PDP1

Plasma Device 1 Dimensional Bounded Electrostatic Code

Plasma Theory and Simulation Group
EECS Department, University of California,
Berkeley, CA 94720

Contents

1 INTRODUCTION

XPDP1 is a bounded electrostatic code for simulating 1 dimensional Plasma Devices running on Unix workstations with X-Windows, and PC's with an X-Windows emulator. The code simulates a bounded plasma plus external circuit system whose characteristics, including particles and electrostatic fields, are specified by the user at run time using an input file. The external circuit includes R, L, and C elements, as well as AC, DC, and ramped current and voltage sources. The simulation proceeds in real-time, with the user viewing output as the code runs in the form of various user specified diagnostics which are updated each time step (animation).

The code compiles with standard C compilers and requires X-Windows libraries (X10 or higher). This version of PDP1 differs from the Cray FORTRAN version (PDW1 - Plasma Device Workshop) originated by W. S. Lawson on several physics issues summarized below. The X-Windows version provides interactive animation whereas the FORTRAN version produces output files requiring post processing.

1.1 Scope

This document describes the XPDP1 programs running on the workstations and UNICOS Cray environment. The general physics issues involved in a bounded plasma simulation are discussed briefly. Program installation, operation, and modification are discussed. In addition, the library of input files accompanying XPDP1 is described, and the guidelines to generate new input files are provided.

This manual makes no attempt to explain exhaustively the physics and computational issues of particle simulation. Many comprehensive texts on particle simulation are available [1][2]. For information on the Cray FORTRAN version of PDW1, see W. S. Lawson [3].

Some familiarity with plasma physics is required to understand the results of the simulations and generate new simulations. Knowledge of numerical analysis and/or particle simulation is useful for modification of the code and understanding of numerical errors which can occur in any computer simulation.

1.2 Objectives

The objectives of this package include distribution of the $XPDP1$ codes in a form which makes it accessible to large and small research facilities, universities, and individuals interested in plasma simulation. The Unix and X-Windows environments provide a vehicle for distribution since nearly every researcher and student has access to this type of machine. The trend in recent years has been one of decreasing cost and increasing performance and capability.

1.3 History

PDW1 was originally written by W. S. Lawson for a plasma device seminar on axially bounded plasma systems at the University of California at Berkeley in 1983. We will discuss differences with the Cray FORTRAN version below as appropriate.

The FORTRAN code was translated into Cray-C by T. L. Crystal in 1987. Crystal made simplifications in the external circuit as well as a number of programming-related changes. PDP1 has significant differences from these previous versions of PDW1, containing more accurate and versatile circuit solvers, simultaneous solution of the circuit and potential equation coupled through boundary conditions at the plasma wall, a basic ionization model, MKS units, and Fourier transform diagnostics as well as many additional diagnostics and tools.

1.4 Enhancements

This section summarizes the enhancements made to the PC version of PDP1 since the original version (PDW1) written in FORTRAN by Lawson. The enhancements are presented in the order they occurred, referenced by version number. Note that the original version has no version number, and the Cray-C version due to Crystal has been given version number PDW1-1.0 although it had no explicit version number.

1.4.1 Version PDW1-2.0

The code was translated into Microsoft C 5. 1 running under MS-DOS on IBM PC compatible microcomputers. The diagnostics code was replaced with real time graphics animation in EGA resolution (640x350 pixels, 16 colors). The code had limitations on particles, diagnostics, etc., and the user interface consisted of mnemonic keys to display snapshots of diagnostics. The code could only run while displaying phase space, and had a maximum time-step limitation.

Version PDW1-2.0 was never officially released due to the rapid development of PDP1 described below.

1.4.2 Version 1.0

Version 1.0 is a major overhaul providing PDP1 with an improved user interface and output capabilities. The core of the user interface, designed and written by J. P. Verboncoeur and V. Vahedi[4], handles the programming issues of the simulation. These include keyboard handling, screen graphics, and printer output for Postscript and IBM Graphics printers (including the Epson FX compatible family of printers) as well as CGM (Computer Graphics Metafile) output for editing by many drawing programs. The code is implemented in an object-oriented style in standard C to facilitate enhancements, including new menu items and diagnostics. The code is separated into a physics application and the windowing core as shown in Figure 1. New physics and diagnostics can be added without altering the windowing code, with the only restriction that any new diagnostic must be a linear, semi-log, or scatter plot. A text plot is currently under consideration which would display parameters from the input file during the simulation.

Using the windowing core, all diagnostics are updated dynamically in time. The core can also update in individual time-steps, pausing for a keystroke before continuing the simulation.

The code runs significantly faster when fewer (or zero) plots are displayed on screen. Therefore, when output is not being viewed (long runs overnight, coffee break, etc.), it is more efficient to remove all plots from the screen.

PDP1 no longer has a time step limitation; it can run indefinitely. All time histories are combed periodically such that there are never more than HISTMAX (a

Figure 1: Schematic representation of the interaction between WinGraphics and the physics kernel.

programmable constant) values stored. Note that after long runs this can result in a loss of high frequency resolution on the time history plots. This has no effect on the physics of the simulation since the diagnostics are simply a view of the physical results.

PDP1 can now generate hard copy output on an IBM Graphics Printer or compatible, including the Epson FX series of dot matrix printers. PDP1 can generate a CGM format file for further editing by many drawing packages which support ANSI standard CGM and for inclusion in many word processors and desktop publishing programs supporting CGM. In addition PDP1 can generate a Postscript file which can be sent to any Postscript printer or included in a Postscript document. The resolution and grey-shading of the CGM and Postscript output is dependent only on the output device, while the dot matrix output is at the screen resolution in monochrome.

PDP1 now employs four circuit solvers to handle the full range of external circuit parameters. The general voltage-driven series RLC case is solved using a second order backward Euler method. For the open circuit case, $C \to 0$, the external circuit no longer needs to be solved; the boundary surface charge influences the potential as always, but it cannot exchange charge (via current) with the other boundary. For $C \to \infty$ and $R = L = 0$, the external circuit becomes a short circuit, so the applied potentials are applied directly to the plasma boundaries. The final case is an ideal current source, which imposes the specified current independent of the external circuit parameters or boundary potentials. For a complete discussion of the simultaneous solution of the external circuit and the spatial plasma potential, refer to the attached paper[5].

FFT routines are used to transform time histories into frequency domain plots. Both amplitude and phase of the transformed quantity are available. This feature is useful when looking for response of a bounded plasma at one or more frequencies, as well as determining the impedance of the plasma to external currents/voltages.

1.4.3 Version 2.0

Version 2.0 supports a number of enhancements, including a Monte Carlo collisional model for electron- and ion-neutral collisions. Dump files are now supported. The current simulation state can be saved to a file for later restart. This is useful for saving long runs toward equilibrium, and changing a few parameters and restarting close to the new equilibrium.

In addition, the right boundary of the simulation is now assumed to be the zero reference potential in PDP1. In PDC1 and PDS1, the left boundary represents the inner (driven) electrode, while the right boundary is the zero reference potential as in PDP1. A flag has been added for refluxing at the right boundary as an alternative to the reference potential; refluxing is explained in detail in 4.1.3 Flags.

1.4.4 Version 2.1

An error was discovered in the Monte Carlo collisions (MCC) between charged particles and neutrals in calculating the scattered velocity from the incident velocity. In this calculation, one must rotate the velocity vector and adjust its magnitude accordingly. In the rotation part, a set of orthonormal vectors is needed to set up a coordinate system to rotate the incident velocity vector by theta and phi to the scattered velocity vector. This set of orthonormal vectors was NOT calculated properly (they were orthogonal to each other but NOT normal). The vector set that we used was [$\vec{v}, (\vec{v} \times \vec{y}), \vec{v} \times (\vec{v} \times \vec{y})$] which should have been [$\vec{v}/|\vec{v}|$, $(\vec{v} \times \vec{y})/|\vec{v} \times \vec{y}|$, $(\vec{v} \times (\vec{v} \times \vec{y}))/|\vec{v} \times (\vec{v} \times \vec{y})|$]. If the vector set is not normalized, the magnitude of the scattered velocity is also affected through the rotation. In fact the scattered velocity will be smaller than the incident velocity. This would act as a cooling mechanism for the particles, and the system would loose energy.

In addition a digital smoothing filter (1, 2, 1) is added to the planar case (PDP1) which requires an flag in the input file (see the list of input parameters).

1.4.5 Version 3.0 (XPDP1)

XPDP1 is the X-Windows version of XPDP1 running under X11. This version makes extensive use of a mouse for selecting menus, moving windows, resizing, etc. In addition, XPDP1 can be run on a remote host while the output displays on a local graphics workstation. XPDP1 uses XGrafix for the graphics display, which requires X11 libraries to compile.

1.4.6 Version 3.1 (XPDP1)

In the previous versions, the code would allocate position and velocity arrays for each species based on a parameter called MAXLEN (defined previously in the header file) which was the maximum number of particles for all species. In that case, the user would have had to recompile the whole code if more particles were desired.

The input files in this version have an extra parameter per species (called max-np) for the maximum number of particles of that species. This avoids the complications of recompiling and at the same time allows the user to define variable array sizes per species.

1.4.7 Version 4.0 (XPDP1)

Xgrafix has been updated to use Tcl/Tk libraries for dialog boxes. The installation of Xgrafix now requires that these libraries be also installed.

The input files in this version has been many new parameters. The major one being **GAS**. This parameter picks out one of the four gases built in to PDP1 along with the respective collisions. As a result of this, the collision part of the input file is obsolete and no longer required. This subsection is explained in later sections for continuity and for help in understanding collision implementation. The gases are enumerated below.

- GAS = 1: Helium
- GAS = 2: Argon
- GAS = 3: Neon

- GAS = 4: Oxygen [1]

In addition to this, an implicit mover *imp*, volume source *psource* and sub cycling k have also been implemented.

1.4.8 Version 4.1 (XPDP1)

A second order accurate injection push has been implemented. Also all components of velocity can be specified independently. The inversion of the distribution functions is now second order accurate. The gas package now includes a fifth option (GAS = 5) which allows the use of the old cross section parameters (which are explained in section 4.1.5-6). These parameters need to be added to mcc.cpp before compilation.

[1] Gas = Oxygen requires that $nsp = 3$

2 INSTALLATION

This section describes the contents of the XPDP1 distribution and its installation procedure.

2.1 Distribution Format

XPDP1 is no longer distributed on floppy diskettes. Instead the preferred distribution procedure is through anonymous FTP. The codes are available in the tar gzipped form from the **PTSG** web site at *http://ptsg.eecs.berkeley.edu*. **xgrafix** distribution is available separately at the same web site and has to be installed for any of the PTSG family of codes to work. The contents of the XPDP1 distribution contains:

README Text file containing the direction for installing XPDP1 on a Unix platform.

XPDP1.TAR This tar file contains the files required for XPDP1.

\xpdp1 directory:

README Text file containing information about licensing.

.c All files with the .c extension are the C language source files for XPDP1. These files should be placed in the xpdp1 directory.

.h All files with the .h extension are the C language header files for XPDP1. These files should be placed in the xpdp1 directory.

makefile The make file for automatically performing conditional compilation /linking of only those files which have been changed. This file should be placed in the xpdp1 directory.

\xpdp1\inp directory:

*.inp All files with the inp extension are input files. For detailed information on

each input file refer to Section 4.2 Input File Library.

A directory called inp is set up under the xpdp1 directory to include all the input files.

\xpdp1\doc directory:

xpdp1.ps	PostScript file containing the documentation.
xpdp1.txt	ASCII file containing the documentation.

The **xgrafix** distribution contains the required files for displaying graphics in the X Windows version. These are:

README Text file containing some directions and information on compiling the XGrafix libraries. When the information in this manual conflicts with the README file, assume the file is correct.

xgrafix.c The source file for the XGrafix graphics display library. This file should be placed in the xgrafix directory.

xgrafix.h Header file for XGrafix. This file should be placed in the xgrafix directory.

xgrafix.ico The XGrafix icon (bitmap). This file should be placed in the xgrafix directory.

xgrafix.str Another header file containing string definitions for XGrafix. This file should be placed in the xgrafix directory.

makefile The make file for XGrafix. This file should be placed in the xgrafix directory.

2.2 Setup and Installation Procedure (X Windows version)

The installation procedure to a workstation must be done manually. Take the tar file along with the README file to your workstation or Unix platform. Follow the directions in the README file for installation and compiling. Read the README file in the xgrafix directory as well as the makefile before compiling XGrafix, then compile xpdp1.

3 X-WINDOWS PROGRAM OPERATION

The X-Windows version of PDP1, XPDP1, is operated in the same manner as discussed in the manual for the PC version, with the exceptions noted below.

3.1 Syntax

```
xpdp1 -i filename[.inp] -d [dumpfile.dmp]
```

where $< filename.inp >$ is the name of the input file. Although we have used *.inp for the input files in the library, the .INP extension is not required. If no filename is provided on the command line, XPDP1 displays an error message. The dumpfile parameter is optional; it must be an existing file created by the same version of the code. If the input files are not in the same directory or are located in a sub-directory, the path must also be specified. For instance, the syntax for starting XPDP1 with the input file vc.inp which is in a sub-directory of xpdp1 called INP is:

```
xpdp1 -1 inp/vc[.inp]
```

The input file is required since XPDP1 determines the parameters of the simulation at run time.

3.2 GUI Support

XPDP1 fully supports a mouse for selection of items, buttons etc. Moving, resizing, and iconifying of windows is supported indirectly via the X window manager (Motif, Open Look, etc.). Keystrokes are not supported for these actions, so a mouse is required. The move, resize, and iconifying buttons and operations are governed by the window manager; consult the window manager manual or guru for details of these procedures.

3.3 Main Menu

The buttons on the main menu can be selected using the mouse. The functions available include RUN, STOP, STEP, SAVE, and QUIT, which all perform the same function described previously in Section 3. Note that the SAVE function is equivalent to the DUMP function in the MS-DOS version which is also NOT implemented in this version.

3.4 Diagnostic Window Buttons

Every diagnostic window in XPDP1 contains four buttons: Rescale, Trace, Print, and Cross-hair.

3.4.1 Rescale

The rescale button pauses the simulation and opens a dialog box containing editable fields for the minimum and maximum labels on the x and y axes. In addition, the dialog box contains buttons for automatic rescaling of the x and y axis. These buttons toggle auto rescaling of the respective axis on and off. When all axes are scaled as desired, select OK to accept the changes or CANCEL to return to the previous status. Note that while rescaling the simulation is paused.

3.4.2 Trace

The trace button turns toggles the plot tracing feature on and off. The previous plots are accumulated, generating a series of lines or dots as described above.

3.4.3 Print

The Print button generates a PostScript plot file of the current window. Pressing the button opens a dialog box containing the file name for the plot and a plot title. Selecting OK generates the plot, CANCEL returns to the simulation. Note that the simulation is paused while the dialog box is open.

3.4.4 Crosshair

The crosshair button activates the crosshair pointer and opens a dialog box displaying the coordinates of the pointer. To display the coordinates of a point move the crosshair pointer to the desired location and click. The simulation is paused until the crosshair is deactivated by selecting the Crosshair button again.

3.5 Diagnostics

In XPDP1 diagnostics there is no diagnostic menu list. Instead, all diagnostics appear as icons at the bottom of the display or in the designated icon area depending on the window manager. To open a diagnostic, simply click on its icon. In addition, some window managers will display a list of the available diagnostics with all other open windows in a window list. Note that this is not a feature of XPDP1, but rather a feature of the X-Windows Manager used on the system.

4 INPUT FILES

XPDP1 obtains its versatility through the use of input files. The input file contains the parameters for the simulation, specifying number of each species, grid spacing, charge to mass ratios, etc. This section describes the contents, use, and modification of input files for XPDP1.

4.1 Input File Parameters

The codes use input files to describe the simulation, including the physical bounded device parameters, external circuit, RF drive, etc. (global parameters), as well as the parameters describing each species of particles. Units, if any, are shown in [ ].

4.1.1 Global Parameters

nsp The number of particle species to simulate (0= no species present, may use this option to check the system, 1= one species in the whole system, etc.). If modifying an input file that has, say, 2 species, to add more species, just copy one of the blocks of parameters corresponding to species 1 or 2, and change the parameters to the desired values. Note that each species added seeks another 100 kByte of memory.

nc The number of spatial cells. The cell width is calculated using $\Delta x = \mathit{length}/\mathit{nc}$.

nc2p The number of physical particles per computer particle. The number of super particles in the simulation is found using $N = \mathit{initn} \cdot \mathit{area} \cdot \mathit{length}\,/\mathit{nc2p}$, where *initn* is the uniform number density.

dt The time step [sec].

length The length of the system (distance between electrodes) [m].

area Electrode area [m^2]. Allows application of real currents and real external circuit parameters.

epsilonr Background relative dielectric constant of system.

B Applied homogeneous magnetic field [Tesla].

PSI Angle the B-field makes with the normal from the electrode, x-axis (B-field is in the x-z plane) [deg].

rhoback Fixed background charge density (non-accelerating) [C/m^3].

backj Background current density (non-accelerating) [Amps/m^2].

dde Sinusoidal perturbation of charge density ($\delta x/l$) at $t = 0$; $\delta x(x) = l \cdot dde \cdot \sin(2\pi x/l)$

extR External circuit resistance [Ohms].

extL External circuit inductance [Henries].

extC External circuit capacitance [Farads].

q0 Initial capacitor charge [C].

4.1.2 Applied Voltage Or Current Sources

When the flag dcramped is off, the general form of the applied source is:

$$S(t) = DC + Ramp \cdot t + AC \cdot \sin(2\pi f_0 t + \theta_0)$$

where $S(t)$, the applied source, is either a current or a voltage source.

The flag dcramped should be turned on (set to 1) when a step function is desired. The step function can have a zero rise time, $Ramp \gg 1$, or can be ramped to its final DC value either with a constant slope or sinusoidally as shown in Figure 2. The rise

Figure 2: When the flag dcramped is set, the signal is ramped to its final DC value sinusoidally (left), or with a constant slope (right).

time of the applied signal is equal to 1/2f0, in the sinusoidal case, or DC/Ramp, in the constant linear slope case.

source Voltage or current source indicator (v=voltage, i=current).

dcramped Flag for ramping external voltage/current source to a final DC value (1=yes, 0=no).

DC DC voltage or current source [V or Amps]. Zero value indicates zero dc voltage.

Ramp Rate of ramping for voltage or current source [V/sec or Amps/sec]. Zero value indicates zero ramping for voltage.

AC AC voltage or current source [V or Amps]. Zero value indicates zero ac voltage and the values of $f0$ and $theta0$ are ignored.

f0 AC source driving frequency [Hz].

theta0 Initial phase angle of AC source [deg].

4.1.3 Flags

secondary Secondary electron emission flag (0=off, 1=species 1 emitted, etc.).
The emitted electron species give the emitted velocity distribution at the surface specified for the species (see SPECIES PARAMETERS).

e_collisional The flag for ionization, elastic, and excitation electron-neutral collisions (0 = off, 1 = species 1 is the colliding electron species, etc.).
Note: Only ONE species can be the colliding electron species.

i_collisional The flag for scattering and charge exchange ion-neutral collisions (0 = off, 2 = species 2 is the colliding ion species, etc.).
Note: Only ONE species can be the colliding ion species.

reflux The flag for refluxing the particles at the right wall (0=off, 1=on). In this case, the particles hitting the right wall are not absorbed but reflected back into the system. Since the right wall in this case does not charge up, it serves only as a symmetry plane allowing for a semi-infinite plasma at the right wall. The particles of each species are refluxed at the temperature specified for the species.

nfft Number of samples for the Fast Fourier Transform analyzer (must be a power of 2). When this parameter is set to zero, no FFT analysis is done, and the diagnostics in the frequency-domain are NOT shown.

n_ave Number of samples for the average diagnostics. When this parameter is set to zero, no averages are not done and NOT shown.

nsmoothing Number of time that a (1, 2, 1) digital smoothing filter is applied to the charge density arrays prior to the field-solve. This filter is only present in PDP1.

4.1.4 Wall Emission Coefficients and Neutral Gas Parameters

seec(elect.) The coefficient of secondary electron emission due to the first species striking the two electrodes. If this parameter is set to say 0.1, on average one electron is injected for every 10 incident particles of this species.

seec(ions) The coefficient of secondary electron emission due to the second species striking the two electrodes.

ion species indicates the ion species created by electron-neutral ionization collisions (2=the created ions are of type species 2, etc.).
Note: this also specifies the type of the background neutral gas particles colliding with electrons.

Gpressure Background neutral gas pressure [Torr].

Gtemp Background neutral gas thermal temperature [eV].

4.1.5 Electron-Neutral Collisional Parameters

The cross-sections [2] used in the code are close fits to the values measured experimentally. The general expression for all electron-neutral collision cross-sections, as seen in Figure 3, is:

$$\sigma(E) = 0 \qquad E < E_0$$
$$\sigma(E) \propto E \quad E_0 \le E \le E_1$$

[2] These details are built-in to XPDP1 in version 4.0

Figure 3: General profile for electron-neutral c versus incident particle energy E.

$$\begin{aligned} \sigma(E) &= \sigma_{max} \quad E_1 \leq E \leq E_2 \\ \sigma(E) &\propto \frac{\ln(E)}{E} \qquad E_2 \leq E \end{aligned}$$

selsmax Maximum electron-neutral elastic cross section [m^2].

elsengy0 Elastic collision threshold energy [eV].

elsengy1 Low energy of plateau for elastic collisions [eV].

elsengy2 High energy of plateau for elastic collisions [eV].

sextmax Maximum excitation cross section [m^2].

extengy0	Excitation threshold energy [eV].
extengy1	Low energy of plateau for excitation [eV].
extengy2	High energy of plateau for excitation [eV].
sionmax	Maximum ionization cross section [m^2].
ionengy0	Ionization threshold energy [eV].
ionengy1	Low energy of plateau for ionization [eV].
ionengy2	High energy of plateau for ionization [eV].

4.1.6 Ion-Neutral Collisional Parameters

The cross-sections [3] for ion-neutral collisions are assumed to be in the form $\sigma = a + b/\sqrt{E}$, as seen in Figure 4, where a and b are determined by finding the best fit to the experimental values. This empirical fit seems to closely resemble the experimental cross-sections measured for helium, argon and neon.

achrgx	Charge exchange cross section [m^2].
bchrgx	Charge exchange cross section [m$^2\sqrt{eVolt}$].
ascat	Scattering cross section [m^2].
bscat	Scattering cross section [m$^2\sqrt{eVolt}$].

4.1.7 Species Parameters

One set for each species should be specified.

[3] These details are built-in to XPDP1 in version 4.0

Figure 4: General profile for ion-neutral collision cross-section versus incident particle energy E.

max-np The maximum number of particles per species.

q Charge per physical particle [C].

m Mass per physical particle [kg].

j0L Magnitude of injected current density from the left electrode [Amps/m^2].

j0R Magnitude of injected current density from the right electrode [Amps/m^2].

initn Initial species physical density in the system [m^{-3}].

max-np The maximum number of particles per species.

4.1.7.1 Velocity Distribution

vx0L Drift velocity for $v > 0$ particles [m/sec].

vx0R Drift velocity for $v < 0$ particles [m/sec].

vxtL Thermal velocity for $v > 0$ particles [m/sec].

vxtR Thermal velocity for $v < 0$ particles [m/sec].

vxcL Cutoff velocity for $v > 0$ thermal distribution [m/sec].

vxcR Cutoff velocity for $v < 0$ thermal distribution [m/sec].

v0y Drift velocity in the y directions for particles [m/sec].

vty Thermal velocity in the y direction for particles [m/sec].

Figure 5: Velocity distribution function in x-direction. The distribution function in the perpendicular direction does not have a cutoff, but may have a drift.

v0z Drift velocity in the z direction for particles [m/sec].

vtz Thermal velocity in the z direction for particles [m/sec].

Note that the thermal velocity components are defined for a Maxwellian distribution as $v_{thermal} = \left(\frac{k_B T}{m}\right)^{1/2}$, where k_B is Boltzmann's constant, T is the temperature, and m is the mass of the species.

4.1.7.2 Energy Distribution Diagnostics

nbin Number of bins for the energy distribution diagnostic of the species at left wall.

Emin The minimum energy seen in the energy distribution diagnostic of the species at left wall [eV].

Emax The maximum energy seen in the energy distribution diagnostic of the species at left wall [eV].

Parameters for an energy distribution function in the system. The parameters XStart and XFinish designate a region (a window) in the space over which the energy distribution is calculated.

nbin Number of bins for the energy distribution diagnostic of the species in the system.

Emin The minimum energy seen in the energy distribution diagnostic of the species in the system [eV].

Emax The maximum energy seen in the energy distribution diagnostic of the species in the system [eV].

XStart The left boundary of the region over which the distribution is calculated.

XFinish The right boundary of the region over which the distribution is calculated.

4.1.7.3 Velocity Distribution Diagnostics

vx_lower Lower velocity for velocity distribution diagnostics [m/sec].

vx_upper Upper velocity for velocity distribution diagnostics [m/sec].

nxbin Number of bins used. (if 0 diagnostics is turned off.)

vy_lower Lower velocity for velocity distribution diagnostics [m/sec].

vy_upper Upper velocity for velocity distribution diagnostics [m/sec].

nybin Number of bins used. (if 0 diagnostics is turned off.)

vz_lower Lower velocity for velocity distribution diagnostics [m/sec].

vz_upper Upper velocity for velocity distribution diagnostics [m/sec].

nzbin Number of bins used. (if 0 diagnostics is turned off.)

4.2 Input File Library

Clearly the number of data sets is virtually unlimited. PDP1 is accompanied by a number of prepared simulations discussed below. The user is encouraged to make a working copy of the data sets to edit using any ASCII word processor or editor. Note that the relative position of the numeric data is important for useful results, but the number of spaces separating each number is unimportant as long as the numbers remain on the same line. Comments may be added on the lines containing the descriptive text (so long as it remains on a single line) and at the end of the input file an unlimited number of text lines are supported.

4.2.1 PIIIA.INP (PIIIH.INP)

This input file simulates plasma immersion ion implantation (sometimes referred to as plasma source ion implantation) in an argon (hydrogen) plasma. A single short step potential is applied to the target. The electrons are pushed away from the target before the ions have responded, generating an ion matrix (uniform) sheath. On a longer time scale, the ions are accelerated by the resultant electric field across the sheath and implanted into the target. As the ions are implanted, the ion density in the sheath drops. This causes the sheath-plasma edge to recede and uncover more ions to increase the ion density in the sheath and sustain the potential drop across the sheath. The velocity of the moving sheath edge depends upon, among other factors, the pressure of the background neutral gas. One can vary the pressure of the neutral gas and observe the difference in the profiles of ion energy and angular

distribution at the target. Electrons and ions are injected and absorbed at the right wall. The rate of injection for electrons is higher than that for ions by the square root of the mass ratio (mi/me) to reduce the source sheath produced at the right wall.

4.2.2 QMACH.INP

This input file simulates a Q-machine, in which Maxwellian distributions of electrons and ions are injected from the right electrode. The electrons reach the left electrode before the more inertial ions, charging the (floating) electrode negatively. After a few ion transit times, a sheath drop region near the wall can be seen in the spatial potential, followed by a flat region of essentially charge-neutral plasma. Depending on the difference in the electron and ion injected current densities, a source sheath may be generated at the injection electrode.

4.2.3 RFDA.INP (RFDH.INP)

This input file simulates a capacitively coupled RF discharge started with a few ionized particles in an argon (hydrogen) plasma. Electrons responding to the RF field applied to the electrodes gain energy and ionize neutrals, building the plasma initially, until some equilibrium is reached when the electron-ion pairs generated just balances the loss to the electrodes. Note the symmetric (but out of phase) sheath near the wall in the spatial plot of potential. The characteristics of the discharge can be changed by varying the neutral gas pressure. Also note that there is no physical difference between the driven electrode and the reference electrode at zero potential in PDP1.

4.2.4 VC.INP

This input file simulates a virtual cathode oscillations due to injection of a cold electron beam between shorted electrodes. The oscillations are periodic after an initial transient, and can be seen in any of the time history plots.

4.3 Format

The input file is currently a fixed format ASCII file. The input files may be edited using any ASCII text editor or word processor which does not insert formatting characters into the file. The text lines can contain any descriptive comments, etc., and the text may continue for as many lines as desired. The lines containing numbers must also remain on a single line and each number can only be separated by white space, including any amount of spaces and tabs. The numbers may be in floating point or exponential format (decimal point and sign are optional).

The preferred method of trying new input parameters is to COPY the original data set to a new file name and edit the working copy. This will leaves original data set intact for future use and reference.

A SYSTEM REQUIREMENTS

The X-Windows version requires X11 libraries (X11R4 or any superset of X11 such as Motif), a C compiler, and an X display or X-terminal. In addition xgrafix requires Tcl/Tk also be installed. The exact version of the (public domain) required graphics libraries can be ascertained by reading the README file presented in the distribution.

To edit the input files, the user must have an ASCII text editor or word processor that can edit and save plain ASCII files without formatting or control characters.

B APPENDIX TECHNICAL SUPPORT

The Plasma Theory amd Simulation Group maintains mailing lists for discussing the codes developed by it. In order to subscribe to a mailing list
send email to
majordomo@langmuir.eecs.berkeley.edu
with the phrase
subscribe ⟨**list**⟩
in the body of the message. This will cause you to be automatically added to the mailing list.
⟨**list**⟩ is the name of the mailing list you want to subscribe to and can be one of the following

- **oopic-users**: mailing list for oopic and xoopic users.
- **pdp2-users**: mailing list for xpdp2 users.
- **pdx1-users**: mailing list for xpdp1 and xpdc1 users.
- **es1-users**: mailing list for xes1 users.

The group also maintains a web site containing home pages of group members and the anonymous FTP directory for accessing codes and can be accessed at *http://ptsg.eecs.berkeley.edu* in the *pub/codes* directory

For technical support, modifications and custom models, contact **CPS** (Computer Physics Software) via email at *cps@langmuir.eecs.berkeley.edu*

References

[1] C. K. Birdsall and A. B. Langdon, *Plasma Physics Via Computer Simulation*, (McGraw-Hill 1985, Adam-Hilger 1991 which has ES1 disk).

[2] R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles*, Adam Hilger (1988).

[3] W. S. Lawson, *PDW1 User's Manual*, Electronics Research Laboratory Report M84/37 (1984).

[4] J. P. Verboncoeur and V. Vahedi, *WinGraphics: An Optimized Windowing Environment for Plasma Simulation*, Proceedings of 13^{th} Numerical Simulation Conference, Sante Fe, NM (1989).

[5] John P. Verboncoeur, M. Virginia Alves, V. Vahedi, and C. K. Birdsall, *J. Comp. Phys.* **104**, 321 (1993).