

# **External Data File (EDF) Package Users' Guide**

The External Data File (EDF) package serves as a utility in MELCOR to allow a general means of communication with external data files containing time history data. In any MELCOR run, one or more such files may be defined. Each file contains values of time and the corresponding values of one or more dependent variables, and therefore defines each dependent variable as a function of time. Data can be either read from or written to each file; the permitted direction is defined in MELGEN. All characteristics of each file are also defined by EDF input. The EDF package assumes all responsibility for opening and positioning the files, and for reading from each file or writing to it, as appropriate.

Data that have been read in by the EDF package may be accessed and used by other packages in MELCOR. Data to be written out are obtained from the databases of other MELCOR packages. The interface may be either through the Control Function package or—in cases where appropriate coding has been provided—through utility-level entries in the EDF package.

The primary use of the EDF package is to facilitate input of data that define sources and/or boundary conditions as a function of time, particularly in cases where the volume of data is so great that the generation of tabular function input would be extremely tedious and error prone, or completely impractical. The ability to write files allows these input data to be generated directly by another MELCOR calculation; files written by other codes (or even by hand) may also be used. In addition, data files written by MELCOR could be used as input to another simulation code or to an output processor.

This document gives an introduction to the package, describes input requirements, lists sensitivity coefficients, plot variables and control function arguments, and describes output. Sample input is provided and discussed.

# EDF Package Users' Guide

## Contents

1.	Introduction .....	5
2.	User Input Requirements .....	7
2.1	MELGEN User Input .....	7
	<b>EDFnnn01</b> – File Specification .....	7
	<b>EDFnnn02</b> – External Data File Format .....	8
	<b>EDFnnn03</b> – Time Offset for External Data File .....	8
	<b>EDFnnn04</b> – Buffer Length for READ File .....	8
	<b>EDFnnn1k</b> – Write Increment Control for WRITE or PUSH File .....	9
	<b>EDFnnnk</b> – Channel Variables for WRITE File .....	9
2.2	MELCOR User Input .....	10
	<b>EDFnnn01</b> – File Specification .....	10
	<b>EDFnnn02</b> – External Data File Format .....	10
	<b>EDFnnn03</b> – Time Offset for External Data File .....	10
	<b>EDFnnn1k</b> – Write Increment Control for WRITE or PUSH File .....	10
3.	Sensitivity Coefficients .....	10
4.	Plot Variables and Control Function Arguments .....	10
5.	Example Input .....	11
6.	Discussion of Output .....	13

# EDF Package Users' Guide

## 1. Introduction

The External Data File (EDF) package serves as a utility in MELCOR to allow other packages to communicate with external data files. One or more such files may be defined in any MELCOR calculation. Each file is identified by a user-assigned number and a user-defined name in addition to the name by which the operating system recognizes the file, and contains values of time and of one or more dependent variables. It therefore defines a function of time, referred to as a "data channel", for each dependent variable in the file. Data can be either read from or written to each file; the permitted direction is defined in MELGEN. A file may not be both read from and written to in a single calculation.

A file from which data are read is termed a *READ* file; the data that have been read in may be accessed by other packages in MELCOR, either as control function arguments through the Control Function (CF) package or directly through utility-level calls to the EDF package. In the former case, only "old" values, appropriate to the start of the current timestep, are defined. (This is the case for *any* control function value.) In the latter case, only values for times within the current timestep are available, since only that portion of the external file is required to be in the EDF database. Simple linear interpolation is used between tabulated points read from the file. Direct access requires specific coding that must be provided by the code developer. This capability is currently used only by the Control Function Hydrodynamics (CVH) and Transfer Process (TP) packages.

As suggested above, the EDF database provides storage for only a limited number of records to be read in from each file. The requirement that these records must span the entire MELCOR timestep may imply a limitation on that timestep in cases where very small time increments are used in portions of a data file. Therefore, the size of each record buffer may be changed (enlarged) by user input if desired.

Data to be written out are obtained from the databases of other MELCOR packages. Time is accessed directly and automatically by EDF; dependent variables may be obtained in either of two ways. For a *WRITE* file, the value of any variable accessible as a control function argument may be "pulled" into the EDF database using methods that exactly parallel those used for arguments in the CF package. For a *PUSH* file, values must be "pushed" into the EDF database by another package through utility-level entries in the EDF package. Such direct access again requires specific coding in the controlling routine, and the option is currently used only by the Transfer Process package. The frequency with which records are written is controlled by the user.

With the current coding of EDF, the values of variables which are to be written to a file are also available as control function arguments. However, use of these control function arguments should be avoided because the values available in EDF at the time the CF package is called are one timestep out of date. (If the order of execution were reversed, the values of any Control Functions written to an external file would be out of date.) In

## EDF Package Users' Guide

essentially all cases, up-to-date information may be obtained by direct reference to a control function argument in the package from which EDF is obtaining the data.

If data are pushed into the EDF database by other packages, EDF has no way of identifying (for edit purposes) what they represent. Therefore, a path is provided to allow the other package to also define character labels for the data channels it pushes. A check is made in the EDF package to verify that values are actually pushed for every channel on every timestep.

The primary use of the EDF package is to facilitate input of data which define sources and/or boundary conditions as functions of time. The most obvious use is to communicate data from one MELCOR run to another. For example, ex-vessel sources can be calculated with a detailed nodalization of the primary system and a very simple containment model. These sources can be stored in a file, and then input to a later calculation with detailed containment nodalization but little or no representation of the primary.

In cases where an external data file specifies a source or sink of mass or energy, it is generally preferable that the file record cumulative (integral) sources rather than rates. This assures that, regardless of the actual timesteps taken by MELCOR, the total source in the calculation will match that in the data file. Construction of an integral source may involve use of the INTEGRAL control function in the MELCOR run which writes the file. If the receiving package does not accept cumulative sources, data read from this file in a later run can be converted to rates using the forward difference (DER-F) control function.

The EDF package can also be used to communicate data to or from another code. An input file need not have been generated by MELCOR; it might be the output of another code, or might even have been constructed by hand. A file written by MELCOR can be used as input to another simulation code or to an output processor such as a special-purpose plot program. To facilitate these uses, the format of records in each file may be specified in MELGEN; the default is for the file to be unformatted.

EDF files have also been used to generate tabular data, in prescribed standard formats, for code comparison exercises. In such cases, the required files can be written directly by MELCOR in the specified format, with no code modifications required, if all desired quantities are available (or can be constructed) as control function arguments.

On each restart, including the initial execution of MELCOR from the restart file generated by MELGEN, a limited check is made of the data in each file to be read in. It is required that the last record before the current time match *some* record read during the previous execution; a warning is issued if the position of this record in the file has been altered. This is intended to detect obvious errors without preventing the user from performing some editing of the file if he so desires.

## 2. User Input Requirements

A full description of all external data files must be provided as part of MELGEN input. Certain elements of the description may be changed on restart through input to MELCOR.

### 2.1 MELGEN User Input

The user input for the External Data File package is described below. One set of records is required for each external data file; the actual records required differ for the different types of files. Although the input formats would permit 1000 different files to be defined, current coding limits the total number to 20 for any single calculation.

#### **EDFnnn00** – External Data File Definition Record

$001 \leq nnn \leq 999$ , nnn is the user-assigned number of the data file.

Required

This record defines a user name for the data file, the number of data channels (dependent variables), and the direction and mode of information transfer.

- (1) EDFNAM - User defined external data file name.  
(type = character\*16, default = none)
- (2) NCHAN - Number of channels (dependent variables) in each record of the file.  
(type = integer, default = none, units = dimensionless)
- (3) MODE - Direction and mode of information transfer. May be 'READ', 'WRITE', or 'PUSH', see Section 1.  
(type = character, default = none)

#### **EDFnnn01** – File Specification

$001 \leq nnn \leq 999$ , nnn is the user-assigned number of the data file.

Required

This record defines the name by which the file is known to the operating system. It is the character string which will be used in the FILE-FILNAM parameter when the file is opened. If FILNAM contains lower case characters which must be preserved, enclose it in single quotes (').

- (1) FILNAM - Name of file on the operating system, such as 'input47.dat'.  
(type = character\*80, default = none)

## EDF Package Users' Guide

### **EDFnnn02** – External Data File Format

$001 \leq nnn \leq 999$ , nnn is the user-assigned number of the data file.

Optional

A format may be defined for each external data file; the default is for the file to be unformatted. The field specifier for TIME in WRITE or PUSH files should provide sufficient significant digits to allow the written time values to be distinguished. The format may specify that a logical record contain more than one physical record ("line" in the file). A physical record length in excess of 133 characters may cause problems on many systems.

- (1) IFMT - Format of records in the external file. If the record is omitted or blank, the file is assumed to be unformatted. If the format contains one or more commas, it must be enclosed in single quotes. The enclosing parentheses may be included '(4E12.4)' or omitted '4E12.4'; in the latter case, the input string is limited to character\*22.  
(type = character\*24, default = " ")

### **EDFnnn03** – Time Offset for External Data File

$001 \leq nnn \leq 999$ , nnn is the user-assigned number of the data file.

Optional

To simplify communications between different codes, the zero of time in a data file need not coincide with that in the MELCOR run. The relationship is  $t_{\text{file}} = t_{\text{MELCOR}} + t_{\text{off}}$ . The offset may be positive or negative.

- (1) TIMOFF - Offset of time in the external data file relative to time in MELCOR.  
(type = real, default = 0.0, units = s)

### **EDFnnn04** – Buffer Length for READ File

$001 \leq nnn \leq 999$ , nnn is the user-assigned number of the data file.

Optional

Only a portion of the data in a READ file is stored in the EDF database. The buffer employed must contain enough records to cover the range of time from the beginning to the end of the current MELCOR timestep; the buffer size may therefore impose a limit on the permissible timestep. The minimum practical number of records in the buffer is 3; the default is 5. If the READ file contains closely spaced records, it may be desirable to increase this number to avoid unnecessarily slowing the calculation.

- (1) NBUFF - Size of READ file data buffer, in records. Must be  $\geq 3$ .  
(type = integer, default = 5, units = dimensionless)

**EDFnnn1k** – Write Increment Control for WRITE or PUSH File

$001 \leq nnn \leq 999$ , nnn is the user-assigned number of the data file.

$0 \leq k \leq Z$  is used for sequencing

Required

The output of records to a WRITE or PUSH file is controlled by the user. The control consists of (start time, time increment) pairs. It is similar to that for MELCOR output as described in the MELCOR/MELGEN EXEC Users' Guide, but *no* output is generated until the first time value has been passed. The "k" character in the record identifier is used only to distinguish records; the (start time, time increment) pairs input will be sorted by the code. At least one such pair must be input. No more than 10 pairs are permitted under current coding.

- (1) TWEDF - Time at which the following output increment takes effect.  
(type = real, default = none, units = s)
- (2) DTWEDF - Time increment between output records.  
(type = real, default = none, units = s)

Several pairs of (TWEDF, DTWEDF) may be entered on a single record; elements of a pair may not be split between two records.

**EDFnnnkk** – Channel Variables for WRITE File

$001 \leq nnn \leq 999$ , nnn is the user-assigned number of the data file.

$A0 \leq kk \leq ZZ$  is used for sequencing

Required

Values for each channel (dependent variable) of a WRITE data file are obtained by reference to control function arguments. The variables available in this way are listed in the Users' Guides for the various packages. Exactly NCHAN arguments are required; they are assigned to channels in the order of appearance on records sequenced according to the fields kk. Note that the independent variable, TIME, is automatically written as the first datum in each record.

- (1) CHARG - Control function argument to identify an element of the database. Refer to the Users' Guides for the various packages for permitted values.  
(type = character\*24, default = none)

More than one control function argument may be entered on a single record.

## 2.2 MELCOR User Input

Certain elements of the input for an external data file may be changed at a restart (this includes the first execution of MELCOR). This is intended to allow the user to replace or rename files, or to alter the frequency with which records are written to WRITE or PUSH files.

The input to MELCOR is a subset of that described for MELGEN; the permitted records are described below (the presence of any others will be treated as a fatal input error, and the calculation will not be run).

### EDFnnn01 – File Specification

The name of the file on the operating system may be changed.

### EDFnnn02 – External Data File Format

The format of the file may be changed. The user should be aware that for a WRITE or PUSH file an attempt will be made to open and read the old file under the new format in order to position it correctly. This could result in a read error if the old file exists and the old and new formats are incompatible.

### EDFnnn03 – Time Offset for External Data File

### EDFnnn1k – Write Increment Control for WRITE or PUSH File

Note that on any restart, a limited check is made for changes in a READ file. An error is assumed if the last record found on the newly connected file before the time of the restart does not match *any* record already in the buffer (read from the file previously connected). If there is a matching record but the record number is different, a warning message is issued to the output and diagnostic files.

## 3. Sensitivity Coefficients

There are no sensitivity coefficients associated with the External Data File package.

## 4. Plot Variables and Control Function Arguments

The variables in the External Data File package which may be used for plot variables and control function arguments are listed and described below. The control function arguments are denoted by a “c”, the plot variables by a “p”, within slashes (“/”) following the variable name

EDF.n.m            /cp/    Value of the  $m^{\text{th}}$  data channel in external data file n. If the data are plotted, the units will be given (by default) as "UNK" because they are not—and cannot be—known by EDF. The user may, of course, specify the correct units by defining a nondefault axis label as part of the input to the plot program.

The use of channels in WRITE of PUSH files as control function arguments should be avoided because the values available at the time the CF package is called are one timestep out of date. In most cases, up-to-date values of the underlying variables may be obtained by direct reference to control function arguments in the packages from which EDF is obtaining the data.

## 5. Example Input

An example of input for a READ file is:

```
EDF00700  MELT-SOURCE      4      READ
EDF00701  RUN27.DAT
EDF00702  5E12.5      *Time and four dependent variables
EDF00703  2600.      *Melcor time 0 corresponds to 2600 s on file
EDF00704  10        *Expand buffer to 10 data points
```

This defines an external data file with the user-specified number 7, and will cause data, consisting of time and four dependent variables, to be read from the file RUN27.DAT under FORMAT 5E12.5. The (interpolated) values of the dependent variables are available as control function arguments with the names EDF.7.1 through EDF.7.4. Space is reserved in the data buffer for 10 time points (for each dependent variable). This allows a MELCOR advancement to "step over" as many as 8 tabulated points while retaining a point before the start of the step and one after its end.

The data from a READ file may always be accessed as Control Function arguments; in general, this is the *only* method available to the user. However, in a few cases specific coding has been included in a package to access EDF data directly. At present only two packages make use of this capability:

- (1) The Control Volume Hydrodynamics (CVH) package allows a property of a time-specified (boundary) volume to be specified by direct reference to the contents of an external data file. The field 'EDF.n.m' on a CVHnnAk record (as described in the CVH Users' Guide) will result in the value of the corresponding property being extracted directly from channel m of external data file n.
- (2) The Transfer Process (TP) package can be instructed to actively construct "parcels" of debris (or of radionuclides) from data contained in a READ file specified on a

## EDF Package Users' Guide

TPINnnn01 record. These parcels are treated exactly as if they had been received from another package (e.g., from Core); they are held until called for by some other package (e.g., by CAV). This allows debris and/or radionuclides to be added to a MELCOR calculation at a rate determined by some other calculation performed by MELCOR, by another code, or by hand. The user has relatively few options in defining such a file. See the Cavity, Core, Radionuclide, and Transfer Process Users' Guides for more information.

An example of input for a WRITE file is:

```
EDF11100  SPECIAL-DATA   3    WRITE
EDF11101  `specdat.dat`
EDF11110   500.        100.
EDF11111  1000.        10.
EDF11112  5000.       1000.
EDF111AA  CVH-P.200
EDF111AB  CFVALU.4
EDF111AC  CAV-MEX-H2.3
```

This defines an external data file with the user-specified number 111, and will cause unformatted records to be written to a file specdat.dat. A record will be written every 100 s starting at 500 s, every 10 s starting at 1000 s, and every 1000 s starting at 5000 s. Each record will contain the time, the pressure in volume 200, the value of control function 4, and the total mass of hydrogen released in control volume 3.

The difference between a WRITE file and a PUSH file is that in the latter the values of NCHAN data channels must be defined *by some other package*. MELGEN will terminate with an error message unless *some* package defines all of the data channels for a PUSH file during initialization. Similarly, MELCOR will terminate without completing the timestep on any step during which a new value for one or more channels fails to be defined before the end of the advancement. Thus, specific coding must be included in some package to transmit data directly to EDF. At this time, only the Transfer Process package has the capability to push data to EDF. It can record data about material parcels transferred to TP by some "IN" process in a file for later use.

An example of input for such a PUSH file is:

```
EDF00500  MELT-EJECTION  16    PUSH
EDF00501  MELTEJ.DAT
EDF00502  5E15.7        *Each logical EDF record requires 4 lines
EDF00510  3000.          10.
```

This defines an external data file with the user-specified number 5 and, in conjunction with the record

```
TPINnnn01 WRITE 5
```

will cause formatted records to be written to a file MELTEJ.DAT every 10 s, starting at 3000 s. (This file may be used as input to define debris sources in a later MELCOR run. A second, similar file would be required to record the associated radionuclide transfers and define sources in the second run.) See the description of input record TPINnnn01 in the Transfer Process Users' Guide for more information.

## **6. Discussion of Output**

The output from the EDF package will be self-explanatory. For each file, it includes a user-defined name, the transfer mode, and the values of all data channels. The channels for WRITE files are labeled by the control function argument involved; those for PUSH channels by names defined by the PUSHing package, or by the label "UNDEFINED\*PUSHED\*DATA" if no other was defined. The channels for READ files are not labeled, as there is no way of knowing their contents.

Additional output provided for the first edit of each run gives the system name of the file, buffer information for READ files, and write increment control information for WRITE and PUSH files.

# EDF Package Users' Guide