_logger[50];ssltemp>>filename_logger;
    logger.open(filename_logger);

    //Print out the particle information in
    //true: each particle has his own file (default)
    //false: print out all information in one .._pAll.txt
    use_particle_file(true);

    //Initialize the ODE. First the filename_prefix, then the Order of the ODE can be 1,4 or 5, next is the initial timestep (should be around 1e-9), next is a boulean
    //true= with adaptive stapsize, false is without adaptive stepsize.
    InitIonFly(filename_prefix,ODEORDER,1e-9,true);

    //print out the ions informations to the file every certain timestep
    SetPrintInterval(1e-5);

    //with or without Coulomb Interaction
    SetCoulomb(false);

    //scaled Coulomb Factor, if used
    //UseScaledCoulomb(10000);
```

Starting up a simulation (3)

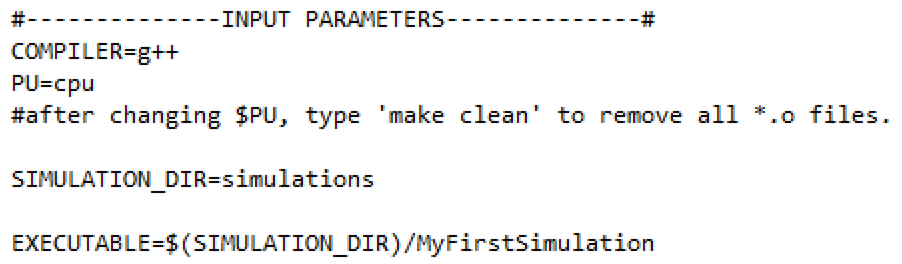
- Change variables in main.cpp



```
 //with or without including the electrode Boundaries. I.e. an ion is lost if it hits the electrode walls
 IncludeElectrodeBoundaries(true);

 //In case you want to include external fieldmaps
 //NoIdealTrap("Er_cooler_trapping.txt","Ez_cooler_trapping.txt","6T fieldmap z-35to35 r0to25.txt");

 //Import the data from previous simulations
 //ImportData("../../../prep/prep_C_cool+dipool_1000x1000p_bundled.txt");

 //Or create the ions according to a maxwell boltzmann distribution with the maxima around 1.5eV
 CreateIons(NRPARTICLES,1.5);

 /* What simulations do you want to do? */
 DoNoExcitation(0.010,true,p_buff_mbar/1000.0);
 //DoDipoleExcitationWithBuffergas(0.005, K39.Getwmin(), 0.5, p_buff_mbar/1000.0);
 //DoQuadrupoleExcitationWithBuffergas(0.200, Cs133.Getwc()+900, 1.5, p_buff_mbar/1000.0);
 //If you want to simulate a transfer between 2 penning traps. For advanced users only :)
 //DoTransfer(0.000065, false, 0.0,"transfer_Er.txt","transfer_Ez.txt","traps_Er.txt","traps_Ez.txt");

  /* close all outputfiles */
  ExitIonFly();

  /* PostProces the data */
  BundleData(filename_prefix,NRPARTICLES);
  //email_it("G100_bundled.txt" , "blabla@blablab.com");
  //see http://cc.byexamples.com/20070120/print-color-string-without-ncurses/
  printf("%c[%d;%dmProgam Ended%c[%dm\n",27,4,31,27,0);
  return 0;
}
```

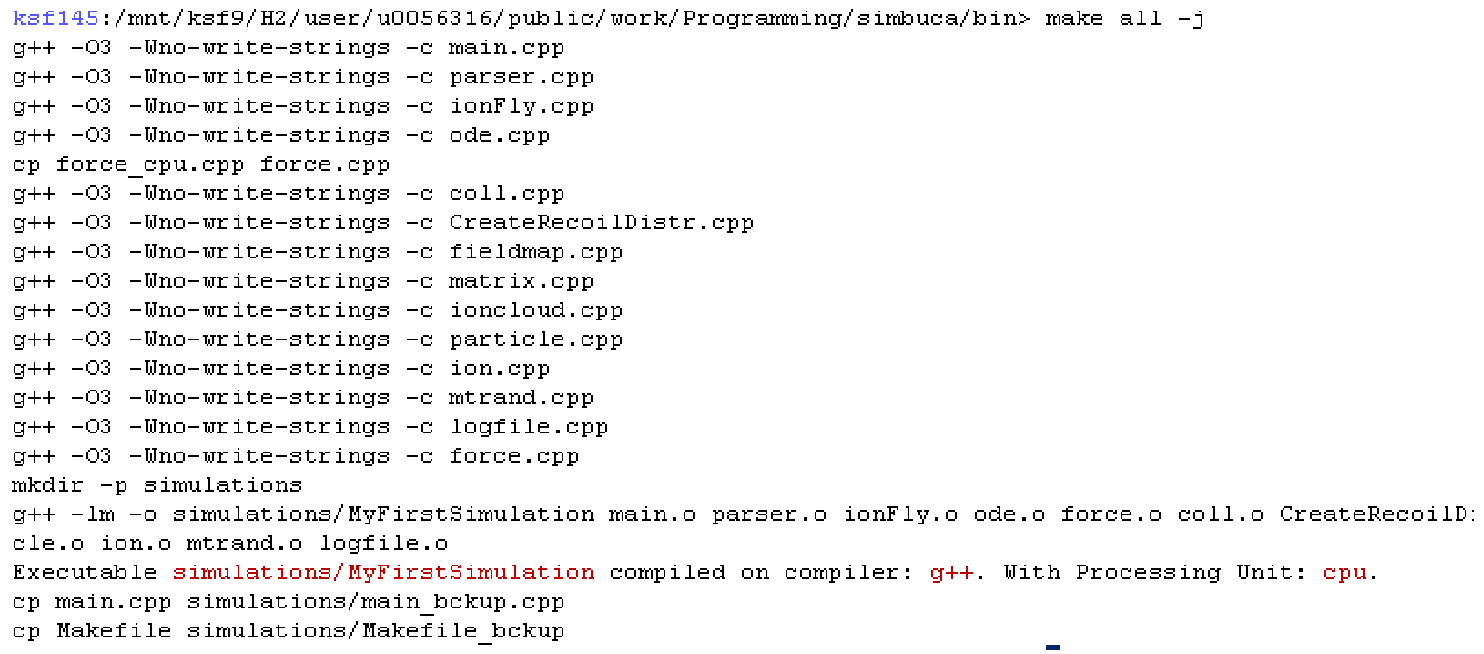
Compile

- Change variables in Makefile



```
#--------------INPUT PARAMETERS--------------#
COMPILER=g++
PU=cpu
#after changing $PU, type 'make clean' to remove all *.o files.

SIMULATION_DIR=simulations

EXECUTABLE=$(SIMULATION_DIR)/MyFirstSimulation
```

- compile the program



```
ksf145:/mnt/ksf9/H2/user/u0056316/public/work/Programming/simbuca/bin> make all -j
g++ -O3 -Wno-write-strings -c main.cpp
g++ -O3 -Wno-write-strings -c parser.cpp
g++ -O3 -Wno-write-strings -c ionFly.cpp
g++ -O3 -Wno-write-strings -c ode.cpp
cp force_cpu.cpp force.cpp
g++ -O3 -Wno-write-strings -c coll.cpp
g++ -O3 -Wno-write-strings -c CreateRecoilDistr.cpp
g++ -O3 -Wno-write-strings -c fieldmap.cpp
g++ -O3 -Wno-write-strings -c matrix.cpp
g++ -O3 -Wno-write-strings -c ioncloud.cpp
g++ -O3 -Wno-write-strings -c particle.cpp
g++ -O3 -Wno-write-strings -c ion.cpp
g++ -O3 -Wno-write-strings -c mtrand.cpp
g++ -O3 -Wno-write-strings -c logfile.cpp
g++ -O3 -Wno-write-strings -c force.cpp
mkdir -p simulations
g++ -lm -o simulations/MyFirstSimulation main.o parser.o ionFly.o ode.o force.o coll.o CreateRecoilD:
cle.o ion.o mtrand.o logfile.o
Executable simulations/MyFirstSimulation compiled on compiler: g++. With Processing Unit: cpu.
cp main.cpp simulations/main_bckup.cpp
cp Makefile simulations/Makefile_bckup
```

execute

- Go to the directory with executable and execute the program

```
ksf8:/mnt/ksf9/H2/user/u0056316/public/work/Programming/simbuca/bin/simulations> ./MyFirstSimulation
#simulation ended after 11 s
files succesfully bundled in myfirstsim_bundled.txt
Progam Ended
ksf8:/mnt/ksf9/H2/user/u0056316/public/work/Programming/simbuca/bin/simulations> █
```

- check the logfile

```
logfile created on:     Mon Feb 13 16:03:15 2012
In case of ideal trap: B = 6 T.U0/d2 = 18000
Choose Runga Kutta 4th order. With adaptive stepsize and abs error: 1e-07, rel error: 1e-07
Without Coulomb Interaction
With Electrode Boundaries
10 particles added. Max-Boltz distribution around 1.5 eV.
cloud with gaussian distribution among x,y,z with stdev (sigma_x,sigma_y,sigma_z) = (0.001;0.001;0.001) and initial offsetL (x;y;z) = (0;0;0.02)
No excitation for 0.01 sec,
with buffergas: 0.0001 bar
Close everything...
#simulation ended after 11 s
```

- check the outputfile

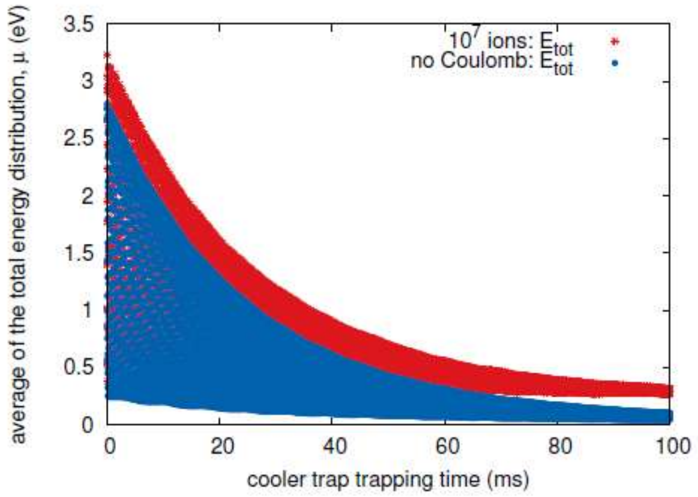
```
ksf8:/mnt/ksf9/H2/user/u0056316/public/work/Programming/simbuca/bin/simulations> cat *p1.txt | head
#Simulation started: Mon Feb 13 16:03:15 2012
#--------------------------------------------------------------------------------#
#                 Penning Trap Simulation Program by Simon Van Gorp              #
#       Dormand-Prince Runga-Kutta with Proportional Integrating controller      #
#--------------------------------------------------------------------------------#
index mass x y z (mm) vx vy vz (m/s)     r+(mm)           r-(mm)           R(mm)    Energy(eV)      Temperature(K)          t(ms)
**********************************************************************************
0  132.905 -0.154131 0.26821 17.8978 -757.764 -2166.05 190.649 0.527274 0.787755 0.309343 3.6519 42378.5 1e-06
0  132.905 -0.120547 0.505676 9.76066 233.883 -2283.08 -1818.66 0.527274 0.787755 0.519846 5.90565 68532.2 0.0100012
0  132.905 -0.187372 0.735128 -8.78982 1191.08 -1962.26 -1886.81 0.527274 0.787755 0.758631 6.08094 70566.3 0.0200002
```

Check the output simulation

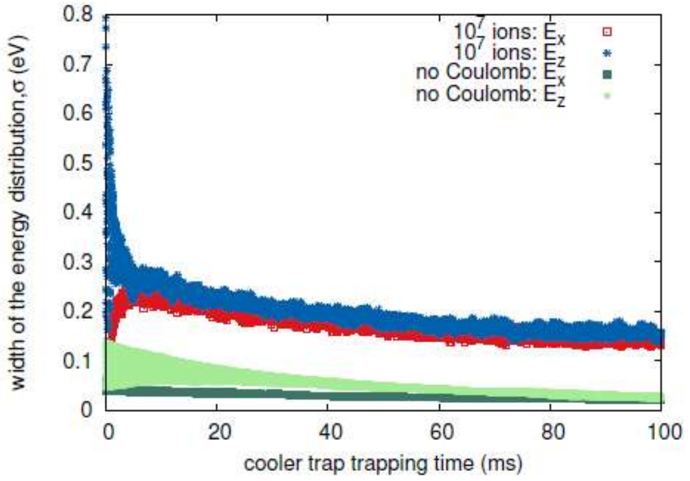
- Optional: Post-process with self-made functions or gnuplot (i.e. the easiest) or with functions parser.cpp or with root / linux bash

```
void BundleData(const char* filenamebegin, int nrparticles);
void PrintIonCloudinfo(const char *beginstream, int nrparticles, double diaphragm_radius_mm);
void PrintIonCloudGaussEvo(const char *beginstream, int nrparticles, double diaphragm_radius_mm);
```

```
ksf8:/mnt/ksf9/H2/user/u0056316/public/work/Programming/simbuca/bin/simulations> cat myfirstsim_cloud_GaussEvo.txt | head
->1.time(ms)     2.#particles    xu       yu       zu       6.vxu   vyu      vzu              9.xs     ys       zs       12.vxs          vys      vzs      15.ru    rmu
           18.rs    rms      rps      21.Eu    Tu       Es       Ts
```



(a) Average energy, μ, of the ion cloud



(b) Spread, σ, of the energy of the ion cloud

- When you want to run a simulation on the Graphics card (GPU). Change the PU variable in Makefile from CPU to GPU. Afterwards, type “make clean” in the console to erase the object files from the CPU before typing “make all”.