In the following, a general format for the input file of MITHRA is presented. The red icons or groups can be repeated in the text. *int* stands for an integer number, *real* represents a real value, and *string* denotes a string of characters. The reference directory in the path locations is the path where the simulation is started. In other words, "./" points to the location where the project is called.

```
MESH
{
    length-scale            = < real |
                                METER |
                                DECIMETER |
                                CENTIMETER |
                                MILLIMETER |
                                MICROMETER |
                                NANOMETER |
                                ANGSTROM >
    time-scale              = < real |
                                SECOND |
                                MILLISECOND |
                                MICROSECOND |
                                NANOSECOND |
                                PICOSECOND |
                                FEMTOSECOND |
                                ATTOSECOND >
    mesh-lengths            = < ( real, real, real ) >
    mesh-resolution         = < ( real, real, real ) >
    mesh-center             = < ( real, real, real ) >
    total-time              = < real >
    bunch-time-step         = < real >
    bunch-time-start        = < real >
    mesh-truncation-order   = < 1 | 2 >
    space-charge            = < true | false >
}

BUNCH
{
bunch-initialization
{
    type                    = < manual |
                                ellipsoid |
                                3D-crystal |
                                file >
    distribution            = < uniform | gaussian >
    charge                  = < real >
    number-of-particles     = < int >
    gamma                   = < real >
    beta                    = < real >
    direction               = < ( real, real, real ) >
    position                = < ( real, real, real ) >
    sigma-position          = < ( real, real, real ) >
    sigma-momentum          = < ( real, real, real ) >
    numbers                 = < ( int, int, int ) >
    lattice-constants       = < ( real, real, real ) >
    transverse-truncation   = < real >
    longitudinal-truncation = < real >
    bunching-factor         = < real between 0 and 1 >
    bunching-factor-phase   = < real >
    shot-noise              = < true | false >
}
bunch-sampling
{
    sample                  = < true | false >
    directory               = < /path/to/location >
    base-name               = < string >
    rhythm                  = < real >
}
bunch-visualization
{
    sample                  = < true | false >
    directory               = < /path/to/location >
    base-name               = < string >
    rhythm                  = < real >
}
bunch-profile
{
    sample                  = < true | false >
    directory               = < /path/to/location >
    base-name               = < string >
    time                    = < real >
    rhythm                  = < real >
}
}
FIELD
{
field-initialization
{
    type                    = < plane-wave |
                                confined-plane-wave |
                                gaussian-beam >
    position                = < ( real, real, real ) >
    direction               = < ( real, real, real ) >
    polarization            = < ( real, real, real ) >
    radius-parallel         = < real >
    radius-perpendicular    = < real >
    signal-type             = < neumann | gaussian |
                                secant-hyperbolic |
                                flat-top >
    strength-parameter      = < real >
    offset                  = < real >
    variance                = < real >
    wavelength              = < real >
    CEP                     = < real >
}
field-sampling
{
    sample                  = < true | false >
    type                    = < over-line | at-point >
    field                   = < Ex | Ey | Ez |
                                Bx | By | Bz |
                                Ax | Ay | Az |
                                Jx | Jy | Jz |
                                F  | Q >
    directory               = < /path/to/location >
    base-name               = < string >
    rhythm                  = < real >
    position                = < ( real, real, real ) >
    line-begin              = < ( real, real, real ) >
    line-end                = < ( real, real, real ) >
    number-of-points        = < int >
}
field-visualization
{
    sample                  = < true | false >
    type                    = < in-plane | all-domain >
    plane                   = < xy | yz | xz >
    position                = < ( real, real, real ) >
    field                   = < Ex | Ey | Ez |
                                Bx | By | Bz |
                                Ax | Ay | Az |
                                Jx | Jy | Jz |
                                F  | Q >
    directory               = < /path/to/location >
    base-name               = < string >
    rhythm                  = < real >
}
field-profile
{
    sample                  = < true | false >
    field                   = < Ex | Ey | Ez |
                                Bx | By | Bz |
                                Ax | Ay | Az |
                                Jx | Jy | Jz |
                                F  | Q >
    directory               = < /path/to/location >
    base-name               = < string >
    rhythm                  = < real >
    time                    = < real >
}
}
UNDULATOR
{
static-undulator
{
    undulator-parameter     = < real >
    period                  = < real >
    length                  = < int >
    polarization-angle      = < real >
    offset                  = < real >
}
static-undulator-array
{
    undulator-parameter     = < real >
    period                  = < real >
    length                  = < int >
    polarization-angle      = < real >
    gap                     = < real >
    number                  = < int >
```

```
tapering-parameter    = < real >
}

optical-undulator
{
beam-type    = < plane-wave |
                 confined-plane-wave |
                 gaussian-beam >

position             = < ( real, real, real ) >
direction            = < ( real, real, real ) >
polarization         = < ( real, real, real ) >
radius-parallel      = < real >
radius-perpendicular = < real >
signal-type          = < neumann | gaussian |
                         secant-hyperbolic |
                         flat-top >

strength-parameter   = < real >
offset               = < real >
variance             = < real >
wavelength           = < real >
CEP                  = < real >
}
}

EXTERNAL-FIELD
{
electromagnetic-wave
{
beam-type    = < plane-wave |
                 confined-plane-wave |
                 gaussian-beam >

position             = < ( real, real, real ) >
direction            = < ( real, real, real ) >
polarization         = < ( real, real, real ) >
radius-parallel      = < real >
radius-perpendicular = < real >
signal-type          = < neumann | gaussian |
                         secant-hyperbolic |
                         flat-top >

strength-parameter   = < real >
offset               = < real >
variance             = < real >
wavelength           = < real >
CEP                  = < real >
}

FEL-OUTPUT
{
radiation-power
{
sample               = < false | true >
type                 = < at-point | over-line >
directory            = < /path/to/location >
base-name            = < string >
plane-position       = < real >
line-begin           = < real >
line-end             = < real >
number-of-points     = < int >
normalized-frequency = < real >
minimum-normalized-frequency = < real >
maximum-normalized-frequency = < real >
number-of-frequency-points   = < int >
}

power-visualization
{
sample               = < false | true >
directory            = < /path/to/location >
base-name            = < string >
plane-position       = < real >
normalized-frequency = < real >
rhythm               = < real >
}
}
```