**bunch module**

The bunch module contains the Bunch class and related classes.

**Classes**

*class* **Analysis**

Read back the files produced by Diagnositics\_particles and obtain some variable of interest.

Currently implemented: just the Betatron Tune, based on the so-called “particle” files.

*Public Functions*

**Analysis**(std::string const & filename, size\_t maxTurn = 0)

**~Analysis**()

double **get\_betatron\_averaged\_tune**(bool isHorizontal)

double **get\_betatron\_tune\_spread**(bool isHorizontal)

std::vector< double > **get\_betatron\_tunes**(bool isHorizontal)

std::vector< double > **get\_XCoords\_forTunes**(size\_t selectedParticle)

std::vector< double > **get\_YCoords\_forTunes**(size\_t selectedParticle)

size\_t **get\_num\_betatron\_tune\_found**(bool isHorizontal)

void **set\_minimum\_required\_turn\_Num**(size\_t n)

size\_t **get\_minimum\_required\_turn\_Num**()

int **get\_num\_turns**()

int **get\_num\_part\_first\_bunch**()

std::vector< double > **get\_transverse\_action\_for\_particle**(bool isH, size\_t selectedParticle, double alpha, double beta)

std::vector< double > **get\_transverse\_action\_for\_bunch**(bool isH, size\_t turnNumber, double alpha, double beta)

*class* **Bunch**

Represents a macroparticle bunch distributed across the processors in a comm\_sptrunicator.

*Public Type*

**State enum**

The state of the bunch is captured at a fixed s (or z, longitudinal coordinate) or at a fixed time. In the former case, particles are found within a range of different time coordinates while in the later case particles position along the beam axis do vary. A change of state is accomplish via the fixed\_t\_z\_converter class.

*Values:*

* fixed\_z = = 1 -
* fixed\_t = = 2 -
* fixed\_z\_lab = = 1 -
* fixed\_t\_bunch = = 2 -
* fixed\_t\_lab = = 3 -
* fixed\_z\_bunch = = 4 -

*Public Functions*

**Bunch**([*Reference\_particle*](http://compacc.fnal.gov/~amundson/html/foundation.html#project0class_reference__particle) const & reference\_particle, int total\_num, double real\_num, [*Commxx\_sptr*](http://compacc.fnal.gov/~amundson/html/utils.html#project0commxx_8h_1a8e4a45ebebd319814e88c81a65f0dbd9) comm\_sptr)

Constructor: Allocates memory for the particles and assigns particle ID’s, but does not fill the phase space values in any way.

To fill the bunch with particles, use the populate methods.

**Parameters**

* reference\_particle -

the reference particle for the bunch.

* total\_num -

the total number of macroparticles in the bunch

* real\_num -

the number of real particles represented by the bunch.

* bucket\_index -

the bucket number the bunch occupies, used for multi-bunch simulations

* comm\_sptr -

the comm\_sptrunicator.

**Bunch**([*Reference\_particle*](http://compacc.fnal.gov/~amundson/html/foundation.html#project0class_reference__particle) const & reference\_particle, int total\_num, double real\_num, [*Commxx\_sptr*](http://compacc.fnal.gov/~amundson/html/utils.html#project0commxx_8h_1a8e4a45ebebd319814e88c81a65f0dbd9) comm\_sptr, int particle\_charge)

Constructor with 5-parameter signature Same as above, but having the flexibility to redefine the charge of a particle.

**Parameters**

* reference\_particle -

the reference particle for the bunch.

* total\_num -

the total number of macroparticles in the bunch

* real\_num -

the number of real particles represented by the bunch.

* comm\_sptr -

the comm\_sptrunicator.

* particle\_charge -

in units of e.

**Bunch**([*Reference\_particle*](http://compacc.fnal.gov/~amundson/html/foundation.html#project0class_reference__particle) const & reference\_particle, int total\_num, double real\_num, [*Commxx\_sptr*](http://compacc.fnal.gov/~amundson/html/utils.html#project0commxx_8h_1a8e4a45ebebd319814e88c81a65f0dbd9) comm\_sptr, double z\_period\_length, int bucket\_index = 0)

**Bunch**()

Default constructor for serialization use only.

**Bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) const & bunch)

Copy constructor.

[*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & **operator=**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) const & bunch)

Assignment constructor.

void **set\_particle\_charge**(int particle\_charge)

Set the particle charge.

**Parameters**

* particle\_charge -

in units of e.

void **set\_real\_num**(double real\_num)

Set the number of real particles represented by the bunch.

**Parameters**

* real\_num -

the new real number of particles

void **set\_local\_num**(int local\_num)

Reduce (set) the number of particles on this processor.

The number of particles can only be lowered by this member function. (In order to add new particles, create another [*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) and use the inject member.) The total number and real number for the bunch will not be correct until [*update\_total\_num()*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch_1a5407e27bc81a1e77fa7a1d06e725ab35) is called. The real number will scale to reflect the change in the total number. n.b.: The only way to change the total number after the bunch has been created is to change the local numbers on each processor.

**Parameters**

* local\_num -

the new number of particles on this processor

void **update\_total\_num**()

Update the total number and real number of particles after the local number has been changed.

Requires comm\_sptrunication.

void **set\_sort\_period**(int period)

Set the period for periodic\_sort and reset the counter Periods less than zero will prohibit sorting.

**Parameters**

* period -

void **sort**(int index)

Sort the particles.

**Parameters**

* index -

the particle index on which to sort

void **periodic\_sort**(int index)

Sort the particles every sort period calls.

**Parameters**

* index -

the particle index on which to sort

void **set\_converter**([*Fixed\_t\_z\_converter*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_fixed__t__z__converter) & converter)

Set the [*Fixed\_t\_z\_converter*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_fixed__t__z__converter) class to be used when converting between fixed-t and fixed-z representations.

**Parameters**

* converter -

the converter class

void **convert\_to\_state**([*State*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch_1abff5638a0453dc549f9936860b9e64da) state)

Convert to (fixed-t or fixed-z) state if necessary.

Does nothing if the bunch is already in the requested state.

**Parameters**

* state -

convert to this state.

[*Reference\_particle*](http://compacc.fnal.gov/~amundson/html/foundation.html#project0class_reference__particle) & **get\_reference\_particle**()

Return the reference particle.

[*Reference\_particle*](http://compacc.fnal.gov/~amundson/html/foundation.html#project0class_reference__particle) const & **get\_reference\_particle**()

[*MArray2d\_ref*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1ae229cef4126da69b5a71fef37eeac3c5) **get\_local\_particles**()

Get the array containing the macroparticles on this processor.

The array has length (length,7), where length of the array may be larger local\_num. The macroparticle state vectors are stored in array[0:local\_num,0:6] and the macroparticle IDs are stored in array[0:local\_num,6]. Use [*get\_local\_num()*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch_1a8d54b9effd4820f4c13cab1807298488) to obtain local\_num.

[*Const\_MArray2d\_ref*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1a0c7d8b792425b7d17a90ca4d88b439d9) **get\_local\_particles**()

int **get\_particle\_charge**()

Get the particle charge in units of e.

double **get\_mass**()

Get the particle mass in units GeV/c^2.

double **get\_real\_num**()

Get the real number of particles represented by the bunch.

double **get\_z\_period\_length**()

Get the period length of the bunch.

bool **is\_z\_periodic**()

Is the bunch periodic?

int **get\_local\_num**()

Get the number of macroparticles stored on this processor.

int **get\_total\_num**()

Get the total number of macroparticles.

int **get\_sort\_period**()

Get the period for periodic\_sort.

void **set\_bucket\_index**(int index)

int **get\_bucket\_index**()

[*State*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch_1abff5638a0453dc549f9936860b9e64da) **get\_state**()

Get the (fixed-t or fixed-z) state.

[*Commxx*](http://compacc.fnal.gov/~amundson/html/utils.html#project0class_commxx) const & **get\_comm**()

Get the communicator.

[*Commxx\_sptr*](http://compacc.fnal.gov/~amundson/html/utils.html#project0commxx_8h_1a8e4a45ebebd319814e88c81a65f0dbd9) **get\_comm\_sptr**()

Get the communicator.

void **inject**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) const & bunch)

Add a copy of the particles in bunch to the current bunch.

The injected bunch must have the same macroparticle weight, i.e., real\_num/total\_num. If the state vectors of the reference particles of the two bunches differ, the particles will be shifted accordingly.

void **check\_pz2\_positive**()

template < class Archive >

void **save**(Archive & ar, const unsigned int version)

template < class Archive >

void **load**(Archive & ar, const unsigned int version)

**~Bunch**()

*Public Static Attributes*

const int **x**

const int **xp**

const int **y**

const int **yp**

const int **z**

const int **zp**

const int **cdt**

const int **dpop**

const int **id**

*class* **Bunch\_train**

*Public Functions*

**Bunch\_train**(Bunches const & bunches, double spacing)

**Bunch\_train**(Bunches const & bunches, std::vector< double > const & spacings)

**Bunch\_train**()

[*Commxx\_sptr*](http://compacc.fnal.gov/~amundson/html/utils.html#project0commxx_8h_1a8e4a45ebebd319814e88c81a65f0dbd9) **get\_parent\_comm\_sptr**()

size\_t **get\_size**()

Bunches & **get\_bunches**()

std::vector< double > & **get\_spacings**()

std::vector< int > & **get\_proc\_counts\_for\_impedance**()

std::vector< int > & **get\_proc\_offsets\_for\_impedance**()

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

**~Bunch\_train**()

*class* **Diagnostics**

[*Diagnostics*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics) is an abstract base class for bunch diagnostics classes.

*Public Functions*

**Diagnostics**(std::string const & name, std::string const & filename, std::string const & local\_dir = “”)

**Diagnostics**()

std::string const & **get\_filename**()

std::string const & **get\_local\_dir**()

void **set\_bunch\_sptr**([*Bunch\_sptr*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0bunch_8h_1a692c3b3b613b19e0fd4c4d2603f2d554) bunch\_sptr)

bool **have\_bunch**()

void **delete\_write\_helper\_ptr**()

[*Diagnostics\_write\_helper*](http://compacc.fnal.gov/~amundson/html/foundation.html#project0class_diagnostics__write__helper) \* **new\_write\_helper\_ptr**()

bool **have\_write\_helper**()

[*Diagnostics\_write\_helper*](http://compacc.fnal.gov/~amundson/html/foundation.html#project0class_diagnostics__write__helper) & **get\_write\_helper**()

[*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & **get\_bunch**()

bool **is\_serial**()

Multiple serial diagnostics can be written to a single file.

void **update**()

Update the diagnostics.

void **write**()

Write the diagnostics to the file.

void **update\_and\_write**()

Update the diagnostics and write them to the file.

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

**~Diagnostics**()

*class* **Diagnostics\_basic**

[*Diagnostics\_basic*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__basic) provides the minimal set of statistical quantities to be calculated for a [*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch).

*Public Functions*

**Diagnostics\_basic**(std::string const & filename, std::string const & local\_dir = “”)

Create a [*Diagnostics\_basic*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__basic) object.

**Parameters**

* filename -

filename for output

* local\_dir -

local directory to use for temporary scratch

**Diagnostics\_basic**()

bool **is\_serial**()

Multiple serial diagnostics can be written to a single file.

The [*Diagnostics\_basic*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__basic) class is serial.

void **init\_writers**([*Hdf5\_file\_sptr*](http://compacc.fnal.gov/~amundson/html/utils.html#project0hdf5__file_8h_1aabf20c82608aa7edaf304a96e5eb56d9) file\_sptr)

void **update**()

Update the diagnostics.

double **get\_s\_n**()

Get the distance from the origin along the reference trajectory in meters.

int **get\_repetition**()

Get the number of complete repetitions.

double **get\_s**()

Get the total distance along the reference trajectory in meters.

int **get\_num\_particles**()

Get the total number of macroparticles in the bunch.

double **get\_real\_num\_particles**()

Get the total number of real particles represented by the bunch.

[*Const\_MArray1d\_ref*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1ae8c173d01855d23993771d2fd98632e9) **get\_mean**()

Get a six-dimensional vector of the means of each phase-space coordinate.

The units are in Synergia units.

[*Const\_MArray1d\_ref*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1ae8c173d01855d23993771d2fd98632e9) **get\_std**()

Get a six-dimensional vector of the standard deviations of each phase-space coordinate.

The units are in Synergia units.

const [*MArray1d*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1a6fdb09aeda0bff08536900e90228f5be) **get\_min**()

const [*MArray1d*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1a6fdb09aeda0bff08536900e90228f5be) **get\_max**()

void **write**()

Write the diagnostics to the file.

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

**~Diagnostics\_basic**()

*Public Static Attributes*

const char **name**[]

*class* **Diagnostics\_bulk\_track**

[*Diagnostics\_bulk\_track*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__bulk__track) records the phase space coordinates of a multiple particles.

Particles will only be tracked if they stay on the same processor. Lost particles that are somehow restored or particles not available when the first update is called will also not be tracked.

*Public Functions*

**Diagnostics\_bulk\_track**(std::string const & filename, int num\_tracks, int offset = 0, std::string const & local\_dir = “”)

Create an empty [*Diagnostics\_bulk\_track*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__bulk__track) object.

**Parameters**

* bunch\_sptr -

the [*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch)

* filename -

the base name for file to write to (base names will have a numerical index inserted

* num\_tracks -

the number of local particles to track

* offset -

id offset for first particle to track

* local\_dir -

local directory to use for temporary scratch

**Diagnostics\_bulk\_track**()

bool **is\_serial**()

Multiple serial diagnostics can be written to a single file.

The [*Diagnostics\_bulk\_track*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__bulk__track) class is serial.

void **update**()

Update the diagnostics.

void **write**()

Write the diagnostics to the file.

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

**~Diagnostics\_bulk\_track**()

*Public Static Attributes*

const char **name**[]

*class* **Track\_status**

*Public Functions*

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

*Public Members*

bool **found**

int **last\_index**

int **particle\_id**

*class* **Diagnostics\_full2**

[*Diagnostics\_full2*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__full2) provides the full set of statistical quantities to be calculated for a [*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) up to the second moments.

*Public Functions*

**Diagnostics\_full2**(std::string const & filename, std::string const & local\_dir = “”)

Create a [*Diagnostics\_full2*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__full2) object.

**Parameters**

* bunch -

the [*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch)

* filename -

filename for output

* local\_dir -

local directory to use for temporary scratch

**Diagnostics\_full2**()

void **init\_writers**([*Hdf5\_file\_sptr*](http://compacc.fnal.gov/~amundson/html/utils.html#project0hdf5__file_8h_1aabf20c82608aa7edaf304a96e5eb56d9) file\_sptr)

bool **is\_serial**()

Multiple serial diagnostics can be written to a single file.

The [*Diagnostics\_full2*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__full2) class is serial.

void **update**()

Update the diagnostics.

double **get\_s\_n**()

Get the distance from the origin along the reference trajectory in meters.

int **get\_repetition**()

Get the number of complete repetitions.

double **get\_s**()

Get the total distance along the reference trajectory in meters.

int **get\_num\_particles**()

Get the total number of macroparticles in the bunch.

double **get\_real\_num\_particles**()

Get the total number of real particles represented by the bunch.

[*Const\_MArray1d\_ref*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1ae8c173d01855d23993771d2fd98632e9) **get\_mean**()

Get a six-dimensional vector of the means of each phase-space coordinate.

The units are in Synergia units.

[*Const\_MArray1d\_ref*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1ae8c173d01855d23993771d2fd98632e9) **get\_std**()

Get a six-dimensional vector of the standard deviations of each phase-space coordinate.

The units are in Synergia units.

const [*MArray1d*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1a6fdb09aeda0bff08536900e90228f5be) **get\_min**()

const [*MArray1d*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1a6fdb09aeda0bff08536900e90228f5be) **get\_max**()

[*Const\_MArray2d\_ref*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1a0c7d8b792425b7d17a90ca4d88b439d9) **get\_mom2**()

Get a 6x6 matrix of the second moments of the phase-space coordinates.

The units are Synergia units.

[*Const\_MArray2d\_ref*](http://compacc.fnal.gov/~amundson/html/utils.html#project0multi__array__typedefs_8h_1a0c7d8b792425b7d17a90ca4d88b439d9) **get\_corr**()

Get a 6x6 matrix of the correlation coefficients of the phase-space coordinates.

double **get\_emitx**()

Get the horizontal emittance.

Currently reported in unnatural Synergia units.

double **get\_emity**()

Get the vertical emittance.

Currently reported in unnatural Synergia units.

double **get\_emitz**()

Get the longitudinal emittance.

Currently reported in unnatural Synergia units.

double **get\_emitxy**()

Get the (4D) transverse emittance.

Currently reported in unnatural Synergia units.

double **get\_emitxyz**()

Get the (6D) full emittance.

Currently reported in unnatural Synergia units.

void **write**()

Write the diagnostics to the file.

bool **get\_have\_writers**()

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

**~Diagnostics\_full2**()

*Public Static Attributes*

const char **name**[]

*class* **Diagnostics\_particles**

[*Diagnostics\_particles*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__particles) dumps the state of particles in a bunch.

*Public Functions*

**Diagnostics\_particles**(std::string const & filename, int min\_particle\_id = 0, int max\_particle\_id = 0, std::string const & local\_dir = “”)

Create a [*Diagnostics\_particles*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__particles) object.

**Parameters**

* bunch\_sptr -

the [*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch)

* filename -

the base name for file to write to (base names will have a numerical index inserted

* min\_particle\_id -

the lowest particle id to write (defaults to 0)

* max\_particle\_id -

the highest particle id to write (0 indicates no limit, hence min,max = 0,0 writes all particles)

* local\_dir -

local directory to use for temporary scratch

**Diagnostics\_particles**()

bool **is\_serial**()

Multiple serial diagnostics can be written to a single file.

The [*Diagnostics\_particles*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__particles) class is not serial.

void **update**()

Update the diagnostics.

void **write**()

Write the diagnostics to the file.

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

**~Diagnostics\_particles**()

*Public Static Attributes*

const char **name**[]

*class* **Diagnostics\_reference\_particle**

*Public Functions*

**Diagnostics\_reference\_particle**(std::string const & filename, std::string const & local\_dir = “”)

Create a [*Diagnostics\_reference\_particle*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__reference__particle) object.

**Parameters**

* bunch -

the [*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch)

* filename -

filename for output

* local\_dir -

local directory to use for temporary scratch

**Diagnostics\_reference\_particle**()

bool **is\_serial**()

Multiple serial diagnostics can be written to a single file.

The [*Diagnostics\_reference\_particle*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__reference__particle) class is serial.

void **init\_writers**([*Hdf5\_file\_sptr*](http://compacc.fnal.gov/~amundson/html/utils.html#project0hdf5__file_8h_1aabf20c82608aa7edaf304a96e5eb56d9) file\_sptr)

void **update**()

Update the diagnostics.

void **write**()

Write the diagnostics to the file.

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

**~Diagnostics\_reference\_particle**()

*Public Static Attributes*

const char **name**[]

*class* **Diagnostics\_track**

[*Diagnostics\_track*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__track) records the phase space coordinates of a single particle.

Particles will only be tracked if they stay on the same processor. Lost particles that are somehow restored or particles not available when the first update is called will also not be tracked.

*Public Functions*

**Diagnostics\_track**(std::string const & filename, int particle\_id, std::string const & local\_dir = “”)

Create an empty [*Diagnostics\_track*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__track) object.

**Parameters**

* bunch\_sptr -

the [*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch)

* filename -

the base name for file to write to (base names will have a numerical index inserted

* particle\_id -

the particle ID to track

* local\_dir -

local directory to use for temporary scratch

**Diagnostics\_track**()

[*Diagnostics\_write\_helper*](http://compacc.fnal.gov/~amundson/html/foundation.html#project0class_diagnostics__write__helper) \* **new\_write\_helper\_ptr**()

bool **is\_serial**()

Multiple serial diagnostics can be written to a single file.

The [*Diagnostics\_track*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__track) class is serial.

void **update**()

Update the diagnostics.

void **write**()

Write the diagnostics to the file.

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

**~Diagnostics\_track**()

*Public Static Attributes*

const char **name**[]

*class* **Fixed\_t\_z\_alex**

*Public Functions*

void **from\_z\_lab\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the accelerator frame.

void **from\_t\_lab\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the accelerator frame to the fixed-z state in the accelerator frame.

void **from\_z\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the beam frame.

void **from\_t\_bunch\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the beam frame to the fixed-z state in the accelerator frame.

void **from\_t\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

void **from\_t\_bunch\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

*class* **Fixed\_t\_z\_ballistic**

[*Fixed\_t\_z\_zeroth*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_fixed__t__z__zeroth) implements a fixed-t-fixed-z converter using the ballistic approximation: longitudinal coordinates are transformed, then transverse coordinates are transformed using the ballistic approximation, i.e., as though the particles were traveling independently in free space.

UNIMPLEMENTED

*Public Functions*

void **from\_z\_lab\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the accelerator frame.

void **from\_t\_lab\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the accelerator frame to the fixed-z state in the accelerator frame.

void **from\_z\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the beam frame.

void **from\_t\_bunch\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the beam frame to the fixed-z state in the accelerator frame.

void **from\_t\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

void **from\_t\_bunch\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

*class* **Fixed\_t\_z\_converter**

[*Fixed\_t\_z\_converter*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_fixed__t__z__converter) is a virtual base class for converters between the bunch fixed-z representation and the bunch fixed-t representation.

There are three converters one can choose:

*Public Functions*

void **from\_z\_lab\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the accelerator frame.

void **from\_t\_lab\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the accelerator frame to the fixed-z state in the accelerator frame.

void **from\_z\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the beam frame.

void **from\_t\_bunch\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the beam frame to the fixed-z state in the accelerator frame.

void **from\_t\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

void **from\_t\_bunch\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

Convert from the fixed-t state to the fixed-z state.

Convert from the fixed-z state to the fixed-t state.

**~Fixed\_t\_z\_converter**()

*class* **Fixed\_t\_z\_synergia20**

transformation as in the old synergia....

*Public Functions*

void **from\_z\_lab\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the accelerator frame.

void **from\_t\_lab\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the accelerator frame to the fixed-z state in the accelerator frame.

void **from\_z\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the beam frame.

void **from\_t\_bunch\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the beam frame to the fixed-z state in the accelerator frame.

void **from\_t\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

void **from\_t\_bunch\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

*class* **Fixed\_t\_z\_zeroth**

[*Fixed\_t\_z\_zeroth*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_fixed__t__z__zeroth) implements a fixed-t-fixed-z converter using the simplest approximation: longitudinal coordinates are transformed, but transverse coordinates are unaffected.

*Public Functions*

void **from\_z\_lab\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the accelerator frame.

void **from\_t\_lab\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the accelerator frameto the fixed-z state in the accelerator frame.

void **from\_z\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-z state in the accelerator frame to the fixed-t state in the beam frame.

void **from\_t\_bunch\_to\_z\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

Convert from the fixed-t state in the beam frame to the fixed-z state in the accelerator frame.

void **from\_t\_lab\_to\_t\_bunch**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

void **from\_t\_bunch\_to\_t\_lab**([*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) & bunch)

template < class Archive >

void **serialize**(Archive & ar, const unsigned int version)

**Typedefs**

typedef boost::shared\_ptr< [*Bunch*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_bunch) > **Bunch\_sptr**

typedef boost::shared\_ptr< [*Diagnostics\_basic*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__basic) > **Diagnostics\_basic\_sptr**

typedef boost::shared\_ptr< [*Diagnostics\_bulk\_track*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__bulk__track) > **Diagnostics\_bulk\_track\_sptr**

typedef boost::shared\_ptr< [*Diagnostics\_full2*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__full2) > **Diagnostics\_full2\_sptr**

typedef boost::shared\_ptr< [*Diagnostics\_particles*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__particles) > **Diagnostics\_particles\_sptr**

typedef boost::shared\_ptr< [*Diagnostics\_reference\_particle*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__reference__particle) > **Diagnostics\_reference\_particle\_sptr**

typedef boost::shared\_ptr< [*Diagnostics*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics) > **Diagnostics\_sptr**

typedef boost::shared\_ptr< [*Diagnostics\_track*](http://compacc.fnal.gov/~amundson/html/bunch.html#project0class_diagnostics__track) > **Diagnostics\_track\_sptr**