Converting SINDA/G® and Hughes' CINDA text input files to SINDA/FLUINT input files

This document describes an automated translation tool that assists conversion from old ASCII text input formats into SINDA/FLUINT.

While this translator helps take care of many onerous tasks, no translation is perfect; some conversions still require your attention. Also, there aren't always one-to-one correspondences between older SINDA or CINDA versions, which lacked submodels and used different network solution approaches. You should carefully check the final model!

Summary of Key Differences

Most parts of a model need little or no translation. Almost all of the basic network descriptions (nodes, conductors, etc.) need little or no modification. Many SINDA/G® (SINDA/G is a registered trademark of Network Analysis, Inc., now owned by MSC Software) and Hughes' (now Boeing) CINDA options are available in SINDA/FLUINT. However, many options are considered archaic and are therefore not documented to discourage their continued use. Although they still might function in SINDA/FLUINT, you should make sure any such deprecated features are converted to their modern counterparts before running the model in SINDA/FLUINT. Most (but not all) of these conversions are handled by the translator.

Some translation differences between SINDA/FLUINT and older SINDA and SINDA-like codes appear in the logic sections. As no translator can be foolproof, you may still be required to make modifications in logic blocks where the translator cannot predict the intent or even action of the logic, and can only translate line by line.

One of the major differences between SINDA/FLUINT and other SINDA-like codes is the presence of FLUINT (addressed by the FloCAD® module of Thermal Desktop®, or "TD"). In other codes, limited fluid flow analysis is performed with one-way conductors. Although one-way conductors are available in SINDA/FLUINT as well, it is *strongly* recommended that fluid flow be modeled with modern flow networks instead.

Other major differences include the availability of registers and the built-in spreadsheet, of submodels, and of the Solver in SINDA/FLUINT. Once a model is converted, the user should consider modifying it to take advantage of these powerful features.

Registers and Expressions

With few exceptions, almost anywhere a value is input in a SINDA/FLUINT data block, an expression can be supplied instead. Expressions can use multiplications ("*"), divisions ("/"), additions ("+"), subtractions ("-"), and exponentiations ("^" or "**") nested within arbitrary levels of parentheses. Furthermore, you can use built-in functions (like sine, cosine, and logarithms), built-in conversion constants and physical constants (pi, and the Stefan-Boltzmann constant).

You can even define your own variables, called *registers*, and make these registers functions of each other. An expression may contain one or more registers and can be used to define SINDA/FLUINT

parameters, such as “my_model.T5000” for the temperature of node 5000 in submodel “my_model.” Expressions may also contain IF/THEN/ELSE-like switching (conditional operators), and can refer to “processor variables” such as problem time and other control parameters, output results (e.g., temperatures, pressures), etc.

Registers and expressions should be used extensively because they …

- make a model more self-documenting. If you inherit a model, or if you are attempting to read and understand someone else’s old model, you will be better able to understand the sources and intents of inputs when they are left as full expressions.
- add spreadsheet like functions to a model. You can define complex interrelationships between inputs (and even between inputs and outputs) to make changes consistently, making it easier to maintain several analysis cases or designs within a single model file (although this functionality is better accomplished in modern usage via TD’s **Case Set Manager**.)
- allow model building to proceed before dimensions and properties have been finalized.
- minimize the use of logic blocks, and the need to remember translation rules, eliminating the inconvenience of defining in two places (input and logic) how a network element behaves (see also “Network Element Logic” in Thermal Desktop.)

Furthermore, registers can be:

- varied dynamically during processor execution, with their effects propagated automatically throughout a model. This capability greatly facilitates execution of parametric analyses and sensitivity studies.
- used as output variables when goal seeking, or reversing the normal input/output sequence of SINDA/FLUINT.
- used as design variables in optimization.
- used as uncertainties or correlating parameters for automated test data correlation.

In summary, you will find it advantageous to make copious use of registers during the construction of a model.

Submodels

Submodels are a way to break up your model into logical parts.

Submodels allow easy model integration and facilitate working in teams. They allow you to combine multiple component models into one large model without having to worry about duplicate conductors or nodes.

Submodels may be added and deleted during run time as needed to swap in or out components, boundary conditions, alternate designs or materials, etc.

Solver

The Solver is a complete design optimization module that works with the built-in spreadsheet options. (Within Thermal Desktop, see the **Dynamic SINDA** tab in the **Case Set Manager**.)

The Solver will automatically vary the model to achieve the user-specified results: SINDA/FLUINT can be tasked to find the best design (least weight, best performance), or the best (correlated) model of a given design.

Moving on to Thermal Desktop®

A successfully executing SINDA/FLUINT input file that satisfactorily matches previous results should be viewed as a step, and not an end product. Text-only models are expensive to maintain and arduous to expand. They are also somewhat dangerous because it can be hard to visualize predictions.

The real goal should be conversion into a Thermal Desktop drawing.

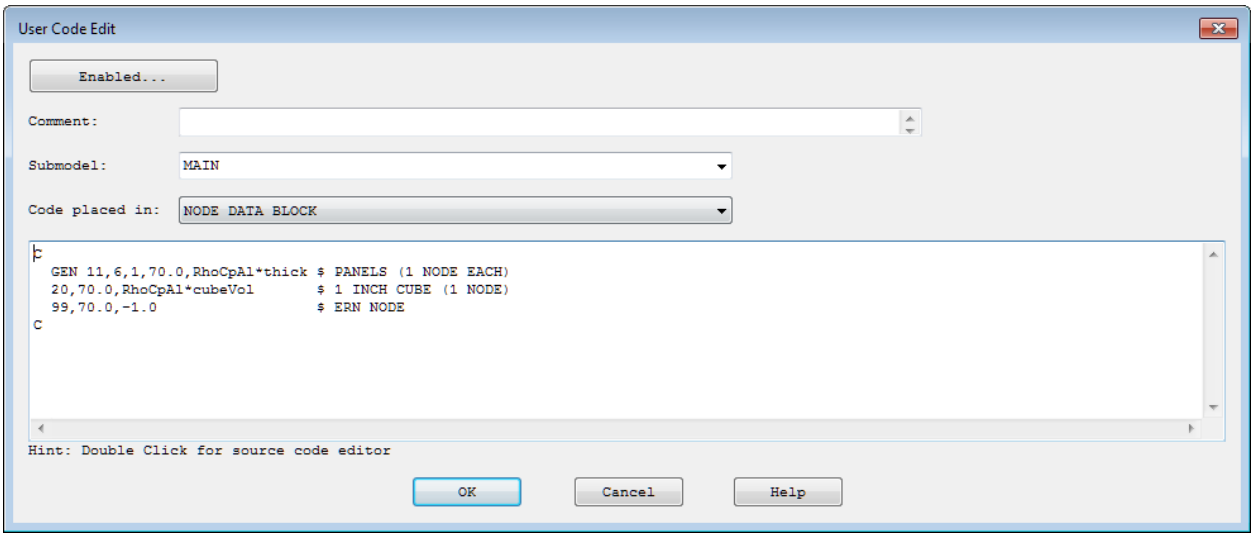
This final conversion can be done in stages, and should not be viewed as an all-or-nothing proposition. Some text-based inputs will perhaps remain in that form permanently.

In SINDA/FLUINT, portions of a model can be inserted from other files using an INSERT command. Two options exist for including text-based model information in Thermal Desktop: Logic Objects and Case Sets.

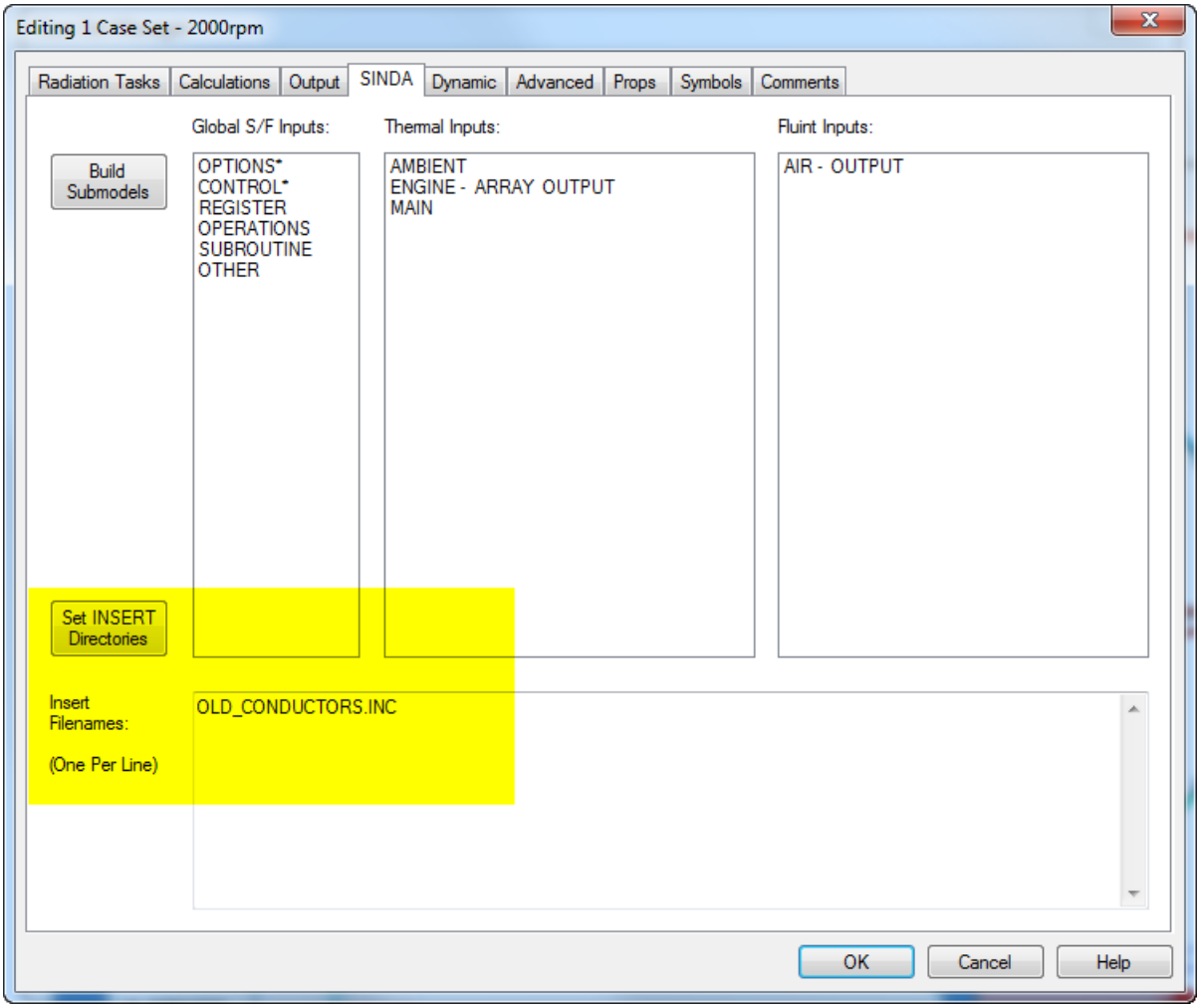
Logic Objects include predefined functions, such as array interpolation or equations of motion, or user-defined data or logic blocks. Logic Objects are available for all cases, but can be excluded from a case if desired. To add a user-defined data or logic block:

1. Select **Thermal > Logic Objects Manager**
2. Choose **User Text Input HEADER/SUBROUTINE** from the **Select Object To Create** drop-down
3. Select **Create**
4. Choose the submodel for the data or logic block from the **Submodel** drop-down
5. Choose the type of data or logic block from the **Code placed in** drop-down
6. Paste the data block, logic block, or INSERT statement into the **Code** field

The HEADER line does not have to be included if the text in the Code field matches the block type selected in the **Code placed in** field (as shown below). If another HEADER line is included, the text following it must match that block's format.



Case Sets contain the definition of specific solution. Case Sets are created or edited through the Case Set Manager (**Thermal > Case Set Manager**). The **SINDA** tab of the **Case Set** (shown below), allows INSERT files to be listed and directories containing INSERT files to be added to a search path.



Even within a block (such as ARRAY DATA or VARIABLES 2 logic), an INSERT file can be added. Each block can be accessed using the above form.

The data for each block need not be located all in one place. When TD sends input files to SINDA/FLUINT, if the final combined file has multiple instances of a block (for example, “HEADER OUTPUT CALLS, THATSTUFF”), then they will all be combined together providing the header “cards” are all identical.

> Note: Because blocks are combined, the user should always reset FAC to 1 using a FAC record at the end of any data block section whenever FAC appears in the data block.

Some of the key differences between a stand-alone SINDA/FLUINT text-based model and a Thermal Desktop drawing are summarized below.

Geometry

SINDA has little or no knowledge of geometry, whereas Thermal Desktop® (and its companion modules FloCAD® and RadCAD®) are powerful precisely because of their knowledge of the model's geometry. A separate module, TD Direct®, is almost exclusively focused on geometric preparation (including geometric modeling, CAD import and clean-up, meshing, and thermal mark-ups).

Nonetheless, nongeometric models *can* be constructed in Thermal Desktop: actual geometry is recommended but not mandatory (unless thermal radiation is involved). This allows you to gradually replace sections of a SINDA model by a geometric TD object (whether finite element or finite difference).

Thermal Radiation

In the days of "text-only" SINDA models, thermal radiation used to be solved separately in programs such as TRASYS, which created text input files for use by SINDA-like programs. TRASYS and many similar models can be imported by Thermal Desktop, where radiation and convection/conduction are no longer separated (though RadCAD is a separately licensed thermal radiation and environmental heating module).

Thermal Properties

Thermal properties are defined in separate data sets, and are referenced by name (e.g., "steel" or "white paint") so that they can be changed rapidly and re-used in different models.

Case Set Manager

A single Thermal Desktop drawing can contain multiple (perhaps hundreds) of SINDA/FLUINT models or "cases." Each case can have custom OPERATIONS, outputs, BUILD sequences. Most commonly, each will have different symbol overrides (see Symbols and Registers below), allowing one or more factors to change with each case, perhaps as driven from an external spreadsheet.

Cases can be edited individually or together, and can be arranged in groups.

OPERATIONS, VARIABLES, and OUTPUT CALLS

Although these blocks can still be customized (meaning: built manually), they are normally constructed by TD according to menu choices in the Case Set Manager's **Calculations**, **Output**, and **SINDA** tabs and by selections in the model (e.g., convection correlations, temperature-variable properties, heat pipe or TEC objects, etc.).

Symbols and Registers

TD has its own set of parameters called "symbols" that are by default independent from SINDA/FLUINT registers, though they can be overlapped.

Why two sets of parameters? Because some parameters need only exist in set-up phases (such as an initial temperature or a dimension) and become *symbols*, and others need only exist during processor execution and are *registers* (such as a thermostatic set point or an effectiveness), and some are needed in both phases (perhaps if they vary parametrically during a run).

In modern usage, almost all variations are described as symbols first, and then a subset of these symbols is then sent to SINDA/FLUINT as registers (perhaps varying by case).

Model Browser

The **Model Browser** is a powerful tool not just for finding and editing objects, but even postprocessing certain data.

Logic Manager

Arrays and interpolation, PID controls, equations of motion, etc. are all best handled by the **Logic Manager**, and not in individual cases.

Domains and Domain Tag Sets

Regions of a model can be declared as domains, and portions within that region (say nodes or surfaces) will then inherit actions applied to those domains. Domains make it easy to change a model (swapping parts or changing resolution) without deleting and re-applying heat sources, contact conductances, etc.

Measures and the Data Logger

Features for defining the location of thermocouples and the test data associated with those measurement points, for the automating of model calibration to test, have no parallel in SINDA files.