**IMPROVED TIMESTEP-SIZE
DIAGNOSTIC EDITS FOR TRAC-P**

by

P. T. Giguere

**Technology and Safety Assessment Division
Los Alamos National Laboratory
Los Alamos, New Mexico 87545**

April 1996

RECEIVED

JUN 11 1996

OSTI

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

MASTER

DISCLAIMER

**Portions of this document may be illegible
in electronic image products. Images are
produced from the best available original
document.**

IMPROVED TIMESTEP-SIZE DIAGNOSTIC EDITS FOR TRAC-P

by

P. T. Giguere

ABSTRACT

Improvements have been made to the timestep-size selection logic diagnostic edits of the Transient Reactor Analysis Code (TRAC), specifically to the TRAC-P version. These include both a precise account of the reason for the selection for individual timesteps and thermal-hydraulic information on mesh cells that control the timestep size. The new edits can be specified by user input as a range of timestep numbers, problem time, or both. A description of the current timestep controls in effect in TRAC-P is also given.

1.0. INTRODUCTION

Reference 1 determined that the runtime for the Transient Reactor Analysis Code (TRAC), specifically, the TRAC-P version, can be improved by two general approaches:

- Run TRAC-P at a larger average timestep size for a given problem. Using fewer Newton iterations for a given timestep size also would be part of this approach.
- Use optimization techniques that make TRAC-P run faster (on a given platform) for a given timestep size and iteration count. Examples are vectorization and parallelization.

Reference 1 states that, "The timestep selection and backup logic have provided good service over the years; however, there clearly is room for improvement." The first approach can be taken by both code users and code developers. The users would attempt to improve their input models and "time domain" input, and the developers would attempt to improve TRAC-P's timestep-size selection logic. Reference 1 recommended a heuristic strategy to achieve improved timestep-size selection. For both TRAC-P users and developers, information on the code's timestep-size selection for the calculation at hand is essential. TRAC-P has for years provided data to its output files on its timestep-size selection logic; however, this information is cryptic and incomplete.

In addition to describing the improvements made to TRAC-P's diagnostics on its timestep-size selection logic, this report also provides a description of the current timestep-size controls in effect (as of TRAC-P Version 5.4.17).

2.0. NEW TIMESTEP-SIZE DIAGNOSTIC EDITS

The new diagnostic information is provided as a "timestep-size diagnostic edit" written to TRAC-P's output file TRCMSG. Printing of the new diagnostic edits is under user-input control; a range of timestep numbers may be selected, a range of problem time, or both. The improvements have been made in two areas: (1) for each timestep the precise reason for the timestep-size selection is provided, including (where appropriate) the identity of the controlling mesh cell and the controlling physical phenomenon; (2) additional thermal-hydraulic information is printed for any controlling cell.

Section 2.1 describes the user-selection of the new edits. Section 2.2 describes their format and content. Section 2.3 gives details on their programming implementation.

2.1. User-Input Control of Diagnostic Edits

The new diagnostic edits are invoked with four variables that have been added to TRAC-P's input file TRACIN, in NAMELIST group INOPTS, which comes after the problem title cards in the Main Data block of TRACIN. The edits can be invoked for a range of specified timestep numbers (NAMELIST variables TSDLS and TSDUS) or an interval of problem time (NAMELIST variables TSDLT and TSDUT). For all timesteps in these ranges, a "timestep-size diagnostic edit" is written to file TRCMSG. If both types of interval are selected, both will apply. Descriptions of the four variables are given in Table 2-1.

NAMELIST variables TSDLS and TSDUS specify lower and upper bounds on the timestep number for which edits are written, according to the value of internal TRAC-P variable NSTEP (note that TSDLS and TSDUS are integers). A calculation always starts with NSTEP initialized to 0 (including restart runs), and NSTEP is incremented by 1 at the end of a successful timestep. The current NSTEP is printed in the header of the TRCOUT large edits. Note that the diagnostic edits are written at the start of the step, before NSTEP is incremented. We have adopted the convention that there is no "timestep number 0". This convention is natural for the TRAC-P user, and corresponds to the step number written to file TRCOUT at the start of all large edits after the "time 0.0" edit (the TRAC-P developer on the other hand would be more comfortable starting at step 0).

The default values of all four variables are set to give no diagnostic edits. Note that negative values for the lower bound variables TSDLS and TSDLT in the actual coding in TRAC-P have the effect of selecting all positive timesteps and times, respectively. Therefore, when the lower bounds are -1 or -1.0, the corresponding upper bounds TSDUS and TSDUT should also be -1 or -1.0.

TABLE 2-1

NEW NAMELIST VARIABLES

Variable	Value Range	Description	Default Value
TSDLS	-1, >0	The first timestep for which detailed diagnostic information is written to file TRCMSG on the logic used to calculate that step's size. Timestep backups that occur in the course of that step will be indicated in the diagnostic for the following timestep. <u>Note:</u> TSDLS is an integer. -1 = > no timestep-size diagnostics; >0 = > timestep-size diagnostics at the selected timesteps.	-1
TSDUS	-1, >0	The last timestep for which detailed diagnostic information is written to file TRCMSG on the logic used to calculate that step's size (not including backups). <u>Note:</u> TSDUS is an integer. -1 = > no timestep diagnostics; >0 = > timestep-size diagnostics at the selected timesteps.	-1
TSDLT	-1.0, > = 0.0	The starting problem time for which detailed diagnostic information is written to file TRCMSG on the logic used to calculate timestep size. -1.0 = > no timestep-size diagnostics; > = 0.0 = > timestep-size diagnostics for the selected time range.	-1.0
TSDUT	-1.0, > = 0.0	The final problem time for which detailed diagnostic information is written to file TRCMSG on the logic used to calculate timestep size. -1.0 = > no timestep diagnostics; > = 0.0 = > timestep-size diagnostics at the selected time range.	-1.0

2.2. Format and Content of Edits

There are 25 possible mechanisms that can determine any given TRAC-P timestep size (all discussion here refers to TRAC-P Version 5.4.17). The new diagnostics monitor all 25 mechanisms; Table 2-2 summarizes them (note that some are effectively disabled). All of the parameters listed in Table 2-2, with the exception of the first two (IOFAIL and ITRPDT) and the last three (DTMIN, DTMAX, and DTSTRT), are evaluated in subroutine NEWDLT (before the PREP stage), which determines the most restrictive controller to use as a prospective timestep size for the forthcoming step. A twenty-sixth method, NAMELIST variable ICDEL, was found to be obsolete with current TRAC-P transient logic and should be removed from the code (this matter is discussed further in Section 3.2).

The exact mechanism for the timestep-size selection is provided. If for any reason more than one mechanism results in the same timestep size, this is indicated (For example, a calculation may be at the maximum timestep size DTMAX and experience an outer iteration count that would result in no timestep-size change). The timestep-size diagnostic is printed at the start of a given timestep, just after the timestep-size selection logic is executed. If a timestep undergoes one or more backups [e. g., caused by excessive Newton ("outer") iterations], this is noted in the timestep-size diagnostic edit at the start of the subsequent step. Figure 2-1 shows annotated sample TRCMSG timestep diagnostics for two TRAC-P runs, which used the LEHIGH and W4LOOPNR decks from the Los Alamos TRAC-P Standard Test Matrix. The timestep numbering convention is according to what a user sees in the present large edits after the first, "time-0," edit (there is no "0th" step). In going from one step to the next, before TRAC-P applies the various possible timestep-size limiters (e.g., fluid temperature or void fraction change), it either advances the timestep size (presently by 5%), holds it constant, or scales it back, based on the number of outer iterations taken by the previous step to converge. For example, note in the LEHIGH sample the case for step 499 where the step size has been limited by excessive vapor temperature change. The limitation logic was imposed in TRAC-P after the timestep size (variable DELT) from step 498 was advanced by 5%; therefore, $DELT_{499} = (DELT_{498})(1.05)(25.0/26.1)$, where the maximum change in vapor temperature during step 498 was 26.1 K, and 25.0 K is the limiter for vapor temperature built into the code (this is shown in Table 2-2 for Monitored Parameter DELCMX—DTVMX).

For cases where changing void fraction, liquid temperature, vapor temperature, or pressure is controlling the timestep size, the edit identifies the limiting component and mesh cell, the amount of change since the previous timestep, and provides additional physical information for the cell in question. This includes old- and new-time void fraction, pressure, vapor temperature, and liquid temperature (see Fig. 2-1). For the material Courant limit, the controlling cell and face are identified, and (1D only) the vapor and liquid velocities are printed. For structural temperature change, the controlling Heat Structure or 1D-component wall node is identified, and the magnitude of the change is printed.

TABLE 2-2
TIMESTEP-SIZE CONTROLS^a

Monitored Parameter	Description	Controlling Limit	Comments
IOFAIL	Flag for timestep backup mode	Timestep size used to start current step is reduced by factor that depends on nature of backup.	Backups of current timestep are counted and reported in edit for next step.
ITRPDT (new variable for diagnostics)	Flag for trip-controlled timestep logic	Timestep size used to start current step is possibly reduced, depending on trip-controlled time domain data.	If set on in routine TRPSET, it is reported in edit for next step.
DELVMX	1D material Courant limit (multiple thereof)	1000 x 1D material Courant limit	Multiplier always used (SETS always on for 1D hydro).
DELV3X	3D material Courant limit (or multiple thereof)	1 x (no 3D SETS) or 1000 x (3D SETS) 3D material Courant limit	Choice of 3D SETS under user control (also with option to toggle during run).
DELDMX	Heat Structure diffusion number, only effectively applied when using explicit numerics for axial conduction	<u>implicit</u> : 1.0×10^8 s (effectively disabled); <u>explicit</u> : DIFMIN, the minimum diffusion number for axial conduction required for stability of conduction solution (if $DIFMIN < 1.05 \times$ old timestep size)	NAMELIST variable NRSLV and Heat Structure input variable IAXCND determine numerics of axial conduction. DIFMIN is calculated in subroutine RODHT.

TABLE 2-2 (Cont'd)

TIMESTEP-SIZE CONTROLS^a

Monitored Parameter	Description	Controlling Limit	Comments
DELEMX	Vessel mass error	Effectively disabled; VMERMX = max. allowed relative error = 10000.0	Obsolete; left over from MOD1 Vessel numerics
DELAMX - DAU	Absolute void fraction change— increase	XDAU = 0.05	
DELAMX - DAL	Absolute void fraction change— decrease	XDAL = 0.20	
DELAMX - OAU	Absolute void fraction change—up oscillation	XOAU = 1.0; effectively disabled	
DELAMX - OAL	Absolute void fraction change— down oscillation	XOAL = 1.0; effectively disabled	
DELRMX	Maximum iterative pressure variation (VARERM)	VRMX = 0.1	
DELTIT (new variable for diagnostics)	Excessive outer iterations	OITNO > MAXITC = 5 outer iterations	See iteration count logic in NEWDLT.
DTDDI (new variable for diagnostics)	Standard timestep- size advancement	DDIx old timestep size, where DDI = constant = 1.05; OITNO < MAXITB = 4 outer iterations	See iteration count logic in NEWDLT.
DELTNC (new variable for diagnostics)	No change	MAXITB <= OITNO <= MAXITC	See iteration count logic in NEWDLT.

TABLE 2-2 (Cont'd)

TIMESTEP-SIZE CONTROLS^a

Monitored Parameter	Description	Controlling Limit	Comments
DELCMX - DTLMX	Absolute liquid temperature change	DTLMC = 20.0 K	
DELCMX - DTVMX	Absolute vapor temperature change	DTVMC = 25.0 K	
DELCMX - DPRMX	Relative pressure change	DPRMC = 0.50	
DELCMX - DTSMX	1D component wall temperature change	DTSMC = 20.0 K; transient only	Outer surface only (see Section 3.2)
DELCMX - DTRMX	Heat Structure temperature change	DTRMC = 20.0 K; transient only	Inner/outer surface in contact with fluid
DELCMX - DTFL (DTFL is new variable for diagnostics)	DELT/2 "floor"	MAX(DELCMX _{int} , DELT _{int} /2), where DELCMX _{int} and DELT _{int} are intermediate values	Most restrictive DELCMX physical limiter also printed.
DELPMX	Relative power change	FPMAX = 0.1	DELPMX is calculated in PREP stage. Backup is forced when fractional power change > 2 x FPMAX (DELPMX < 0.5 x DELT), using DELPMX for timestep size (transients only).

TABLE 2-2 (Cont'd)

TIMESTEP-SIZE CONTROLS^a

Monitored Parameter	Description	Controlling Limit	Comments
DELXMX	Valve adjustment rate	FXMAX = 0.4; GXMAX = 0.05	DELXMX is calculated in PREP stage. Backup is forced when $\text{DELXMX} < 0.5 \times$ DELXMX for timestep size.
DTMIN	TRACIN minimum timestep size for current time domain	N. A.	See Section 3.2 for use in restart runs for first timestep.
DTMAX	TRACIN maximum timestep size for current time domain	N. A.	See Section 3.2 for use in restart runs for first timestep.
DTSTRT	Optional NAMELIST variable for first timestep size	N. A.	See Section 3.2 for comments on related NAMELIST variable ICDELX.

^aTRAC-P Version 5.4.17

LEHIGH, before PREP stage (start) of timestep 499 ("old time" refers to step 497, "new time" to step 498)

***** timestep size diagnostic edit *****

at start of timestep: 499
current problem time: 7.26184 seconds
timestep used for step 498 was 0.034143 seconds

timestep planned for step 499: 0.034295
reason for selection of this timestep size:

limited by pressure or temperature change
details:

vapor temperature in component	1, level	6, cell	1 changed by	26.1 k
old-time void fraction =	9.539E-01	new-time void fraction =	9.592E-01	
old-time pressure =	1.01805E+05	new-time pressure =	1.01823E+05	
old-time vap. temp. =	6.567E+02	new-time vap. temp. =	6.828E+02	
old-time liq. temp. =	3.736E+02	new-time liq. temp. =	3.736E+02	

***** end timestep diagnostic *****

W4LOOPRN

***** timestep size diagnostic edit *****

at start of timestep: 1
current problem time: 0.00000 seconds

timestep planned for step 1: 0.010000
reason for selection of this timestep size:

first step, using namelist variable dtstrt
dtstrt = 0.010000

***** end timestep diagnostic *****

•
•

***** timestep size diagnostic edit *****

at start of timestep: 2
current problem time: 0.01000 seconds
timestep used for step 1 was 0.010000 seconds

timestep planned for step 2: 0.010500
reason for selection of this timestep size:

1.05x advancement from last step

<<<--- no limiter in effect

***** end timestep diagnostic *****

Fig. 2-1. Sample edits from LEHIGH and W4LOOPRN runs.

```

.
.
***** timestep size diagnostic edit *****

at start of timestep:      3
current problem time:      0.02050 seconds
timestep used for step    2 was    0.010500 seconds

timestep planned for step   3:    0.009751
reason for selection of this timestep size:

limited by core power change
  delpmx =    0.009751

*****      end timestep diagnostic      *****
*****
.
.
***** timestep size diagnostic edit *****

at start of timestep:    141
current problem time:    1.90897 seconds
timestep used for step  140 was    0.025947 seconds

timestep planned for step  141:    0.025957
reason for selection of this timestep size:

1000.0x material Courant limit in 1D hydro,
  component number      6, face number      2
  limit =    0.000026 seconds, delvmx =    0.025957
  liquid velocity =    52.943 m/s
  vapor velocity  =    69.515 m/s

*****      end timestep diagnostic      *****
*****<<<-- IADDED edit follows
  140      1.908969  0.025957      2      998.4341  0.8883  1      6      2

***** timestep size diagnostic edit *****
.
.

```

Fig. 2-1 (Cont). Sample edits from LEHIGH and W4LOOPRN runs.

Each edit provides the timestep size intended for the current timestep and the actual size used for the previous step. Any differences in these numbers should be accounted for by intervening timestep backups or trip control.

2.3. Programming Implementation

The new edits were implemented with update TSDIAG2, which is compatible with TRAC-P Version 5.4.17. TSDIAG2 uses subroutine FIND to provide physical information on the controlling mesh cell or face. An improvement to FIND to allow passing of a request for absolute-numbered cell-face data was developed to be used by TSDIAG2 (this facilitated locating Courant-limiting velocities); this was implemented with update FXFACE2 (originally developed by S. Jolly-Woodruff). Complete listings of TSDIAG2 and FXFACE2 are given in Appendix A.

All logic to determine the timestep-size selection and print diagnostic information is contained in new subroutine DTDIAG, which is called from subroutine TIMSTP after TIMSTP's call to timestep-size-selection routine NEWDLT. A number of new Common Block variables were added to the code to provide information to DTDIAG on the timestep selection in NEWDLT and the identity of the limiting cell (if any). A new Common Block variable was also added to flag the use of trip-controlled timestep logic in subroutine TRPSET. The TSDIAG2 header provides complete documentation on all new Common Block variables. The Fortran-Comment header of DTDIAG describes all local variables.

There is logic in subroutine DTDIAG to inform the user if the overall reason for the timestep-size selection is not found ("*** timestep control not found***"), or if the void fraction, pressure, or fluid temperature controlling cell is not identified ("***cell not found***"). If the problem ends before the upper bound of the diagnostic range, the "timestep control not found" message will be printed. If for any reason subroutine FIND fails to find the controlling location, the message "***error in finding data***" will be written (see Section 3.2 for a discussion of dump/restart issues).

3.0. TESTING THE NEW DIAGNOSTICS

3.1. Test Runs

Tests were conducted with five problems from the TRAC-P Standard Test Matrix. The tests validated that TSDIAG2 had found the correct reason for timestep-size selection and that the physical diagnostic data printed were correct. Most of the possible controlling conditions were represented in the test runs; these included the 1D Courant limit, void fraction change (increase), "DELCMX" (vapor temperature change and the DELT/2 "floor"), Heat Structure temperature change, power change, outer iteration-count logic that resulted in normal advancement by factor DDI (currently 1.05) and no change, time-domain lower and upper limits DTMIN and DTMAX, and initial NAMELIST-

TABLE 3-1
SUMMARY OF TEST RUNS

Test Deck	Comments
TFPIPE2	1D blowdown; test basic functionality. See Section 3.2 for comment on DTMIN = DTMAX for first timestep.
W4LOOPN	3D/1D plant model; steady state
W4LOOPRN	LBLOCA restart from steady state
W4LOOPRN - restart	Restart of W4LOOPRN
LEHIGH	Reflood; 3D Vessel w/1 ring, 1 sector
UPTF8B2	Cold-leg flow test; 3D Vessel (run after the completion of TSDIAG2)

specified timestep-size DTSTRT. Backups were also accurately flagged. Table 3-1 summarizes the test runs.

3.2. Errors and Additional Notes

After TSDIAG2 was completed, three errors were found:

- Values of NAMELIST variables TSDLS, TSDLT, TSDUS, and TSDUT were not echoed to output file TRCOUT. This has been fixed by update IDIAGS.
- For cases of pressure or fluid temperature change in the 3D VESSEL component, the wrong cell index was reported and passed to FIND (for VESSELS with more than one ring. This is to be fixed by update FXTSD.
- For cases of pressure change in 1D components, fractional change is reported as percent change ("DELTA/2 floor" only). This is to be fixed by update FXTSD (percent change will be written).

In the course of this effort, some points of possible confusion in the current timestep edits turned up:

- The large edit header has the statement, "time-step size was limited by component. . .at cell. . .to. . .s." Actually, the value printed is the 1D material Courant limit, which may or may not be controlling timestep size (also, because of the SETS numerics, this value is multiplied by a factor currently set at 1000 to obtain the 1D Courant limiter).

- The NAMELIST input variable IADDED controls optional output showing the progress of a calculation. The edits that show current timestep number, step size, and Courant numbers are printed by subroutine HOUT before the current timestep is completed. Therefore, there is an inconsistency between these edits and the large edits, which are done after the timestep is finished and the step number and problem time are updated. This is shown in the sample output in Fig. 2-1.

Comments are also in order on the logic used to select an initial timestep size to start a calculation. When TRAC-P initializes a run from a restart dump, it will by default use the timestep size in that dump as the old-time size and proceed to the usual advancement/control logic in subroutine NEWDLT. This avoids discontinuities in a linked series of restarts (steady state or transient). When starting a transient from a steady-state dump, it is often desirable to start with a smaller timestep size. This can be achieved with the use of the NAMELIST variable DTSTRT, which will override the restart-dump value and subsequent NEWDLT logic. Another way is to set up an initial input time domain with DTMIN and DTMAX set to the desired value. For **non-restart** runs, TRAC-P will use DTMIN for the first step (the TFPIPE2 test problem, which is not a restart, has an initial DTMIN = DTMAX time domain; this causes the new diagnostics to indicate that both variables controlled the first step size; in fact, DTMIN would always initialize the run). NAMELIST variable ICDELTA is an obsolete feature that is not supported by the current diagnostics. TRAC-P determines if it is calculating the first step of a restart from the value of variable DSTEP, and the logic on ICDELTA is not needed (also, Section VI of the User's Guide incorrectly has "ICDELTA = 0 forces. . .". It should read "ICDELTA=1 forces . . .").

Certain location information used by the diagnostics is not currently written to the dump file (e. g., the Courant-limiting location). In these situations, for the first step of a restart, the message "***error in finding data***" will be written.

In the course of testing with the LEHIGH deck, we noticed in one case that the reported vapor temperature change did not exactly correspond to the values printed in the TRCOUT large edits. This happened because the timestep in question backed up from subroutine POST. The most restrictive limiter, which by chance happened on the first pass, was used for future timestep control logic (and printed in the diagnostic edit), but this was not the final temperature change when the step was completed. This could cause overly restrictive timestep control. Also, it was noted that the DELCMX limiters (DTLMX, etc.) were not written to dump file TRCDMP, which could cause an inexact restart. Both situations have been fixed by update DELCMX1. The technical notes header comments subroutine DTDIAG on timestep backups are therefore no longer current; they will be deleted by updated FXTSD.

3.3. Future Improvements

Two areas for additional development are:

- Improved, user-friendly, timestep backup diagnostics. The timestep-size diagnostics presented here do inform the user when (and how many) backups occur (as shown in the LEHIGH sample in Fig. 2-1), but do not identify the reasons for the backups. Of course a great deal of information on backups is already written to file TRCMSG, but it would be very desirable to organize it into the framework of the improved timestep-size diagnostics.
- Presently, only the most limiting cell/variable is identified in the diagnostics. This could be broadened to include other potential limiters in a user-friendly diagnostic.

4.0. SUMMARY

TRAC-P's runtime can be decreased by (1) reducing the average timestep size for a given calculation, and (2) use of optimization techniques such as vectorization and parallelization. Improved diagnostics on the code's timestep-size selection have been developed and incorporated in a recent version of TRAC-P. These diagnostics are expected to be very useful for the first approach to runtime improvement.

Acknowledgment

The author is grateful to Susan Jolly-Woodruff for valuable discussions, and for developing the initial version of update FXFACE2.

Reference

1. P. T. Giguere, "Plan for Speedup of TRAC Runtime Phase One: Timestep Size," Los Alamos National Laboratory document LA-UR-95-3963 (November 1995).

DRAFT

APPENDIX A

LISTING OF UPDATE TSDIAG2

```
*ident tsdiag2
*/ -----Update Header Rev. 02-01-96-----
*/
*/   File & Update Ident name (must be the same): tsdiag2
*/   TSHOOTER identifier: N.A.
*/
*/   Code developer: Paul Giguere           Date developed: 03/01/96
*/   Acknowledgements: Susan Jolly-Woodruff
*/   Update reviewer:                      Date reviewed:
*/
*/   Update to be applied to MOD2 _X_ or other:
*/
*/   CFS file path: /084220/updates
*/
*/   Purpose of update:  Add additional diagnostics for TRAC's timestep
*/                       selection logic.
*/
*/   Basis for update:  The diagnostics are written to file TRCMMSG.
*/                       The edits are under control of four new
*/                       NAMELIST variables; default is no diagnostics.
*/                       For a given run, the user may select a time
*/                       range or a timestep range for the edits, or
*/                       both.  If both ranges are selected, both
*/                       will apply (a union will be done).
*/
*/                       There is a very small but non-zero chance that
*/                       more than one of the timestep controllers may
*/                       give the same result.  In such a case each is
*/                       reported (this is not the case in the current
*/                       large edits).
*/
*/                       If the current timestep undergoes one or more
*/                       backups, such is indicated in the diagnostic
*/                       for the subsequent timestep.  Note that a
*/                       backup may make the change in, e.g., fluid
*/                       temperature, that controlled the step size
*/                       not be consistent with what finally appears
*/                       in TRCOUT for the new and old timesteps.
*/                       This is not an error, but only a reflection
*/                       of the timestep control logic.
*/
*/                       When pressure, liquid temperature, vapor
*/                       temperature, or void fraction change in Tee
*/                       side tube controls timestep size, absolute
*/                       cell index is reported (including "phantom
*/                       cell").  When Courant number controls, only
*/                       velocities for 1D-controlling face are given.
*/
*/   Dependencies on other updates: Needs update fxface2.
*/
*/   Justification of non-ANSI coding: N.A.
*/
*/   Does this update generate new unit labels? YES _X_ NO ____
*/   If yes, path and name of LABNEW file: /084220/updates/labtsdiag2
*/
*/   Does the TRAC I/O change because of this update? YES _X_ NO ____
*/   TRAC I/O includes input, dump, output, and graphics files.  If
```

DRAFT

```

*/      yes, then submit the necessary manual changes.
*/
*/      See User's Guide changes below.
*/
*/      Does this change require that any of the TRAC support codes, e.g.
*/      EXTRACT, GOCNVRT, or EXCON, be changed?  YES ____ NO X
*/      If yes, then include a description of the necessary changes.
*/      This description will be passed onto the custodian of the TRAC
*/      support codes.
*/
*/      Does the TRAC User's Guide, Theory Manual, or Programmer's Guide
*/      change because of this update?  YES X NO ____
*/      If yes, then submit the necessary manual changes.
*/
*/      *****
*/      User's Guide changes:
*/      *****
*/      Add the following to Chapter VI ("INPUT-DECK PREPARATION"),
*/      Section 1 ("Main Data"), in "NAMELIST Data Cards for
*/      Group INOPTS", in alphabetical order:
*/
*/      Value                                     Default
*/      Variable Range Description Value
*/
*/      NOTE: Detailed diagnostics for TRAC's timestep size selection
*/      can be invoked for a range of specified timestep numbers
*/      (NAMELIST variables TSDLS and TSDUS) or an interval of
*/      problem time (TSDLT and TSDUT).  If both types of interval
*/      are selected, both will apply.  Descriptions of these four
*/      variables follow in alphabetical order.  Note that negative
*/      values for the lower bound variables TSDLS and TSDLT
*/      in the actual coding in TRAC have the effect of selecting
*/      all positive timesteps and times, respectively.  Therefore,
*/      when the lower bounds are -1 or -1.0, the corresponding
*/      upper bounds TSDUS and TSDUT should also be -1 or -1.0.
*/
*/      TSDLS      -1,>0      The first timestep for which          -1
*/                               detailed diagnostic information is
*/                               written to file TRCMSG on the logic
*/                               used to calculate that step's size.
*/                               Timestep backups that occur in the
*/                               course of that step will be indicated
*/                               in the diagnostic for the following
*/                               timestep.
*/                               Note: TSDLS is an integer.
*/                               -1 = no timestep size diagnostics;
*/                               >0 = timestep size diagnostics at
*/                               the selected timesteps.
*/
*/      TSDLT      -1.,       The starting problem time for which      -1.
*/                               >=0. detailed diagnostic information is
*/                               written to file TRCMSG on the logic
*/                               used to calculate timestep size.
*/                               -1. = no timestep size diagnostics;
*/                               >=0.= timestep size diagnostics for
*/                               the selected time range.
*/
*/      TSDUS      -1,>0      The last timestep for which          -1
*/                               detailed diagnostic information is
*/                               written to file TRCMSG on the logic
*/                               used to calculate that step's size (not
*/                               including backups).
*/                               Note: TSDUS is an integer.
*/                               -1 = no timestep diagnostics;

```

DRAFT

```

*/              >0 = timestep size diagnostics at
*/              the selected timesteps
*/
*/
*/      TSDUT      -1.,      The final problem time for which      -1.
*/              >=0.      detailed diagnostic information is
*/              written to file TRCMSG on the logic
*/              used to calculate timestep size.
*/              -1. = no timestep diagnostics;
*/              >=0.= timestep size diagnostics at
*/              the selected time range.
*/
*/      *****
*/      Programmer's Guide changes:
*/      *****
*/
*/      Appendix B:
*/
*/      A new subroutine has been added:      DTDIAG
*/              calls FIND
*/              called by TIMSTP
*/
*/      Appendix D:
*/
*/      A new common block has been added: DTINFO
*/
*/      COMDECK DTINFO
*/
*/      NOTE: With the exceptions of variables DDI and DELV3X,
*/              variables in Common Block DTINFO are only used
*/              to pass information to the timestep size
*/              diagnostic subroutine DTDIAG.
*/
*/      common/dtinfo/ ddi      ,deltit,deltnc,delv3x,dprsv ,
*/              dtddi ,dtfl ,dtlsv ,dtrsv ,dtssv ,
*/              dtvsv ,prmxsv,svdelc,tlmxsv,trmxsv,
*/              tsmxsv,tvmxsv
*/
*/      common/dtinfo/ idcdal,idcdau,idcoal,idcoau,itrdpt,
*/              izdal ,izdau ,izoal ,izoau ,kccmxt
*/
*/      REAL VARIABLES:
*/
*/      DDI:      Timestep size-advancement factor.
*/      DELTIT: Saves timestep size from iteration-count logic
*/              that results in scaling back of step size.
*/      DELTNC: Saves timestep size from iteration-count logic
*/              that results in no change to step size.
*/      DELV3X: Timestep size from 3D material Courant limit
*/              (or multiple thereof).
*/      DPRSV: Saves pressure-change DELCMX limiter.
*/      DTDDI: Saves timestep size from iteration-count logic
*/              that results in advancement by factor DDI.
*/      DTFL: Saves value of 0.5*DELT "floor" used in DELCMX
*/              timestep size control logic.
*/      DTL SV: Saves liquid temperature-change DELCMX limiter.
*/      DTRSV: Saves Heat Structure temperature-change DELCMX
*/              limiter.
*/      DTSSV: Saves 1D wall temperature-change DELCMX limiter.
*/      DTVSV: Saves vapor temperature-change DELCMX limiter.
*/      PRMXSV: Saves maximum pressure change (DPRMX) used in
*/              DELCMX logic.
*/      SVDEL C: Saves current value of DELCMX (before 0.5*DELT
*/              "floor" logic applied).
*/      TLMXSV: Saves maximum liquid temperature-change (DTLMX)
*/              used in DELCMX logic.

```

DRAFT

```

*/      TRMXSV: Saves maximum Heat Structure temperature-change
*/      (DTRMX) used in DELCMX logic.
*/
*/      TSMXSV: Saves maximum 1D wall temperature-change (DTSMX)
*/      used in DELCMX logic.
*/
*/      TVMXSV: Saves maximum vapor temperature-change (DTVMX)
*/      used in DELCMX logic.
*/
*/
*/      INTEGER VARIABLES:
*/
*/      IDCDAL: Cell number in 3D level for void fraction-change
*/      timestep size control variable DAL.
*/
*/      IDCDAU: Cell number in 3D level for void fraction-change
*/      timestep size control variable DAU.
*/
*/      IDCOAL: Cell number in 3D level for void fraction-change
*/      timestep size control variable OAL.
*/
*/      IDCOAU: Cell number in 3D level for void fraction-change
*/      timestep size control variable OAU.
*/
*/      ITRPDT: Flag to indicate trip-controlled timestep size
*/      logic used in subroutine TRPSET.
*/
*/      IZDAL: 3D level number for void fraction-change timestep
*/      size control variable DAL.
*/
*/      IZDAU: 3D level number for void fraction-change timestep
*/      size control variable DAU.
*/
*/      IZOAL: 3D level number for void fraction-change timestep
*/      size control variable OAL.
*/
*/      IZOAU: 3D level number for void fraction-change timestep
*/      size control variable OAU.
*/
*/      KCCMXT: Absolute cell-face index for 1D material Courant
*/      limit logic.
*/
*/
*/      Variables have been added to two common blocks: DLIM and
*/
*/      ERRCON
*/
*/
*/      Add to COMDECK DLIMIT (COMMON/DLIM/):
*/
*/      real variables SVMAXT,SVMXT3
*/
*/      SVMAXT: Saves reciprocal of timestep size from 1D material
*/      Courant limit (variable VMAXT). Multiplier not
*/      included.
*/
*/      SVMXT3: Saves reciprocal of timestep size from 3D material
*/      Courant limit (variable VMAXT3). Multiplier not
*/      included.
*/
*/
*/      Add to COMDECK ERRCON:
*/
*/      real variables TSDLT ,TSDUT
*/
*/      TSDLT: If TSDLT .LE. ETIME .LE. TSDUT, detailed timestep
*/      size diagnostic edits are printed to file TRCMSG.
*/
*/      TSDUT: If TSDLT .LE. ETIME .LE. TSDUT, detailed timestep
*/      size diagnostic edits are printed to file TRCMSG.
*/
*/
*/      integer variables TSDLS ,TSDUS
*/
*/
*/      TSDLS: If TSDLS-1.LE.NSTEP.LE.TSDUS-1, detailed timestep
*/      size diagnostic edits are printed to file TRCMSG.
*/
*/      TSDUS: If TSDLS-1.LE.NSTEP.LE.TSDUS-1, detailed timestep
*/      size diagnostic edits are printed to file TRCMSG.
*/
*/
*/      *****end of documentation changes*****
*/
*/
*/      NAME OF
*/      SUBROUTINE

```

DRAFT

```

*/      OR COMDECK
*/      CHANGED      DESCRIPTION OF CHANGE
*/
*/      DLIMIT      Add variables for 1D and 3D material Courant limits.
*/
*/      ERRCON      Add variables for timestep number and problem time
*/                  ranges for control of timestep diagnostic edits.
*/
*/      BKSTB3      Set variables that identify Vessel level and cell
*/                  numbers for most-limiting void fraction changes.
*/
*/      BLKDAT      Initialize 4 variables that specify timestep and time
*/                  ranges for timestep size diagnostic edits (default
*/                  to no edits), 3 variables that pass timestep size
*/                  information from NEWDLT to DTDIAG, a flag for trip-
*/                  controlled timestep size (ITRPDT), and the existing
*/                  timestep size advancement variable DDI (DATA stmt
*/                  moved from NEWDLT because DDI is now in a COMMON blk).
*/
*/      EVALDF      Ensure that variables for Vessel level number (LTLTST,
*/                  etc.) are set only for Vessels, and set to 0
*/                  otherwise. This information is needed by new routine
*/                  DTDIAG for correct use of routine FIND.
*/
*/      INPUT      Add 4 variables that specify timestep and time ranges
*/                  for timestep size diagnostic edits to NAMELIST group
*/                  INOPTS.
*/
*/      NAMLST      Add input-error checks for new NAMELIST variables
*/                  TSDLS, TSDLT, TSDUS, and TSDUT.
*/
*/      NEWDLT      Provide information for the timestep size diagnostic
*/                  edits done by routine DTDIAG. Because timestep size
*/                  advancement variable DDI is now in a common block,
*/                  move its DATA statement from NEWDLT to BLKDAT.
*/
*/      PLEN3      Set variables IZDAU, IZDAL, IZOAU, and IZOAL to 0.
*/                  Used by DTDIAG for correct (non-Vessel) use of FIND.
*/
*/      POSTER      Set variables IZDAU, IZDAL, IZOAU, and IZOAL to 0.
*/                  Used by DTDIAG for correct (non-Vessel) use of FIND.
*/
*/      TF1DS1      Add variable KCCMXT for Courant-limiting face. Used
*/                  by DTDIAG (existing KCCMX=J+1-JSTART, which is
*/                  relative to start of Tee side tube).
*/
*/      TIMSTP      Call new subroutine DTDIAG for timestep size
*/                  diagnostic edits when in range of timestep numbers
*/                  or problem times specified by user.
*/
*/      TRPSET      Add logic for flag ITRPDT, for case where a trip
*/                  controls timestep size (used in subsequent step by
*/                  DTDIAG).
*/
*/      DTINFO      New comdeck with variables describing timestep size
*/                  selection logic; some are existing local variables,
*/                  some are new (see changes to Programmer's Guide).
*/
*/      DTDIAG      New subroutine to provide timestep size diagnostic
*/                  edits to file TRCMSG.
*/
*/      -----
*/
*/      *compile dlimit,errcon

```

DRAFT

```

*compile bkstb3,blkdat,input,namlst,newdlt,plen3,poster,tflds1,timstp
*compile trpset
*/
*delete dlimit.5                                     dlimit
    * dtbkup,fpmax ,fxmax ,gxmax ,svmaxt,svmxt3
*/
*delete errcon.5                                     errcon
    3 dtvu ,dtvum ,timdl ,timdu ,
    4 tsdlt ,tsdut
*delete errcon.12                                    errcon
    6 nsdl ,nsdu ,tsdls ,tsdus
    integer tsdls,tsdus
*/
*insert bkstb3.19                                     bkstb3
*call dtinfo
*insert bkstb3.222                                    bkstb3
    izdau=izz
    idcdau=it+(ir-1)*ntsx
*insert bkstb3.226                                    bkstb3
    izdal=izz
    idcdal=it+(ir-1)*ntsx
*insert bkstb3.230                                    bkstb3
    izoau=izz
    idcoau=it+(ir-1)*ntsx
*insert bkstb3.234                                    bkstb3
    izeal=izz
    idcoal=it+(ir-1)*ntsx
*/
*insert fprad.1                                       blkdat
*call dtinfo
*insert blkdat.288                                    blkdat
    data tsdls,tsdus,tsdlt,tsdut/-1,-1,-1.0,-1.0/
    data ddi,deltit,dtddi,deltnc/1.05,-1.0,-1.0,-1.0/
    data itrpd/0/
*/
*insert evaldf.12                                     evaldf
*call holerith
*delete evaldf.39                                     evaldf
    if (type .eq. vsslh) then
        ltlst=iz
    else
        ltlst=0
    endif
*delete evaldf.54                                     evaldf
    if (type .eq. vsslh) then
        ltvst=iz
    else
        ltvst=0
    endif
*delete evaldf.68                                     evaldf
    if (type .eq. vsslh) then
        ltmst=iz
    else
        ltmst=0
    endif
*delete evaldf.84                                     evaldf
    if (type .eq. vsslh) then
        lprtst=iz
    else
        lprtst=0
    endif
*/
*insert units.3018                                    input
    *                                     tsdls ,tsdlt ,tsdus ,tsdut ,

```

DRAFT

```

*/
*delete namlst.43                                namlst
c      timdl, timdu, tsdlt, tsdus
c      nsdl, nsdu, tsdls, tsdus
*insert namlst.908                                namlst
c
c      check the timestep numbers to begin and end detailed
c      timestep size diagnostic edits
c
      if((tsdls.lt.-1).or.(tsdus.lt.-1).or.(tsdls.eq.0).or.
1      (tsdus.eq.0).or.(tsdls.gt.tsdus)) then
        write(imout,700) tsdls,tsdus
        write(iout,700) tsdls,tsdus
*if -def,unicos,1
        write(itty,700) tsdls,tsdus
*if def,unicos,1
        write(*,700) tsdls,tsdus
700      format(/' *namlst* namelist variables tsdls =',i9,
1      ' and tsdus =',i9/11x,'should be -1 or positive in value, ',
2      'with tsdls.le.tsdus')
        if (tsdls.lt.-1) tsdls=-1
        if (tsdus.lt.-1) tsdus=-1
        if (tsdls.gt.tsdus) tsdls=tsdus
        if (tsdls.eq.0) tsdls=1
        if (tsdus.eq.0) tsdus=1
        write(imout,705) tsdls,tsdus
        write(iout,705) tsdls,tsdus
*if -def,unicos,1
        write(itty,705) tsdls,tsdus
*if def,unicos,1
        write(*,705) tsdls,tsdus
705      format(11x,'trac has reset tsdls =',i9,' and tsdus =',i9)
        call error(2,'*namlst* tsdls,tsdus have out-of-range values',5)
      endif
c
c      check the problem time range to begin and end detailed
c      timestep size diagnostic edits
c
      if(((tsdlt.ne.-1.0).and.(tsdlt.lt.0.0)).or.
1      ((tsdut.ne.-1.0).and.(tsdut.lt.0.0)).or.
2      (tsdut.lt.tsdlt)) then
        write(imout,720) tsdlt,lus,tsdut,lus
        write(iout,720) tsdlt,lus,tsdut,lus
*if -def,unicos,1
        write(itty,720) tsdlt,lus,tsdut,lus
*if def,unicos,1
        write(*,720) tsdlt,lus,tsdut,lus
720      format(/' *namlst* namelist variables tsdlt =',1p,e14.6,a,
1      ' and tsdut =',e14.6,a/11x,'should be -1.0, 0.0, or',
2      ' positive in value, with tsdlt.le.tsdut')
        if (tsdlt.lt.-1.0) tsdlt=-1.0
        if (tsdut.lt.-1.0) tsdut=-1.0
        if (tsdlt.gt.tsdut) tsdlt=tsdut
        write(imout,725) tsdlt,tsdut
        write(iout,725) tsdlt,tsdut
*if -def,unicos,1
        write(itty,725) tsdlt,tsdut
*if def,unicos,1
        write(*,725) tsdlt,tsdut
725      format(11x,'trac has reset tsdlt =',i9,' and tsdut =',i9)
        call error(2,'*namlst* tsdlt,tsdut have out-of-range values',5)
      endif
*/
*insert newdlt.4                                newdlt

```


DRAFT

```

c-----subroutine dtdiag must be kept current with changes to
c-----this subroutine and its common block variables
*insert newdlt.12                                newdlt
*call dtinfo
*delete fxnst1.25                                newdlt
      data maxitb/4/,maxitc/5/
*delete newdlt.59,newdlt.60                       newdlt
c-----variables deltit, dtddi, and deltno are used by routine
c-----dtdiag for timestep size diagnostic edits
      if(oitno.gt.maxitc) then
        deltno=deltno*float(maxitc)/float(oitno)
        deltit=deltno
        dtddi=-1.0
        deltno=-1.0
      endif
      if(oitno.lt.maxitb) then
        deltno=deltno*ddi
        dtddi=deltno
        deltit=-1.0
        deltno=-1.0
      endif
      if(oitno.ge.maxitb .and. oitno.le.maxitc) then
        deltno=deltno
        deltit=-1.0
        dtddi=-1.0
      endif
*delete newdlt.81                                newdlt
      if(delcmx.ne.1.0e+8) then
c-----save current values of delcmx and 0.5*delt for timestep
c-----size diagnostic edits
        svdelc=delcmx
        dtfl=0.5*delt
c
        delcmx=amax1(delcmx,0.5*delt)
c
c-----save other delcmx limiters for timestep size diagnostics
        dtlsv=dtlms
        tlmxsv=dtlms
        dtvsv=dtvms
        tvmxsv=dtvms
        dprsv=dprms
        prmxsv=dprms
        if(istdy.eq.0) then
          dtssv=dtssms
          tsmxsv=dtssms
          dtrsv=dtrms
          trmxsv=dtrms
        endif
      endif
*insert newdlt.164                                newdlt
c-----save vmaxt and vmaxt3 for timestep size diagnostic edits
c
      svmaxt=vmaxt
      svmaxt3=vmaxt3
c
*/
*insert plen3.8                                    plen3
*call dtinfo
*insert plen3.51                                    plen3
      izdau=0
*insert plen3.55                                    plen3
      izdal=0
*insert plen3.59                                    plen3
      izoau=0

```

DRAFT

```

*insert plen3.63                                plen3
      izoal=0
*/
*insert poster.9                                poster
*call dtinfo
*insert poster.167                              poster
      izdau=0
*insert poster.171                              poster
      izdal=0
*insert poster.175                              poster
      izoau=0
*insert poster.179                              poster
      izoal=0
*/
*insert tflds1.37                              tflds1
*call dtinfo
*insert tflds1.930                              tflds1
      kccmxt=j
*/
*insert timstp.27                               timstp
*call errcon
*insert timstp.273                             timstp
C
C-----check for edit of detailed timestep size diagnostics
C
      if ((nstep.ge.tsdl-1 .and. nstep.le.tsdu-1) .or.
1      (etime.ge.tsdl .and. etime.le.tsdu)) then
      call dtdiag(iofail)
      endif
C-----
*/
*insert trpset.14                               trpset
*call dtinfo
*insert trpset.770                             trpset
C-----set timestep size diagnostic flag itrpdt (used by dtdiag)
      itrpdt=1
*/
*addfile,,dskptp
*comdeck dtinfo
      common/dtinfo/ ddi ,deltit,deltnc,delv3x,dprsv ,
$                  dtddi ,dtfl ,dtlsv ,dtrsv ,dtssv ,
$                  dtvsv ,prmxsv,svdelc,tlmxsv,trmxsv,
$                  tsmxsv,tvmxsv
      common/dtinfo/ idcdal,idcdau,idcoal,idcoau,itrpdt,
$                  izdal ,izdau ,izoal ,izoau ,kccmxt
*/
*addfile,,drodl
*deck dtdiag
      subroutine dtdiag (iofail)
C
C
C
C          P U R P O S E   O F   S U B R O U T I N E
C
C          Determine reason for selection of timestep size to be used
C          at start of current step. Write same to TRCMSG file with
C          additional diagnostic information. Also count any timestep
C          backups that occur during current step and write backup note
C          in diagnostic edit for following step.
C          Subroutine DTDIAG is called from subroutine TIMSTP, under
C          control of NAMELIST variables TSDLS, TSDUS, TSDLT, and TSDUT.
C
C

```

DRAFT

TECHNICAL NOTES

Allows for possibility of more than one controller giving same result (there are no "RETURNS" or "GO TOs" after a timestep size match is found). If for any reason no match is found, such is written to TRCMMSG.

Note that a backup may make the change in, e.g., fluid temperature, that controlled the step size not be consistent with what finally appears in TRCOUT for the new and old timesteps. This is not an error, but only a reflection of the timestep control logic.

INPUT ARGUMENTS

IOFAIL Flag that indicates current timestep is in backup mode.
INTEGER IOFAIL

OUTPUT ARGUMENTS

none

*** **

COMMON BLOCKS

*call blankcom
*call chgalp
*call contrllr
*call dlimit
*call dtinfo
*call ediff
*call emot
*call iounits
*call iterstat
*call rsparm

LOCAL VARIABLES

DTFLAG Indicates if overall timestep control has been found.
INTEGER DTFLAG

PTFLAG Indicates if pressure/temperature controlling cell found.
INTEGER PTFLAG

AFLAG Indicates if void fraction controlling cell found.
INTEGER AFLAG

BKPCTR Counts number of backups for current step.
INTEGER BKPCTR

DRAFT

```
C      ID           Cell/level/face numbers passed to subroutine FIND.  
C                 INTEGER ID(3)  
  
C  
C      MXVARS       Maximum number of data items that can be asked for in  
C                   single call to subroutine FIND.  
C                   INTEGER MXVARS  
C                   PARAMETER (MXVARS = 10)  
  
C  
C      NVARS        Actual number of data items asked for in call to  
C                   subroutine FIND.  
C                   INTEGER NVARS  
  
C  
C      VARS         Data identifiers passed to subroutine FIND.  
C                   CHARACTER*8 VARS(MXVARS)  
  
C  
C      NTYPE        Node-type indicator passed to subroutine FIND.  
C                   CHARACTER*8 NTYPE  
  
C  
C      LIST         Indices into A-array returned by subroutine FIND.  
C                   INTEGER LIST(MXVARS)  
  
C  
C      INC          Increment value returned by subroutine FIND.  
C                   Currently not used here.  
C                   INTEGER INC  
  
C  
C      CS           Completion-status code returned by subroutine FIND.  
C                   INTEGER CS  
  
C  
C      DAUTMP       Scratch variables used to determine specific cause,  
C      DALTMP       component, and cell when void fraction is controller.  
C      OAUTMP       Use data in COMMON block CHGALP.  
C      OALTMP  
C      ATMP  
C  
C                  REAL DAUTMP,DALTMP,OAUTMP,OALTMP,ATMP  
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>  
C  
C  
C      data bkpctr/0/  
C      save bkpctr  
  
C  
C-----see if current step is in backup mode  
C-----if so, increment bkpctr and RETURN  
C      if (iofail .eq. 1) then  
C          bkpctr = bkpctr + 1  
C          return  
C      endif  
  
C  
C      dtflag = 0  
  
C  
C      write (imout,100)  
100 format (/,'*****  
$ /'***** timestep size diagnostic edit *****')  
    write (imout,200) nstep+1,etime  
200 format (/,'at start of timestep: ',i4,',','current problem time: ',  
$ f11.5,' seconds')  
    if (nstep .gt. 0) then  
        write (imout,210) nstep,odelt  
210   format ('timestep used for step ',i4,', was ',f11.6,' seconds')  
        if (bkpctr .gt. 0) then  
            write (imout,220) nstep,bkpctr  
220   format (('***note: step ',i4,', backed up ',i2,' times'))  
            bkpctr = 0  
        endif
```

DRAFT

```

        if (itrpdt .gt. 0) then
            write (imout,230) nstep
230      format ('***note: step ',i4,' invoked trip-controlled ',
$          'timestep logic')
            itrpdt = 0
        endif
    endif
    write (imout,250) nstep+1,delt
250 format ('timestep planned for step ',i4,': ',f11.6,/,
$          'reason for selection of this timestep size:')

c
c-----1D material Courant limit (multiple thereof)
    if (delt .eq. delvmx) then
        write (imout, 400) csf*csfld,iccmx,kccmxt,1.0/svmxt,delvmx
400      format (/,f6.1,'x material Courant limit in 1D hydro',/,/,10x,
$          'component number ',i4,', face number ',i3,/,10x,
$          'limit = ',f10.6,' seconds,', delvmx = ',f10.6)

c
c-----variables for subroutine find
        nvars = 2
        vars(1) = 'vln'
        vars(2) = 'vvn'

c
c-----find and write velocities at face kccmx, using negative 3rd arg.
c-----to tell routine find index is for face ('lface' is arbitrary,
c-----wrt 'rface'; kccmx is actual face index, right or left)
        id(1) = -kccmxt
        id(2) = 0
        id(3) = 0
        ntype = 'lface'
        call find (iccmx,ntype,id,nvars,vars,list,inc,cs)
        if (cs .eq. 0) then
            write (imout,410) a(list(1)),a(list(2))
410      format (10x,'liquid velocity = ',f10.3,' m/s',
$          /,10x,'vapor velocity = ',f10.3,' m/s')
        else
            write (imout,8500)
        endif
        dtflag = 1
    endif

c
c-----3D material Courant limit (or multiple thereof)
    if (delt .eq. delv3x) then
        write (imout, 450) csf*csf3d,iccmx,kccmx,1.0/svmxt3,delv3x
450      format (/,f6.1,'x material Courant limit in 3D hydro',/,/,10x,
$          'component number ',i4,', face number ',i3,/,10x,
$          'limit = ',f10.6,' seconds,', delv3x = ',f10.6)
        dtflag = 1
    endif

c
c-----limit caused by heat structure diffusion number
    if (delt .eq. deldmx) then
        write (imout,500) deldmx
500      format ('limited by heat structure diffusion number',
$          /,10x,'deldmx = ',f11.6)
        dtflag = 1
    endif

c
c-----limit caused by vessel mass error
    if (delt .eq. delemx) then
        write (imout,600) delemx
600      format ('limited by vessel mass error',
$          /,10x,'delemx = ',f11.6)
        dtflag = 1
    endif

```

DRAFT

```

endif
c
c-----limit caused by absolute void fraction change
  if (delt .eq. delamx) then
    write (imout,700)
    700   format (/'limited by absolute void fraction change')
    dtflag = 1
c-----determine limiting component and cell, using information
c-----in common block chgalp, and print same (allow for small
c-----possibility of more than one controller)
    aflag = 0
    if (dau .gt. xdau) then
      dautmp = xdau/dau
    else
      dautmp = 1.0
    endif
    if (abs(dal) .gt. xdal) then
      daltmp = xdal/abs(dal)
    else
      daltmp = 1.0
    endif
    if (oau .gt. xoau) then
      oautmp = xoau/oau
    else
      oautmp = 1.0
    endif
    if (abs(oal) .gt. xoal) then
      oaltmp = abs(xoal/oal)
    else
      oaltmp = 1.0
    endif
    atmp = amin1(dautmp,daltmp,oautmp,oaltmp)
c
c-----variables for subroutine find
  nvars = 8
  vars(1) = 'alp'
  vars(2) = 'alpn'
  vars(3) = 'p'
  vars(4) = 'pn'
  vars(5) = 'tv'
  vars(6) = 'tvn'
  vars(7) = 'tl'
  vars(8) = 'tln'
  ntype = 'cell'
c
  if (atmp .eq. dautmp) then
    aflag = 1
    if (izdau .eq. 0) then
c-----1d component (or plenum)
      write (imout,710) dau,ndau,jdau
      710   format (10x,'void fraction increased by ',f5.3,
        $         ' in component ',i4,', cell ',i3)
      id(1) = jdau
      id(2) = 0
      id(3) = 0
    else
c-----3d component
      write (imout,711) dau,ndau,izdau,idcdau
      711   format (10x,'void fraction increased by ',f5.3,
        $         ' in component ',i4,', level ',i3,', cell ',i3)
      id(1) = izdau
      id(2) = 0
      id(3) = idcdau

```

DRAFT

```

endif
call find(ndau, ntype, id, nvars, vars, list, inc, cs)
if (cs .eq. 0) then
  write (imout, 740) a(list(1)), a(list(2)), a(list(3)),
$   a(list(4)), a(list(5)), a(list(6)), a(list(7)), a(list(8))
  else
    write (imout, 8500)
  endif
endif

c
  if (atmp .eq. daltmp) then
    aflag = 1
    if (izdal .eq. 0) then
c-----1d component (or plenum)
      write (imout, 730) dal, ndal, jdal
730   format (10x, 'void fraction decreased by ', f5.3,
$       ' in component ', i4, ', cell ', i3)
      id(1) = jdal
      id(2) = 0
      id(3) = 0
    else
c-----3d component
      write (imout, 731) dal, ndal, izdal, idcdal
731   format (10x, 'void fraction decreased by ', f5.3,
$       ' in component ', i4, ', level ', i3, ', cell ', i3)
      id(1) = izdal
      id(2) = 0
      id(3) = idcdal
    endif
    call find(ndal, ntype, id, nvars, vars, list, inc, cs)
    if (cs .eq. 0) then
      write (imout, 740) a(list(1)), a(list(2)), a(list(3)),
$   a(list(4)), a(list(5)), a(list(6)), a(list(7)), a(list(8))
    else
      write (imout, 8500)
    endif
740   format (10x, 'old-time void fraction = ', 1pe10.3, 3x,
$   'new-time void fraction = ', 1pe10.3, '/', 10x,
$   'old-time pressure = ', 1pe12.5, 6x,
$   'new-time pressure = ', 1pe12.5, '/', 10x,
$   'old-time vap. temp. = ', 1pe10.3, 6x,
$   'new-time vap. temp. = ', 1pe10.3, '/', 10x,
$   'old-time liq. temp. = ', 1pe10.3, 6x,
$   'new-time liq. temp. = ', 1pe10.3)
  endif

c
  if (atmp .eq. oautmp) then
    aflag = 1
    if (izoau .eq. 0) then
c-----1d component (or plenum)
      write (imout, 750) oau, noau, joau
750   format (10x, 'void fraction oscillating, up by ', f5.3,
$       ' in component ', i4, ', cell ', i3)
      id(1) = joau
      id(2) = 0
      id(3) = 0
    else
c-----3d component
      write (imout, 751) oau, noau, izoau, idcoau
751   format (10x, 'void fraction oscillating, up by ', f5.3,
$       ' in component ', i4, ', level ', i3, ', cell ', i3)
      id(1) = izoau
      id(2) = 0
      id(3) = idcoau
    endif
  endif

```

DRAFT

```
endif
call find(noau,ntype,id,nvars,vars,list,inc,cs)
if (cs .eq. 0) then
  write (imout,740) a(list(1)),a(list(2)),a(list(3)),
$   a(list(4)),a(list(5)),a(list(6)),a(list(7)),a(list(8))
  else
    write (imout,8500)
  endif
endif

c
  if (atmp .eq. oaltmp) then
    aflag = 1
    if (izoal .eq. 0) then
c-----1d component (or plenum)
      write (imout,770) oal,noal,joal
770   format (10x,'void fraction oscillating, down by ',f5.3,
$       ' in component ',i4,', cell ',i3)
      id(1) = joal
      id(2) = 0
      id(3) = 0
    else
c-----3d component
      write (imout,771) oal,noal,izoal,idcoal
771   format (10x,'void fraction oscillating, down by ',f5.3,
$       ' in component ',i4,', level ',i3,', cell ',i3)
      id(1) = izoal
      id(2) = 0
      id(3) = idcoal
    endif
    call find(noal,ntype,id,nvars,vars,list,inc,cs)
    if (cs .eq. 0) then
      write (imout,740) a(list(1)),a(list(2)),a(list(3)),
$   a(list(4)),a(list(5)),a(list(6)),a(list(7)),a(list(8))
    else
      write (imout,8500)
    endif
  endif
endif

c
  if (aflag .eq. 0) then
    write (imout,790)
790   format (10x,'***cell not found***')
  endif

c-----
c-----end delamx logic
c-----
  endif

c
c-----limit caused by iterative pressure variation
  if (delt .eq. delrmx) then
    write (imout,800) delrmx
800   format (/,'limited by iterative pressure variation',
$       ',10x,'delrmx = ',f11.6)
    dtflag = 1
  endif

c
c-----excessive outer iterations (see iteration count logic in newdlt)
  if (delt .eq. deltit) then
    write (imout,900) deltit
900   format (/,'limited by excessive outer iterations',
$       ',10x,'deltit = ',f11.6)
    dtflag = 1
  endif

c
c-----healthy 5% (ddi) increase (see iteration count logic in newdlt)
```


DRAFT

```

        if (delt .eq. dtddi) then
            write (imout,1000) ddi
1000    format (/f4.2,'x advancement from last step')
            dtflag = 1
        endif
c
c-----no change (see iteration count logic in newdlt)
        if (delt .eq. deltn) then
            write (imout,1100)
1100    format (/ 'no change from last step, intermediate outer ',
$         'iteration count')
            dtflag = 1
        endif
c
c-----limit caused by change in pressure or temperature (with floor
c-----dtfl of then-current value of 0.5*delt in routine newdlt)
        if (delt .eq. delcmx) then
            write (imout,1200)
1200    format (/ 'limited by pressure or temperature change'
$         ',,5x,'details:')
            dtflag = 1
            ptflag = 0
c
c-----variables for subroutine find
        nvars = 8
        vars(1) = 'alp      '
        vars(2) = 'alpn     '
        vars(3) = 'p        '
        vars(4) = 'pn       '
        vars(5) = 'tv       '
        vars(6) = 'tvn     '
        vars(7) = 'tl       '
        vars(8) = 'tln     '
        ntype = 'cell
c
c-----determine limiting cell and variable, write information
        if (delcmx .eq. dtlsv) then
            if (ltltst .eq. 0) then
c-----1d component
                write (imout,1210) ntlstst,jtlstst,tlmxsv
1210    format (10x,'liquid temperature in component ',i4,
$         ', cell ',i3,' changed by ',f6.1,' k')
                id(1) = jtlstst
                id(2) = 0
                id(3) = 0
            else
c-----3d component
                write (imout,1211) ntlstst,ltltst,jtlstst,tlmxsv
1211    format (10x,'liquid temperature in component ',i4,
$         ', level ',i3,', cell ',i3,' changed by ',f6.1,' k')
                id(1) = ltltst
                id(2) = 0
                id(3) = jtlstst
            endif
            call find(ntlstst,ntype,id,nvars,vars,list,inc,cs)
            if (cs .eq. 0) then
                write (imout,740) a(list(1)),a(list(2)),a(list(3)),
$         a(list(4)),a(list(5)),a(list(6)),a(list(7)),a(list(8))
            else
                write (imout,8500)
            endif
            ptflag = 1
        endif
c

```

DRAFT

```

        if (delcmx .eq. dtvsv) then
            if (ltvtst .eq. 0) then
c-----1d component
                write (imout,1220) ntvstst,jtvstst,tvmxsv
1220         format (10x,'vapor temperature in component ',i4,
$             ', cell ',i3,' changed by ',f6.1,' k')
                id(1) = jtvstst
                id(2) = 0
                id(3) = 0
            else
c-----3d component
                write (imout,1221) ntvstst,ltvtst,jtvstst,tvmxsv
1221         format (10x,'vapor temperature in component ',i4,
$             ', level ',i3,', cell ',i3,' changed by ',f6.1,' k')
                id(1) = ltvtst
                id(2) = 0
                id(3) = jtvstst
            endif
            call find(ntvtst,ntype,id,nvars,vars,list,inc,cs)
            if (cs .eq. 0) then
                write (imout,740) a(list(1)),a(list(2)),a(list(3)),
$             a(list(4)),a(list(5)),a(list(6)),a(list(7)),a(list(8))
            else
                write (imout,8500)
            endif
            ptflag = 1
        endif

c
        if (delcmx .eq. dprsv) then
            if (lpvtst .eq. 0) then
c-----1d component
                write (imout,1230) npvtst,jpvtst,100.0*prmxsv
1230         format (10x,'pressure in component ',i4,
$             ', cell ',i3,' changed by ',f6.1,' percent')
                id(1) = jpvtst
                id(2) = 0
                id(3) = 0
            else
c-----3d component
                write (imout,1231) npvtst,lpvtst,jpvtst,100.0*prmxsv
1231         format (10x,'pressure in component ',i4,', level ',
$             i3,', cell ',i3,' changed by ',f6.1,' percent')
                id(1) = lpvtst
                id(2) = 0
                id(3) = jpvtst
            endif
            call find(npvtst,ntype,id,nvars,vars,list,inc,cs)
            if (cs .eq. 0) then
                write (imout,740) a(list(1)),a(list(2)),a(list(3)),
$             a(list(4)),a(list(5)),a(list(6)),a(list(7)),a(list(8))
            else
                write (imout,8500)
            endif
            ptflag = 1
        endif

c
        if (istdy .eq. 0) then
            if (delcmx .eq. dtssv) then
                write (imout,1240) ntmstst,jtmstst,tsmxsv
1240         format (10x,'wall temperature in component ',i4,
$             ', cell ',i4,', changed by ',f6.1,' k')
                ptflag = 1
            endif
            if (delcmx .eq. dtrsv) then

```

DRAFT

```

        write (imout,1250) ntmtst,jtmtst,trmxsv
1250    format (10x,'inner/outer temperature in Heat Str. ',i4,
        $      ', node row ',i4,', changed by ',f6.1,' k',/,
        $      10x,'(includes any fine mesh node rows)')
        ptflag = 1
    endif
endif

c
    if (delcmx .eq. dtfl) then
        write (imout,1260)
1260    format (10x,'delt/2 "floor", most restrictive ',
        $      'limiter would have been: ')
        if (svdelc .eq. dtlsv) then
            if (ltltst .eq. 0) then
                write (imout,1210) ntlstst,jtlstst,tlmxsv
            else
                write (imout,1211) ntlstst,ltltst,jtlstst,tlmxsv
            endif
            ptflag = 1
        endif
        if (svdelc .eq. dtvsv) then
            if (ltvtst .eq. 0) then
                write (imout,1220) nvtstst,jvtstst,tvmxsv
            else
                write (imout,1221) nvtstst,ltvtst,jvtstst,tvmxsv
            endif
            ptflag = 1
        endif
        if (svdelc .eq. dprsv) then
            if (lprtst .eq. 0) then
                write (imout,1230) nprtst,jprtst,prmxsv
            else
                write (imout,1231) nprtst,lprtst,jprtst,100.0*prmxsv
            endif
            ptflag = 1
        endif
        if (istdy .eq. 0) then
            if (svdelc .eq. dtssv) then
                write (imout,1240) ntmtst,jtmtst,tsmxsv
                ptflag = 1
            endif
            if (svdelc .eq. dtrsv) then
                write (imout,1250) ntmtst,jtmtst,trmxsv
                ptflag = 1
            endif
        endif
        write (imout,1265)
1265    format (10x,'(delt used in division was intermediate ',
        $      'value based on',/,10x,'last step iteration count)')
    endif

c
    if (ptflag .eq. 0) then
        write (imout,790)
    endif

c-----
c-----end delcmx logic
c-----
endif

c
c-----limit caused by power change (which also may force backup)
    if (delt .eq. delpmx) then
        write (imout,1300) delpmx
1300    format (/,'limited by core power change',
        $      /,10x,'delpmx = ',f11.6)

```

DRAFT

```
        dtflag = 1
      endif

c
c-----limit caused by valve adjustment
      if (delt .eq. delxmx) then
        write (imout,1400) delxmx
1400    format (/'limited by valve adjustment rate',
      $      /,10x,'delxmx = ',f11.6)
        dtflag = 1
      endif

c
c-----user-specified minimum timestep size
      if (delt .eq. dtmin) then
        write (imout,1500) dtmin
1500    format (/'at input-specified minimum timestep (dtmin)',
      $      /,10x,'dtmin = ',f11.6)
        dtflag = 1
      endif

c
c-----user-specified maximum timestep size
      if (delt .eq. dtmax) then
        write (imout,1600) dtmax
1600    format (/'limited by input-specified maximum timestep (dtmax)',
      $      /,10x,'dtmax = ',f11.6)
        dtflag = 1
      endif

c
c-----set by namelist variable dtstrt (1st step)
      if (delt .eq. dtstrt .and. nstep .eq. 0) then
        write (imout,1700) dtstrt
1700    format (/'first step, using namelist variable dtstrt',
      $      /,10x,'dtstrt = ',f11.6)
        dtflag = 1
      endif

c
      if (dtflag .eq. 0) then
        write (imout,8000)
8000    format (10x,'***timestep control not found***')
      endif

c
8500 format (10x,'***error in finding data***')

c
      write (imout,9000)
9000 format (/'*****          end timestep diagnostic          *****'
      $      /'*****')
      return
      end
```

DRAFT

APPENDIX B

LISTING OF UPDATE FXFACE2

```
*ident fxface2
*/ -----Update Header Rev. 02-01-96-----
*/
*/   File & Update Ident name (must be the same): fxface2
*/   TSHOOTER identifier: N.A.
*/
*/   Code developer: S. Jolly-Woodruff      Date developed: 02/28/96
*/                       P. Giguere
*/   Acknowledgements:
*/   Update reviewer:                      Date reviewed:
*/
*/   Update to be applied to MOD2 _X_ or other:
*/
*/   CFS file path: /084220/updates
*/
*/   Purpose of update:  Add capability to pass cell-face index to
*/                       subroutine FIND (1D only).
*/
*/   Basis for update:  Previously FIND required cell-center index.
*/                       New cell-face logic is flagged by negative
*/                       value of FIND input argument ID.
*/
*/   Dependencies on other updates: None.
*/
*/   Justification of non-ANSI coding: N.A.
*/
*/   Does this update generate new unit labels?  YES ___ NO _X_
*/   If yes, path and name of LABNEW file:
*/
*/   Does the TRAC I/O change because of this update?  YES ___ NO _X_
*/   TRAC I/O includes input, dump, output, and graphics files.  If
*/   yes, then submit the necessary manual changes.
*/
*/   Does this change require that any of the TRAC support codes, e.g.
*/   EXTRACT, GOCNVRT, or EXCON, be changed?  YES ___ NO _X_
*/   If yes, then include a description of the necessary changes.
*/   This description will be passed onto the custodian of the TRAC
*/   support codes.
*/
*/   Does the TRAC User's Guide, Theory Manual, or Programmer's Guide
*/   change because of this update?  YES ___ NO _X_
*/   If yes, then submit the necessary manual changes.
*/
*/   NAME OF
*/   SUBROUTINE
*/   OR COMDECK
*/   CHANGED      DESCRIPTION OF CHANGE
*/
*/   AICOMP      Use negative value of ID to get cell-face data.
*/
*/   FIND        Add comments and new completion-status code for
*/               incorrect use of negative ID (Vessels and Plena
*/               not included in this update).
*/
*/ -----
```

DRAFT

```
*/
*compile aicompile,find
*/
*I aicompile.197          aicompile
    if (id .lt. 0) then
        if (node .ne. 'cell' ) then
            joff = iabs(id) - 1
        else
            cscode = -2
            go to 9999
        endif
    else
        *I aicompile.207          aicompile
        endif
    *I find.49              find
c        For a 1-D cell, if id<0 it is taken to be a cell-face
c        index. The value of node is ignored. This use is only
c        valid for locating cell-face variables.
c        node should not be given value of 'cell' .
c
    *I find.139              find
c        cscode = -30    bad value for id.
    *I find.194              find
        if (id(1) .lt. 0) then
            cscode = -30
            go to 9999
        endif
    *I find.197              find
        if (id(1) .lt. 0) then
            cscode = -30
            go to 9999
        endif
*/ -----
```