

# **IMINTDYN81 (Version 8.2.1)**

## **Ion Matter Interaction - Dynamic**

A versatile and fast Monte Carlo Binary Collision Approximation simulation software

### **Command Parameters**

Prof. Dr. Hans Hofsäss

2<sup>nd</sup> Institute of Physics

University Göttingen

Germany

# 1 Program Structure

Main program IMINTDYN:

1. Default\_init
2. Init-parallel\_environment
3. Read namelist imint.inp
4. Setup simulation mode and free flight path option
5. Check\_series\_mode
6. Zero\_init
7. Default\_init
8. Read namelist imint.inp
9. Check symbols
10. Set series\_mode
11. Read\_all\_tables
12. Set scattering\_species
13. Set nra\_species
14. Read\_r33 cross section files
15. iba\_init
16. main broadcast
17. set\_matrix\_ranges
18. initialize random, number generator
19. set max. atomic fraction
20. initialize vacancy mode
21. initialize noblegas treatment
22. angle\_init
23. energy\_init
24. chemical\_init
25. outgas\_init
26. diff\_ko\_init
27. set OR\_loss electronic energy loss
28. target\_init
29. setup\_isotopes
30. main broadcast
31. create\_layer\_profile
32. sbv\_init – surface binding energy model
33. create free\_flight path tables
34. create energy\_loss tables
35. create sublimation enthalpy table
36. allocate memory
37. setup particle info details
38. setup trajectory details
39. create angle vs impact parameter tables
- 40. call histories, projectile and recoil**
41. gsum\_statr, gsum\_fleucne, gsum\_mat
42. out\_particles, out\_trajectories
43. outp
44. deinit\_parallel\_environment

**Subroutines history, projectile and recoil:**

1. Seed random number generator
2. init\_dynamic\_load\_balancer
3. sbv\_init, sbe\_init
4. initialize parallel processing
5. loop for each processor core an number of histories
  - a. call projectile
  - b. call recoil
  - c. call oligomer\_sputtering
6. end loop
7. update target structure and composition and update output data files after every history

## 2 Global parameters

**Imintdyn82.1 changes**

numslices > target\_slices

ioutput\_part > ioutput\_particles

subroutine setup\_histories updated

../../bin/linux.PRO/hostfile on new computer updated

**Imintdyn82 changes**

numslicesm > slicesmax

numslices > numslices

ncpm > speciesmax

ncp > numspecies

e0 > energy0

e0\_inc > energy0\_inc

dns0 > atomic\_density

name changed: param.F90 -> parameters.F90

imint.sh script file updated

**imintdyn 81 changes**

script batch file updated

output of atomic concentrations into logbook

postprocessing updated and minor bugs eliminated

layer-input modified, so that a too high number of layers is not possible

identifier for coincidence events in post processing updated

option compound\_dns\_mode removed

Update of oligomer\_sputtering. Some bugs were corrected

Position of Subroutine OR\_loss was changed to occur after the setup of isotopes, because the latter one changes ncp and thus the size of various arrays. After that a call to mainbroadcast occurs.

Subroutine setup\_isotopes was moved before iba\_init

Postprocessing programs were updated and small bugs corrected.

Main script file imint.sh was updated for the test mode

**Imintyn80 changes**

New subroutines set\_vac\_stopping, get elements contributing to vacancy stopping

New subroutine get\_AtNr\_tmax, get atomic numbers of elements for vacancy stopping

Vacancy\_stopping\_fraction introduced, default = 1.0

Vacancy\_species introduced. Up to ncp elements used for electronic stopping for vacancies. The contribution is based on the atomic fractions.

Vacancy stopping introduced for stopping mode 6 and 7 (SRIM and ZBL)

Target\_init moved into tableread.F90 subroutine before loading of stopping power data

MPI broadcast extended for string vectors

The weighting in subroutine profile was corrected

parts of imintdyn are shifted in to subroutines in init\_all

timestamp routines are updated to show total time for a series simulation

postprocessing with faulty line detection implemented

oligomer\_sputtering is now a subroutine and part. of sub.F90; call from recoil

trajec\_out(i) und partic\_out(i) are defined as character vectors

Andock model removed from recoil

MPI\_filewrite(fh,buffer,nchars) und  
MPI\_filewrite1(fh,buffer(:),nlines,nchar) implemented

definition of allocatable array is done more flexible

mpi\_wait added in routine mpi\_filewrite and mpi\_filewrite1 to ensure thread safety

MPI\_Bcast extended for string vectors

trajectory\_all.dat removed

variable "trajek" replaced by "traject"

for MPI writing we need the linefeed character: linefeed = char(10)

Imoments set als .false. default

no more sequential mode

only intel and oneapi compilers are supported

no debug option in makefile supported

only machtype=x86\_64 is supported

debugging is now possible using the commands

debugging\_pid=pid1,pid2 and

debugging\_history= h1,h2

sbv and sbvtt not anymore a matrix but a 1D array

chemical sputtering and diffusion package taken from SDTrimSP 6.05

## Imintdyn72 changes

NRA postprocessing corrected

Several routines were put into lba\_init.F90

Buffered output to out\_particles implemented.

Bufsize=nnn, default 100 (lines of output)

Particles\_per core=nn, default 10

Out\_particles optimized for new oneAPI Compiler from INTEL for AMD Ryzen Threadripper of Intel I9 processors

Program can now run with all cores on these processors, time for data output is now negligible

-----  
Dimer\_sputtering, Dimer compound, table-dimers.txt

series\_mode: New: "angle\_lin", "angle\_posneg", "energy\_lin", "energy\_log", "angle&energy", "history\_log", "angle\_random", "none"

energy\_mode: "input\_file", "3D\_Maxwell", "1D\_Maxwell", "fixed", "none", Works only for series\_mode="none"

angle\_mode: "input\_file", "cosine", "cosine\_2pi", "random", "posneg", "fixed", "none"; Works only for series\_mode="none"

Case\_e0: only used as internal parameter

Case\_alpha: only used as internal parameter

“two\_comp” replaced by “binary\_compound”

“i\_two\_comp=” replaced by “compound\_dns\_mode”

e\_bulkb(1:num\_species) is now obsolete

“Kijflag” replaced by “curvature\_flag”

“proj\_per\_history” replaced by “proj\_per\_history”

### Imintdyn71 changes

File\_r33\_nra as namelist input

rra\_target\_species(ncpm) array

nra\_product\_species(ncpm) array

several different nra reactions and different products can be specified

nra(5, ncpm) as array

heavy(ncpm) array

Update of random number generator

More efficient handling of nraflag and enforced scattering, weight factor includes concentration. Output only if weightfactor > 0.

Read\_r33 separated from read\_all\_tables

Matrix output: small changes and in commands and some minor bugs fixed.

### Imintdyn70 changes:

- New scattering mode “nra”
- Read nra IBANDL R33 files
- New postprocessing option for ebs\_scattering: “esa”
- New postprocessing option for erda\_scattering: “nra”
- New subroutine “interpolate\_fractions” in program projectile and recoil
- fix\_layer\_density

option Intel Fortran random numbers, use IFPORT, result = RAND([ iflag]), result= RANDOM(iflag), RANDOM\_INIT,RANDOM\_NUMBER, RANDOM\_SEED, SRAND

### Imintdyn65: changes:

- Testmode, get command argument
- Particle data output also for series
- Post processing also for series output
- Cleaning up the series mode simulations:
- Vacancy\_mode = 1,2,3,...
- Vacancy\_sequence
- Call pvfktn(l,cx) calculates vacancy formation probability
- Fluence steps out
- No restart option

- shth = sheath\_potential
- Ltime\_red not used anymore
- use\_isotopes
- atomic\_fraction
- atomic\_fraction
- max\_atomic\_fraction
- max\_atomic\_fraction\_flag
- target\_thickness
- cutoff\_from\_sbe
- post processing program update
- free path display update
- layer definition file .def
- layer output file .dat
- lorentz\_flag
- matric output extended! `echo 0 | sudo tee /proc/sys/kernel/yama/ptrace_scope`
- add\_bulk\_layers
- main\_broadcast as new subroutine
- iba\_init as new subroutine
- use\_isotope option was corrected. Now setup\_isotopes subroutine appears after target\_init subroutine. After that all parameters are broadcast again. This is necessary because setup\_isotopes reorders all arrays of size ncpm because the value ncp changes
- allocate\_memory as a new subroutine
- improved output for logbook.dat
- set\_series\_mode as new subroutine
- set\_scattering\_species as new subroutine
- set\_matrix\_ranges as new subroutine
- create\_layer\_profile as new subroutine
- dsf changed to surface\_thickness
- postprocessing output files now all have 6 head lines and 3 subheader lines
- Estimate of dynamic thickness has been upgraded (or\_loss now before target\_init)
- postprocessing output has been modified for 5 decimal digit output

## 2.1 How to compile and run the program

The IMINTDYN package is stored in folder imintdyn82 with subfolders bin, case, post, src, tables, template inp.files, and template layer.def.

The subfolder bin contained a folder linux.PRO and a folder testnml.

The subfolder src contains all F90 source files as well as the files Makefile and Makefile\_testnml.mak. It also contains a folder SRIM-sources with source programs to create the SRIM stopping data base for IMINTDYN.

To compile the program

- (i) open a terminal screen and change to imintdyn82/bin/linux.PRO
- (ii) execute the Makefile using the command “sh mk”. Makefile automatically identifies the INTEL compiler and the compile and generates the program inimdyn.exe.

To compile the postprocessing programs

- (i) open a terminal screen and change to imintdyn82/post
- (ii) execute the Makefile using the command “sh mkpost”. Makefile will then create all postprocessing programs which have been modified since the last compilation process.

The folder tables contains all required tables for imintdyn, as well as a folder SRIM\_tables and SRIM\_raw\_data. In folder SRIM\_raw\_data is the application “read\_srim\_stopping” which reads all stopping .txt files and created the SRIM2013-nn stopping data files. These should then be copied into the SRIM\_tables folder.

The folder “case” contains the simulation data, each simulation has its own subfolder. In this subfolder there is the simulation script file “imint-filename1.inp” and optional script files “filename2.inp” and “layerdefinition.def”. To run a simulation one needs to start the script file “imint.sh” using the command “bash imint.sh”. After the simulation is finished, the results are stored in folders with name “filename1-filename2”.

In the file “imint.sh” one can define several scriptfiles and support files. Before a simulation is started, one should set the script file to testmode. In testmode only the setup part of imintdyn is executed and one gets detailed error response in case the input script files \*.inp contain some errors. Only if the testmode finishes without errors, one should deactivate the testmode and start the simulation.

The simulation uses a temporary directory in the folder case.

## 2.2 Parameters defined in module param.F90:

ncpm = 15            maximum number of elements (projectiles and target elements)    [up to 999]



numslicesm = 2000            maximum number of depth intervals  
 thick\_min = 1.0            minimum target thickness [A] (sufficient for a 1D layer of 2D material)  
 default\_cutoff\_fraction = 0.333    default fraction to calculate the cutoff energy from the sublimation enthalpy

## 2.3 Parameters for random numbers

If ART\_RAND is selected, then            iwpi = 4 long integer format for random seed number  
 Else                                        iwpi = 8 double long integer format for random seed number

## 2.4 Other parameters defined in module param.F90 (and used in dlb.F90):

pemax = 128            maximum number of Processors  
 nalpha = 180            max number of data in angular input file (a precision of 0.5 deg for angles between 0 and 90 deg seems sufficient)  
 nemax = 200            max number of data in energy input file  
 max\_generation = 1000            maximum number of recoil generations  
 non\_RBS = 1024            number of elements in a non\_RBS cross section file  
 yzlost = 115.            Factor to find lost particles (y,z) >= yzlost\*ttarget

## 2.5 Parameters defined in module default\_init, used in module task\_description.F90 and dlb.F90 (only MODE: PAR)

dntmax=100000            memory size of global task queue  
 ntqmax=100000            memory size of local task queue

Allocated memory:  $(ntqmax+dntqmax) \times (13 \times (4 \text{ Bytes}) + 18 \times (8 \text{ Bytes})) / 1024. / 1024.$  ( in Mbytes) = 37.38 MBytes

MEMORY for trajectory: assumption

Integer = 4 byte

Real = 8 byte

one trajectory ->  $(ncpm+13) \times (4) + 18 \times (8) = 256$  Bytes for  $ncpm=15$

## 2.6 Parameter ptime is a variable calculated in subroutine "inelast.F90"

It describes the flight time between two collisions. It is calculated from the pathlength divided by velocity and is given in units of femtoseconds.

### 3 Input file 'imint.inp' structure for an imintdyn.exe simulation

IMINTDYN uses one or two input files to define a simulation. The first and primary file has the name "imint-nameX.inp" with the precursor "imint". It contains all necessary information of a simulation, for example imint-ArSi1keV.inp to simulate sputter erosion of Si with 1 keV Ar ions. Upon execution of the simulation, this file is copied to a temporary directory under the name "imint.inp".

The second file has the name "subnameX.inp" (without a precursor "imint") and contains specific details of the simulation which can be changed easily without modifying the primary file. The content of "subnameX.inp" overwrites the previous data from file "imint-nameX.inp". For example, n001.inp to define a negative curvature of the sample surface. Upon execution of the simulation this file is copied to a temporary directory under name "imintb.inp"

Within the execution script file "imint.sh" one can define several primary and secondary input files, for each primary file a secondary file is required.

The form of imint-nameX.inp file is that for an input file (header + namelist):

**The 1st line which does not start with a '!' character is the Headline (Text), format Ascii A80.**

The following lines are defined by the namelist IMINT\_INP (see subroutine WORK.F90) . The variables and values follow the initial line &IMINT\_INP and close with the line " / ". The script file may contain FORTRAN Comments starting with " ! " as 1st character . The head line appears before &IMINT\_INP it is a line which does not start with a "!" character.

Namelist is a list of variables or arrays, **separated by commas**

The Namelist in program imintdyn.exe is IMINT\_INP . The structure is start line "&IMINT\_INP" , end line "/" .

```
&IMINT_INP
```

```
<variables and values>
```

```
/
```

The sequence of the input variables and values in the input file is arbitrary (namelist).

### 3.1 Testmode

To test if the input script file contains nor errors, one can run IMINTDYN in testmode. In this mode I)NIMTDYN setup is processed, but no simulation is started. Furthermore, only one processor is used. The testmode can be activated with a command line extension “ -testmode”. For example a simulation using MPI routine and a number \$np of processors can be tested using the following line in the bash script file imint.sh.

```
mpirun -np "$NP" ../../bin/linux.PRO/imintdyn.exe -testmode
```

The bash script file has a variable TM defined at the beginning of the file as TM="-testmode" or TM="-off". The variable is added to the program execution command line as

```
mpirun -np "$NP" ../../bin/linux.PRO/imintdyn.exe „$TM”
```

## 4 Necessary input variables (in some cases no default values exist)

| Command                    | Default value | Description   |
|----------------------------|---------------|---|
| alpha0(1:num_species)      | none          | See: series_mode="angle", see: angle_mode=  |
| alpha0_inc(1:num_species)  | none          | angle increment (deg) for series_mode="angle"   |
| energy0(1:num_species)     | none          | See: series_mode="energy", see: energy_mode=  |
| energy0_inc(1:num_species) | none          | energy increment (eV) for series_mode="energy" and for projectiles (beam_fraction > 0.)   |
| fluence                    | 0.1           | incident fluence in units $10^{16}$ atoms/cm <sup>2</sup> or 1 atom/Å <sup>2</sup> in case of a dynamic simulation<br>default is 0.1 atoms/cm <sup>2</sup> or $1 \cdot 10^{15}$ atoms/cm <sup>2</sup> |
| potential                  | 'KrC'         | selects the interaction potential   |

| Command | Default value  | Description  |
|---------|--|--|
|         | ASCII code<br>Keeps the old<br>ipot variable<br>internally | "KrC ", best suited for sputter erosion and low energy collisions (default)<br>"Moliere", "ZBL", "Nakagawa" = Nakagawa-Yamamura<br>"Si-Si" , "power" |

| Command          | Default value | Description   |
|------------------|---------------|---|
| projectiles      | None          | If defined with value > 0, then histories and proj_per_history are calculated based on num_cores aand dynamic or static simulation mode.  |
| histories        | 24            | number of histories (projectiles)<br>each history consists of proj_per_history individual projectiles. The total number of projectiles is then histories*proj_per_history . In dynamic mode : target update occurs after each history<br>proj_per_history(default) =34 is the default value, but may be changed by the program based on the number of processors used. This ensures that all processors have the same computational load.<br>Proj_per_history should not be too high to avoid significant incremental changes of the stoichiometry<br>proj_per_history should be <= num_cores in dynamic mode. Then it is:<br>proj_per_history = proj_per_history(default) + num_cores - mod(proj_per_history(default) , num_cores) |
| target_slices =  | 100           | number of slices of the target (discretization). The maximum value maxslices = 2000 (default) set in param.F90 .If target_slices < 0, then the positive absolute value of target_slices is used.<br><br>Target_slices = 0 is not allowed.<br><br>If a file e.g. "layer.def" defines the number of target slices, then target_slices is set from this layer.def file.<br><br>Target_slices should not be too large, or a layer thickness should not be smaller than an atomic spacing or 1 Å. In this case the concentration of atoms in a layer may get zero in case of a dynamic simulation.   |
| add_bulk_layers= | .true.        | If the number target_slices of target layers decreases below the initial number specified in the input script file, then a layer with the composition and thickness of the last layer is  |

|                                |      |   |
|--------------------------------|------|---|
|                                |      | <p>added. In this way the target gets an infinitely thick bulk substrate and target_slices does not decrease below its initial value.</p> <p>This option prevents a target from being completely eroded.</p>  |
| beam_fraction(1:num_species)   | none | <p>projectile atomic fractions (in the incident beam) of num_species species. Num_species is determined automatically based on the symbols specified.</p> <p>For a projectile: beam_fraction &gt; 0. ,</p> <p>Note: it is not necessary to normalize beam_fraction values to <math>\text{sum}(\text{beam\_fraction}(1:\text{num\_species})) = 1</math> ! This is done by the program.</p> <p>For target atoms: beam_fraction = 0.</p> |
| atomic_fraction(1:num_species) | none | <p>initial target atomic fractions of num_species species in case of a homogenous initial composition.</p> <p>Note: <math>\text{sum}(\text{qu}(1:\text{num\_species})) = \text{not necessary to be normalized.}</math></p> <p>Example: Ar on Ta<sub>3</sub>O<sub>7</sub> : <math>\text{qu}(1:\text{num\_species}) = 0., 3., 7.</math></p>   |
| symbol(1:num_species)          | none | <p>numspecies chemical symbols of elements according to table1. Symbol(1:numspecies) is a character*5 string.</p> <p>(special symbols: H2, D, T, He3, He, C_a, C_g, C_d, ta-C, C_f, Sp2, Sp3, Sp3H)</p> <p>(other special symbols: Vac (vacancy) , Ar-l, Ar-s, Ar-g (liquid,solid,gas), same for other noble gases)</p>   |

## 5 Optional variables:

These values have default values defined by subroutines default\_init and zero\_init (see default\_init.txt). If values different from the default values are needed, then these values have to be given explicitly in the input file. Other choices are provided in the description column

| 5.1.1 Definition of input data file names and directories and output file options |                |   |
|---|----------------|---|
| Command   | Default values | Description   |
| dir_angleinp =  | './'           | directory of inputfile 'angle.inp' as default name (or another specified name), default is the current directory (see also: dir_layerinp, dir_tableinp, dir_energyinp)  |
| dir_energyinp=  | './'           | directory of inputfile 'energy.inp' as default name (or another specified name) , default is the current directory (see also: dir_layerinp, dir_tableinp, dir_angleinp)   |
| dir_layerinp=   | './'           | directory of inputfile 'layer.inp' as default name (or another specified name); the default directory is './' . (see also: dir_tableinp, dir_angleinp, dir_energyinp).<br>example: home/imintdyn/case/specific_case/  |
| dir_tableinp=   | './tables'     | directory of inputfile for tables; (see also: dir_layerinp, dir_angleinp, dir_energyinp)  |
| file_layerinp=  | 'layer.def'    | filename of layered target structure defintions:<br>The output files have names "layeroutput-000.dat" and "layeroutinp-000.def"<br>The number is filled in case of a series simulation with the series index.<br>The file layeroutinp-000.def has an additional empty 1 <sup>st</sup> column for a projectile element an can be directly used as input file for e.q. a SIMS analysis simulation using Cs O or other ions as target species. |
| file_angleinp=  | 'angle.def'    | filename of angular input data defintions. maximum 180 angular values (set in subroutine "work" the file has 2 header lines and the data pairs of angle[deg], probability (not normalized)  |
| file_energyinp=   | 'energy.def'   | filename of energy input data definitions<br>maximum 200 values (set in subroutine "work", only energies 0.1 eV < E < 10 <sup>9</sup> eV are accepted, 2 header lines, pairs of energy[eV], probability (not normalized)  |



| Command                     | Default values | Description  |
|-----------------------------|----------------|--|
| fluence_steps_out =         | 100            | Determines the output for Number of fluence steps.<br>The old parameter idout is used internally and is calculated as<br>idout=histories/fluence_steps_out<br>A large value of fluence_steps_out causes output files of very large size  |
| ioutput_trajectories (1:6)= | 10             | number of traced trajectories for: stopped, backscattered and transmitted projectiles, stopped, backspattered, transmission sputtered recoils (see also: ltraj_p, ltraj_r)<br>The value represents the number per processor core should not exceed 5-10 because trajectory calculation require very much allocated memory  |
| ioutput_particles (1:6)=    | 10000          | number of traced particles for: stopped, backscattered and transmitted projectiles, stopped, backspattered, transmission sputtered recoils (see also: lparticle_p, lparticle_r). The file size for 100,000 particles may reach 30 Mbytes.  |
| layer_input =               | .false.        | initial target composition flag<br><br>if layer_input=.true., then read the layer structure from layer input file<br>see also dir_layerinput and file_layerinput<br>true : initial depth dependent composition taken from layer input file ( e.g. layer.def)<br>false : (default) initial composition is homogeneous, target_slices layers with constant layer interval thickness<br>layer input file starts with 2 text lines, followed by several lines: j, xxx, atomic_fractionx(1:num_species)<br><br>j > 0 : number of layers ; xxx layer thickness in Å<br>j = 0 : duplicate previous structure until target_slices defined in imint.inp |

| Command   | Default values | Description   |
|---|----------------|---|
|   |                | j < 0 ; end reading input file. In the case the other input values in the line are ignored<br>calculates also new values for target_slices and new ttarget !  |
| lenergy_distr=                                    | .false.        | output of the energy distribution in the target<br>(stop energy ,electronic loss and elastic nuclear loss)<br>E_distr_all.dat E_distr_stop.dat E_distr_inel.dat E_distr_nucl.dat  |
| dist_nx=60  | 60             | x-size of the matrix of energy distribution in target   |
| dist_ny=60  | 60             | y-size of the matrix of energy distribution in target   |
| dist_nz=60  | 60             | z-size of the matrix of energy distribution in target   |
| dist_delta=2.0                                    | 2.0            | distance between the matrix points of energy distribution in target   |
| elosstables=                                      | .false.        | [default: .false.], if true, then electronic energy loss data tables are created.<br>The default data file name in elosstable(n).dat with n=1,...num_species<br>The table consist of 301 energy values in the range 10 eV to 2 GeV                        |
| impactflag  | .false.        | If .true., then an output file of scattering angle as function of impact parameter for the incident energy energy0 and p <sub>impact</sub> in the range 10 <sup>-6</sup> to 10 Å is created. This table allows to adjust the “enforce_scattering” option. |
| shtable=  | .false.        | If .true., then create a table of sbe[eV] versus concentration (for binary compounds)   |
| lorentz_flag                                      | .false.        | If .true. then a relativistic correction is included for energy straggling and relativistic mass of projectiles   |
| <b>5.1.2 Matrix output distribution commands:</b> |                |   |
| lmatrices=  | .false.        | .true. : output of matrices for emitted particles, only for static simulations<br>.false. : no matrix output  |
| matrix_transmitted =                              | .false.        | create matrices for transmitted particles   |
| matrix_backscattered =                            | .true.         | create matrices for back-emitted particles  |
| matrix_cumulative =                               | .true.         | cumulative or differential data collection, only for dynamic simulations  |
| matrix_log_energy =                               | .false.        | Output type of energy in matrix energy vs depth for emitted particles   |

| Command                       | Default values | Description  |
|-------------------------------|----------------|--|
|                               |                | .false. : linear energy intervals<br>.true. : logarithmic energy intervals   |
| matrix_cos_angle =            | .false.        | Output of emission angle in a matrix for emitted particles<br>.false. : angle in degree intervals<br>.true. : cosine intervals   |
| matrix_e_min_r=               | 0              | minimum of lin. energy distribution of recoils in matrices [eV]  |
| matrix_e_max_r=               | none           | max(energy0) maximum of lin. energy distribution of recoils in matrices [eV]   |
| matrix_e_min_p=               | 0              | minimum of lin. energy distribution of projectiles in matrices [eV]  |
| matrix_e_max_p=               | none           | max(energy0) maximum of lin. energy distribution of projectiles in matrices [eV]   |
| matrix_e_steps_lin_r=         | 10             | Number of recoil energy steps between min and max value-linear scale   |
| matrix_e_steps_log_r=         | -1             | Number of recoil energy steps between min and max value-log scale, 9 steps per decade  |
| matrix_e_steps_lin_p=         | 10             | Number of projectile energy steps between min and max value-linear scale   |
| matrix_e_steps_log_p=         | -1             | Number of projectile energy steps between min and max value-log scale, 9 steps per decade  |
| File names for matrix output: |                | meagb_p_file ='matrix_energy_angle_back_proj.dat'<br>meagt_p_file ='matrix_energy_angle_trans_proj.dat'<br>meagb_r_file ='matrix_energy_angle_back_sputt.dat'<br>meagt_r_file ='matrix_energy_angle_trans_sputt.dat'<br>mepb_p_file ='matrix_energy_path_back_proj.dat'<br>mept_p_file ='matrix_energy_path_trans_proj.dat'<br>mpe_ex_p_file='matrix_start energy_vs max_penetration_back.dat'<br>polar and azimuthal angular_matrix_file_br ='meagb_r_000000.dat'<br>polar and azimuthal angular_matrix_file_bp ='meagb_p_000000.dat'<br>polar and azimuthal angular_matrix_file_tr ='meagt_r_000000.dat'<br>polar and azimuthal angular_matrix_file_tp ='meagt_p_000000.dat' |

| Command   | Default values | Description  |
|---|----------------|--|
| 5.1.3 Output of detailed projectile and recoil information: |                |  |
| Imoments=   | .false.        | output of moments for energy distributions (linear and logarithmic) of projectiles and recoils and for range distributions (linear) of projectiles.<br>.true. : moments are written in to a output file<br>.false. : moments are not written |
| lparticle_p=  | .false.        | .true. : output of projectile information<br>.false. : no output of projectile information<br>(see also: ioutput_particles)  |
| lparticle_r=  | .false.        | .true. : output of recoil information<br>.false. : no output of recoil information<br>(see also: ioutput_particles)  |
| out_angle_pt =  | 0.,90.         | Defines minimum and maximum emission angle of projectiles for output data<br>Forward transmitted: $0 \leq \text{angle} \leq 90$ deg  |
| out_angle_pb =  | 90.,180.       | Back scattered : $90 \leq \text{angle} \leq 180$ deg   |
| out_angle_rt =  | 0.,90.         | Defines minimum and maximum emission angle of recoils for output data<br>Forward transmitted: $0 \leq \text{angle} \leq 90$ deg  |
| out_angle_rb =  | 90.,180.       | Back sputtered : $90 \leq \text{angle} \leq 180$ deg   |
| out_energy_p =  | 0.,            | Defines minimum energy of emitted projectiles for output data  |
| out_energy_r =  | 0.,            | Defines minimum energy of emitted recoils for output data  |
| ltableread =  | .true.         | Default strongly recommended is .true.<br>.true. : read from table_elements, table_compounds, table_isotopes, table_Hstopping table_Hestopping1<br>.false. : no table read, a_num_z, a_mass, dns0, e_surfb, e_displ have to be given         |

| Command  | Default values | Description   |
|--|----------------|---|
|  |                | <p>tables:</p> <p>table_elements .txt: chemical symbol (symbol), nuclear charge (a_num_z), atomic mass (a_mass), mass density, atomic density (dns0), surface binding energy (e_surfb), displacement energy (e_displ), cutoff energy (e_cutoff)</p> <p>table_compounds.txt : symbol of binary compound target and physical values</p> <p>see also: nm, tabe.compound, binary_compound, dimer_sputtering, rho_mean</p> <p>table_isotopes.txt : chemical symbol, nuclear charge, isotope mass, atomic weight (in amu), natural abundance</p> <p>table_Hstopping.txt : inelastic stopping coefficients for hydrogen: symbol, nuclear charge, inelastic stopping coefficients a1 to a12 (ch_h), ck</p> <p>table_Hestopping1.txt , table Hestopping2.txt: inelastic stopping coefficients for helium: symbol, nuclear charge, inelastic stopping coefficients a1 to a9 (ch_he)</p> |
| ltraj_p=   | .false.        | .true. : output of projectile trajectories<br>.false. : no output of projectile trajectories  |
| ltraj_r=   | .false.        | .true. : output of recoil trajectories<br>.false. : no output of recoil trajectories  |
| <b>5.1.4 Definition of ion species and target elements</b> |                |   |
| symbol(1:num_species)                                      | none           | <p>num_species chemical symbols of elements according to <b>“table1-elements.txt”</b></p> <p>(special symbols: H2, D, T, He3, He, C_a, C_g, C_d, ta-C, C_f, Sp2, Sp3, Sp3H, a-C1D). Other special symbols: Vac, Vac1D (vacancy) , Ar-l, Ar-s, Ar-g (liquid,solid,gas), same for other noble gases).</p>   |
| beam_fraction(1:num_species)                               | 0.             | projectile atomic fractions (in the incident beam) of num_species species.  |

| Command                            | Default values | Description   |
|------------------------------------|----------------|---|
|                                    |                | For a projectile: beam_fraction > 0. , Note: sum(beam_fraction (1:num_species)) = 1 is normalized by the program. For target atoms: beam_fraction = 0.  |
| atomic_fraction(1:num_species)     | none           | initial target atomic fractions of num_species species in case of a homogenous initial composition (iq0 = 0)<br>Note: sum(atomic_fraction (1:num_species)) = 1 is normalized by the program   |
| max_atomic_fraction(1:num_species) | none           | maximum atomic fractions in the target for num_species species, if dynamic simulation. Example: for SiO <sub>2</sub> one should set max_atomic_fraction = 1.0 ; 0.667 so that excess oxygen diffuses out  |
| max_atomic_fraction_flag           | .false.        | .true. , then calculate max_atomic_fraction = max_atomic_fraction (0 deg) * cos(alpha0) only if case_apha=5 and only for species with beam_fraction > 0. This is an option, that adjusts the retained noble gas concentration for varying incidence angles using an empirical cosine law.<br>.false., default |
| charge(1:num_species)=             | 0.             | elementary charge of ion species if case_e0 = 2 or 3 and sheath_potential > 0 (plasma). Otherwise it is not used.<br>>= 1. for beam_fraction > 0 (projectiles) charge state should at least be 1+ .<br>= 0. for beam_fraction = 0 (target atoms)  |
| alpha0(1:num_species)              | none           | angle of incidence (in deg) of num_species species used depending on angle_mode and series_mode<br>alpha0 is measured in neg y-direction with respect to the surface normal (-x) if alpha0 < 0, then azimuth angle phi is set to 180 deg  |
| alpha0_inc(1:num_species)          | none           | angle increment (deg) for linear angle series   |
| energy0(1:num_species)             | none           | Incident ion energy   |
| eenergy0_inc(1:num_species)        | none           | energy increment (eV) for projectiles (beam_fraction > 0.) in case of linear energy series  |

| Command   | Default values | Description  |
|---|----------------|--|
| series_mode                                       | "none"         | "angle_lin", "angle_posneg", "energy_lin", "energy_log", "angle&energy", "history_log", "angle_random", "none"   |
| angle_mode  | "none"         | "input_file", "cosine", "cosine_2pi", "random", "posneg", "fixed", "none"<br>Works only for series_mode="none"   |
| energy_mode                                       | "none"         | "input_file", "3D_Maxwell", "1D_Maxwell", "fixed", "none"<br>Works only for series_mode="none"   |
| series_steps=                                     | 1              | number of calculations if a series of calculations is carried out (for a given selected series mode)<br>Histories = log scale with steps 5,16,50,160,500,1600,5000,....<br>energy log scale: 3 values per decade: 1,2,5,10,20,50, etc  |
| case_e0<br><b>only used as internal parameter</b> | 0              | flag for the choice of the incident energy:<br>= 0 : fixed incident energies (eV) of projectiles (beam_fraction > 0.)<br>= 1 : input of a given energy distribution from file energy.inp<br>= 2 : temperature (eV) of plasma of a 3D Maxwellian velocity distribution of projectiles, particle energy e_part comes from subroutine mxvelo<br>= 3 : temperature (eV) of plasma of a 1D Maxwellian energy distribution of projectiles, particle energy e_part comes from subroutine energ<br>= 4 : does not yet exist<br>= 5 : series of calculations with different projectile energies<br>linear series with fixed incremental energy<br>energy = energy0 + (i-1) * eenergy0_inc ; i = 1, series_steps )<br>if also case_alpha = 5, then the program ignores case_e0 !!<br>output: output???.dat with ??? = i<br>default setting: lmatrices = .false. ; ltraj_p = .false. ; ltraj_r = .false. ;<br>lparticle_r = .false.<br>lparticle_p = .false. ;<br>case_alpha = 0<br>note: all *.dat files from last calculation |

| Command   | Default values | Description   |
|---|----------------|---|
|   |                | <p>= 6 : series of calculations with different angles of incidence and different energies (also sets case_alpha0 =6) INPUT OF ANGULAR AND ENERGY DISTRIBUTION FROM FILE<br/>                     see subroutine ene_ang_init in init_all</p> <p>= 7 : series of calculations with different projectile energies<br/>                     logarithmic series given by 3 values per decade: 1,2,5,10,20,50, etc<br/> <math>kk = 1 + \text{mod}(\text{iseries\_steps} - 1, 3)</math><br/> <math>\text{energy0}(:) = \text{energy0}(1) * 10^{((\text{iseries\_steps} - 1) / 3) * \text{int}(10^{(kk * 0.35)})}</math><br/>                     example: energy0=20, series_steps=6 : 20,40,100,200,400,1000 eV</p> <p>output: output???.dat with ??? = i<br/>                     default setting: lmatrices = .false. ; lttraj_p = .false. ; lttraj_r = .false. ;<br/>                     lparticle_r = .false. , lparticle_p = .false. ; case_alpha = 0<br/>                     (note: all *.dat files from last calculation)</p> |
| <p>case_alpha =<br/> <b>only used as internal parameter</b></p> | <p>0</p>       | <p>flag for the choice of the angle of incidence :</p> <p>= 0 : angle of incidence (degree) counted from the surface normal<br/>                     alpha0 = 0... 90 (starting above surface)<br/>                     alpha0 = 90...180 (starting in solid)<br/>                     (azimuthal angle phi = 0)</p> <p>= 1 : input of a given incident angular distribution from file angle.inp</p> <p>= 2 : cosine distribution of angles of incidence (only from above surface)</p> <p>= 3 : cosine distribution of angles of incidence<br/>                     alpha=0...Pi/2,max: at 0 phi= 0...2Pi</p> <p>= 4 : random distribution of angles of incidence (only above surface) (alpha and phi random)</p> <p>= 5 : series of calculations with different angles of incidence</p>  |



| Command                      | Default values                   | Description  |
|------------------------------|----------------------------------|--|
|                              |                                  | <p>( alpha = alpha0 + (i-1) * alpha_inc; i = 1, series_steps )<br/> output : output.*dat<br/> default set :lmatrices = .false.<br/> ltraj_p = .false.<br/> ltraj_r = .false.<br/> lparticle_r = .false.<br/> lparticle_p = .false.<br/> case_e0 = 0</p> <p>(note: all *.dat output files from last calculation)<br/> if also case_e0 = 5, then the program ignores case_e0 !!</p> <p>= 6 : series of calculations with different angles of incidence and different energies (also sets case_e0 = 6) INPUT OF ANGULAR AND ENERGY DISTRIBUTION FROM FILE see subroutine ene_ang_init in init_all<br/> = 10 : fixed polar angle of incidence (same as case_alpha = 0) but alternating opposite azimuthal angle phi=0 deg or phi=180 deg<br/> = 15 : series polar angles of incidence (same as case_alpha = 5) but alternating opposite azimuthal angle phi=0 deg or phi=180 deg</p> |
| phi0 = (1:num_species)=      | 0                                | azimuth angle of ion incidence in deg; 0 deg = along y-axis; 90 deg = along z-axis   |
| atomic_mass(1:num_species)=  | > = 0<br>< 0: use table_elements | mass (in amu) of num_species elements; default from <b>"table_elements.txt"</b><br>atomic_mass = 0 is a vacancy, then also atomic_number = 0.  |
| atomic_number(1:num_species) | See table_elements               | atomic number of num_species elements; default from <b>"table_elements.txt"</b>  |

| Command                      | Default values | Description  |
|------------------------------|----------------|--|
| use_isotopes(1:num_species)= | 0              | <p>flag for isotope mass</p> <p>= 0 : natural isotope mixture (mass from <b>"table_elements.txt"</b>)</p> <p>= 1 : isotope masses and natural abundances from <b>"table_isotopes"</b><br/>(valid for projectiles as well as for target species)<br/>the masses are set by a_mass(1:num_species) or taken from <b>"table_elements.txt"</b></p> <p>use_isotopes(:) = 1 increases the number of species to account for the number of isotopes. Each isotope is included as a new target species. Make sure that ncpm (default 15) is not exceeded !</p> |

|                      |         |  |
|----------------------|---------|--|
| compound_flag =      | .false. | .false.: not a binary molecular target<br>If .true. then simulation is for a binary compound defined by "binary_compound" and rho_mean, the number nm of atoms in a molecule is taken from " <b>table.compound</b> "<br>There may be more than one projectile, but only two target elements which appear last, i.e. num_species-1 and num_species !!!!!<br>Thjer compound sublimation enthalpy is defined in the table, but it is not used by default( the table contains a correction factor set to zero.<br>If a compound formation enthalpy shall be use, the correction factor should be set > 0. Then, for a binary compound a new model is used to calculate the sublimation enthalpy automatically for bvariable stoichiometry of the compound! |
| binary_compound =    | none    | symbol of binary compound target according to " <b>table_compounds</b> "<br>(e.g. binary_compound ="Ta2O5"). binary_compound is a character*10 string<br>Note: only selected compounds in " <b>table_compounds</b> "<br>see also: nm, " <b>table_compounds</b> ", binary_compound, rho_mean<br>There may be more than one projectile, but only two target elements which appear last, i.e. num_species - 1 and numspecies !!!!!  |
| rho_mean =           | none    | mass density of a binary compound target;<br>default from " <b>table_compounds</b> " [g/cm**3)<br>see also: nm, " <b>table_compounds</b> ", binary_compound, rho_mean<br>There may be more than one projectile, but only two target elements which appear last, i.e. num_species-1 and num_species !!!!!   |
| delta_hf_corr =      | 0.      | Correction factor to scale the compound formation enthalpy $0 < \text{delta\_Hf\_corr} < 1$<br>This overrides the factor from the compound table   |
| target_thickness =   | none    | total target thickness in Angstrom (Å)<br>The minimum default value is 3 Å, set in "param.F90"   |
| target_temperature = | 300.    | target temperature, only of interest at high temperatures, it reduces the surface binding energy   |

|   |                                    |  |
|---|------------------------------------|--|
|   |                                    | according to a Maxwellian energy distribution  |
| x0(1:num_species)=                                | 0.                                 | starting position (impact) of projectile (see also curvature Kij) in A<br><= 0. : outside the surface at x=xc<br>> 0 : inside the solid<br>In a surface curvature is defined using the curvature parameters Kij, then the value of x0 should be set to typically + 100 A.  |
| deltahd(1:num_species)                            | See<br>table_elements              | heat of dissociation (eV) of elements dns0(1:num_species) of a molecular target  |
| deltahf   | See<br>table_elements              | heat of formation (eV) of a molecular target   |
| atomic density (1:num_species) =                  | > 0<br>< 0 : use<br>table_elements | atomic density in units (atoms/Å <sup>3</sup> or 10 <sup>24</sup> atoms/cm <sup>3</sup> ) of num_species elements;<br>Example: 0.05 for Si<br>if atomic_density <= 0 or not specified, then the table_elements values is taken   |
| fix_layer_density                                 | .false.                            | If .true., then keep the initial density of each target layer fixed. If layers are dynamically added or deleted, then the new layer density of the affected layers are interpolated. This option avoids drastic density changes if a very low density gas atom is implanted into a dense solid (example: O into SiO <sub>2</sub> ) |
| <b>5.1.5 Definition of specific energy values</b> |                                    |  |
| e_cutoff(1:num_species)=                          | 1.                                 | cutoff energy (eV) of target species; defaults from table1 (1 eV for noble gases; 1 eV for H, D, T; Normally a value of 1 eV is sufficiently low.<br>If values of zero are specified, then a value of 1.0 eV is chosen as default.   |
| cutoff_from_sbe                                   | .true.                             | .true., then take cutoff_energy = cutoff_fraction * sbe, if simulation is in SBE_mode. If .false., the use values from table or imint.inp  |
| cutoff_fraction                                   | 0.333                              | Fraction of the sublimation enthalpy to calculate the cutoff energy in sublimation mode<br>0 < cutoff_fraction <= 1 ; in case of another value the default value 0.333 is chosen in sublimation mode, the cutoff is taken as fraction of the sublimation   |

|                                       |            |   |
|---------------------------------------|------------|---|
|                                       |            | enthalpy. For some gases, the sublimation enthalpy is zero. In this case, the cutoff is set to the corresponding <code>e_cutoff</code> value to ensure that a particle comes to rest.   |
| <code>e_displ(1:num_species)</code>   | 0.         | displacement energy (eV); default from table1. The displacement energy can be set to zero, but the simulation speeds up if a larger value is chosen. Note, that a displacement energy is only defined in a crystalline matrix and is used to estimate the amount of damage created (similar to SRIM software)   |
| <code>e_thres_r(1:num_species)</code> | 0.         | threshold energy (eV); The threshold energy is used to define a minimum energy transfer to the recoil for bookkeeping of projectile and recoil data for CERDA   |
| <code>e_surfb(1:num_species)</code>   | See table1 | <p>surface binding energy (eV) (sublimation enthalpy) <b>for standard conditions</b>, RT and standard pressure; default from table1</p> <p>Imintdyn.exe uses the tabulated sublimation enthalpies from table1 (new: table1-elements.txt). Or the values specified in the .inp file. This was the previous option <code>isbv=1</code> in SDTrimSP and TRIDYN.</p> <p>In case of a binary compound with <code>compound_flag=.true.</code>, the surface binding energy is modified based on the compound formation enthalpy tabulated in table.compound (new: table-compound.txt). In dynamic simulation and for sputtered recoils, the sublimation enthalpy is calculated based on the local concentration values in up to 10 layers to the front or back surface. For projectiles a sublimation enthalpy is set to zero, because they are never bound to target atoms.</p> <p>If a value &gt; 0 is specified, then the correction enthalpy for 6000 K is scaled based on the tabulated value in table1. This scaling ensures, that the correction enthalpy remains smaller than the sublimation enthalpy at standard RT.</p> |
| <code>temp_cascade=</code>            | 298.15.    | Assumed temperature of the collision cascade in K. This value is used to reduce surface binding energy (the sublimation enthalpy) and the binary compound formation enthalpy according to a linear approximation of the Shomate equation between 298K and 6000 K.   |

|                |         |  |
|----------------|---------|--|
|                |         | If temp_cascade <= 0., then the temperature is approximated dependent of the projectile energy by $T = R_t + (6000 - R_t) \cdot \exp(-\text{energy0(keV)/2.})$   |
| sbe_mode =     | .true.  | If .true., then the surface binding energies are used in the usual way: A particle is only sputtered, if its kinetic energy normal to the surface is larger than the sublimation energy.<br>If .false., then the sublimation energy is subtracted from the kinetic energy of recoils immediately after the recoil is created. To be sputtered, an energy > 0 is sufficient. A particle whose energy was reduced initially gets the sublimation enthalpy added before it comes to rest. |
| temp_cascade = | 298.15  | Set the fixed temperature of the collision cascade for the temperature dependent sublimation enthalpy and surface binding energy calculation. Value is in Kelvin   |
| refraction =   | .true.  | Switches the refraction effect on an off   |
| temp_auto =    | .false. | .true.: Automatic calculation of the cascade temperature after 100 histories , or update every idout ihistories for a dynamic simulation   |
| tilt_flag =    | .false. | If .false., then Sigmunds Gaussian ellipsoid is calculated in the x,y,z coordinate system.<br>If .true., then Sigmunds Gaussian ellipsoid is calculated in the titled coordinate system with x' along the beam direction.  |

|                                |        |  |
|--------------------------------|--------|--|
| stopping_mode(1:num_species) = | 3      | <p>inelastic loss model - how to calculate the electronic energy loss</p> <p>= 1 : Lindhard-Scharff; necessary condition: <math>E &lt; 25 * Z^{4/3} * M</math> (in keV), where E, Z, M are the energy, the atomic number and the atomic mass of the moving particle, respectively</p> <p>= 2 : Oen-Robinson; necessary condition: <math>E &lt; 25 * Z^{4/3} * M</math> (in keV)</p> <p>= 3 : Equipartition of 1 and 2 [DEFAULT]</p> <p>= 4 : high energy hydrogen (H,D,T) (<math>25 \text{ keV} &lt; E &lt; 475 \text{ MeV}</math>), values from table3</p> <p>= 5 : high energy helium (He3,He4) (<math>E &gt; 100 \text{ keV}</math>), values from table4</p> <p>= 6 : Ziegler-Biersack stopping based on the tables scoef95A and scoef95b. This stopping model option is very similar to the data used in SRIM-2013</p> <p>= 7 : SRIM-2013 electronic stopping power data. This option may require to extract stopping data from the SRIM Software. Stopping powers are interpolated from a logarithmic table with values in the range 0 eV to 2 GeV. Raw stopping data files are stored in subdirectory tables in a folder "SRIM_raw_data". The program "read_srim_stopping.exe" reads the raw data files and creates the SRIM2013-nn.dat files, where nn is the atomic number of the projectiles. Each file contains up to 92 stopping tables for each target element. The stopping unit is eV per <math>10^{15} \text{ at/cm}^2</math> and is independent of the actual target density.</p> <p>Imintdyn selects the stopping data for the num_species specified target atoms from the SRIM2013-nn.dat files. The SRIM2013-nn.dat files contain stopping powers for target atoms with Z=1 to 92. If a vacancy with Z=0 is specified as a target atom using symbol "Vac" the program set the electronic loss for such a "target atom" to zero.</p> |
| sheath_potential =             | 0.0125 | <p>= 0.0125 eV : weak sheath potential sheath_potential= <math>k_B T_e / 2e</math> for <math>T=300\text{K}</math></p> <p>&gt; 0 : sheath potential (eV), usually = <math>3 *  \text{energy0} </math>, only if case_e0=2,3 (Maxwellian distribution, plasma)</p>  |

|  |                             |  |
|--|-----------------------------|--|
| surf_inel_loss =   | 0                           | <p>= 0 : no inelastic energy loss outside the target surface (<math>x \leq 0</math> , <math>x &gt; ttarget</math>)<br/>                 = 1 : inelastic energy loss outside the target surface (<math>-suu &gt; x &gt; 0</math>. And <math>ttargets &lt; x &lt; ttarget + suu</math>)<br/>                 If (kijflag) is set, then the lower value <math>x=0</math> is shifted to <math>x0(jin)</math></p>   |
| <p><b>5.1.6</b> Definition of simulation options</p>             |                             |  |
| dynamic_simulation=<br>static_simulation=<br>cascade_simulation= | .false.<br>.true.<br>.true. | <p><b>mode of simulation, parameter idrel=</b><br/>                 full dynamic calculation (TRIDYN), internally: dynamic_flag=.true., cascade flag =.true.</p> <p>cascade_flag=.false : only projectiles (no recoils) are followed. internally: static_flag=.true.,</p> <p>Typical parameter for a dynamic simulation: dynamic_flag=.true., cascade flag =.true.</p> <p>Typical parameter for a static simulation is static flag =.true., cascade flag =.true.</p>   |
| proj_per_history=  | 12,24                       | <p>number of projectiles between two target updates (histories) for dynamic simulation. In a static simulation, the total number of simulated ions is histories * proj_per_history.</p> <p>each history consists of proj_per_history individual projectiles. The total number of projectiles is then histories* proj_per_history . In dynamic mode: target update occurs after each history</p> <p>proj_per_history (default) =12 is the default value, but may be changed by the program based on the number of processors used. This ensures that all processors have the same computational load.</p> <p>proj_per_history should not be too high (<math>\leq 12</math>) to avoid significant incremental changes of the stoichiometry</p> |



|                               |                 | <p>proj_per_history should be <math>\leq</math> num_cores. Then it is:<br/> <math display="block">\text{proj\_per\_history} = \text{proj\_per\_history}(\text{default}) + \text{num\_cores} - \text{mod}(\text{proj\_per\_history}(\text{default}), \text{num\_cores})</math></p> <table border="0"> <thead> <tr> <th># of processors</th> <th>num_cores</th> <th>proj_per_history</th> <th>proj_per_history(default)=12</th> </tr> </thead> <tbody> <tr> <td>1,2,3,4</td> <td>12</td> <td></td> <td></td> </tr> <tr> <td>5</td> <td>10</td> <td></td> <td></td> </tr> <tr> <td>6</td> <td>12</td> <td></td> <td></td> </tr> <tr> <td>7</td> <td>14</td> <td></td> <td></td> </tr> <tr> <td>8</td> <td>16</td> <td></td> <td></td> </tr> <tr> <td>9</td> <td>18 == &gt; 9</td> <td></td> <td></td> </tr> <tr> <td>10</td> <td>20 == &gt; 10</td> <td></td> <td></td> </tr> <tr> <td>11</td> <td>22 == &gt; 11</td> <td></td> <td></td> </tr> <tr> <td>12</td> <td>12</td> <td></td> <td></td> </tr> <tr> <td>13</td> <td>13 == &gt; warning</td> <td></td> <td></td> </tr> <tr> <td>.....</td> <td></td> <td></td> <td></td> </tr> </tbody> </table> | # of processors              | num_cores | proj_per_history | proj_per_history(default)=12 | 1,2,3,4 | 12 |  |  | 5 | 10 |  |  | 6 | 12 |  |  | 7 | 14 |  |  | 8 | 16 |  |  | 9 | 18 == > 9 |  |  | 10 | 20 == > 10 |  |  | 11 | 22 == > 11 |  |  | 12 | 12 |  |  | 13 | 13 == > warning |  |  | ..... |  |  |  |
|-------------------------------|-----------------|--|------------------------------|-----------|------------------|------------------------------|---------|----|--|--|---|----|--|--|---|----|--|--|---|----|--|--|---|----|--|--|---|-----------|--|--|----|------------|--|--|----|------------|--|--|----|----|--|--|----|-----------------|--|--|-------|--|--|--|
| # of processors               | num_cores       | proj_per_history   | proj_per_history(default)=12 |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 1,2,3,4                       | 12              |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 5                             | 10              |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 6                             | 12              |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 7                             | 14              |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 8                             | 16              |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 9                             | 18 == > 9       |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 10                            | 20 == > 10      |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 11                            | 22 == > 11      |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 12                            | 12              |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| 13                            | 13 == > warning |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| .....                         |                 |  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| particles_per_core            | 10              | Number of particles distributed to each core. This value is generally used for static simulations. In dynamic simulations the value depends on the number of cores_num_cores and the desired proj_per_history.   |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| bufsize                       | 100             | 100 line of buffered output to out_particles. 100 is sufficient for storing projectile and sputter data. For stopping of recoils the number may be even larger.  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| surface_thickness =           | 3.              | Defines and average thickness of a surface layer (Å). This layer is used to calculate the surface composition as function of fluence   |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| flux=                         | 1.0             | flux of incident atoms ( $\text{atoms}/\text{Å}^2/\text{s} = 10^{20} \text{ atoms}/\text{m}^2/\text{s}$ )  |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| lchem_ch=                     | .false.         | calculation with chemical erosion for H on C(SP <sup>2</sup> ,SP <sup>3</sup> ,SP <sup>3</sup> H), D on C(SP <sup>2</sup> ,SP <sup>3</sup> ,SP <sup>3</sup> H) using a mix of methods by Hopf, Roth and Mech   |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |
| atomic_fraction_interpolate = | .false.         | If .true. then linear interpolation of atomic fractions between the depth intervals. Only required if larger depth intervals and significant concentration gradients   |                              |           |                  |                              |         |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |    |  |  |   |           |  |  |    |            |  |  |    |            |  |  |    |    |  |  |    |                 |  |  |       |  |  |  |

|                                       |        |  |
|---------------------------------------|--------|--|
| integration_mode =                    | 1      | integration method (default=1)<br>= 0 : MAGIC, only valid for KrC, ZBL, Moliere<br>= 1 : Gauss-Mehler quadrature, ipivot >= 8 recommended<br>= 2 : Gauss-Legendre quadrature, ipivot <= 16<br>= 3 : Gauss-Legendre quadrature 8 (double precision), ipivot <= 16   |
| ipivot=                               | 8      | number of pivots in the Gauss-Mehler and Gauss-Legendre integration, the minimum number is 4 (larger numbers increase the computing time)  |
| case_layer_thick=                     | 0      | mixing scheme of target<br>= 0 : mixing layer with neighbor layer, if thickness is 150% or 50%<br>= 1 : mixing to constant layer thickness   |
| ca_scre(1:num_species,1:num_species)= | 1.0    | correction factor for the screening length in the interaction potential (not applicable for KrC and ZBL potentials)  |
| ck_elec(1:num_species,1:num_species)= | 1.0    | correction factor for the inelastic energy loss; correction factors for hydrogen (below 25 keV) are given in table_Hstopping.txt   |
| diff_koeff1(1:num_species)=           | 0.     | transport coefficient if loutgas=.true. [ $\text{\AA}^3/\text{ion}$ ]  |
| diff_koeff2(1:num_species)=           | 0.     | diffusions coefficient if loutgas=.true. [ $\text{\AA}^4/\text{ion}$ ]   |
| imcp=                                 | 0      | flag indicating whether (flib)-moments of depth distributions are calculated<br>= 0 : no moment calculation<br>= 1 : calculate moments of depth distributions for all projectiles with (qubeam>0.)   |
| subthr_rec_bound                      | .true. | flag for subthreshold recoil atoms<br>.false. : subthreshold recoil atoms free<br>>= 0 : subthreshold recoils bound  |
| weak_coll_p=                          | 2      | number of ring cylinders for weak simultaneous collisions <b>for projectiles</b> ; for high energies (MeV H or He) weak_coll_p can be reduced to 1 or 0 to reduce computing time.<br>The inner ring with $p=r1*pmax$ ( $r1$ random number) calculates the primary collision. The first ring has $p=pmax*\sqrt{r1+1}$ . The 2 <sup>nd</sup> ring has $p=pmax*\sqrt{r1+2}$ , etc. For each ring the number $r1$ and also the collision partner is individually determined. For 2 rings, $p$ may reach a value of $pamx*\sqrt{3}$ , so that the tails of the screened potential becomes relevant. |

|                |          |  |
|----------------|----------|--|
| weak_coll_r=   | 2        | <p>number of ring cylinders for weak simultaneous collisions <b>for recoils</b>; for high energies (MeV H or He) weak_coll_p can be reduced to 1 or 0 to reduce computing time.</p> <p>The inner ring with <math>p=r1*pmax</math> (<math>r1</math> random number) calculates the primary collision. The first ring has <math>p=pmax*\sqrt{r1+1}</math>. The 2<sup>nd</sup> ring has <math>p=pmax*\sqrt{r1+2}</math>, etc. For each ring the number <math>r1</math> and also the collision partner is individually determined. For 2 rings, <math>p</math> may reach a value of <math>pmax*\sqrt{3}</math>, so that the tails of the screened potential becomes relevant.</p> |
| Kij =          | 0.,0.,0. | <p>defines a surface curvature by <math>h(y,z) = h0 - 1/2 K11 y^2 - K12 yz - 1/2 K22 z^2</math><br/>Kij is given in units 1/Angstrom.</p> <p>A positive curvature describes a valley-like surface, note that the surface normal is along -x.</p> <p>If Kij not equal 0, then one has to set proper <math>x0</math> values, i.e. the curved surface at any point should be positioned below <math>x0 = 0</math> (values of <math>x0 &gt; 0</math>). Kij should be small enough, at most <math> K  &lt; 0.002</math></p>   |
| curvature_flag | .false.  | <p>If any of the Kij values is non-zero, the curvature is set .true. The flag can be set to .true. in the input file to enforce a shift of the surface to <math>x0 &gt; 0</math>, even if no curvature is defined. This is only used for test purpose.</p> <p>.Not.curvature_flag also influences the inelastic loss calculation (parameter surf_inel_loss) by shifting the target limits from <math>x=0</math> to <math>x=x0</math>.</p>  |
| debugging=     | 0        | <p>a parameter to help debugging program errors with different levels of debugging output</p> <p>0: debugging is off</p> <p>1: output from subroutines imintdyn.F90 and histories.F90 (logical flag "debugging1")</p> <p>2: only some output from subroutine histories.F90 (logical flag "debugging2")</p> <p>3: only output from subroutines imintdyn.F90 and projectile.F90 (logical flag "debugging3")</p> <p>4: only output from subroutines imintdyn.F90 and recoil.F90 (logical flag "debugging4")</p>   |

|                      |             |   |
|----------------------|-------------|---|
|                      |             | <p>5: only debugging of subroutines set_sublenergy in sub.F90<br/> 6: only debugging of subroutines set_temp_cascade in sub.F90<br/> 7: only debugging of subroutine free_path in sub.F90<br/> 8: only debugging for gsum_fluence and gsum_fluence2</p> <p>Implemented by H. Hofsaess 06-5-2021</p> |
| debugging_pid=       | 0,num_cores | Only then debugging output is created (imintdyn80)  |
| debugging_hist=      | 1,histories | Only then debugging output is created (imintdyn80)  |
| craterflag=          | .true.      | <p>Activates calculation of crater function moments and writing of crater function output files.</p> <p>See files: craterfile, cratermoments, balancefile, impl_crater_moments, zero_moments</p>  |
| monolayer=           | .true.      | <p>true: (default) uses the shortest mean free path length <math>l_m=1/(dns0)^{1/3}</math><br/> .false: uses <math>-l_m*\ln(1 - \text{random number})</math> . Can be used for a larger target thickness<br/> This option can be circumvented by free_path_length !!</p>                            |
| free_flight_path     | .false.     | If true, then calculate a free path length based on the scattering cross section and the fractional electronic energy loss. Basis is a minimum deflection angle (default 1 deg) between collisions and a minimum relative electronic energy loss (default 1%) between collisions                    |
| ffp_accuracy         | 0.01        | Set the accuracy of the free path length approximation. A value of 0.01 sets a max deflection angle of 1 deg and a max. rel energy loss of 1%. The value should not exceed 0.05(or 5deg and 5%)   |
| huge_angle =         | 15.         | (default value) threshold for counting events with large scattering angle. The number of events with scattering angle above "huge_angle" are listed in particle*.dat files  |
| small_angle =        | 2.          | (default value) threshold for counting events with small scattering angle. The number of events with scattering angle larger than "small_angle" and below "huge_angle" are listed in particle*.dat files as medium angles   |
| enforce_scattering = | .false.     | If .true., then enforced scattering is activated. The program calculates the average number of collisions for transmitted and backscattered projectiles. The  |

|                     |         |   |
|---------------------|---------|---|
|                     |         | <p>value num_scattering defines the number of large angle scattering events each projectile shall undergo. Bases on the average number of collisions and num_scattering a random number is selected, defining the probability to reduce the max. impact parameter. The reduction factor is set by pmax_coeff, which may be as low as 5E-6.</p> <p><b>A scattering_mode must be selected</b></p>   |
| single_scattering = | .true.  | <p>true: a random number r is chosen to find a target position <math>x_s = r * t_{target} / ffp</math> where enforced scattering takes place. Only one scattering event is considered</p> <p>false: plural scattering mode is active. The probability for enforced scattering <math>p = num\_scattering * ffp / t_{target}</math>. A random number r is chosen and if <math>r &lt; p</math>, then enforced scattering is activated. There may be more than one scattering event, so that plural scattering takes place. With proceeding path step n, the probability for a projectile to survive is <math>(1-p)^n</math>.</p>                                 |
| enforce_rec_scatt = | .false. | <p>If true, then enforced scattering also applies to recoils. This option is only used in C-ERDA simulations like HH- scattering or Li-Li scattering and only when single_scattering=.false.</p>  |
| scattering_mode =   | 'none'  | <p>'none', or 'cerda', 'ercs', 'ebs', 'erda', 'nra'</p> <p>cerda: coincidence ERDA: coincident scattering of identical projectile and target species</p> <p>ercs: elastic-recoil-coincidence-spectroscopy: coincident scattering/recoil of light projectiles and heavier targets species</p> <p>ebs: elastic backscattering: Rutherford or non-Rutherford backscattering</p> <p>erda: detection of recoil emitted under small backscattering angles</p> <p>nra: detection of nra reaction products</p> <p><b>only if one of the scattering modes is specified, then enforce_scattering remains active. Otherwise enforce_scattering is set to .false.</b></p> |
| cerda_angle         | 0.      | the minimum and maximum scattering angle can be defined in the range $0 <$  |
| cerda_angle2 =      | 90.     | $\theta < 89$ deg. Used for identical projectile and scatter atom mass !! C-ERDA like   |

|                                |              |  |
|--------------------------------|--------------|--|
|                                |              | H-H, He-He or Li-Li scattering. From that the impact parameters for enforced scattering are calculated   |
| ercs_angle<br>ercs_angle2 =    | 0.<br>90.    | the minimum and maximum recoil angle can be defined in the range $0 < \theta < 89$ deg. Used for light projectile and heavier scatter atom mass !! ERCS like He-B, He-C, He-N, He-O scattering. From that the impact parameters for enforced scattering the recoils are calculated   |
| ebs_angle<br>ebs_angle2 =      | 0.<br>180.   | the minimum and maximum scattering angle can be defined in the range $0 < \theta < 179$ deg. Used for light projectiles and heavy scatter masses mainly for backscattering. From that the impact parameters for enforced scattering the projectiles are calculated   |
| erda_angle<br>eerda_angle2 =   | 90.<br>180.  | the minimum and maximum recoil angle can be defined in the range $90 < \theta < 179$ deg. Used for heavy projectiles and light scatter masses mainly for ERDA simulations. From that the impact parameters for enforced scattering their recoils and projectiles are calculated  |
| erda_azimuth<br>azimuth_range  | 180°<br>180° | Restriction of azimuth angle for ERDA scattering default: $180^\circ \pm 180^\circ$  |
| no_e_dependence =              | .false.      | if true, then $p_{\text{impact}}$ is adjusted with a factor $\text{energy}_0/E$ . This eliminates an energy dependence in cross section for CERDA and EBS. If false, then max. $p_{\text{impact}}$ is chosen assuming up to 70% residual projectile energy after passing the target.   |
| num_scattering =               | 1.0          | Number of large angle scattering events per path of a projectile through the target  |
| scatt_species(1:num_species) = | '...'        | num_species ASCII characters identical to Symbol, specifying for which collision partner the scattering is enforced. Example: "H" selects collisions with H-Atoms e.g. in a compound TiH <sub>2</sub> .<br>If the first string is set to "all", then large angle scattering applies to a target elements<br>Isotopes must not be specified, just the name of the elemental species |
| non_rutherford                 | .false.      | If .true., then non Rutherford cross sections will be used   |

|   |         |  |
|---|---------|--|
| file_r33  | none    | An IBANDL non-RBS cross section files up to 3 files per target species can be used, total 3*ncpm files   |
| file_r33_nra  | none    | An IBANDL NRA cross section files up to 3 files per target species can be used, total 3*ncpm files   |
| nra_target =<br>nra_product =                           |         | Up to ncpm target and corresponding product elements for NRA simulations.<br>Example: mra_target = "Li7","Li6"<br>nra_product="He4","He3"  |
| l_threshold=  | .false. | (default: .false.) if true, then all events with energy above E_thres are ignored, i.e. only recoils with energies above the threshold are considered. Also projectile data output can be selected according to e_thresh<br>If .false, then E_displ is only used to reset recoils to its initial position if Ekin < E_displ. |
| lpart_r_ed=   | .false. | if true, then do not write recoil particle info data to file partic*.dat if initial recoil energy < E_displ.<br>This avoids storage of too many low energy collision events  |
| lpart_pb_ed=  | .false. | if true, then do not write backscattered projectile (pb) info data to file partic*.dat if energy transfer to recoil is DE < e_thresh_r. This avoids storage of too many low energy collision events  |
| lpart_pt_ed=  | .false. | if true, then do not write transmitted projectile (pt) info data to file partic*.dat if energy transfer to recoil is DE < _e_thresh_r . This avoids storage of too many low energy collision events  |
| loutgas=  | .false. | calculation with outgassing transport and diffusion (see also: diff_koeff1, diff_koeff2)   |
| <b>5.1.7 Treatment of vacancies and noble gas atoms</b> |         |  |
| i_vac_coord=  | 4       | coordination number of atom stop positions. If one of the neighboring atoms is a vacancy, then a stopped particle annihilates the vacancy. Annihilation probability saturates at min(1.,atomic_fractionx(vacancy)*i_vac_ccord )  |
| q_vacancy (1:num_species) =                             | 1.0     | characteristic probability for vacancy formation. 0 < q_vacancy < 1.   |

|  |      |   |
|--|------|---|
|  |      | Formation of vacancies given by the probability and the local vacancy concentration $c_v$ . Default mode <code>vacancy_mode = 1</code> :  |
| <code>vacancy_stopping_fraction</code> | 1.0  | Fraction of the elemental stopping power contributing to electronic stopping of vacancies   |
| <code>vacancy_species=</code>          | none | Target elements used for electronic stopping calculation for vacancies. Up to <code>ncp</code> elements can be specified and are selected based on the global atomic fraction and the value <code>q_vacancy</code> . If nothing is specified, then only the major target element is used for vacancy stopping calculation.  |
| <code>vacancy_mode</code>              | 1    | How to calculate $p_{vac}$ with $0 \leq q_{vac} \leq 1; 0 \leq c_{vac} \leq 1$ :<br><br>0: $p_{vac} = 0$ no vacancy generation at all, also no vacancies are annihilated<br><br>1: default quadratic - $p_{vac} = q_{vac} \cdot (1 - c_{vac})^2$<br>2: quadratic + $p_{vac} = q_{vac} + (1 - q_{vac}) \cdot (1 - (1 - c_{vac})^2)$<br>3: linear - $p_{vac} = q_{vac} \cdot (1 - c_{vac})$<br>4: linear + $p_{vac} = q_{vac} + c_{vac} (1 - q_{vac})$<br>5: cubic - $p_{vac} = q_{vac} \cdot (1 - c_{vac})^3$<br>6: cubic + $p_{vac} = q_{vac} + (1 - q_{vac}) \cdot (1 - (1 - c_{vac})^3)$<br>7: sine - $p_{vac} = q_{vac} \cdot \frac{1}{2} (1 + \sin(\pi(c_{vac} + 0.5)))$<br>8: sine + $p_{vac} = q_{vac} + (1 - q_{vac}) \cdot \frac{1}{2} (1 - \sin(\pi(c_{vac} + 0.5)))$<br>9: constant $p_{vac} = q_{vac}$<br><br>Other values: mode 0 |
| <code>vacancy_sequence</code>          | 0    | Anti coincidence_mode: -1 , then not any two vacancies are chosen in a sequence, except there is a layer filled with vacancies to larger than 80%. This mode is used to satisfy the stacking sequence of graphene layers  |



|                |        |  |
|----------------|--------|--|
|                |        | Sequence_mode: = n integer number. Then up to n vacancies are selected in a sequence, except there is a layer which does not contain vacancies. This mode is used to satisfy the stacking of Metal-Dichalcogenide triple-layers. |
| lgaserf=       | .true. | activates a surface error function profile for qmax of vacancies ensures that qmax for vacancies goes to zero near the surface. Default value is .true.  |
| gaserf_sigma = | 2.     | width of error function in unit of layers for noble gas concentration profile (default 2 layers)   |
| gaserf_pos =   | 4.     | layer position of error function center for noble gas concentration profile (default: layer 4)   |
| lvacerf=       | .true. | activates a surface error function profile vor qmax of vacancies and ensures that qmax for vacancies goes to zero near the surface. Default value is .true.  |
| vacerf_sigma = | 2.     | width of error function in unit of layers for vacancy concentration profile (default 2 layers)   |
| vacerf_pos =   | 4.     | layer position of error function center for vacancy concentration profile (default: layer 4)   |

## 6 Default filenames and directories

These names are set in subroutine default\_init.F90. filenames have a maximum default length of 50 characters and directory names a default maximum length of 100 characters

```
! SET MAIN PROGRAM NAME and PROGRAM VERSION
!
  programname='imintdyn82'
  programversion='Ver 8.2000'
!
! -----
! SET input and output filenames and directories
!
  imint_out_target='imint_output_target-000.dat'
  imint_out_moments='imint_output_moments-000.dat'
  imint_out_moments1='imint_output_momentsA-000.dat'
  imint_out_sputter='imint_output_sputter-000.dat'
  trajec_all='trajec_all.dat'
  trajec_out(1)='trajec_stop.p.dat'
  trajec_out(2)='trajec_back_p.dat'
  trajec_out(3)='trajec_tran_p.dat'
  trajec_out(4)='trajec_stop_r.dat'
  trajec_out(5)='trajec_back_r.dat'
  trajec_out(6)='trajec_tran_r.dat'

  partic_out(1)='partic_stop_p.dat'
  partic_out(2)='partic_back_p.dat'
```

```
  partic_out(3)='partic_tran_p.dat'
  partic_out(4)='partic_stop_r.dat'
  partic_out(5)='partic_back_r.dat'
  partic_out(6)='partic_tran_r.dat'
```

```
  series_filename='serie.dat'
  energy_analysis='energy_analysis.dat'
  timestamp='time.dat'
  time_run='time_run.dat'
  rf_filename1='Rstart_1.inp'
  rf_filename2='Rstart_2.inp'
```

```
  output_filename='output-000.dat'
```

```
  craterfile='craterfunction.dat'
  cratermoments='cratermoments.dat'
  balancefile='balancefile.dat'
  impl_crater_moments='crater-moments-implantation.dat'
```

```
  zero_moments='zero-moments-file.dat'
```

vacancyfile='vacancyfile.dat'

file\_elosstable='energy\_loss\_table0.dat' ! 0 = 1,num\_species  
file\_pimpact\_angle='pimpact\_vs\_angle.dat'

layerinput='layer.inp' ! not used at the moment  
layeroutput='layer\_profile.dat'  
layeroutinp='layer\_profile.def  
dir\_layerinp = './'  
file\_layerinp = 'layer.def'  
dir\_energyinp = './'  
file\_energyinp = 'energy.def'  
file\_energyout\_test = 'energyout\_test.dat'  
dir\_angleinp = './'  
file\_angleinp = 'angle.def'  
file\_angleout\_test = 'angleout\_test.dat'  
dir\_tableinp = '../tables/'

! SRIM data are in default subdirectory 'SRIM-tables'

e\_act\_HCW = 'e\_act\_HCW.inp'  
a0\_HCW = 'a0\_HCW.inp'

scoef95a = 'SCOEf.95A'  
scoef95b = 'SCOEf.95B'

SRIMfile='SRIM2013-nn.dat'  
dir\_SRIMinp='SRIM\_tables/'

! filenames used in subroutine "output"

meagb\_p\_file = 'matrix\_energy\_angle\_back\_proj.dat'  
meagt\_p\_file = 'matrix\_energy\_angle\_trans\_proj.dat'  
meagb\_s\_file = 'matrix\_energy\_angle\_back\_sputt.dat'  
meagt\_s\_file = 'matrix\_energy\_angle\_trans\_sputt.dat'  
mepb\_p\_file = 'matrix\_energy\_path\_back\_proj.dat'  
mept\_p\_file = 'matrix\_energy\_path\_back\_proj.dat'  
morigin\_ex\_bs = 'matrix\_energy\_depthorig\_back\_sputt.dat'  
morigin\_ex\_ts = 'matrix\_energy\_depthorig\_trans\_sputt.dat'  
E\_distr\_stop = 'energy\_distribution\_stop'  
E\_distr\_inel = 'energy\_distribution\_electronic'  
E\_distr\_nucl = 'energy\_distribution\_nuclear'  
E\_distr\_all = 'energy\_distribution\_all'  
depth\_proj = 'depth\_distribution\_projectiles.dat'  
depth\_recoil = 'depth\_distribution\_recoils.dat'

!

! matrix filename in subroutine "histories"

!

angular\_matrix\_file = 'meagb\_s\_000000.dat'

!

! =====

**io numbers used for different input and output files:**

io = 6 !6 Standard output

io7 = 7 !6 time output

io10 = 10 !  
 io21 = 21 ! --- angle\_input,energy\_input  
 io22 = 22 ! --- table\_read  
 io22 = 22 ! --- read table.compound  
 io23 = 23 ! --- ioSCOEf\_95A Ziegler Biersack stopping power table  
 io24 = 24 ! --- ioSCOEf\_95B Ziegler Biersack stopping power data  
 io29 = 29 ! ---general output  
 io30 = 30 ! ---imint.inp or imintb.inp  
 io32 = 32 ! ---crater function file output  
 io31 = 31 ! ---general output, series\_filename, imint\_out\_target  
 io33 = 33 ! ---balance file output  
 io34 = 34 ! ---crater function moments output  
 io35 = 35 ! ---implantation crater function output  
  
 io35 = 35 ! --- imint\_out\_moments1,  
 ! --- E\_distr\_stop.dat, \_distr\_inel.dat,E\_distr\_nucl.dat,E\_distr\_all.dat  
  
 io36 = 36 ! -- zero-moments-file output  
 io37 = 37 ! -- dimer-sputtering file output  
  
 io41 = 41 ! ---energy analysis, depth proj, depth recoil  
 io50 = 50 ! --- output file for energy loss tables  
 io55 = 55 ! ---layer input and output data  
  
 io56 = 56 ! ---layer output data for layer.inp  
 io57 = 57 ! ---matrix output  
 io67 = 67 ! ---matrix output  
 io67 = 76 ! ---matrix output

! these 7 io values should be in a row:  
 io80 = 80 ! --- only as dummy for io80 + j

io81 = 81 ! --- trajec\_stop\_p  
 io82 = 82 ! --- trajec\_back\_p  
 io83 = 83 ! --- trajec\_tran\_p  
 io84 = 84 ! --- trajec\_stop\_r  
 io85 = 85 ! --- trajec\_back\_r  
 io86 = 86 ! --- trajec\_tran\_r

! these 6 io values should be in a row:

io90 = 90 ! --- only as dummy for io90 + j  
 io91 = 91 ! --- partic\_stop\_p  
 io92 = 92 ! --- partic\_back\_p  
 io93 = 93 ! --- partic\_tran\_p  
 io94 = 94 ! --- partic\_stop\_r  
 io95 = 95 ! --- partic\_back\_r  
 io96 = 96 ! --- partic\_tran\_r

io321 = 321 !  
 io331 = 331 ! vacancyfile, imin\_out\_sputter  
 io341 = 341 ! imint\_out\_moments  
 io351 = 351 ! imint\_out\_moments1  
 io100 = 100 ! --- time\_run.dat, rw restart

## 7 Sample input file (imint-template.inp)

```

1 keV Ar -> Si + Vac test simulation
&IMINT_INP

!      debugging= nn      ! set debugging flag to level nn 0,..10, to print output infos
!      debugging_history = 1,10
!      debugging_pid= 1,4
!      impactflag=.true.  ! create a file impact parameter vs. scattering angle

energy_mode = "none"      ! use a constant energy
angle_mode = "none"      ! us a constant incidence angle
series_mode = "none"
series_steps = 1
curvature_flag=.false.
!
!      set projectile and target element definitions:

target_thickness = 1000E+0      ! target depth (Angstroems)
target_slices    =   500E+0      ! number of target intervals

symbol =          "Ar",        "Si",        "Vac"      ! names of projectile and target elements
atomic_density =  0.05,        0.05,        0.05      ! set atomic densities
energy0  =          1000,        0.00,        0.00      ! ion energy (eV)
!energy0_inc =          0.,        0.,        0.,
alpha0 =          0.0,          0.0,          0.0      ! incidence angle (deg)
!alpha0_inc =          5.0 ,        0.0,          0.0
!phi0 =          0.0 ,          0.0,          0.0      ! azimuthal angle (deg)
!x0 =          0.,            0.,            0.0      ! origin of impact [A]

```

```
stopping_mode =      7,          7,          7,
atomic_fraction =   0.0,         1.0,         0.0      ! initial concentration in the target
beam_fraction =     1.000,       0.000,       0.000    ! fraction of incident beam

max_atomic_fraction =0.2,        1.0,         1.0      ! max concentration
charge =            1. ,         0.0,         0.00     ! electrical charge
e_cutoff =          1.0,         1.0,         1.0      ! cut off energies [eV]
e_surfb =           0.0,         4.7,         0.0      ! surface binding energies [eV]
e_displ =           0.0,         0.0,         0.0      ! displacement energies [eV]

q_vacancy =         0.7,         0.7,         0.7,     ! characteristic probability for vac formation

!fil_angleinp='angle_test.inp'

Kij = 0.00, 0., 0.      ! define the surface curvature [1/A]

Projectiles = 2.eE5
! histories = 10000,      ! number of histories
! Projectiles_per_history = 24
fluence_steps_out = 100
fluence = 1              ! maximum fluence in units 1E16/cm²

dynamic_simulation =.true. ! SBE model
! static_simulation=.false.
! cascade_simulation=.false. ! always true in dynamic mode
surf_inel_loss = 0
potential="KrC"          ! "KrC", "Moliere", "ZBL", "Nakagawa"-Yamamura, "Si-Si", "power"
integration_mode = 2     ! 0 = Magic, 1 = Gauss-Mehler, 2=Gauss-Legendre
                          ! 3 = Gauss-Legendre 8 (double precision)
```

```
subthr_rec_bound = .true.! sub threshold atoms are bound

weak_coll_p = 2,          ! nr of weak collisions for projectiles
weak_coll_r = 2,          ! number of weak collisions for recoils

lpart_r_ed =.true.       ! store only events for E > E_displ

!enforce_scattering=.true. ! enforce large angle scattering
!num_sacattering = 1.     ! scattering events per impact
!pmax_coeff = 1E-4        ! reduction factor for pmax value
!scatt_species ="Si"      ! restrict large angle scattering to this element

!free_path_length=.true. ! enable larger free_path length
!ffp_accuracy =0.01      ! accuracy is 1% rel energy loss or 1 deg max deflection
!
i_vac_coord = 1          ! coordination number around stopped atoms
vacany_stopping_fraction = 1.0
lvacerf = .true.         ! erf profile for qumax(num_species_vacancy)
lgaserf = .true.         ! erf profile for qumax(num_species_noblegas)
vacerf_pos = 6.          ! center layer position of error function profile
vacerf_sigma = 3.        ! width of error function profile
gaserf_pos = 6.          ! center layer position of error function profile
gaserf_sigma = 3.        ! width of error function profile

lmatrices = .false. ! matrix output

ltraj_p = .false. ! projectile trajectories
ltraj_r = .false. ! recoil trajectories
```

```
! number of traced trajectories for:
! stopped, backscattered and transmitted projectiles,stopped,backspattered,
! transmission sputtered recoils
ioutput_trajectories = 5, 5, 0, 5, 5, 0,
lparticle_r = .false.    ! create projectile data files
lparticle_p = .false.    ! create recoil data files

! number of traced particles for:
! stopped, backscattered and transmitted projectiles,stopped,backspattered,
! transmission sputtered recoils
!
ioutput_particles = 100000, 100000, 0, 100000, 100000, 0,

/
```



## 8 Example for a shell script file (imint-template.sh)

```
#!/bin/bash
DIRECTORY=$(pwd)
ERRCODE=0

TM='-off'
#TM='-testmode' # if set, then IMINTDYN is run in testmode

# list of input files imint-namex.inp
INPUT=(name1 name2) # primary input script file, default: "imint-name1.inp"
INPUTB=(subname1 subname2) # 2nd input script file, overwrites data from imint.inp, default: "subname1.inp"
# list of input files namey.def !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
SUPPORT=(support1 support2) # example: layer definition file "sample_layer.def"

POST="none" # ebs, hrrbs, hrebs, esa, tofrbs, ercs, erda, toferda, cerda, matrix or none post processing flag
BINSIZE=2048 # binning size of postprocessing spectra
SMOOTH=2 # smoothing of postprocessing spectra

WORK=temp # temporary directory

if [ $TM = "-testmode" ]; then
WORK=temptest
else
WORK=temp # temporary directory
fi # - use a special temp folder for testmode not to mess up with data in the temp folder

# set number NP of processors and simulation mode parallel PAR, or sequential SEQ
echo =====
if test -d /opt/intel/oneapi; then
echo "Intel OneAPI compiler"
COMPILER=oneapi
NP=24
```

```
PPN=48
if [ $TM = "-testmode" ]; then
NP=1
PPN=48
fi
else
echo "Older Intel Compiler"
COMPILER=intel
NP=4
PPN=8
if [ $TM = "-testmode" ]; then
NP=1
PPN=8
fi
fi
MODE=PAR # PAR,SEQ

echo =====
echo start new simulation

echo case directory is $DIRECTORY
echo input script files: ${INPUT[*]}
echo inputB script files: ${INPUTB[*]}
echo support files: ${SUPPORT[*]}.def
echo post processing programs: ${POST[*]}
echo simulation mode is "$MODE"
echo number of processors "$NP"
echo ----- setup finished -----

# process all input files
index=-1
for name in ${INPUT[*]} # -----i---
do
index=$((index+1))
nameb=${INPUTB[$index]}
cd $DIRECTORY # change to specific case directory
file=imint-${name}.inp
fileb=${nameb}.inp
```

```

echo =====
if [ -f "$file" ]; then # if file exists, then
  echo input script file $file exists, start processing !
  nameout=${name}
  echo =====

  if [ -f "$fileb" ]; then # if file exists, then
    echo input script file $fileb also exists !
    nameout=${name}-${nameb}
    echo =====
  fi
  echo output directory name is: $nameout
  cd .. # go back to case directory
  CASEDIR=$(pwd)
  # create directory temp-namout or temptest if it does not exist
  if [ $TM = "-testmode" ]; then
    mkdir -p "$WORK"
    cd $WORK
  else
    mkdir -p "$WORK"- "${nameout}"
    cd $WORK-${nameout}
  fi

  # clean up working directory if not empty
  if [ -z "$(ls -A )" ]; then # -----if--
    echo directory $WORK-${nameout} is empty
  else
    echo delete als files in directory $WORK
    rm --force *.* # force delete files
  fi # -----end if --
  DEFAULT=$(pwd) # now use this temporarydirectory as the default one
  echo working directory is: $DEFAULT
  cd $DIRECTORY # change to specific case directory

cp $file $DEFAULT/imint.inp
# copy related angle, energy and layer definition files
# file extensions are typically .inp or.def
if [ -f "$fileb" ]; then # if file exists, then

```

```

cp $fileb $DEFAULT/imintb.inp
echo 2nd input file $fileb was copied
fi

for j in ${SUPPORT[*]} # -----j-----
do
cp ${j}.def $DEFAULT/${j}.def
done # -----j-----
echo support files copied if available
cd $DEFAULT
# run parallel on NP processors , or sequential
if [ $MODE = "PAR" ] # -----if ---
then
echo ----- start mpirun in parallel mode -----
  if [ $COMPILER = "oneapi" ]; then
    mpirun -n "$NP" -ppn "$PPN" -f ../../bin/linux.PRO/hostfile ../../bin/linux.PRO/imintdyn.exe "$TM"
    ERRCODE=$?
  elif [ $COMPILER = "intel" ]; then
    mpirun -n "$NP" ../../bin/linux.PRO/imintdyn.exe "$TM"
    ERRCODE=$?
  fi
else
echo ----- start mpirun in sequential mode -----
  ../../bin/linux.SEQ/imintdyn.exe "$TM"
  ERRCODE=$?
fi # -----end if---
echo ===== imintdyn program execution finished =====
echo error code = $ERRCODE
  if [ $TM != "-testmode" ]; then
    mkdir -p $DIRECTORY/$nameout
    cp *.* $DIRECTORY/$nameout
    rm *.*
    echo ===== data copied to $DIRECTORY =====
    cd $CASEDIR
    rmdir "$WORK"-"${nameout}"
    echo ===== temporary directory deleted =====
    cd $DIRECTORY
  else

```

```

        echo directory $WORK contains test output data !!
    fi

    if [ $TM != "-testmode" ]; then
    if [ $ERRCODE = "0" ]; then # -----if---
        echo ===== start postprocessing program read_imintdyn_target.exe =====
# goto results directory and start the post processing programs
        cd $DIRECTORY
        cd ${nameout} # goto sub-directory for post processing
# create all files of postprocessing_data

        ../../../../post/read_imintdyn_target.exe

# create all matrix file of angular and energy distributions of emitted particles
        if [ $POST = "matrix" ] ; then
            echo ===== start postprocessing program read_imintdyn_matrix.exe =====
            ../../../../post/read_imintdyn_matrix.exe
            elif [ $POST = "ebs" ] || [ $POST = "rbs" ] ; then
# create the outebs.dat file of energy histograms for RBS or non-Rutherford BS
# options: -s s1 size of histogram; -a a1,a2 angluar regime
            echo ===== start postprocessing program histogram_ebs.exe =====
            ../../../../post/histogram_ebs.exe -b $BINSIZE -s $SMOOTH
            elif [ $POST = "hrrbs" ] || [ $POST = "hrebs" ] || [ $POST = "esa" ] ; then
# create the outebs.dat file of energy histograms for ESA detector resolution default 0.005
# options: -s s1 size of histogram; -a a1,a2 angluar regime -esamode 0.005
            echo ===== start postprocessing program histogram_ebs.exe with ESA mode =====
            ../../../../post/histogram_ebs.exe -esamode 0.005 -b $BINSIZE -s $SMOOTH
            elif [ $POST = "tofrbs" ] ; then
# create the outebs.dat file of energy histograms for TOF detector resolution default 300ps 1m
# options: -s s1 size of histogram; -a a1,a2 angluar regime -tofmode 300.,1.
            echo ===== start postprocessing program histogram_ebs.exe with ESA mode =====
            ../../../../post/histogram_ebs.exe -tofmode 300 1.0 -b $BINSIZE -s $SMOOTH
            elif [ $POST = "cerda" ] ; then
# create the outcerda.dat file of energy histograms of coincidence CERDA events
# options: -s s1 size of histogram; -a a1,a2 angluar regime
            echo ===== start postprocessing program histogram_cerda.exe =====
            ../../../../post/histogram_cerda.exe -b $BINSIZE -s $SMOOTH -c $NP
            elif [ $POST = "ercs" ] ; then

```

```

# create the outerecs.dat file of energy histograms of coincidence ERCS events
# options: -s s1 size of histogram; -a a1,a2 angluar regime
      echo ===== start postprocessing program histogram_ercs.exe =====
      ../../../../post/histogram_ercs.exe -b $BINSIZE -s $SMOOTH -c $NP
      elif [ $POST = "erda" ] ; then
# create the outerda.dat file of energy histograms of ERDA events
# options: -s s1 size of histogram; -a a1,a2 angluar regime
      echo ===== start postprocessing program histogram_erda.exe =====
      ../../../../post/histogram_erda_nra.exe -b $BINSIZE -s $SMOOTH
      elif [ $POST = "toferda" ] ; then
# create the outerda.dat file of energy histograms of TOF-ERDA events 300ps, 1m
# options: -s s1 size of histogram; -a a1,a2 angluar regime
      echo ===== start postprocessing program histogram_erda.exe =====
      ../../../../post/histogram_erda_nra.exe -tofmode 300., 1.0 -b $BINSIZE -s $SMOOTH
      elif [ $POST = "nra" ] ; then
# create the outerda.dat file of energy histograms of NRA events
# options: -s s1 size of histogram; -a a1,a2 angluar regime
      echo ===== start postprocessing program histogram_erda.exe =====
      ../../../../post/histogram_erda_nra.exe -nramode 0
      fi # -----endif-----
echo ===== all done without error =====

      elif [ $ERRCODE > "0" ]; then # -----if---
      echo ==== Because of ERRCODE no post processing done ====
      fi # -----end if ---
      fi # --- endif testmode
      cd $DIRECTORY
else
      echo =====
      echo $file does not exist!!
      echo =====
fi
done

```

## 9 Example of an input file for an angular distribution (template-angle.inp)

```
template for angular distribution as input (may be generated with ORIGIN)
angle[deg] (0 < alpha < 90deg), probability(not normalized), up to 180 lines
28 0
28.5 0
29 0
29.5 0
30 1
30.5 1
31 1
31.5 2
32 3
32.5 4
33 5
33.5 7
34 8
34.5 11
35 13
35.5 16
36 19
36.5 23
37 27
37.5 32
38 37
38.5 42
39 48
39.5 54
40 60
40.5 66
```

|      |     |
|------|-----|
| 41   | 72  |
| 41.5 | 78  |
| 42   | 83  |
| 42.5 | 88  |
| 43   | 92  |
| 43.5 | 95  |
| 44   | 98  |
| 44.5 | 99  |
| 45   | 100 |
| 45.5 | 99  |
| 46   | 98  |
| 46.5 | 95  |
| 47   | 92  |
| 47.5 | 88  |
| 48   | 83  |
| 48.5 | 78  |
| 49   | 72  |
| 49.5 | 66  |
| 50   | 60  |
| 50.5 | 54  |
| 51   | 48  |
| 51.5 | 42  |
| 52   | 37  |
| 52.5 | 32  |
| 53   | 27  |
| 53.5 | 23  |
| 54   | 19  |
| 54.5 | 16  |
| 55   | 13  |
| 55.5 | 11  |



56 8  
56.5 7  
57 5  
57.5 4  
58 3  
58.5 2  
59 1  
59.5 1  
60 1  
60.5 0  
61 0  
61.5 0  
62 0  
62.5 0  
63 0  
63.5 0  
64 0  
64.5 0  
65 0

## 10 Example of an input file for an energy distribution (template-energy.inp)

template for energy distribution input (may be generated with ORIGIN  
energy [eV] (0.1 eV < E > 1E9 eV), probability (not normalized), up to 200 lines

|       |     |
|-------|-----|
| 50.   | 1.  |
| 100.  | 2.  |
| 120.  | 5.  |
| 150.  | 7.  |
| 400.  | 12. |
| 500.  | 17. |
| 800.  | 22. |
| 1000. | 27. |
| 1200. | 35. |
| 1500. | 54. |
| 1800. | 76. |
| 2400. | 81. |
| 2500. | 74. |
| 2800. | 52. |
| 2900. | 34. |
| 3000. | 26. |
| 3100. | 15. |
| 3400. | 12. |
| 3500. | 5.  |
| 3600. | 2.  |
| 3700. | 1.  |

## 11 Example of an input file for a layered structure (template-layer.def)

This is an example of a multilayer where each layer has a thickness of Å. We define a structure for three species 1,2,3. We start with two layers of species 2, followed by 3 layers of species 3, followed by 2 layers of species 2, etc. At the end we have 50 layers of species 2.

We have an alternating composition of 100% for element species 2 and 3. Element species 1 is supposed to be the incident ion and has initial composition 0.

**Note that the number of layers must be given as Integer and the other values as floating point values (which must contain a decimal ".").**

In total there are 65 layers distributed over the target thickness  $\text{target\_thickness} = 65 \times 3 \text{ \AA} = 195 \text{ \AA}$ . -1 in the last line indicates the end of the layered structure. If this is replaced by 0, then the previous layered structure is repeated until the maximum of  $\text{target\_slicesmax}$  layers are filled. Based on the input, the new target thickness  $\text{target\_thickness}$  with the sum  $\text{target\_slices}$  of the specified layers is calculated.

```
number n of      thick-  composition of target (1:num_species); sum( qu(1:num_species)) = 1
  layers        ness [A]  1    ... num_species ; n<0: end, n=0: copy until target_slicesmax
    2            3.00    0.00  1.00  0.00
    3            3.00    0.00  0.00  1.00
    2            3.00    0.00  1.00  0.00
    3            3.00    0.00  0.00  1.00
    2            3.00    0.00  1.00  0.00
    3            3.00    0.00  0.00  1.00
    50           3.00    0.00  1.00  0.00
    -1
```

In contrast to SDTrimsP, the line with "-1" finished the layer profile input. No further values are required in that line. A value "0" in the last line copies the last layer until the maximum number of layers is reached.

## 12 Compiling the program

### The Make file mk.sh:

```
#!/bin/sh -x
#
# bourne shell compatible script. We can't be sure that it's a bourne shell,
# it can be any shell depending on where /bin/sh soft link is pointing to usually.
# For instance, With the latest Ubuntu "sh" by default pointing to "dash".
# -x
#
SRC_DIR=../../src
make clean
# if makefile does not exist as symbolic link, then make a link to src directory
if [ ! -L Makefile ]; then
    ln -s $SRC_DIR/Makefile Makefile
fi
#
#           OSTYPE= linux
#           RAND=RAND_INTEL or RAND=RAND_CRAY or RAND=NR_RANDQ1 random number generation
#           COMPILER=INTEL or COMPILER=ONEAPI (new intel compiler) select the FORTRAN compiler

echo =====
if test -d /opt/intel/oneapi; then
    echo "Intel OneAPI compiler"
    FC='ONEAPI'
else
    echo "Older Intel Compiler"
    FC='INTEL'
fi
echo =====

make OSTYPE=linux SRC_DIR=$SRC_DIR RAND=NR_RANDQ1 COMPILER=$FC -e -f $SRC_DIR/Makefile
# -e overrides environment variables
# -f specifies the location of Makefile
```

**Makefile:**

```
EXEC = imintdyn.exe

# Versions can be compiled for 64 bit machines from the source code:
# MODE = PAR parallel processing
# Furthermore random number generators can be chosen:
# RAND = ( CRAY | NR_RANDQ1 | RAND_INTEL |
# Possible communication libraries are:
# COMM = ( MPI )
# Possible compiler for linux are:
# COMPILER = ( INTEL | ONEAPI)
# OSTYPE: LINUX: linux

MODE = PAR
COMM    = MPI
SRC_DIR = .
VPATH  = $(SRC_DIR) # use this path to find certain files
#
# catch error if OSTYPE is not defined:
#
FF      = @echo "makefile not configured for OSTYPE \"$(OSTYPE)\"; false

### =====
### ---  Macros for Linux PC -----
ifeq ($(OSTYPE), linux)

# COMPILER = INTEL
# COMPILER = ONEAPI (New Intel Compiler)
```

```
#---INTEL ONEAPI ---
  ifeq ($(COMPILER), ONEAPI)
    #---openMPI
    FF= mpiifx # New OneAPI Intel Fortran Compiler

    ##FFLAGS = -v 9.0 -w -FR -r8 -O2 -ip -tpp7 -xN
    ##FFLAGS = -v 9.0 -w -FR -r8 -O2 -ip -tpp6 -tune pn4 -pg -pc64 -pc80 -check all
    ## -pg -check all
    ## -Fr source file free format
    ## -w disables all warnings
    ## -axN optimisation for all computer
    ## -xN optimisation for computer where compile
    ## -check none
    ## -check noarg_temp_created
    ## -r8 use double precision as default, replaced by -double-size
    ## -real-size 64 8 byte real an complex declarations

    FFLAGS =-w -FR -real-size 64 -O3 -align=all -ansi -axcore=CORE-AVX2,SSE4.2,SSSE3 \
      -falign-functions -fast-transcendentals -finline -no-inline-factor -fp-model=fast=2 -ip \
      -ipo -mcode=avx2,sse4.2,ssse3 -march=x86-64-v3 -mtune=core-avx2 -qopt-matmul \
      -qopt-mem-layout-trans=3 -qopt-multi-version-aggressive -qopt-prefetch=4 -pad -rcd -safe-cray-ptr \
      -scalar-rep -unroll-aggressive -parallel -qopenmp -g -traceback \
      -I$(SRC_DIR) -I/usr/local/lib -D$(OSTYPE) -D$(COMPILER)

    # FFLAGS =-w -FR -real-size 64 -O3 -align=all -ansi-alias -assume buffered_io -axcore=CORE-AVX2,SSE4.1,SSSE3 \
    # -falign-functions -fast-transcendentals -finline -no-inline-factor -fp-model=fast=2 -ip \
    # -ipo -mcode=avx2,sse4.1,ssse3 -march=core-avx2,sse4.1,ssse3 -mtune=core2 -qopt-matmul \
    # -qopt-mem-layout-trans=3 -qopt-multi-version-aggressive -qopt-prefetch=4 -pad -rcd -safe-cray-ptr \
    # -scalar-rep -unroll-aggressive -xSSSE3 -parallel -qopenmp -g -traceback \
    # -I$(SRC_DIR) -I/usr/local/lib -D$(MODE) -D$(OSTYPE) -D$(DEBUG)

  endif # endif of compiler OneAPI INTEL
```

```
#---INTEL---
ifeq ($(COMPILER), INTEL)

    #-openMPI
        FF= mpiifort # New Intel Fortran Compiler
#        FF= mpif90 # (Obelix and Old Intel Fortran compiler)
#        FF= mpifort # (GWDG Intel compiler)

##FFLAGS = -v 9.0 -w -FR -r8 -O2 -ip -tpp7 -xN
##FFLAGS = -v 9.0 -w -FR -r8 -O2 -ip -tpp6 -tune pn4 -pg -pc64 -pc80 -check all
## -pg -check all
## -Fr source file free format
## -w disables all warnings
## -axN optimisation for all computer
## -xN optimisation for computer where compile
## -check none
## -check noarg_temp_created
## -r8 use double precision as default, replaced by -double-size
## -real-size 64 8 byte real an complex declarations

FFLAGS =-w -FR -real-size 64 -O3 -align=all -ansi-alias -assume buffered_io -axcore=CORE-AVX2,SSE4.1,SSSE3 \
        -falign-functions -fast-transcendentals -finline -no-inline-factor -fp-model=fast=2 -ip \
        -ipo -mcode=avx2,sse4.1,ssse3 -march=core-avx2,sse4.1,ssse3 -mtune=core2 -opt-matmul \
        -opt-mem-layout-trans=3 -opt-multi-version-aggressive -opt-prefetch=4 -pad -rcd -safe-cray-ptr \
        -scalar-rep -unroll-aggressive -xSSSE3 -parallel -openmp -g -traceback \
        -I$(SRC_DIR) -I/usr/local/lib -D$(OSTYPE) -D$(COMPILER)

endif # endif of compiler INTEL
```

```
# -- set some compiler flags -----  
  
ifeq ($(RAND), NR_RANDQ1)  
    FFLAGS += -D$(RAND)  
endif  
ifeq ($(RAND), RAND_INTEL)  
    FFLAGS += -D$(RAND)  
endif  
ifeq ($(RAND), RAND_CRAY)  
    FFLAGS += -D$(RAND)  
endif  
ifeq ($(COMM), MPI)  
    FFLAGS += -D$(COMM)  
endif  
  
#-----OneAPI INTEL-----  
ifeq ($(COMPILER), ONEAPI)  
  
    LDR = $(FF) $(FFLAGS)  
    #LIBS = -static  
  
endif  
  
#-----INTEL-----  
ifeq ($(COMPILER), INTEL)  
  
    LDR = $(FF) $(FFLAGS)  
    #LIBS = -static  
  
endif
```



```
endif #$(OSTYPE), linux)
```

```
### -----
```

```
MODULES = MPIfunc.o parameters.o random_numbers.o task_description.o particle_description.o \  
          trajectory_description.o work.o dlb.o zb_stoppingtable_data.o zb_stopping_calc.o \  
          \
```

```
BINS = imintdyn.o history.o recoil.o projectile.o vfmt.o output.o sub.o \  
       sub_chem_diff.o inelast.o assign.o zero_init.o interpolate_fractions.o \  
       pots.o integrations.o tableread.o read_r33.o cputim.o pimpactscan.o \  
       default_init.o gsum_fluence.o gsum_stat.o gsum_mat.o \  
       out_particles.o out_trajectories.o out_datfile.o init_all.o \  
       iba_init.o main_broadcast.o allocate_memory.o set_matrix_ranges.o t3e.o\  
       \
```

```
default: $(EXEC)
```

```
$(EXEC): $(BINS)
```

```
$(LDR) -o $(EXEC) $(MODULES) $(BINS) $(LIBS)
```

```
sub.o: work.o random_numbers.o
```

```
sub_chem_diff.o: work.o MPIfunc.o
```

```
random_numbers.o: work.o MPIfunc.o
```

```
imintdyn.o: work.o parameters.o random_numbers.o alloc_types.o zb_stoppingtable_data.o MPIfunc.o \  
           trajectory_description.o particle_description.o
```

```
zero_init.o: work.o parameters.o
```

```
default_init.o: work.o parameters.o
```

```
history.o: work.o random_numbers.o parameters.o dlb.o MPIfunc.o trajectory_description.o \  
           \
```

```
particle_description.o sub_chem_diff.o
pimpactscan.o: work.o MPIfunc.o
recoil.o: work.o random_numbers.o parameters.o dlb.o MPIfunc.o
projectile.o: work.o random_numbers.o parameters.o dlb.o MPIfunc.o
gsum_fluence.o: work.o MPIfunc.o parameters.o
gsum_stat.o: work.o MPIfunc.o parameters.o
gsum_mat.o: work.o MPIfunc.o parameters.o
out_particles.o: work.o MPIfunc.o
out_trajectories.o: work.o MPIfunc.o output.o: work.o
task_description.o: MPIfunc.o parameters.o
particle_description.o: parameters.o
work.o: particle_description.o task_description.o trajectory_description.o vfmt.F90
dlb.o: task_description.o MPIfunc.o
inelast.o: work.o zb_stopping_calc.o
out_datfile.o: work.o MPIfunc.o
tableread.o: work.o MPIfunc.o zb_stoppingtable_data.o
read_r33.o: work.o MPIfunc.o
init_all.o: work.o parameters.o MPIfunc.o random_numbers.o
iba_init.o: work.o MPIfunc.o parameters.o
main_broadcast.o: work.o MPIfunc.o parameters.o
allocate_memory.o: work.o parameters.o MPIfunc.o dlb.o
set_matrix_ranges.o: work.o parameters.o MPIfunc.o
interpolate_fractions.o: work.o random_numbers.o dlb MPIfunc.o
pots.o: work.o

.SUFFIXES:
.SUFFIXES: .o .F90 .f90 .c
.F90.o:
    $(FF) -c $(FFLAGS) $<
.f90.o:
```

```
$(FF) -c $(FFLAGS) $<
.PHONY: clean update
clean:
    rm -f *.[fio] *.mod $(EXEC) aus
```

## 13 Running the program

IMINTDYN simulations are defined in a sub-folder called “case” in the main folder “imintdyn81”. There each simulation is stored in a specific subfolder, e.g. “1 keV Ar in Si”. This subfolder contains all simulation specific information, such as the main simulation script file “imint-namex” and 2<sup>nd</sup> order script file “namey”. In addition there might be supporting files such as angular profiles, energy profiles or a file defining a layered target structure. These supporting files are text files and have the extension “.def.”. The script files “imint-namex.inp” and “namey.inp” are read using the FORTRAN namlist feature, which is a easy but rough way to initialize a simulations. All variables defined in the .inp files must be correctly specified in a way they are defined in the imintdyn subroutine work, or specified in the imintdyn command parameter description. Any other variable definitions or typing errors lead to an error when reading the namelist.

A script file “imint.sh” contains all information on how to run the program and how to do the post-processing of output files. Within imint.sh one defines the script files “imint-namex” and “namey” as well as the supporting files. Several files “imint-namex” and “namey” can be defined which are then processed as a batch process. One also defines the type of post processing such as “ebs”, “nra”, “erda” etc.

Some parameters for postprocessing can be defined in “imint.sh”, such as

```
POST="ebs" # ebs, hrrbs, hrebs, esa, tofrbs, ercs, erda, toferda, cerda, matrix or none post processing flag
BINSIZE=2048 # binning size of postprocessing spectra
SMOOTH=2 # smoothing of postprocessing spectra
```

POST defines the post processing requested

BINSIZE defines the size of the created postprocessing energy spectra.

SMOOTH= 2 is required to calculate the correct statistical error in the post precessing energy spectra. This is by default 2, but can be set a bit larger if necessary.

The script file "imin.sh" also contains information about the number of cores which are used for the simulation.

In the first few lines of "imint.sh" one can and should activate the "testmode". Then the program runs through its initialization process but does not start a simulation. If an error in the namelist occurs the program shows the line in the ".inp" script file where an error occurred. In this way it is easy to check and correct the consistency of the ".inp" script files.

When using the INTEL Oneapi compiler, one must specify computer ID in a text file called "hostfile". The command which is executed by "imint.sh" is

```
mpirun -n "$NP" -ppn "$PPN" -f ../../bin/linux.PRO/hostfile ../../bin/linux.PRO/imintdyn.exe "$TM"
```

with number of processors NP, number of Threads PPN, location of the file "hostfile" and flag TM if testmode is activated

"hostfile" is here located in the imintdyn82/bin/linux.PRO directory where the also "imintdyn.exe and the compilation script files are located.

When using the older INTEL compiler, the following command is executed:

```
mpirun -n "$NP" ../../bin/linux.PRO/imintdyn.exe "$TM"
```

with number of processors NP and flag TM if testmode is activated.