PU/NE-98-8

# SOFTWARE REQUIREMENTS SPECIFICATION FOR THE GENERAL INTERFACE IN THE COUPLED CODE

**Douglas A. Barber, Thomas J. Downar**

School of Nuclear Engineering
Purdue University
W. Lafayette, IN 47907-1290

submitted to:

Division of Systems Technology
Office of Regulatory Research
Nuclear Regulatory Commission
Washington, D.C. 20555-001

Contract NRC-04-96-060, Task 5c
Performed Under Subcontract to SCIENTECH, Inc.

May 1998

## I. Introduction - General Design

The implementation of a thermal-hydraulic/neutronic interface for the reactor vessel is depicted in Figure 1. The design is based on an internal integration approach, which implies that the neutronics code, which will maintain both point and spatial kinetics solution capabilities, solves for the neutron behavior only, and the system code solves for both the system and the core thermal-hydraulics. It should be noted that the design shown in Figure 1 is simply representational; detail concerning calculation flow control will be discussed in Section V.

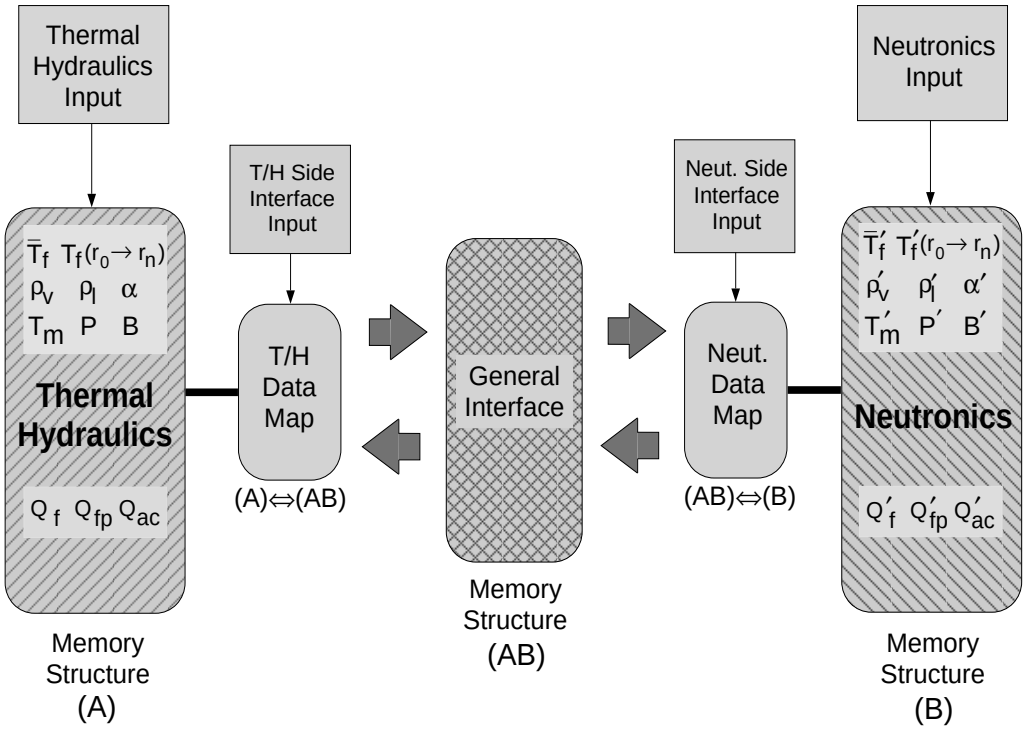


Figure 1: Diagram of Interface Implementation

This concept utilizes independent input processing for the thermal-hydraulic and neutronic codes, which insures that modifications to existing input decks will be minimal. In addition, the input for the interface is processed separately by the code-specific mapping routines. This provides flexibility for inputting code-specific mapping and calculational control information separately, the nature of which will be discussed in Sections IV and V, respectively.

The basic design of the interface includes an immutable generalized interface unit which provides the mapping of property data between thermal-hydraulic ***zones***, neutronic ***nodes***, and heat structure (HS) ***components***. Memory mapping between the interface module and the thermal-hydraulic and neutronic modules is accomplished with customized mapping routines which are specific to the thermal-hydraulic and neutronic codes used in the coupled code system. The description of these routines and the variables to be transferred will be discussed in the following sections.

## II. Variable Transfer

The coupling of the neutronics module to the thermal-hydraulics module is accomplished via the incorporation of thermal-hydraulic feedback effects into the cross sections. For an explicit coupling approach, space-dependent fluid and conduction property data, such as fuel and moderator temperature and moderator density, from the end of the time step are passed through the interface to the neutronics module. The coupling of the thermal-hydraulics module to the neutronics module is achieved with the node-averaged *fission*, *fission product decay*, and *actinide decay powers* ($Q_f$, $Q_{fp}$, $Q_{ac}$), which are used in the heat transfer calculation. For transient control in the coupled code, *time-dependent information*, such as the logic for control rod movement and the time step data, are also passed through the interface.

In an effort to fully generalize the treatment of the fuel temperature feedback, the space-dependent *average fuel temperature* ($\bar{T}_f$) and *fuel temperature profile* in the pin ($T_f(r_0 \rightarrow r_n)$) are transferred to the interface, as shown in Figure 1. This provides flexibility for the neutronics codes which vary in the method of calculating doppler temperature for feedback into the cross sections. To describe the profile, information concerning the pin geometry and discretization is passed through the interface to the neutronics code at the beginning of the calculation. The treatment of the moderator density feedback is generalized to allow the use of mixture density and/or void density correlations. This is accomplished by transferring the space-dependent *liquid and vapor densities*, along with the *void fraction* ($\rho_l$, $\rho_v$, $\alpha$), from the thermal-hydraulics module to the interface. Finally, the space-dependent *moderator temperature* ($T_m$), *pressure* ($P$), and *Boron concentration* ($B$) are also passed through the interface. It should be noted that all property data mapped by this interface utilize SI units.

In practice, the variables described above are sufficient to couple a thermal-hydraulics code to a neutronics code. Nonetheless, flexibility is incorporated to allow the transfer and mapping of any space-dependent information. With regard to output editing and generation of restart files, it is assumed that the thermal-hydraulics and neutronics codes will maintain independent control. However, it is necessary to insure that the frequency of output and restart updates is synchronized for the coupled code. Therefore, sufficient control information should be passed through the interface and processed by either of the code-specific mapping routines.

## III. General Interface Structure and Data Transfer

The structure of the General Interface unit is dependent on the nature of the coupling used to link the thermal-hydraulics, neutronics, and interface modules. In general, this module contains three functional units as shown in Figure 2. In the initialization stage, all necessary mapping and geometry information is transferred to the General Interface and stored for use in the two subsequent variable mapping units. It should be noted that this general structure also applies to the code-specific thermal-hydraulic and neutronic data mapping routines.

As will be discussed in Section IV, the mapping functions to be passed to the General Interface are represented as two permutation matrices, one for each mapping direction. These two per-

mutation matrices, which are stored using the Coordinate Storage Format[1], are processed by either the thermal-hydraulic or neutronic data mapping routines, and are sent to the General Interface during the initialization. For the unit controlling the mapping of thermal-hydraulic fluid and conduction data to neutronic nodes, a vector of thermal-hydraulic zone-wise and component-wise variables are passed to the General Interface. Conversely, for the unit controlling the mapping of neutronic data to heat structure components and thermal-hydraulic zones, a vector containing all neutronic nodal data is passed to the General Interface.

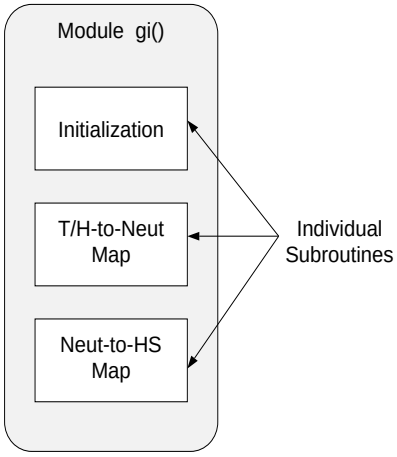


Figure 2: Structure of General Interface Module

For a **multiple-process** coupling strategy, as will be discussed in Section V, the transfer of this data through the interface routines is performed with *send* and *receive* functions, and essentially, two pieces of information are transferred. The first involves information which is actually used by General Interface, and consists of the permutation matrices and the vectors to be mapped. The second consists of any additional information which is used for calculational coherency between the thermal-hydraulic and neutronic processes. As shown in Figure 3, this information is packed into a data structure consisting of characters, logicals, integers, and reals, which is used to transfer initial control information through interface as well as time-dependent information during the calculation, such as control rod and time step logic.

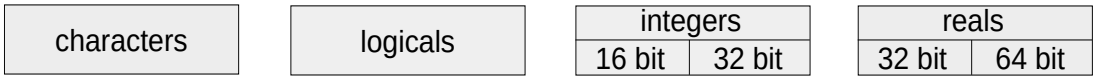


Figure 3: Auxiliary Data Structure

Most of the information stored in these buffers is needed only by the thermal-hydraulic and neutronic processes. However, the following locations are set aside for access by the General Interface:

Initial Thermal-Hydraulic Control Buffer:

logical(1): Indication of an error in the T/H code
logical(2): Indication of a data error in the T/H-Specific Data Map Routine
logical(3): Indication of a PVM error in the T/H-Specific Data Map Routine
logical(4): Indication of a data error in the General Interface
logical(5): Indication of a PVM error in the General Interface
logical(6): Indication of normal calculation termination
logical(7): Indication of whether T/H-Specific Data Map Routine is sending the permutation matrices

Initial Neutronic Control Buffer:

logical(1): Indication of an error in the Neutronic code
logical(2): Indication of a data error in the Neutronic-Specific Data Map Routine
logical(3): Indication of a PVM error in the Neutronic-Specific Data Map Routine
logical(4): Indication of a data error in the General Interface
logical(5): Indication of a PVM error in the General Interface
logical(6): Indication of normal calculation termination
logical(7): Indication of whether Neutronic-Specific Data Map Routine is sending the permutation matrices

Time-Dependent Thermal-Hydraulic Control Buffer:

logical(1): Indication of an error in the T/H code
logical(2): Indication of a data error in the T/H-Specific Data Map Routine
logical(3): Indication of a PVM error in the T/H-Specific Data Map Routine
logical(4): Indication of a data error in the General Interface
logical(5): Indication of a PVM error in the General Interface
logical(6): Indication of normal calculation termination

Time-Dependent Neutronic Control Buffer:

logical(1): Indication of an error in the Neutronic code
logical(2): Indication of a data error in the Neutronic-Specific Data Map Routine
logical(3): Indication of a PVM error in the Neutronic-Specific Data Map Routine
logical(4): Indication of a data error in the General Interface
logical(5): Indication of a PVM error in the General Interface
logical(6): Indication of normal calculation termination

## IV. Design of the Mapping Function

The function used by the General Interface to map zone-wise and component-wise data to neutronic nodes is represented as a single permutation matrix, ***P***, composed of variable-dependent submatrices of either size ***n***-by-***m*** or size ***n***-by-***m′***, where ***n*** is the number of neutronic nodes, ***m*** is the number of thermal-hydraulic zones, and ***m′*** is the number of heat structure components. In practice, a different submatrix is employed for each of the variables passed from the thermal-hydraulics code to allow for separate averaging techniques. The submatrices for the fuel temperatures, which correspond to heat structure components, have dimension ***n***-by-***m′***, and the submatrices for the zone-wise thermal-hydraulic property data have dimension ***n***-by-***m***. In an effort to ease the input requirements, the user is given the option of assigning an input permutation submatrix to multiple state variables. In addition, the mapping function between neutronic nodes and both heat structure components and thermal-hydraulic zones is represented as a single permutation matrix, ***P′***. This permutation matrix is composed of submatrices of size ***m′***-by-***n*** for mapping powers (i.e. fission, decay, and gamma) to components, and submatrices of size ***m***-by-***n***, which are used to map a fraction of the node-wise powers directly to the thermal-hydraulic zones.

As mentioned earlier, these matrices are processed from user input in the code-specific T/H or Neutronics Data Map routines. An error checking module is provided for the input permutation matrices, however, the primary responsibility is on the user to insure their consistency with the input thermal-hydraulic, neutronic, and heat structure nodalization. Detail on the error checking module will be discussed in Section VI.

The use of the permutation matrix, ***P***, in mapping zone-wise and component-wise thermal-hydraulic variables to neutronic nodes is given by:

$$\boldsymbol{P}\,\boldsymbol{x}^{HS+T/H} = \boldsymbol{x}^{Neut.} \;;\; \boldsymbol{x}^{HS+T/H} \in \Re^{im'+jm} \;;\; \boldsymbol{x}^{Neut.} \in \Re^{(i+j)n} \;, \tag{1}$$

where the dimension of the vector, $\boldsymbol{x}^{HS+T/H}$, is based on ***i*** component-wise variables and ***j*** zone-wise variables, and the dimension of $\boldsymbol{x}^{Neut.}$ is based on ***i+j*** total variables. Conversely, the use of the permutation matrix, ***P′***, in mapping neutronic nodal powers to corresponding heat structure components and thermal-hydraulic zones is given by:

$$\boldsymbol{P}'\boldsymbol{x}^{Neut.} = \boldsymbol{x}^{HS+T/H} \;;\; \boldsymbol{x}^{HS+T/H} \in \Re^{im'+jm} \;;\; \boldsymbol{x}^{Neut.} \in \Re^{(i+j)n} \;, \tag{2}$$

where $\boldsymbol{x}^{HS+T/H}$ consists of ***i*** component-wise powers for the heat structures, and ***j*** powers for the fractional deposition in thermal-hydraulic zones, which is used to account for direct heating to the coolant.

The form of these permutation matrices is easily demonstrated with the simple thermal-hydraulic and neutronic planar nodalizations shown in Figure 4.

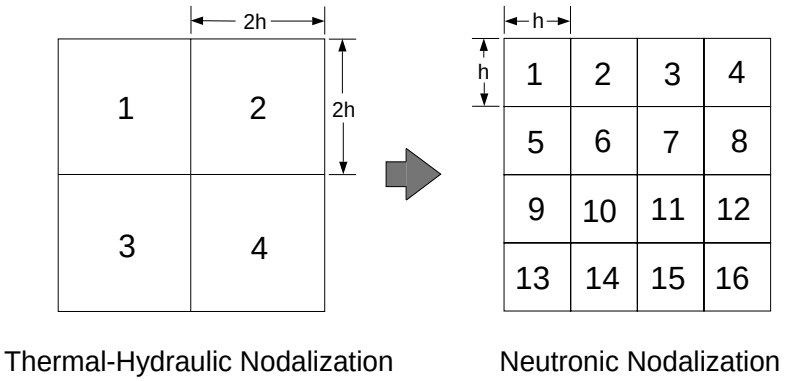


Figure 4: Cartesian Thermal-Hydraulic to Neutronic Mapping

The corresponding permutation for a single thermal-hydraulic variable would be,

$$x^{Neut.} = \begin{bmatrix} 1&0&0&0 \\ 1&0&0&0 \\ 0&1&0&0 \\ 0&1&0&0 \\ 1&0&0&0 \\ 1&0&0&0 \\ 0&1&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&1&0 \\ 0&0&0&1 \\ 0&0&0&1 \\ 0&0&1&0 \\ 0&0&1&0 \\ 0&0&0&1 \\ 0&0&0&1 \end{bmatrix} x^{T/H} \quad ; \quad \begin{matrix} x^{Neut.} \in \Re^{16} \\ x^{T/H} \in \Re^{4} \end{matrix} \quad . \tag{3}$$

This example assumes that the boundaries of the thermal-hydraulic zones and neutronic nodes are congruent, which is the case for most practical applications. However, when the boundaries are not congruent, resulting in overlapping thermal-hydraulic and neutronic regions, the user is responsible for incorporating appropriate weighting factors (e.g., volume fractions) into the permutation matrix. The general form of the permutation matrix is now given as:

$$P_i = \sum_{k(i) \in j} w^{i}_{k(i)} \, e_{k(i)} \quad ; \quad e_{k(i)} \in \Re^{(1,j)} \quad ; \quad \textbf{for } i=1 \ldots n \quad , \tag{4}$$

where ***k(i)*** designates the thermal-hydraulic zone(s) belonging to neutronic node ***i***, and $\boldsymbol{e}_{\boldsymbol{k}(\boldsymbol{i})}$ is a row vector with 1 in the ***k(i)***-th position, and zeros everywhere else. The dimension represented by ***j*** corresponds to either ***m*** or ***m′***, depending on the variable-dependent submatrix. The weighting factor, $\boldsymbol{w}^{\boldsymbol{i}}_{\boldsymbol{k}(\boldsymbol{i})}$, is a scalar which represents the weighting of the ***k(i)***-th thermal-hydraulic zone on the ***i***-th neutronic node. The restriction on the weighting factor, is given as

$$\sum_{\boldsymbol{k}(\boldsymbol{i}) \in \boldsymbol{m}} \boldsymbol{w}^{\boldsymbol{i}}_{\boldsymbol{k}(\boldsymbol{i})} = \mathbf{1.0} \quad . \tag{5}$$

It should be noted that for ***P***, Eq. (5) represents the summation of elements in *row* ***i***, and for ***P′***, Eq. (5) represents the summation of elements in *column* ***i***. An example which would require the use of weighting factors is shown in Figure 5, where a planar mapping from cylindrical thermal-hydraulic coordinates to cartesian neutronic coordinates is required.

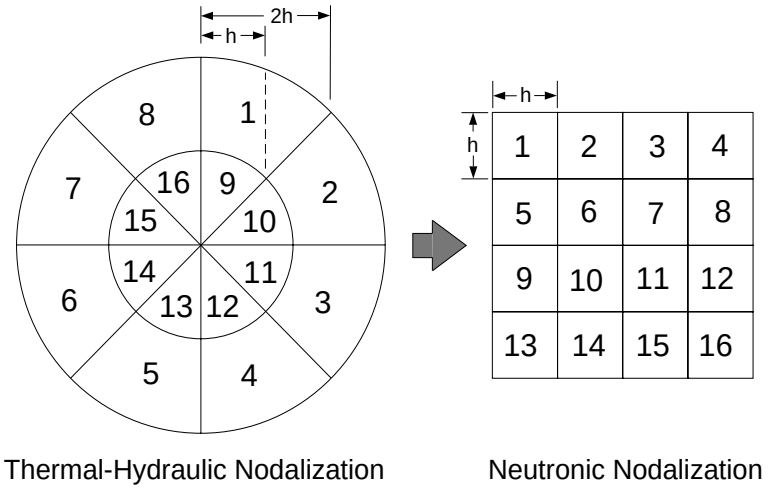


Figure 5: Cylindrical to Cartesian Mapping

By taking the weighting factors for this example to be simple area fractions, the permutation submatrix for a single thermal-hydraulic variable would be of the form:

$$\boldsymbol{P} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.71 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.29 \\
0.71 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.29 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.71 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.29 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.71 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.29 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.71 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.29 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.71 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.29 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.71 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.29 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.29 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} . \qquad (6)$$

## V. Calculational Control

The calculational control for the thermal-hydraulic, neutronic, and interface modules is accomplished with independent processes for each module. For this **multiple-process** approach, the coupling of the modules is achieved with the use of either the Parallel Virtual Machine (PVM) package, which is distributed with the General Interface software. Specifically, the entry points into and out of the interface are managed with *sends* and *receives*, as shown in Figure 6. This coupling strategy has the advantage that minimal coding modifications to the thermal-hydraulic and neutronic modules are required. However, the user is required to manage an additional coding package for PVM.

In the calculational procedure, the thermal-hydraulic and neutronic modules independently perform input processing, which also includes calls to the code-specific data mapping routines to process the input required by the General Interface (e.g. the permutation matrices). Once the permutation matrices have been processed in the code-specific data mapping routines, these matrices are passed to the initialization unit of the General Interface routine to be stored for future use. Following the input processing, the neutronics code calculates the steady-state power distribution based on the initial thermal-hydraulic condition of the core. Using this power distribution, either the steady-state initialization or the transient calculation can be performed. The decision of whether to use point or spatial kinetics during the steady-state initialization is left to the user. However, to avoid a non-initialized core condition, the kinetics used in the transient calculation should be consistent with that employed for the steady-state initialization.

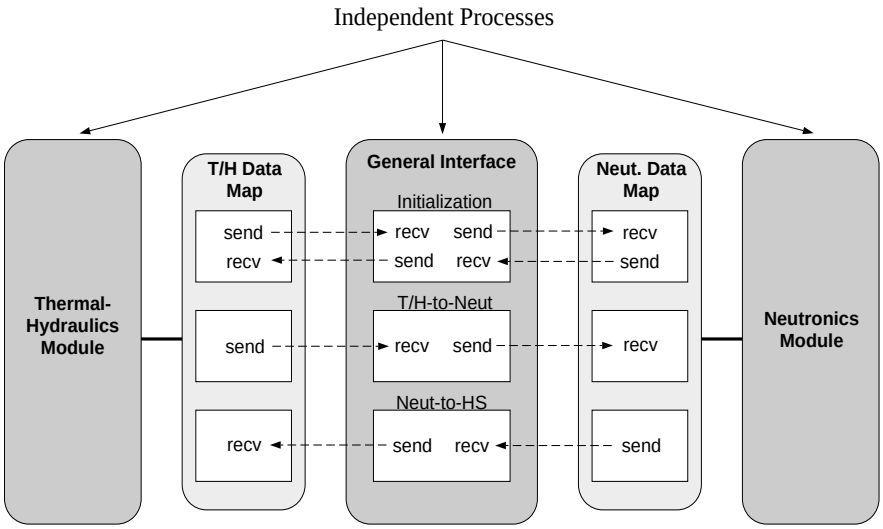


Figure 6: Coupling with PVM

Once the input has been processed and each module has been initialized, the flow through the interface routines is the same regardless of whether a steady-state or transient calculation is being performed. As shown in Figure 6, this flow proceeds in two directions, dictated by the PVM/MPI *sends* and *receives*.

Following the advancement of the heat conduction and hydrodynamic calculation, the zone-wise and component-wise variables stored in the thermal-hydraulic memory structure, denoted by (*A*) in Figure 1, are packed into a single vector in the T/H Data Map routine, converted to the units utilized by the interface, and sent to the General Interface module to be unpacked and stored in the memory structure denoted by (*AB*). Once this data has been received, the General Interface module performs all data mapping from both thermal-hydraulic zones and heat structure components to neutronic nodes. As described in Section IV, this is accomplished with a single matrix-vector multiply. The vector of thermal-hydraulic data corresponding to neutronic nodes is then sent to the Neutronics Data Map routine, where it is unpacked and stored in the neutronics memory structure, denoted by (*B*). At this point, all necessary unit conversion needed for consistency in the neutronics module is performed.

Upon the completion of the neutronics calculation, the neutronic powers are packed into a single vector by the Neutronics Data Map routine, and all necessary unit conversion and/or normalization is performed. This vector of nodal neutronic data is then sent to the General Interface Module, where it is mapped to thermal-hydraulic zones and heat structure components. The resulting vector of neutronic powers corresponding to zones and components is sent to the T/H Data Map routine, where it is unpacked and stored in the thermal-hydraulic memory structure. Any additional normalizations or conversions are performed at this point.

## VI. Time Step Control

Time step control for the coupled code is handled by the T/H Data Map routine and is based on the control information passed through the interface from the thermal-hydraulics and neutronics modules. Because of the difference in characteristic times between the thermal-hydraulics and neutronics during severe transients (e.g. super-prompt critical events or events involving a phasic transition), the flexibility exists to allow either module to subcycle the time step. However, following a time step subcycling, it is necessary to re-synchronize the time steps, and this is controlled by the T/H Data Map routine. In addition, if the time step size sent to the neutronics is such that the change in the core condition is too large, logic control is provided for the neutronics code to reject the time step data, and request a new thermal-hydraulic calculation at a finer time step size. As an example, this logic control is depicted in Figure 7, where the time step size used by the thermal-hydraulics is too large (1), and the kinetics sends logic information back to the thermal-hydraulics code to re-perform the calculation at the time step $t^{n+1/2}$ (2).

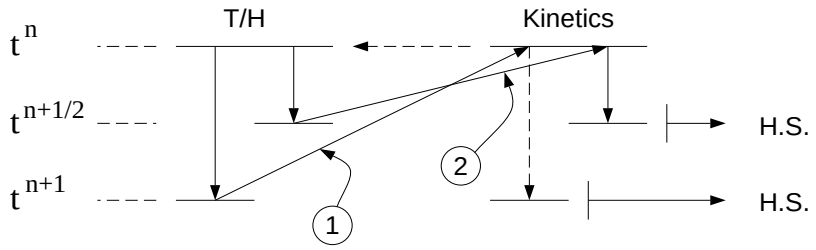


Figure 7: Example of Kinetics Controlling Time Step

## VII. Programming Standard for Interface Routines

The design of the General Interface routine, which adheres to the FORTRAN-90 standard, is modular in nature and maintains portability across all computer platforms. In addition, an error checking module is included to help ensure that the input permutation matrices are consistent with specified requirements. Specifically, the dimension of the input matrices is compared to the dimension of the input vectors, the restriction given in Eq. (5) is checked, and the elements of the matrices and vectors are compared against pre-determined upper and lower bounds. These bounds will be outlined in the Software Design and Implementation Document (SDID). This error routine does not verify that the relation between node numbers and zone/component numbers is consistent with the models input to the thermal-hydraulics and neutronics codes.

## VIII. Interface Summary

The coupling of a thermal-hydraulics and neutronics code described here is accomplished with the use of an immutable generalized interface unit. To accommodate this design, it is necessary to utilize code-specific data mapping routines for both the thermal-hydraulics and neutronics

codes. This allows the two modules to be coupled with only minor programming modifications to each.

The interface design employs a basic internal coupling strategy, and the calculational control is explicit in nature. However, the design is flexible enough to accommodate more sophisticated thermal-hydraulic/neutronic coupling strategies (e.g. implicit, semi-implicit). In addition, this coupled code will have the capability to accurately predict a wide range of transient scenarios, including BWR stability problems.

## IX. References

1. Y. Saad, "Numerical Methods for Large Eigenvalue Problems," Manchester University Press, Manchester, UK., pp. 40-41 (1992).