Thermal Modeling and Analysis

Hume Peabody
Staff Engineer
Goddard Space Flight Center
Code 545
Hume.L.Peabody@nasa.gov
(301) 286-9141

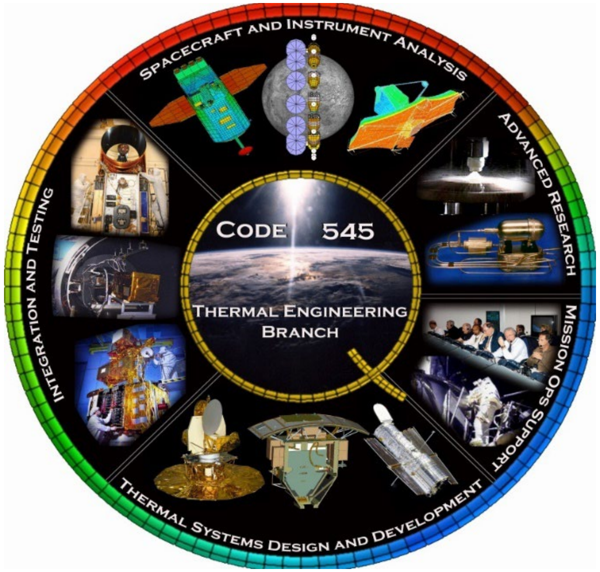
Disclaimer

- This package describes the Thermal Modeling and Analysis process used at the NASA Goddard Space Flight Center
- Much of the Thermal Modeling portion focuses on the capabilities of two commercial tools for performing Thermal Analysis
  - Thermal Desktop® for radiative modeling
  - SINDA/FLUINT® for thermal modeling
- ***<u>This should not be considered as an endorsement of any particular tool by NASA or the United States Government</u>***
- The illustrations of capabilities are meant to be informational and other tools may have similar or superior capabilities

Thermal Analysis Agenda

- *What is Thermal Analysis and Why do we do it?*
- *Overview of Available Tools*
- *Thermal Desktop*
    - *Introduction*
    - *Graphical User Interface*
- *Model Building Process*
- *Building a Thermal Desktop Model*
- *Modeling Specific Component Types*
- *Radiation Computations*
- *SINDA/FLUINT*
    - *Basic*
    - *Intermediate*
    - *Advanced*
- *Thermal Computations*
- *OpenTD API*
- *Best Modeling Practices and Miscellaneous Tips*
- *Analyzing the Model Predictions*

What is Thermal Analysis?



What is Thermal Analysis and Why do we do it?

- ***Thermal Analysis is using computer simulations to predict the thermal performance of a system subjected to various applied constraints and boundary conditions***
- ***Spacecraft thermal analysis is typically a two part process:***
  - *Part 1: using the geometrical design (shapes, locations, orientations, orbits, coatings) compute the energy exchange factors between nodes (representing the temperature of the surface) and the absorbed energy from orbital sources (Sun, Planet, Moon, direct and reflected [albedo])*
  - *Part 2: combining the values computed in part 1 with conduction couplings, additional heat sources, boundary conditions, and control logic, compute the expected temperatures and control heater power*
- ***Why do we do thermal analysis?***
  - *Cost efficient means of exploring design space without the need to build and test every potential design*
  - *Allows various design configurations to be modeled and evaluated*
  - *Allows various questions to be asked of the design: Are all temperature predicts maintained within limits? How much heater power is needed by the system? How long will it take to cool down? How stable are the temperatures? etc.*
- ***What are the downsides to thermal analysis?***
  - *Predictions are just that. It is up to the thermal engineer to evaluate and understand the predictions*
  - *Trust but verify!*
  - *Trust your own judgement. What did you expect from the predicts? Did it match those expectations? If not, do you understand why? Are the physics still true (e.g. is energy conserved)?*
  - *Models are grounded to actual performance through testing and correlation*

Thermal Models

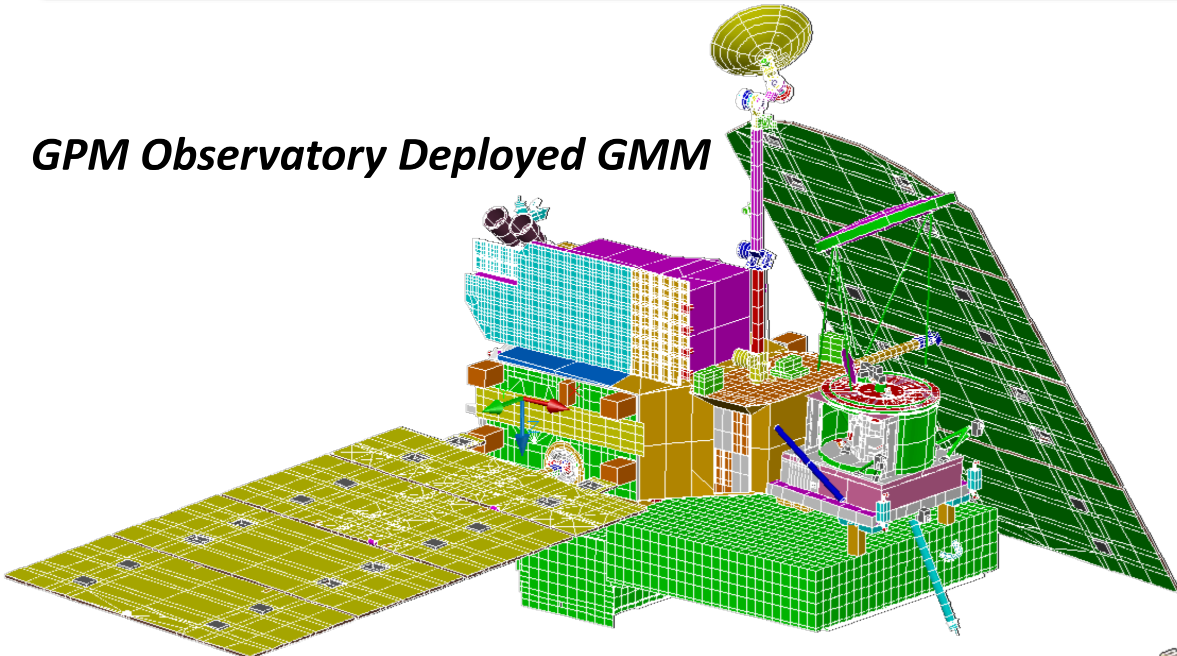
- *Thermal models are a two part process: Geometric Math Model (GMM) and Thermal Math Model (TMM)*
- *The GMM includes all radiating surface locations in their correct spatial position*
- *GMM is used to produce Interchange Factors between surfaces (expressed as Radiation Couplings between nodes in TMM) and Solar/Albedo/Planetshine Environmental Backloading (expressed as sources on nodes in TMM)*
- *The TMM includes all nodal definitions and their interdependence*
- *The TMM also contains instructions for what is to be solved (steady state, transient), simulation duration, solution accuracy control, result output frequency, etc.*
- *GMM generates Environmental Loading and Radiative Interchange Factors as inputs to TMM; TMM calculates temperatures and heater powers*

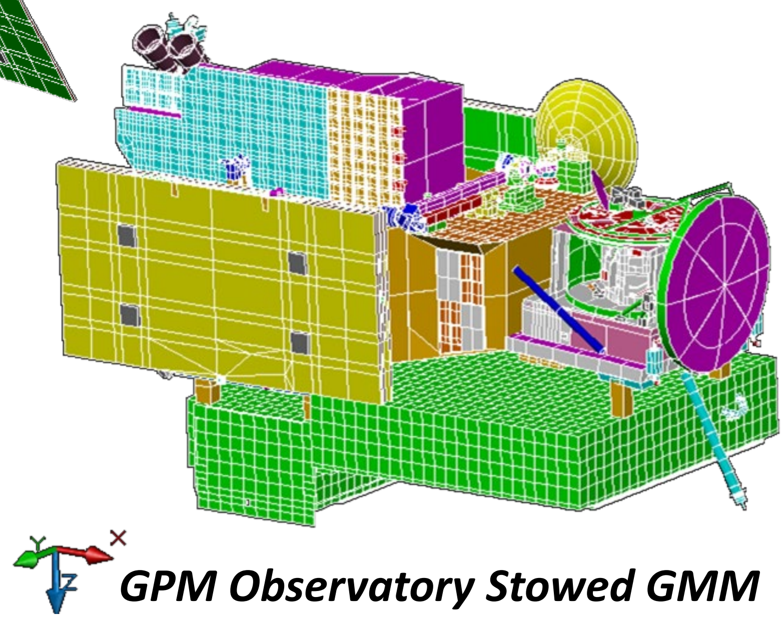
Geometric Math Models (GMM)

- ***The GMM includes all radiating surface locations in their correct spatial position***
  - Surfaces are typically primitive shapes: rectangles, triangles, cones, cylinders, spheres, etc
  - Can use Finite Elements, but if not careful, can lead to oversized models…
  - TMM Nodal assignments, Radiation Active Sides and Thermo-optical Properties (α,ε) must also defined
  - Surface may also be subdivided into smaller nodes
- ***GMM is used to produce Interchange Factors between surfaces (expressed as Radiation Couplings between nodes in TMM) and Solar/Albedo/Planetshine Environmental Backloading (expressed as sources on nodes in TMM)***
  - Articulation/motion of assemblies can be modeled as well as variation due to orbital effects
  - Monte Carlo ray trace most typical solution approach
    - *Random location and direction selected for ray starting with 100% energy*
    - *Ray is propagated until it intersects another surface along its path*
    - *Energy is deposited from the ray onto the absorbing surface based on its properties; the ray is then reflected specularly ($\angle_{incidence} = \angle_{reflection}$) or diffusely (random direction selected) based on properties*
    - *Process continues until ray has minimal energy, after which it is completely absorbed or reflected*
    - *Then another ray is fired from original surface. Process continues for all surfaces until acceptable statistical error is reached for desired accuracy or maximum number of rays have been fired*
  - Fortunately, accuracy tends to be higher for larger view factors, so dedicated radiators do not need as many rays to be fired if they have a large view to space
    - *Internal radiation is typically a secondary means of heat transport (conduction dominates)*
- ***GMM may also be used to define material properties, thicknesses, heat loads, heaters, contact, etc that will be used to generate appropriate inputs to TMM***
  - Nodes, sources, conductors, arrays…(these terms defined in two slides)

Geometric Math Model Example

GPM Observatory Deployed GMM





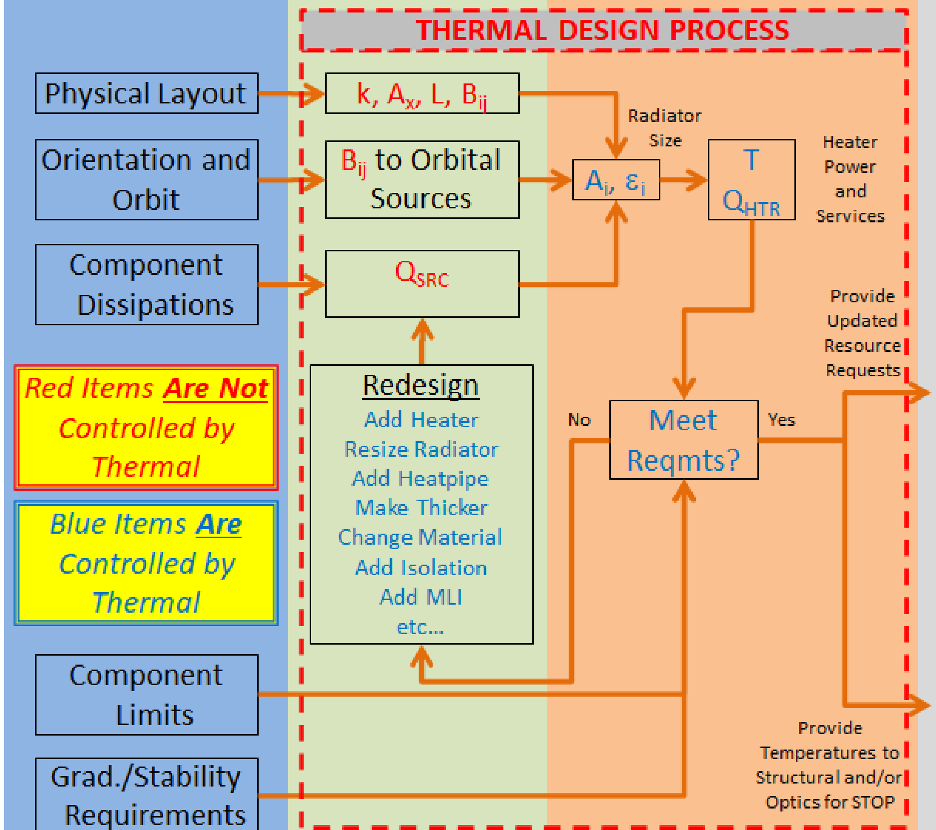
GPM Observatory Stowed GMM

Thermal Math Models (TMM)

- ***The TMM includes all nodal definitions and their interdependence***
  - Nodes may have mass, be massless, or be held as a boundary (space temperature is typical boundary)
    - *Node mCp (Mass x Specific heat) may vary with time, temperature, or be arbitrarily user modified*
  - Nodes are connected by conductors, which may be radiative or linear
    - *Conductance may vary with time, temperature, or be arbitrarily user modified*
  - Nodes may have sources applied
    - *Could be electrical dissipation, heater, environmental backloading, etc.*
    - *Sources may vary with time, temperature, or be arbitrarily user modified*
  - Arrays may be defined for time or temperature related inputs
    - *Arrays are often referenced in the assignment of conductances, mCps, or sources*
- ***The TMM also contains instructions for what is to be solved (steady state, transient), simulation duration, solution accuracy control, result output frequency, etc.***
  - TMM runs are generally based on FORTRAN libraries which are included with the user data (Nodes, Conductors, etc) to compile an executable program which generates the requested data
- ***TMM Output is then evaluated to see if:***
  - Temperature predictions are within limits with margin
  - Heater power predictions are within budgets
  - Stability and Gradient requirements are met
  - Results may be mapped to structural model for thermal distortion/STOP analyses
- ***Thermal typically starts with bounding hot case to size the radiator, followed by bounding cold case to size heaters – !! instantaneous mission temperatures are not predicted !!***
  - Bounding case predicts and off-nominal cases (launch, calibrations) are evaluated as design matures

Thermal Analysis Process

Requirements | Subsystem Inputs | Thermal Outputs



GMM – Geometric Math Model – Used for calculating B_{ij}s between surfaces/nodes and Surfaces and Celestial Sources

TMM – Thermal Math Model – Network Analyzer for calculating Temperatures based on input Nodes, Conductors, and Sources

Basic Spacecraft Thermal Analysis

1D Conduction

$$Q_{ij} = k\, A_x / L * (T_i - T_j)$$

Q_{ij} – Heat Transfer between i and j (W)
k – Thermal Conductivity (W/m K)
A_x – Cross Sectional Area [w x t] (m^2)
L – Distance between T_i and T_j (m)
Isolation is Small k, Small A, and Large L

Radiation

$$Q_{ij} = \sigma\, \varepsilon_i A_i B_{ij}\, (T_i^4 - T_j^4)$$

σ – Stefan Boltzman (5.67e-8 W/m^2 K^4)
ε_i – IR Emissivity of Node i
A_i – Area of Node i – !! RADIATOR !!
B_{ij} – Interchange factor between i and j
*We let the **GMM** calculate these with a ray trace for both B_{ij} and Orbital Sources!!!*

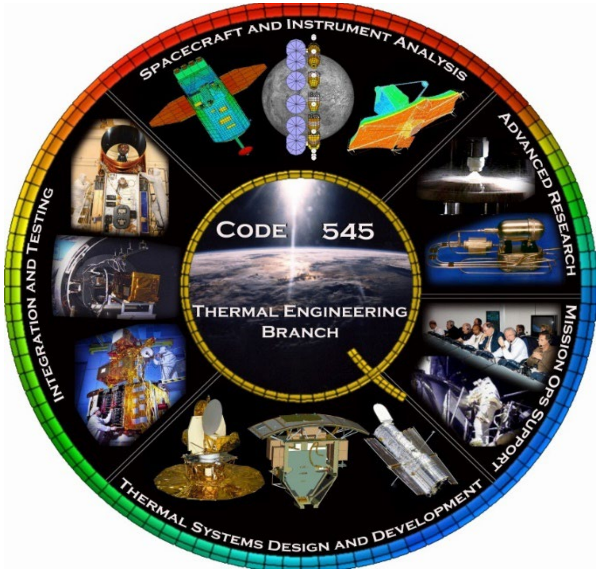
Energy Balance

$$mC_p dT/dt = Q_{SRC} + Q_{HTR} + Q_{IN} - Q_{OUT}$$

mC_p – Mass x Specific Heat (J/K)
dT/dt – Rate of Temp change wrt time (K/s)
Q_{SRC} – Input Heat Loads (W)
Q_{HTR} – Heater Powers (W)
*We enter all the Conductors, Sources, and Nodes into the **TMM** and the computer solves for temperatures*

We may have some convection, aero-heating, $k = f(T)$, $B_{ij} = f(t)$ or $Q_{SRC} = f(t)$, but essentially this is it !!!

Thermal Tool Overview



Thermal Analysis Tools

- ***A variety of commercial software tools are available to perform space based thermal analysis for the verification of thermal designs***
- ***Nearly all these tools use a similar approach:***
  - Compute Radiation Exchange between surfaces and represent them as heat flow paths between computation points (i.e. nodes)
  - Compute Radiation sources from Celestial Objects onto surfaces and represent them as applied heat loads onto computation points (i.e. nodes)
  - Combine the Radiation Conductors (Radks or GRs) and Celestial heat sources with: (1) additional conductive/convective and/or radiative couplings representing the heat flow paths throughout the design, (2) applied heat dissipations to represent the thermal design, (3) thermal capacitance of the design, (4) logic to simulate the behavior of thermal components (e.g. heaters), and (5) instructions governing the simulation boundary conditions and inputs
  - This combined model of sources, sinks, conductors, and capacitances forms an electrical network analogy which can be solved for temperatures at a nodal level as a function of time or under steady state conditions
- ***For Radiation Computations, the most common solution algorithm is the Monte Carlo Ray Trace (MCRT)***
- ***For Thermal Computations, the system of equations relating nodal temperature, heats, and conductors is often represented in matrix form as [G][T]=[Q], where the solution is either iterative or a matrix inversion approach to yield $[G]^{-1}[Q]=[T]$***

Current Commercial Thermal Analysis Tools

(Alphabetical Order, No hierarchy implied…)

- ***ESATAN-Thermal Modeling Suite (Radiation: ESARAD + Thermal: ESATAN)***
  - *Maintained by ITP Engines, primarily used by ESA*
- ***Space Systems Thermal [aka TMG] (Radiation + Thermal)***
  - *Maintained by Maya Heat Transfer Technologies in collaboration with Siemens*
- ***Systema (Radiation: Thermica + Thermal: Thermisol)***
  - *Maintained by Airbus, primarily used by ESA for projects with Airbus support*
- ***Thermal Desktop (Radiation: RadCAD + Thermal: SINDA/FLUINT)***
  - *Maintained by Cullimore and Ring Technologies, used by NASA*
- ***Thermal Synthesizer System [aka TSS] (Radiation: Radk,HeatRate + Thermal: SINDA/FLUINT)***
  - *Maintained by SpaceDesign Corporation, used by NASA*

- *Thermal Analysis Kit 2000 [aka TAK 2000]: (Thermal)*
  - *Maintained by K&K Associates, most often used by Ball Aerospace*
- *TRASYS [not really commercial but still available I think] (Radiation)*

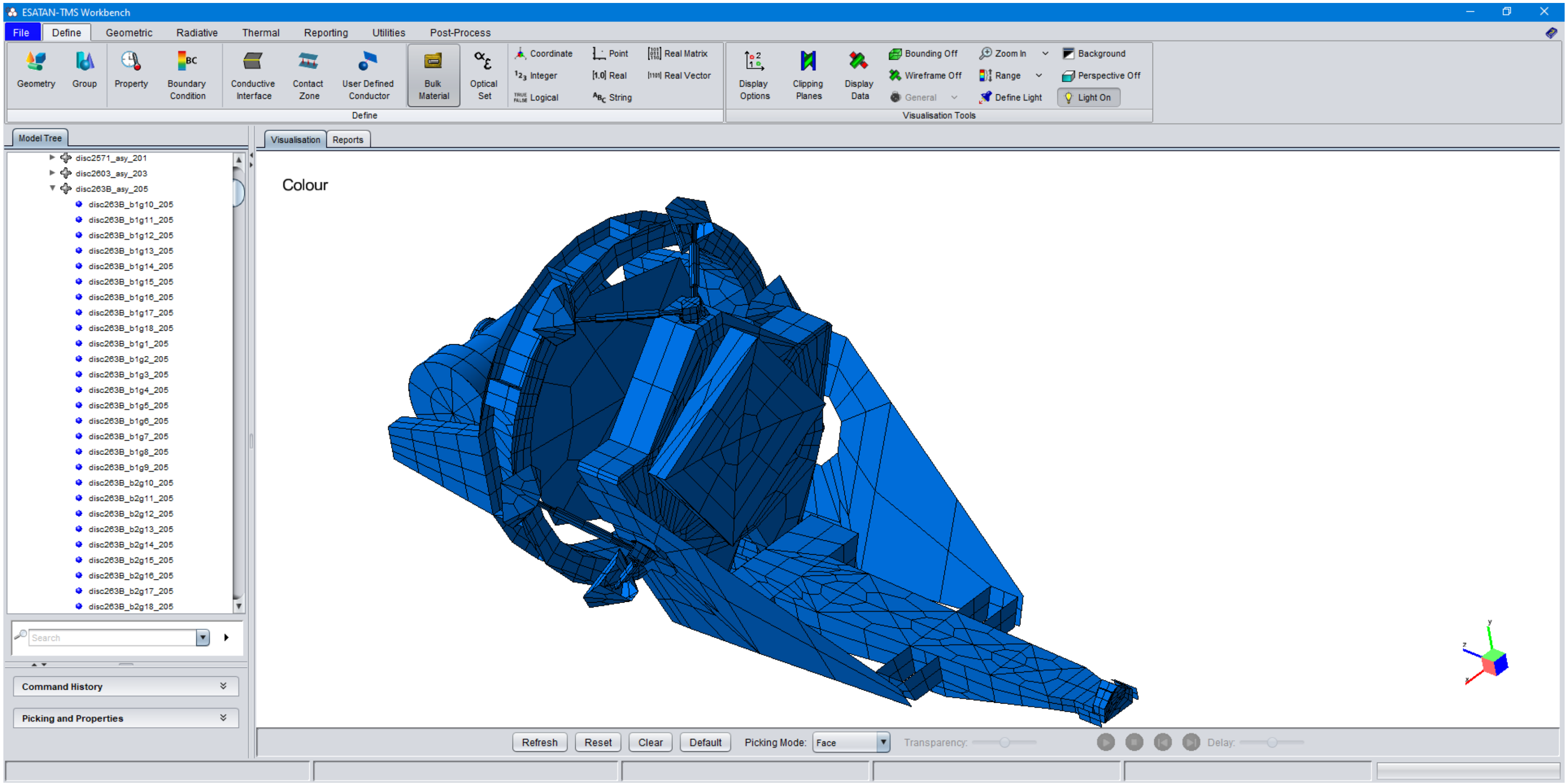
Tool Comparison Metrics

- ***Tool Name (Radiation: Sub-Name + Thermal: Sub-Name)***
  - *ENV: Software environment for tool (e.g. stand-alone, within Product XYZ, etc)*
  - *RAD: Analytical approach to solving radiation problem*
  - *CUT: Support for CAD Boolean subtraction operations (e.g. cutting)*
  - *COND: Analytical approach for conduction generation*
  - *OBJ: Support for thermal objects beyond internal surface conduction (e.g. Heaters)*
  - *FORMAT: Model file formats (ASCII/binary), file structure*
  - *CODE: Ability to add user customized code*
  - *SIM: Ability to define simulation cases (varying loads, orbits, properties, etc)*
  - *THERMAL: Model format (e.g. node numbers, submodel+nodes), compiler needs*

ESATAN-TMS Overview

- ***ESATAN-Thermal Modeling Suite (Radiation: ESARAD + Thermal: ESATAN)***
  - *ENV: Stand alone environment for model construction, visualization, and execution*
  - *RAD: Shape based modeling with MCRT for radiation*
  - *CUT: Extensive support for cutting operations*
  - *COND: Automatic Conductor Generation across interfaces, Some FE support – but automatic merging/renumbering of nodes*
  - *OBJ: Strong support for Contactor/Couplings, Conductors, Heatloads, Heaters*
  - *FORMAT: ASCII based input files, generally easy to follow Object Class structure*
  - *CODE: Extensibility through User Logic in Template file*
  - *SIM: Some customizability of configuration through Radiative/Thermal Cases*
  - *THERMAL: ASCII file - Submodel:node based solution using FORTRAN as underlying compiler along with application library*

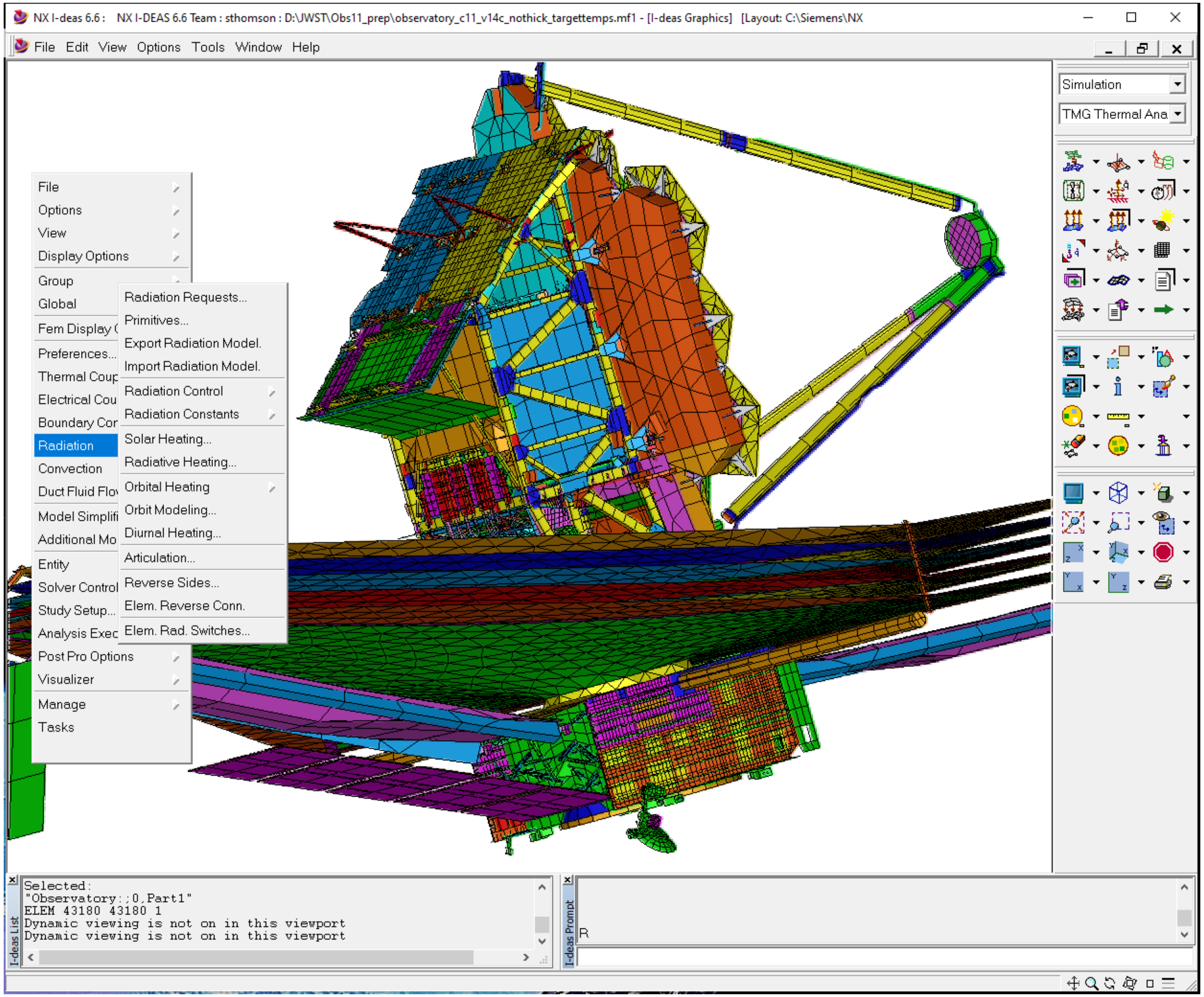
ESATAN-TMS


ESATAN-TMS Workbench
File
Define
Geometric
Radiative
Thermal
Reporting
Utilities
Post-Process
Geometry
Group
Property
Boundary Condition
Conductive Interface
Contact Zone
User Defined Conductor
Bulk Material
Optical Set
Coordinate
Integer
Logical
Point
Real
String
Real Matrix
Real Vector
Define
Display Options
Clipping Planes
Display Data
Bounding Off
Wireframe Off
General
Zoom In
Range
Define Light
Background
Perspective Off
Light On
Visualisation Tools
Model Tree
Visualisation
Reports
Colour
Search
Command History
Picking and Properties
Refresh
Reset
Clear
Default
Picking Mode:
Face
Transparency:
Delay:

Space Systems Thermal Overview

- ***Space Systems Thermal [aka TMG] (Radiation + Thermal)***
  - *ENV: Integrated as part of FEMAP or NX Environments*
  - *RAD: Radiation is generally diffuse view factors (VF) between elements. Hemicube method uses graphics card for quick VF calculations. Some support to use Surfaces, which are then meshed internally*
  - *CUT: No cutting support, but meshing can be used instead*
  - *COND: Uses Finite Elements to define Finite Volume for conduction computations*
  - *OBJ: Strong support for thermal objects (Couplings/Contactors, Heaters, HeatLoads, MLI)*
  - *FORMAT: Binary based input files for models, ASCII for some intermediate files*
  - *CODE: Likely possible to include user files, but not as frequently used*
  - *SIM: Extensive customizability of configuration through Study Setup*
  - *THERMAL: ASCII file – Node number-based solution (No Submodels)*

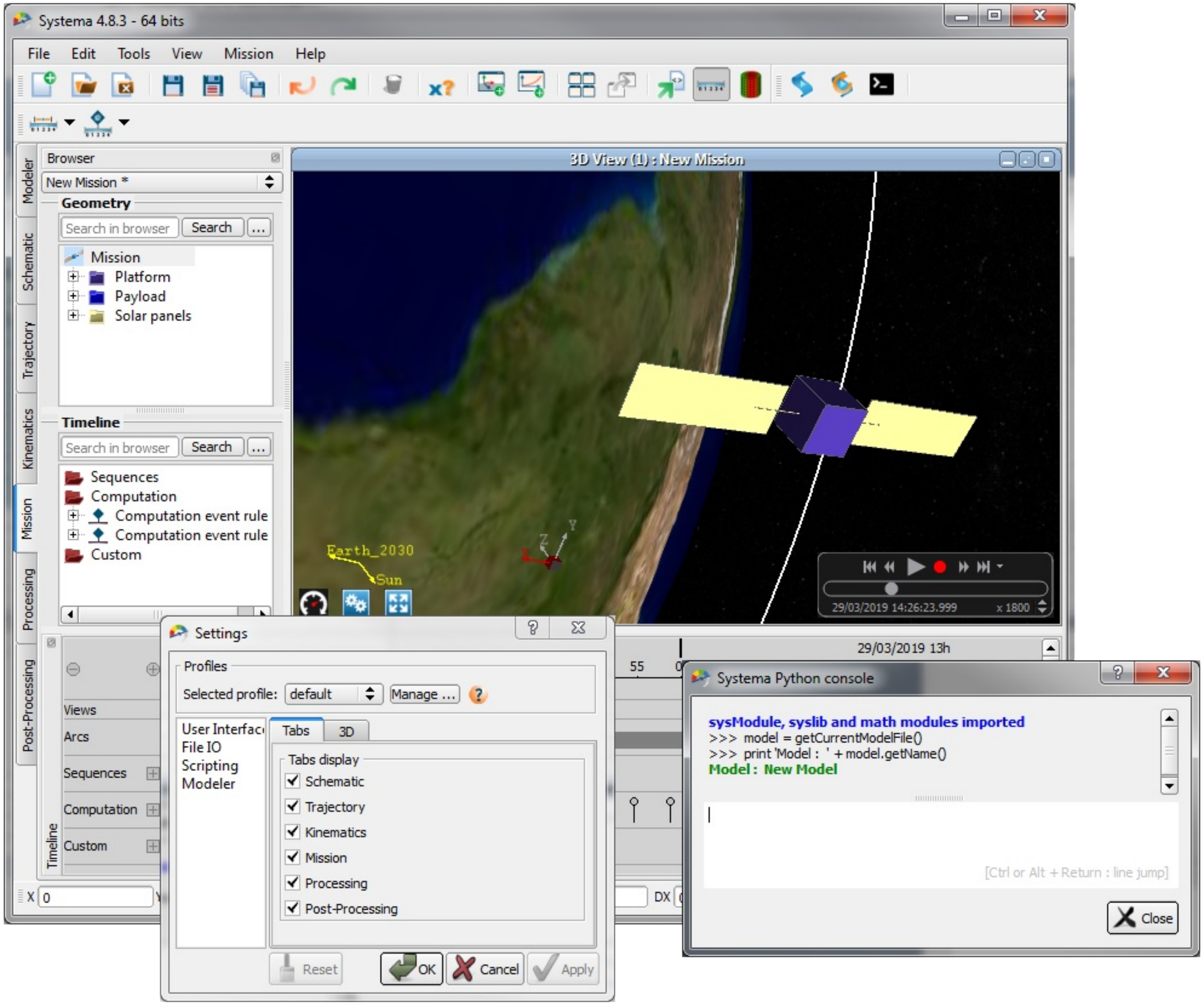
Space Systems Thermal



SYSTEMA

- ***Systema (Radiation: Thermica + Thermal: Thermisol)***
  - *ENV: Stand-alone environment for model construction, visualization, execution and postprocessing. Part of Systema framework which offers applications for analysis of power systems, space environment (ATOX, particles, solar protons, …), outgassing, Plume, Debris…*
  - *RAD: Shape based modelling with accurate analytical algorithm, based on advanced Quasi-Monte-Carlo raytracing (using Halton sequences)*
  - *CUT: Extensive support for cutting operations*
  - *COND: Advanced automatic conductor generation (RCN) incl. cross-element and surface contacts (not for cut elements)*
  - *OBJ: Support for Contactor/Couplings, Conductors, Heatloads, Heaters, convective couplings and aerothermal fluxes*
  - *FORMAT: ASCII based input files, xml format for Thermica, ESATAN/SINDA like format for Thermisol (MORTRAN)*
  - *CODE: Extensibility through User Logic in Skeleton files, Python API for batch processing*
  - *SIM: Extensive customizability of configuration through kinematics, high-precision trajectory (Orekit), mission cases. Solar system fully implemented (e.g. for Jupiter moon missions).*
  - *THERMAL: ASCII file, Submodel:node logic, FORTRAN as underlying compiler along with application library. Compatible with ESATAN input files. Option to use SINDA as thermal solver.*

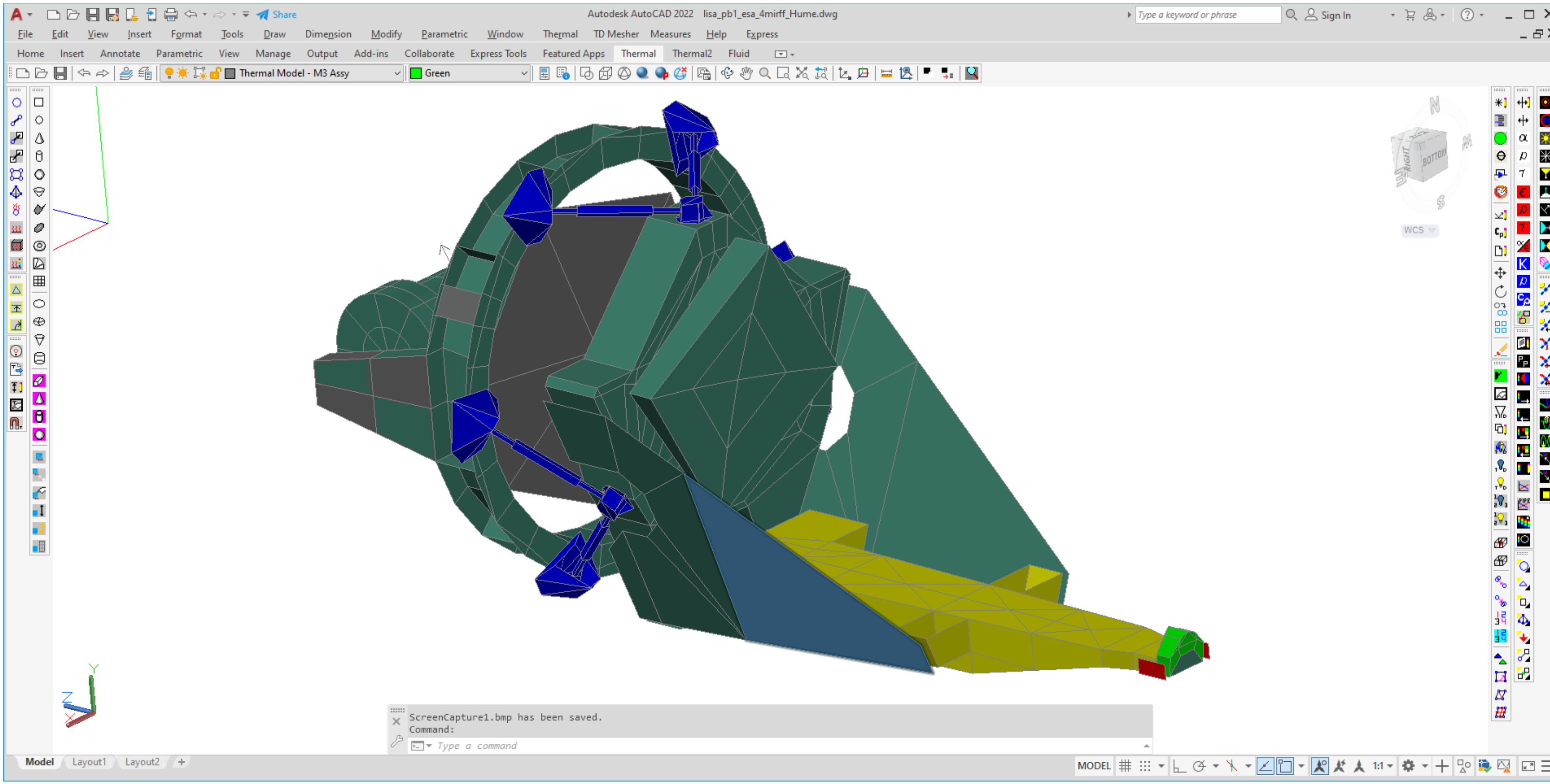
SYSTEMA


Systema 4.8.3 - 64 bits
File
Edit
Tools
View
Mission
Help
Browser
New Mission *
Geometry
Search in browser
Search
Mission
Platform
Payload
Solar panels
Timeline
Sequences
Computation
Computation event rule
Computation event rule
Custom
Modeler
Schematic
Trajectory
Kinematics
Mission
Processing
Post-Processing
3D View (1) : New Mission
Earth_2030
Sun
29/03/2019 14:26:23.999
x 1800
29/03/2019 13h
Views
Arcs
Settings
Profiles
Selected profile:
default
Manage ...
User Interface
File IO
Scripting
Modeler
Tabs
3D
Tabs display
Schematic
Trajectory
Kinematics
Mission
Processing
Post-Processing
Reset
OK
Cancel
Apply
Systema Python console
sysModule, syslib and math modules imported
>>> model = getCurrentModelFile()
>>> print 'Model : ' + model.getName()
Model : New Model
[Ctrl or Alt + Return : line jump]
Close

Thermal Desktop Overview

- ***Thermal Desktop (Radiation: RadCAD + Thermal: SINDA/FLUINT)***
  - *ENV: Integrated into AutoCAD environment for model construction, visualization, and execution*
  - *RAD: Shape or Finite Element based modeling with MCRT for radiation*
  - *CUT: Very minimal support for cutting operations (can disable node in surface)*
  - *COND: Surface Edge nodes/Finite element with node merging for conduction, solid tetrahedron mesher included*
  - *OBJ: Support for Contactor/Couplings, Conductors, Heatloads, Heaters, Measures*
  - *FORMAT: Binary dwg format for input files, but API allows access*
  - *CODE: Logic Objects to add User Code, Ability to add Code unique to CaseSets*
  - *SIM: Extensive customizability of configuration through Case Set Manager*
  - *THERMAL: ASCII file - Submodel:node based solution using FORTRAN as underlying compiler along with application library*

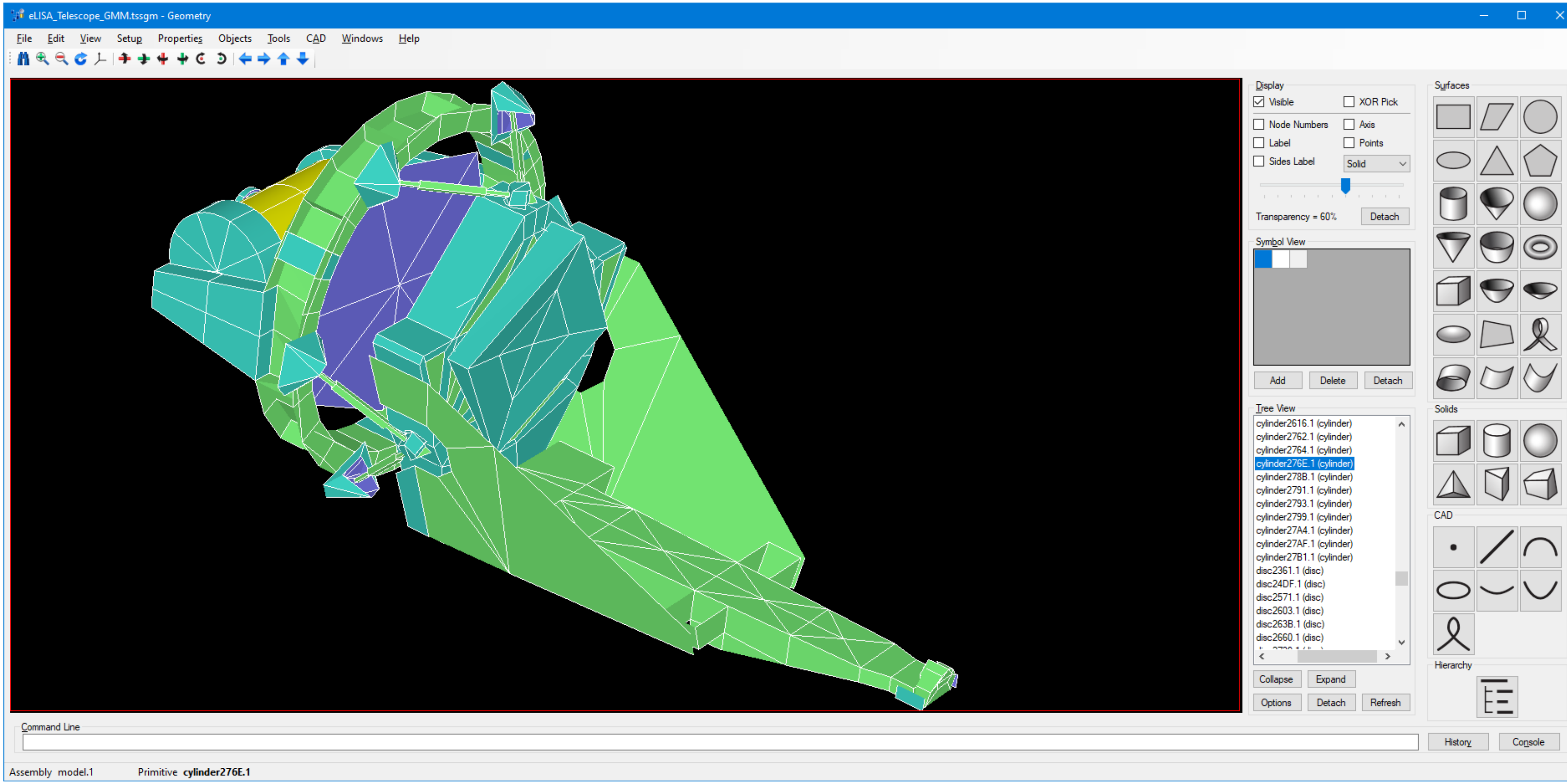
Thermal Desktop



Thermal Synthesizer System Overview

- ***Thermal Synthesizer System [aka TSS] (Radiation: Radk,HeatRate + Thermal: SINDA/FLUINT)***
  - *ENV: Stand alone environment, but different applications for administration, construction, visualization, and execution*
  - *RAD: Shape based modeling with MCRT for radiation, shapes may be FE or Centroid*
  - *CUT: Support for cutting operations, but only for radiation computations or FE*
  - *COND: CondCap support for centroids, FE support for FECC, mesher included*
  - *OBJ: No graphical objects for Support for Contactor/Couplings, Conductors, Heatloads, Heaters, Measures – users adds these to Thermal Math Model file*
  - *FORMAT: ASCII format for all input files*
  - *CODE: SINDA/FLUINT file requires considerable manual assembly for logic*
  - *SIM: No higher level simulation management provided*
  - *THERMAL: ASCII file - Submodel:node based solution using FORTRAN as underlying compiler along with application library*

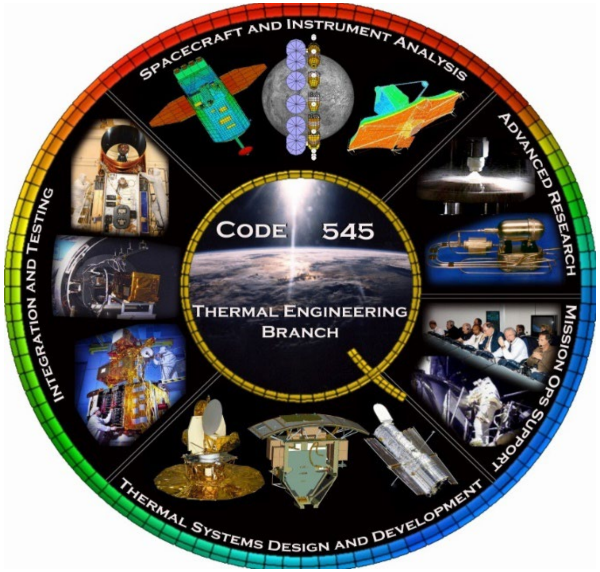
Thermal Synthesizer System



Thermal Desktop

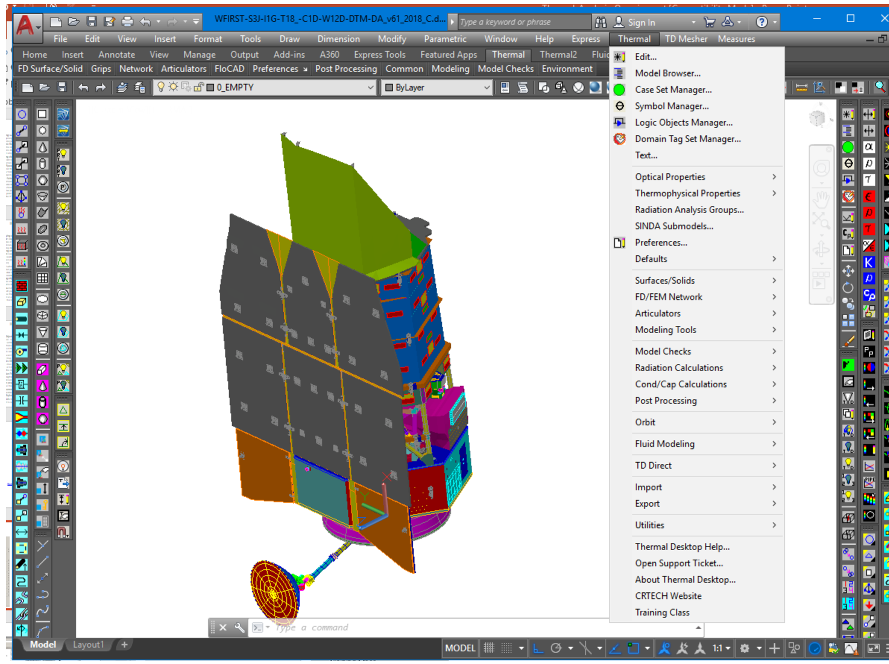
- ***It should be noted that there is considerable overlap amongst the tools in their capabilities and this package does not imply or specify an endorsement or recommendation of any particular tool***
- ***However, in the interest of demonstrating some of the techniques and capabilities available to the thermal analysis community, the subsequent material will focus on Thermal Desktop, as it is the most commonly utilized software at the Goddard Space Flight Center***
  - *The following slides will outline many of the capabilities utilizing screen shots from the Thermal Desktop software*
  - *Other tools may have superior, similar, or identical capabilities, but it is left as an exercise to the user to locate similar features in other tools*

Thermal Desktop Capabilities Overview



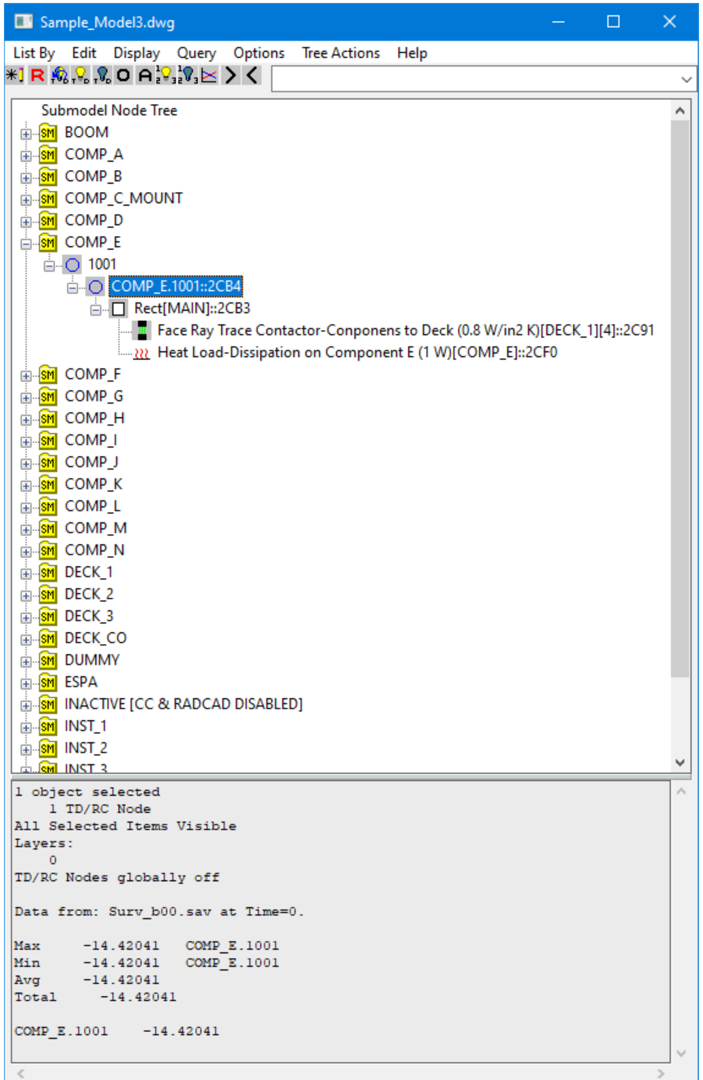
Thermal Desktop at a Glance

- Support of CAD import and operations via AutoCAD
- Thermal Desktop is a plug-in to AutoCAD and uses the graphics engine to display **thermal objects**
- This includes Surfaces, Assemblies, Nodes, Conductors, **Contactors**, Heat Loads, Heaters, etc.
- **Conduction formulation for centroid, edge-node, and finite elements**
- **Orbit** definition and visualization
- **Assemblies and Trackers** for moving components
- The RadCAD module computes radiative exchange factors and absorbed orbital heating using a Monte Carlo Ray Trace algorithm
- The FloCAD module case be used to simulate coupled thermal/1D flow type problems (pipe flow)
- The SINDA/FLUINT module computes temperatures based on an electrical network analogy (Conductor/Source/ Capacitor)
- Thermal Desktop can also display XY scatter plots and graphical contour plots of thermal results
- Temperature mapping to structural FEM for thermal distortion analysis
- **Mesh Controller** object for solid CAD parts
- **Measurement** locations for sensor representation



- Extensive parameterization and configuration options via **Symbols**, **Aliases**, and **Radiation Analysis Groups**
- **Model Browser** to manipulate and modify the model including visibility control via a number of methods
- Solver aspects are included in the **Case Set Manager** which allows definition of the thermal case(s) which generates and executes the SINDA thermal model based on surface properties and thermal objects

What is the Model Browser?



- Model browser is more than just surfaces and assemblies…
- User can show
  - Submodels/Nodes
  - Radiation analysis group
  - Optical properties
  - Material Properties
  - Assemblies/Trackers
  - Contactors
  - Conductors
  - Heaters
  - Heat Loads
- Any TD objects related to these are shown under the related branch
  - To Node and From nodes for contactor
  - Surfaces for Submodels/Nodes
  - Surfaces for Optical Properties
  - Symbols
  - Etc
- Information also displayed for selected objects
  - Temperatures, Heat Flows
  - Layer

What are Radiation Analysis Groups?

- Radiation Analysis Groups are simply enclosures
  - Only surfaces that are in the radiation analysis group can be "seen". They may be inactive or active, but if they are not in the Radiation Analysis Group, they do not exist to the ray trace
  - Why bother with this? It allows for multiple configurations to be included in a single model file. Internal and External models, varying instrument configurations such as reduced X,Y,Z and detailed W or detailed X and reduced W, Y, Z.

- When a radiation run is specified, the Radiation Analysis Group must also be specified

- Can run radiation directly through menu or through Case Set Manager

- Orbital view only displays surfaces in current Radiation Analysis Group

- Displaying active sides is only for current Radiation Analysis Group

- Can merge existing Radiation Analysis Groups to form new ones (boolean OR)

What are Symbols?

- Symbols are variables in Thermal Desktop
  - Defined in Symbol Manager and may be inter-dependent
  - Referenced just about anywhere in Thermal Desktop (opt property, length, rotation, power, etc)
  - Accessed by double clicking field where expression is to be entered
  - Wise to keep them to 32 characters or less if they are used in SINDA/FLUINT, which has 32 character limit on REGISTER DATA
  - May specify which symbols should pass through as REGISTERS to SINDA
    - Some symbols may not be referenced by TD entities, but could be by SINDA logic
- Symbol values may be over-ridden in Case Set Manager
  - Hot Case power vs. Cold Case power
- Symbols may communicate between Desktop and SINDA through the *Solver*
  - Can be used for optimization, but not overly well documented
  - Solver seeks max or min value by varying parameters within constraints set by user
  - Could solve for minimum emissivity allowed to maintain Temp > X: solver would go back and adjust emissivity and rerun radiation calculations, passing results through to SINDA for temperature solution.

What are Aliases?

- Aliases allow an Optical or Material property to be over-ridden for a particular case
  - Useful for trade studies or “what-ifs”
  - May also allow BOL and EOL to all be contained in the same model

- Can also over-ride entire Optical or Material property database for particular case

- To employ, follow these steps:
  - Define the alias and the default property associated with it (e.g White Paint = Z93p_BOL)
  - Assign the Alias (**not the property!**) to a surface
  - In the Case Set Manager, click on Props tab and then Alias button
  - Select the Alias and then the new value to over-ride the default for this case

What are Thermal Objects?

- Thermal Desktop allows a user to add graphical Thermal Objects
  - Conductors: user specified links between nodes/surfaces
  - Heat Load: fixed heat value
  - Heater: Thermostatically or proportionally controlled, SS behavior
  - Contactor (interface conductances based on overlapping length/area): see next slide
  - TECs: thermo electric cooler
  - Measures: locations in XYZ where nodal temperatures are interpolated to find measure value
  - Subdivided HeatPipe (with FloCAD module)

- Each of these objects is translated into SINDA logic in the SINDA output file

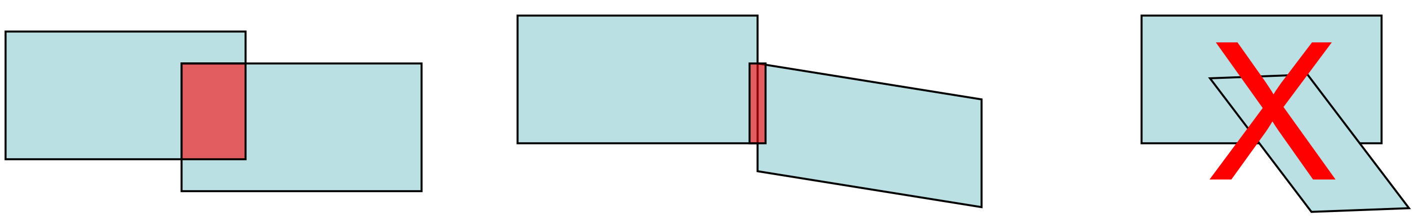
- Specific SINDA code may also be added
  - Logic Manager allows user to define code to be executed at defined points (e.g. before build, end of run, VARIABLES 0,1,2 for specified submodel)
  - Each case set may also include user defined code for all SINDA blocks
  - Anything in Logic Manager applies to ALL CASE SETS
  - Anything specific to a Case Set applies ONLY TO THAT CASE SET

What are Measurement Points?

- Measurement points are discrete locations intended to represent a sensor location
- Measures are located in 3D space and are separate from the underlying model mesh. A measure locates the projected point onto test surfaces and, if within tolerance, performs interpolation to determine the temperature at that location. As such, the mesh or nodalization may change, even if the measure does not
- Measures may:
  - Be processed in Thermal Desktop only (post run and based on results)
  - Output as Registers to SINDA
  - Output as Nodes to SINDA

What are Contactors?

- Contactors find a common length or area between two sets of surfaces
  - Useful for “chip on board” or “panel to panel” type interfaces
  - Overlapping area/length multiplied by user interface (W/m^2 K or W/m K) to generate SINDA conductors
- From Set and To Set
  - From Set, lengths: edges to consider must also be specified for surfaces
  - From Set broken down into smaller sub-surfaces/sub-lengths
  - Each sub-surface/sub-length is evaluated to determine which surface/length of the To Set is closest. Node from this is then assigned as Node j of SINDA conductor
  - Multiple conductors between same nodes merged in final output
  - Best to use Ray Trace instead of Closest Point algorithm for areas
- Cannot be used to match edge to area (i.e. card into motherboard)
- Wise to specify tolerance for what is considered overlapping (typically < 1 cm)



What is the Difference Between Centroid, Edge Node, and FEM?

- Many older models utilize centroids
  - Temperature is solved for the "center" of the surface/object
  - Conductors often manually input

- FEM more commonly used now
  - Temperatures solved at "corners"
  - N x M subdivision yields (N+1) x (M+1) nodes
  - Diagonal terms included and may be negative
  - Resulting Conduction network no longer represents "kA/L" heat flow between nodes
  - Node merging necessary to "connect" sides of elements
  - Desktop includes option to merge coincident nodes, TSS does not
  - TSS subdivides surface internally for ray trace, Desktop uses Shape functions to apportion appropriate energy to each node when ray intersects surface
  - Can surface coat free faces of solid elements with zero thickness elements
    - Only 2D elements can radiate

- Edge Nodes are only available in Thermal Desktop
  - Similar to FEM nodes except:
    - (1) NO Diagonal terms included. Only kA/L relations to adjacent nodes.
    - (2) Resulting Conduction network DOES represent heat flow between nodes
  - Can convert edge node solids/surfaces to FEM 2D/3D elements
    - But you cannot go back… (other than an UNDO just after command)

What are Mesh Controllers?

- Thermal Desktop includes a mesh controller that can be associated with a solid CAD part

- The mesher has very coarse control over mesh density and exclusively utilizes tetrahedron solid elements and triangular elements for the surface coat
  - Best used for simple parts
  - Complex parts can quickly create an unwieldy mesh for thermal purposes

- The user has control over the display (e.g. wireframe, shading) and the properties of the solid elements as well as coatings/materials for a surface coat

- Four layers created for each Mesh Controller:
  - Mesh Controller
  - Solid Part
  - 3D elements
  - 2D elements

- Different mesh controllers may be used for multiple parts

What are Assemblies and Trackers?

- Assemblies represent related components whose position may be adjusted by modifying the assembly
  - Translation X, Y, and Z
  - Rotations about X, Y, and Z
  - Can be tied to symbol related to orbit position for slowly slewing components or deployment simulations
  - Can be tied to symbol and use Fast Spin computation option to simulate objects completing many revolutions between orbit calculation points (e.g. scan mirror)

- Trackers are similar to assemblies, but actively rotate to point towards specified object (e.g. sun)
  - May be disabled during eclipses
  - Rotations may be constrained to specified angle range
  - Can be programmed to be active or locked
  - May be nested for Azimuth/Elevation type pointing

What Orbit Options are Available?

- Thermal Desktop has a variety of features supporting orbital mechanics including:
  - Basic Beta Angle orbits
  - Keplerian Orbit definitions
  - Trajectory Data (Solar, Planet vector list)
  - Surface and sky modeling
  - Free Molecular Heating

- Solar, Albedo and Planet IR fluxes may be Time dependent

- Albedo and Planet IR fluxes may be Latitude/Longitude dependent

- Visualize vehicle in orbit
  - Single or Multiple positions
  - Animations
  - Moving geometry as a function of trackers/assemblies displayed
  - Display maps of planets/moon/sun for presentation quality images

- Visualize vehicle only from orbital location (e.g. Sun, Planet)

What is the Case Set Manager?

- Case Set Manager allows full thermal cases to be defined
  - Radiation Tasks to be run (Radks, HeatRates, Articulated runs, etc)
  - Symbols to be over-ridden
  - Properties to be over-ridden
  - SINDA file properties to be specified
    - Submodel specific data (Node, Conductor, Variables, etc)
    - Output parameters and intervals
    - Definition of run parameters (Convergence, SS/TR, End Time, etc)
    - Submodels to BUILD for solution
  - SINDA file to be specified, run and post-processed
  - Hand shaking between SINDA and Thermal Desktop

- Multiple Cases can be defined and submitted in a single “run”

- Intelligent logic determines if Radiation results need to be recalculated
  - Geometry or Optical Property Change

Do I need to learn AutoCad to use Desktop?

- Technically, no, but it sure does help!!! There are some key things that it helps to understand (in no particular order)…
  - Snap points: cursor snaps to endpoint, center, intersection, node, etc when close enough. User can set this with the OSNAP command and F3 to turn on/off
  - User Coordinate System (UCS): when selecting an arbitrary point in 3D space (i.e. not an endpoint, center, intersection, etc) on a 2D screen, there is a 2D plane defined by the user coordinate system. This plane can be moved as necessary by rotating about X, Y, or Z or alternatively by specifying an origin, a point on X and a point in the XY plane. Use the UCS command to define a new UCS or go back to the World Coordinate system.
  - Entering points via text: a point may be specified by entering its world X/Y/Z coordinates or using the @ prefix to be relative to the last point entered. Polar coordinates may be specified using < between the distance and angle. So @1<90, would be 1 unit in the Y direction relative to the last point defined
  - Layers: layers are a way (outside of Thermal Desktop commands) to control the visibility of Desktop or other drawing objects. Any objects on layers that are off or frozen are not displayed. The layer must both be on and thawed for the object to be displayed. You can make quick changes to existing layers through the combo box at the top or select the layers button for full access (create, delete, etc.)
  - Properties: every object has some basic AutoCad properties, such as color and layer. Access these by selecting the objects you want and typing in PROPERTIES or right click-Properties

You mean there's still more I need to know?

- Zooming: AutoCad provides some quick zoom commands. Z-E zooms to drawing extents, Z-W zooms prompts for a zoom window, Z-0.9x (or any other number) zooms to that percentage of screen size (in this case 0.9 times smaller)
- Dynamic Rotation: 3DORBIT brings up the model within a green circle. Moving the mouse within the circle will dynamically rotate the model. Moving the mouse outside the green circle will rotate about the current screen Z axis. There are also 4 circles at 90° increments that allow rotation only about the screen X or Y axis.
- Rotating objects in 3D space (not rotating the viewpoint): AutoCad's 3DROTATE (NOT ROTATE3D!!!) command allows the user to specify the axis of rotation and the amount to rotate the selected objects
- Paper Space vs. Model Space: When TD goes into post processing mode, your are viewing the model through a view port defined in paper space. The legend is in paper space as is the window frame, but the model is *through* the viewport. PS and MS toggles between paper and model space. Sometimes it is necessary to zoom out/in while in Paper space
- Object Selection: AutoCad commands can work as a Noun/Verb (Objects are already selected, now perform some action) or Verb/Noun (Perform some action and prompt for which objects). When selecting objects, if the object is within the pickbox, then that object is selected (might need to have edges within pickbox). If no object is found within the pickbox upon clicking, then a Window (everything completely within the Window) may be selected by moving right of the 1st pick point or a Crossing (everything completely within or crossing the boundary) by moving left of the 1st pick point.

How much more do I have to learn?!

(Not Much more now)

- Modifying objects: AutoCad native commands let you MOVE, ROTATE, COPY, ERASE, MIRROR, or ARRAY objects. Furthermore, when an object is selected, grips are displayed (small rectangles at midpoints, origins, end points, etc). By selecting and dragging a grip, you can also resize, move, or rotate objects. F8 can be used to toggle orthogonal point selection when modifying objects.
- Inserting one Drawing into Another: before the days of cut-and-paste, AutoCad had the WBLOCK command to export selected objects to a .dwg file and INSERT to import them. These are still available, but Copy with Basepoint and Paste are easier. Note that the AutoCad COPY command (copies within drawing) is different that the COPYCLIP command (copies to clipboard)
- Blocks: when importing CAD geometry (STEPIN, IGESIN, ACISIN) the entire model and any associated parts comes in as a single object. Then EXPLODE this object to remove the assembly and be able to access the subparts. May need to do this multiple times to drill down to solids from assemblies which import as blocks
- Aligning objects: AutoCad provides a handy command (3DALIGN and ALIGN) to allow the user to specify 3 points on a base object and 3 points on destination object(s) and rotate/translate the selected object into the new position
- Units: AutoCad does not necessarily have a specified unit set. That said, Desktop does. So, if you have built your geometry in inches (from a CAD import) but Desktop had units of meters, you could use the Desktop Preferences-Units option with the Don’t Scale Model to new Units checked or use the AutoCad SCALE command. The AutoCAD scale command has the added benefit that it can work on any objects, not just Desktop ones.

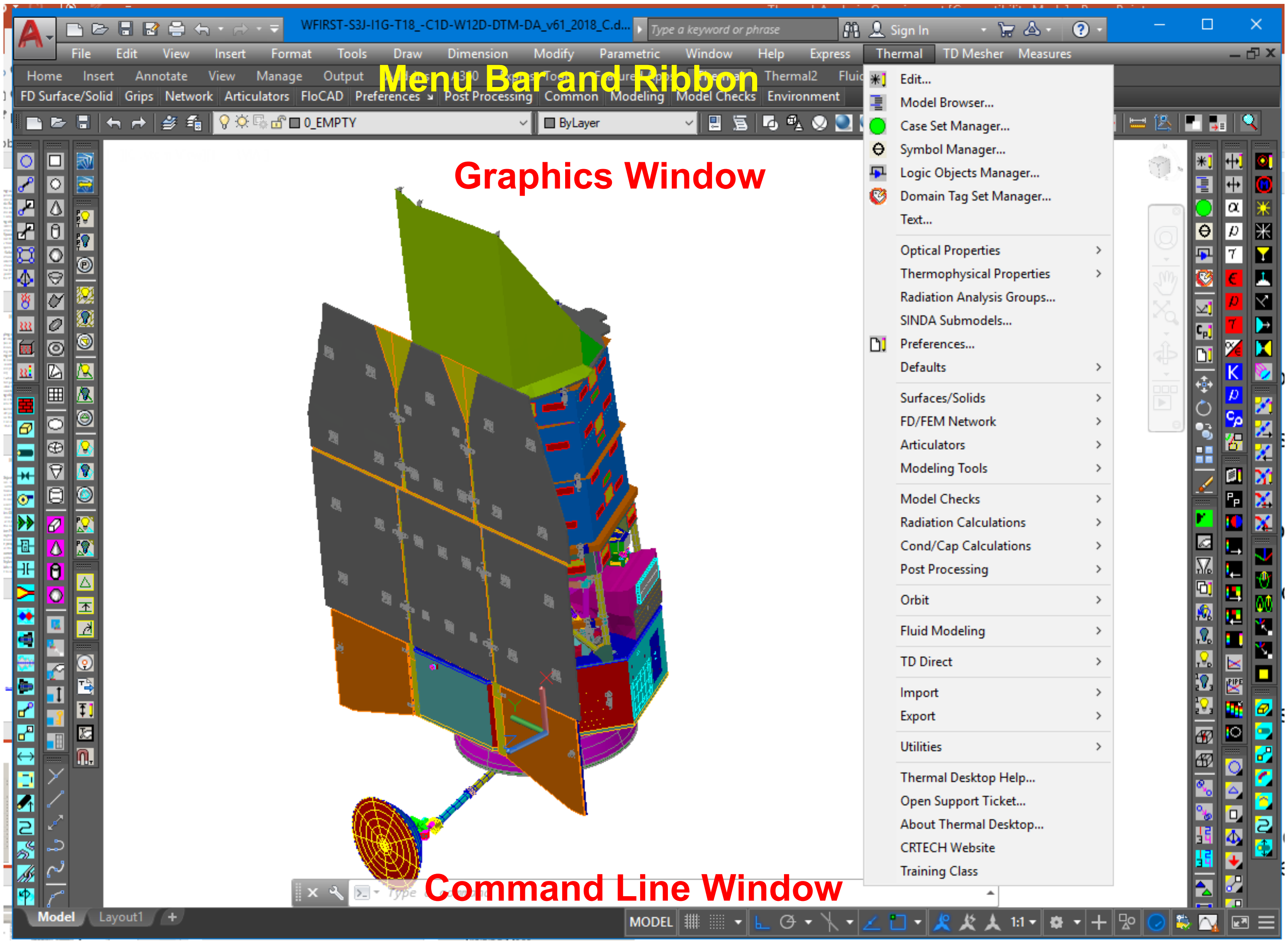
How much more do I have to learn?!

(Almost to the end now)

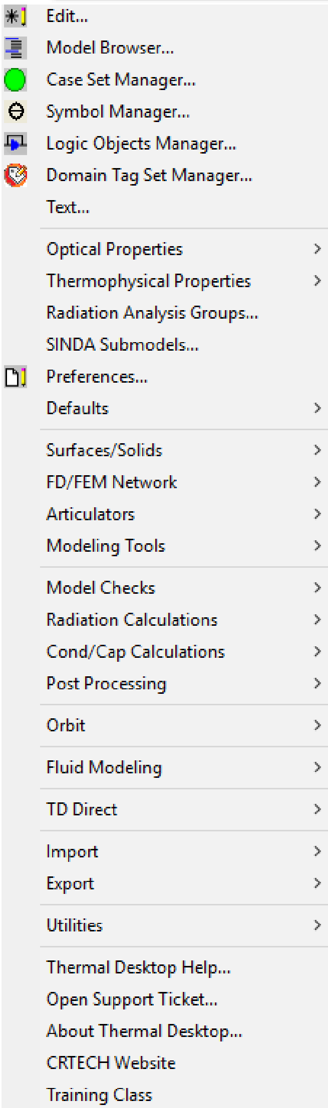
- Solid Object glitch: Sometimes CAD objects just don't show up, even though the layer is turned on. Using the PDMODE command with either 2, 1, or 0 (something different than the current setting) makes the screen refresh and show the objects properly. Sometimes displaying as wireframe can "kick" the system into working as well
- LIST: the LIST command will echo to the text window some details on what was selected. Useful to see if something is a block or solid
- SLICE: used to cut a solid using boolean/cutting operations. Works only on 3Dsolid type objects from other CAD programs
- Graphics Glitch: sometimes the viewpoint is so far away from the scene that zooming to extents does not work. Use the CAMERA command to re-position the viewpoint at 0,0,1 looking at 0,0,0 and switch to camera view. Then zoom to extents and see if this works. If so, delete the camera.
- Selection Preview or Hot Tracking: under Tools-Options Selection Preview tab, options exist for highlighting an object when a command is active or when no command is active when the cursor moves. Use this to disable hot-tracking if desired
- Bylayer properties: selecting the color as bylayer will make the color of the selected object be based on the color assigned to the layer on which that object resides
- Filter command: filter command may be used when selecting to narrow selections to meet user specified criteria (e.g. layer, object type, etc)
- VisualStyles: this command lets you tailor the view as you like it (i.e. show edges)

Thermal Desktop Interface

Thermal Desktop Interface in AutoCAD



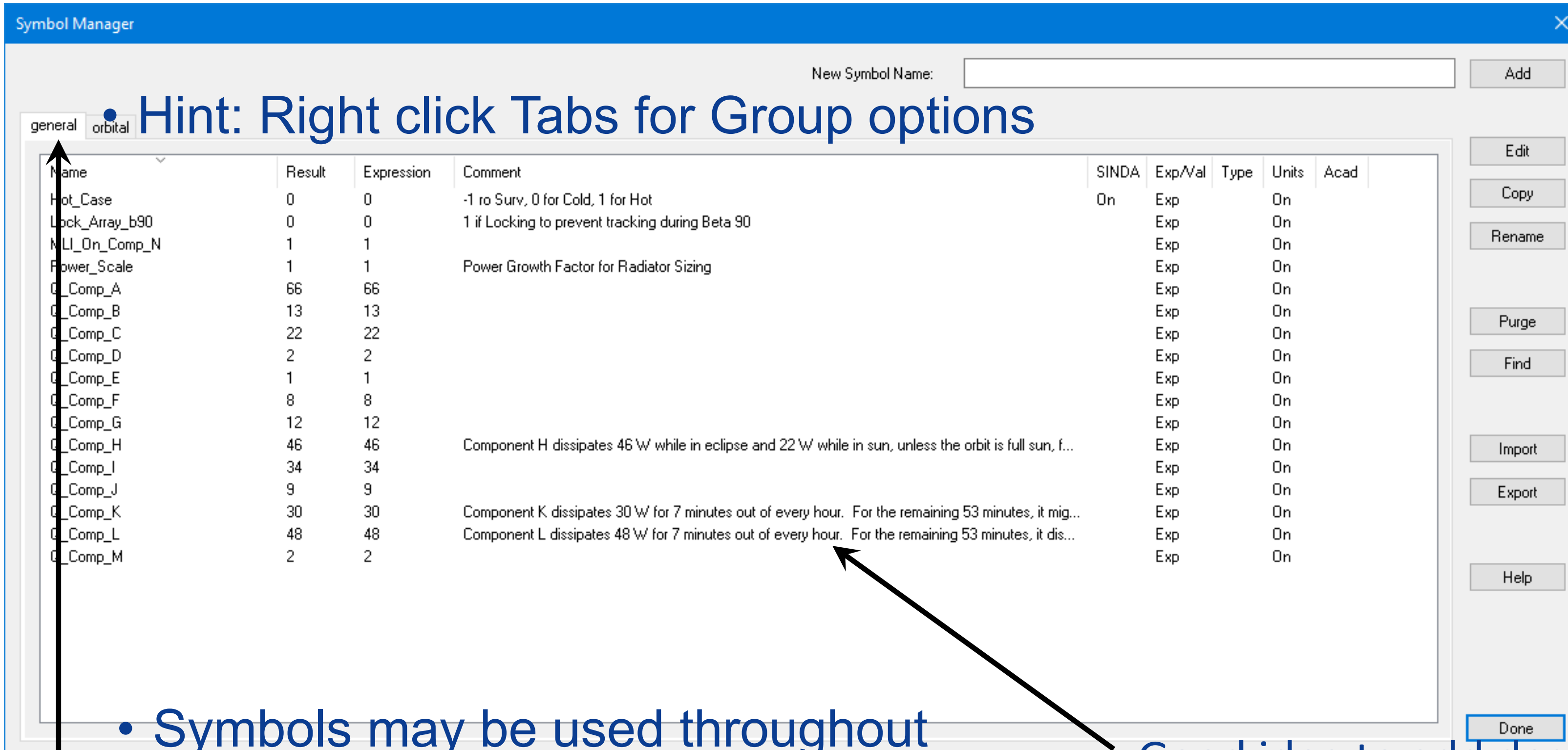
How can I find what I'm looking for in the Desktop Menu?



Desktop Menu follows a fairly logical flow…

1. Most common commands are at the top (Edit, Model Browser, Case Set Manager, Symbol Manager, Logic Objects)
2. Definition of Opt Props, Mat Props, Radiation Groups (Enclosures)
3. Preferences (Units, Graphics Size, Graphics Visibility, etc)
4. Defaults properties for Surfaces, Nodes, Conductors, etc.
5. Define Surfaces
6. Define Nodes and/or Network (Elements, Conductors, Contactors, Heat Load, Heaters, etc)
7. Define Articulation/Assembly Hierarchy
8. Modeling Tools (Renumber, FE Connectivity, etc.)
9. Model Checks (Active Sides, Opt Props, Thickness, Materials)
10. Offline Radiation or Conduction Runs (not through Case Set Manager)
11. Post Processing of Results
12. Orbit definition (may or may not be needed by general users)
13. Fluid Modeling (FloCAD/FLUINT)
14. General Utilities (Import, Export, Screen Capture, etc.)

Setting up Symbols



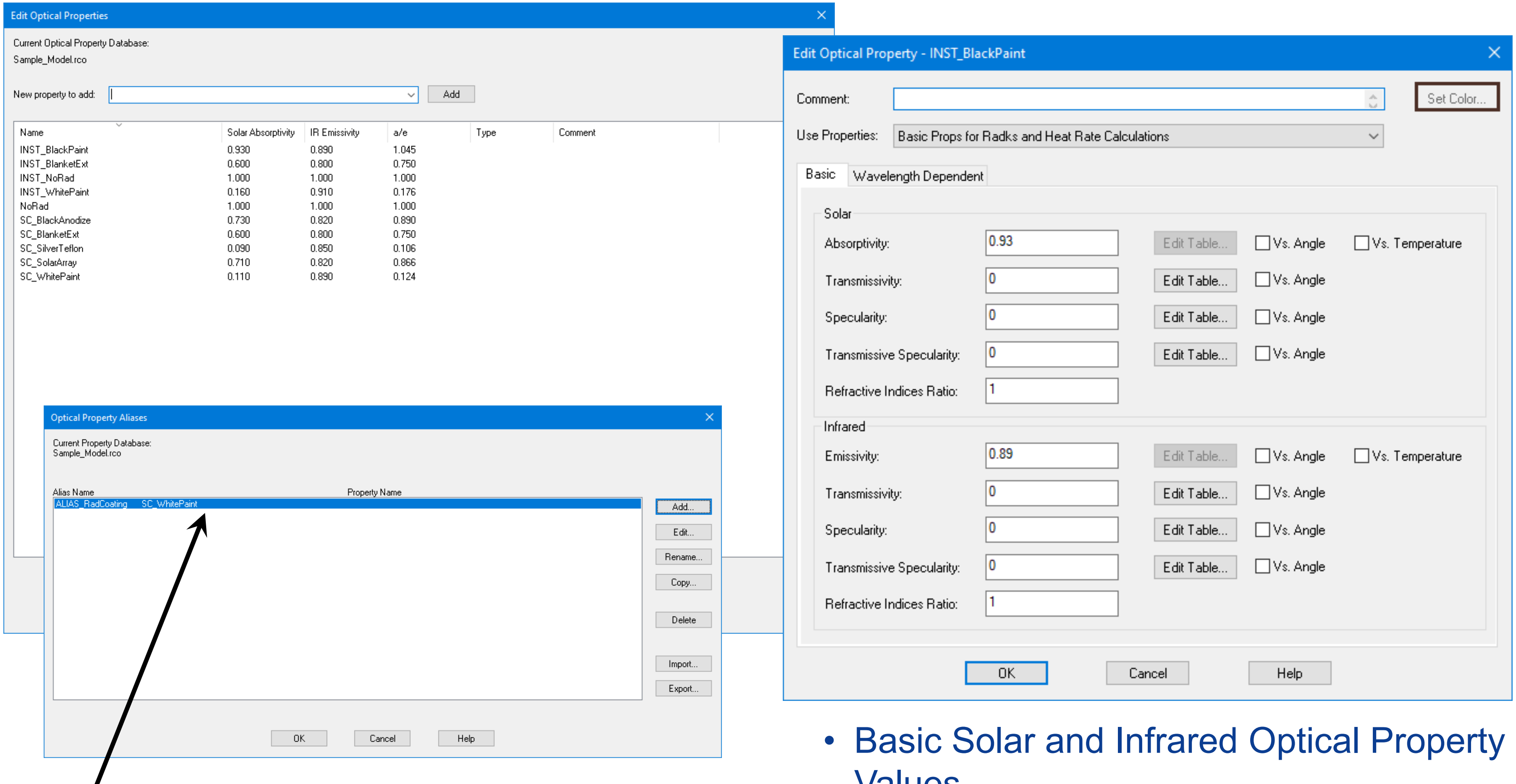
- Hint: Right click Tabs for Group options

- Can purge unused symbols
- Can import from external files

- Symbols may be used throughout Thermal Desktop
  - Symbols may depend on other symbols
  - Symbols may affect geometry (size, rot)
  - Can group Related Symbols together

- Good idea to add descriptions/ comments as you go
- (Test Condition) ? TrueValue : FalseValue allowed
  - (HGA_Depl == 1) ? 90 : 0

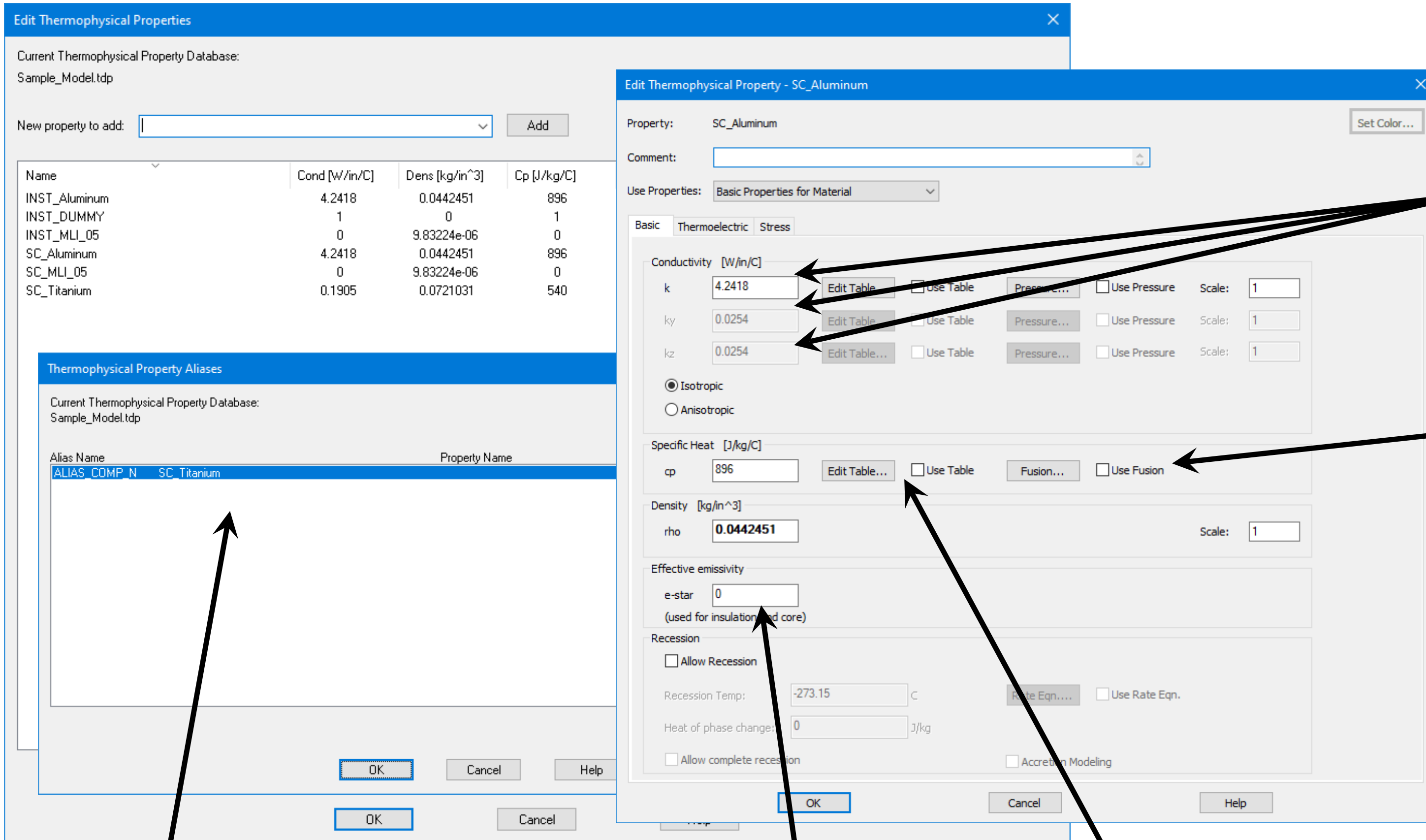
Defining Optical Properties



- Basic Solar and Infrared Optical Property Values
  - Angular or Temperature dependence is there (if you can really get that kind of data…)

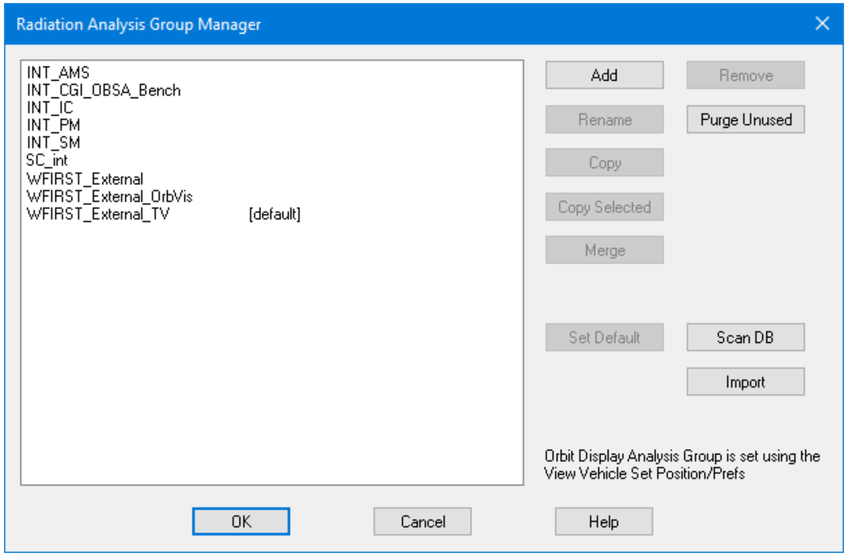
- Property Aliases let you redefine the properties as part of the Case Set by referencing a different Property

Defining Material Properties



- Anisotropic Properties
  - For solid Finite Elements, a material orienter is required to define X, Y, and Z
- Phase Change (Solid-Liquid) using FUSION
- Property Aliases let you redefine the properties as part of the Case Set by referencing a different Property
- Nothing really exotic with Material Properties
  - Temperature dependence is available if you have the data for k and Cp
  - ε^* also included for MLI

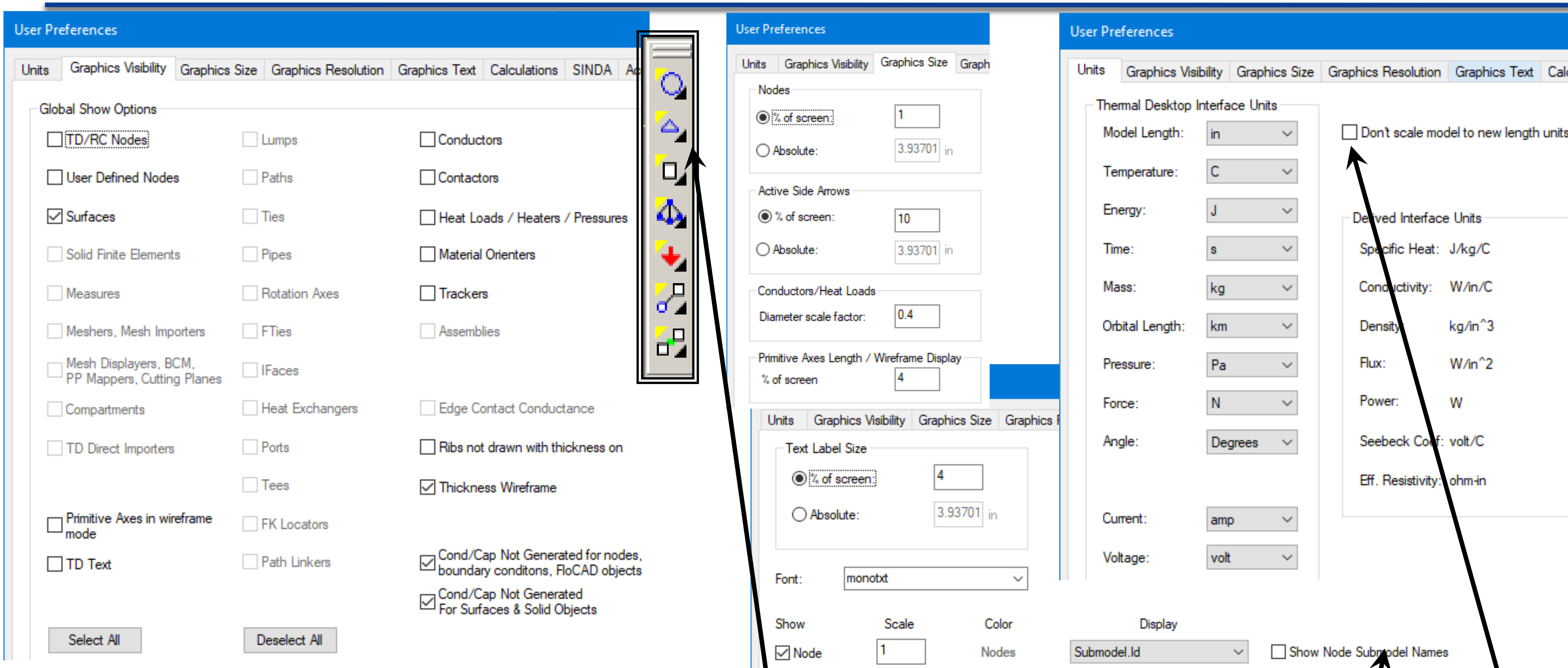
Defining Radiation Groups



- May Merge together multiple groups
- May Copy existing for future modification
- Default group often used for display of Active Sides or in Orbit Visualization to determine visual states for surfaces and radiative activity

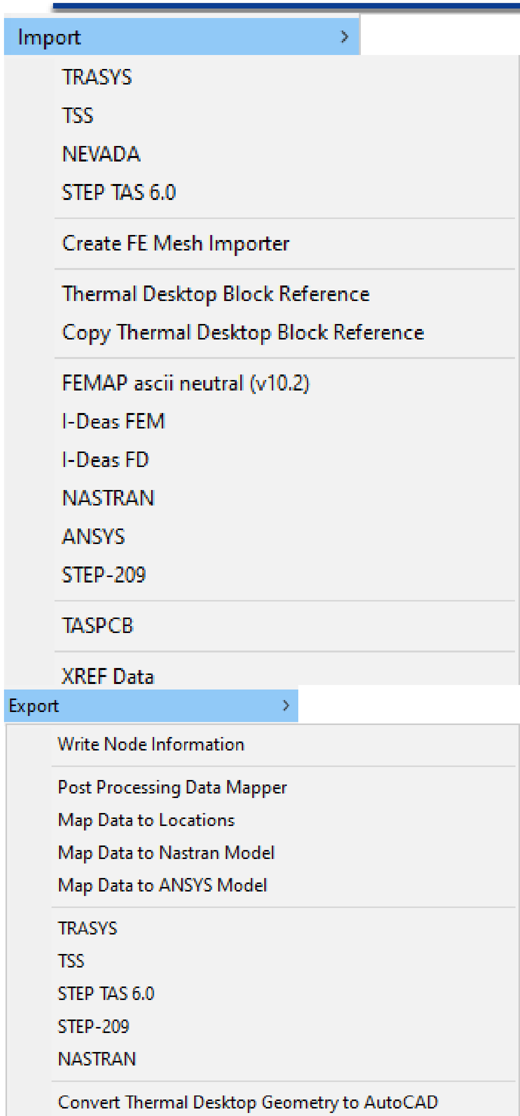
- Radiation Analysis Groups are a somewhat new concept to TSS / TRASYS users
  - Simply put, a Radiation Group is an enclosure
  - Only surfaces (or more specifically sides of surfaces) defined in an Analysis Group can “see” each other during the computations
- This allows multiple configurations to exist in a single model
  - Stowed and Deployed
  - Internal and External Couplings
  - TV and Flight
- So, Active Up/Down/Both/None may vary for each defined Analysis Group
- When specifying a Radiation Task, the Analysis Group must be specified
- May present problems when exporting to other codes, which do not support this feature…

Setting Preferences



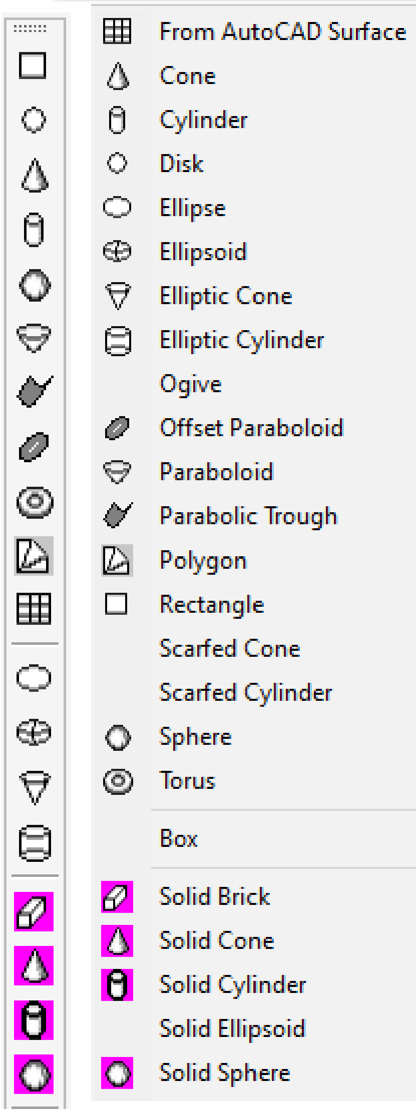
- Units, *Global* Object Visibility, Size
  - Toolbar toggles for most commonly used objects (Node, Planar FE, Surfaces, Solid FE, Heat Loads, Conductors, Contactors)
- Option to Include Submodel name when displaying node numbers
- Scale or not scale model when changing units

Importing Geometry or FE Models



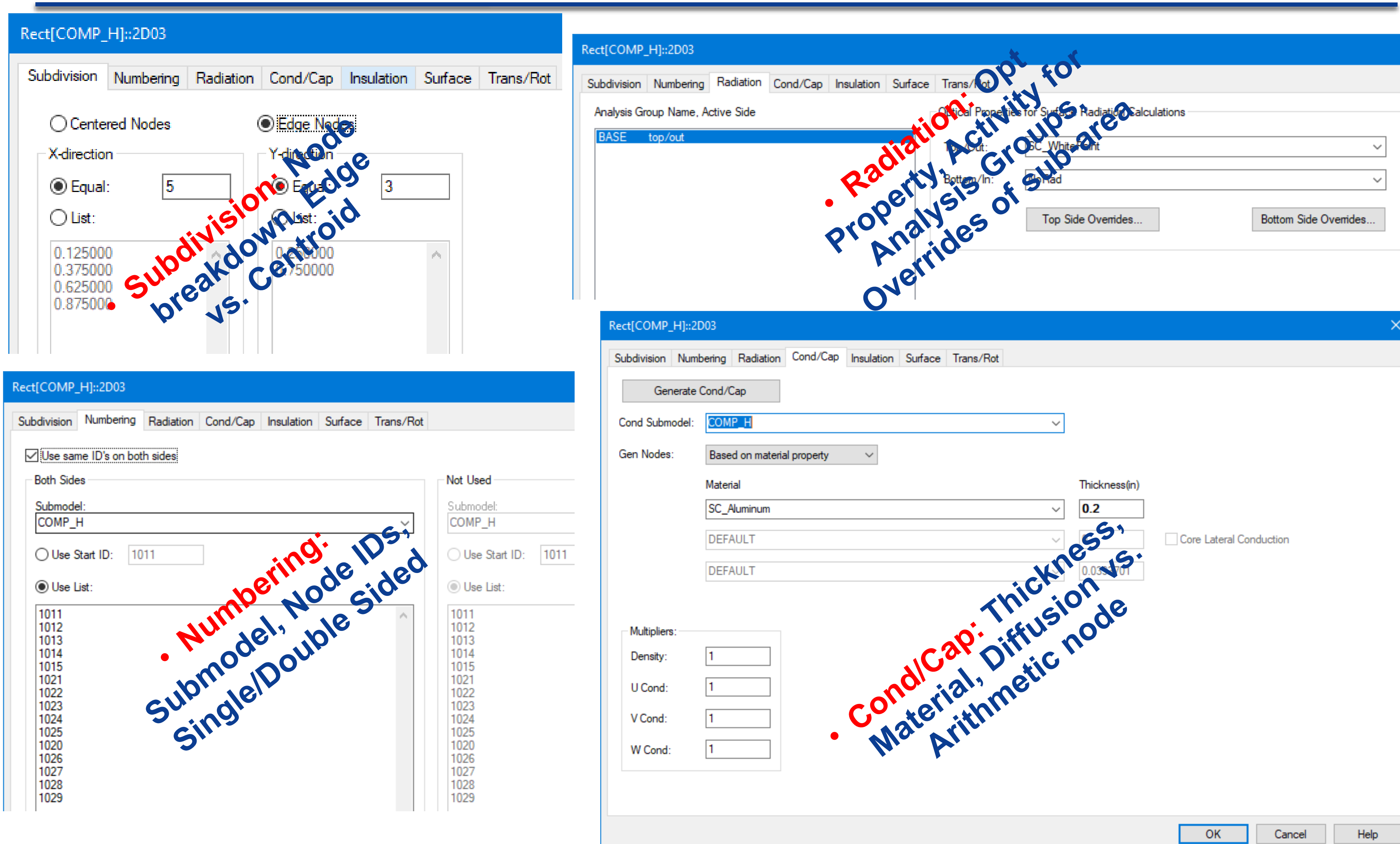
- 2 Methods to Import: Native AutoCAD and Thermal Desktop
- AutoCad imports geometry only, NOT THERMAL GEOMETRY (i.e. does not include optical properties, nodes, materials, etc.)
  - IMPORT (user specifies file type, e.g. IGES, STEP, ACIS)
  - Useful to have underlying geometry over which thermal surfaces will be added by the user
  - There is no magic button that will turn a solid model into a thermal model
- Thermal Desktop imports model geometry
  - TSS, TRASYS, FEMAP most common
  - Note: Thermal Desktop does not support the STEP-TAS converter itself and re-directs to an ESA website
  - Only includes properties that exist in imported code (e.g. NASTRAN may not have opt properties)

Adding Thermal Surfaces



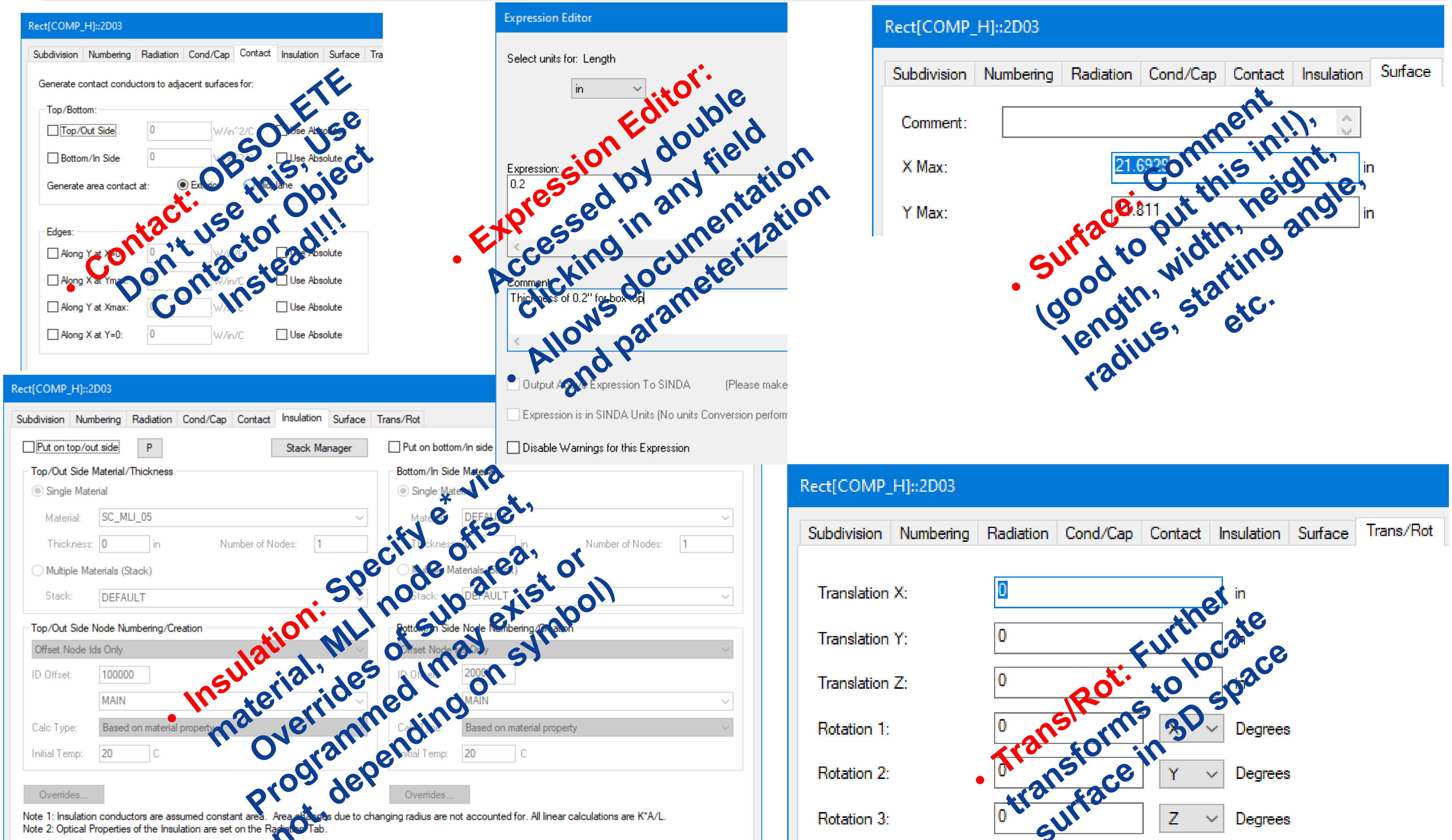
- Various Surface and Solid types
  - Finite elements are created differently (by selecting nodes)
- All surface properties are defined on tabs under Surface Properties
  - Subdivision: Node breakdown, Edge vs. Centroid
  - Numbering: Submodel, Node IDs, Single/Double Sided
  - Radiation: Optical Property, Activity for Analysis Groups, Overrides of sub-area
  - Cond/Cap: Thickness, Material, Diffusion vs. Arithmetic node
  - Contact (Don't use this!!! – may not even be displayed)
  - Insulation: Specify e* via material, MLI node offset, Overrides of sub area, Programmed (may exist or not depending on symbol)
  - Surface: Comment (good to put this in!!), length, width, height, radius, starting angle, etc.
  - Trans/Rot: Further transforms to locate surface in 3D space

Adding Thermal Surfaces (cont'd)



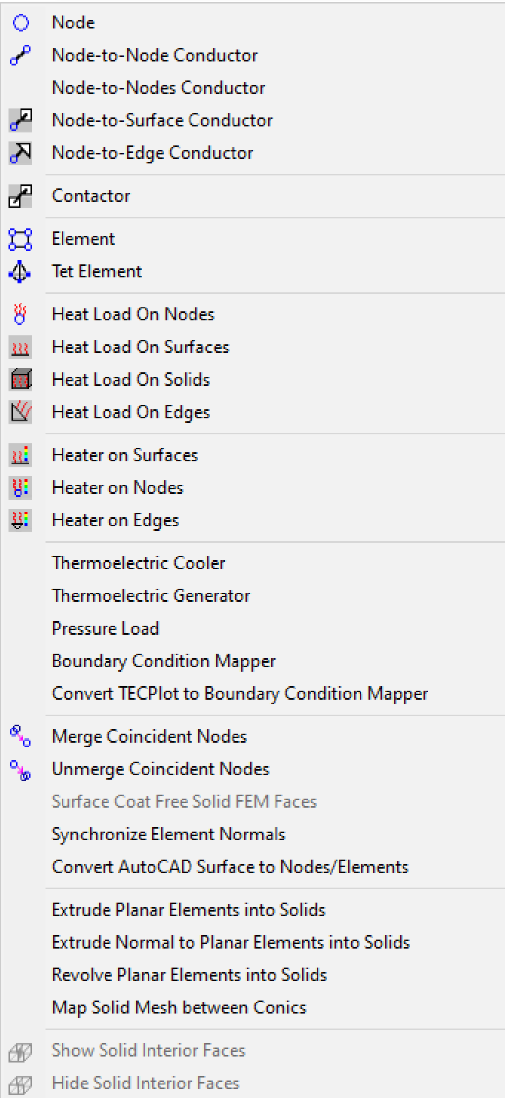
- **Subdivision:** Node breakdown Edge vs. Centroid
- **Radiation:** Opt Property, Activity for Analysis Groups, Overrides of Sub-area
- **Numbering:** Submodel, Node IDs, Single/Double Sided
- **Cond/Cap:** Thickness, Material, Diffusion vs. Arithmetic node

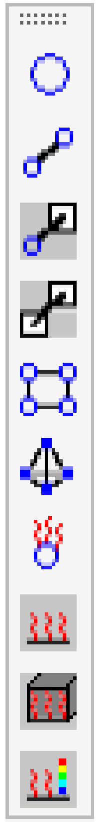
Adding Thermal Surfaces (cont'd)



- **Contact:** OBSOLETE Don't use this, Use Contactor Object Instead!!!
- **Expression Editor:** Accessed by double clicking in any field
  - Allows documentation and parameterization
- **Surface:** Comment (good to put this in!!), length, width, height, radius, starting angle, etc.
- **Insulation:** Specify e* via material, MLI node offset, Overrides of sub area, Programmed (may exist or not, depending on symbol)
- **Trans/Rot:** Further transforms to locate surface in 3D space

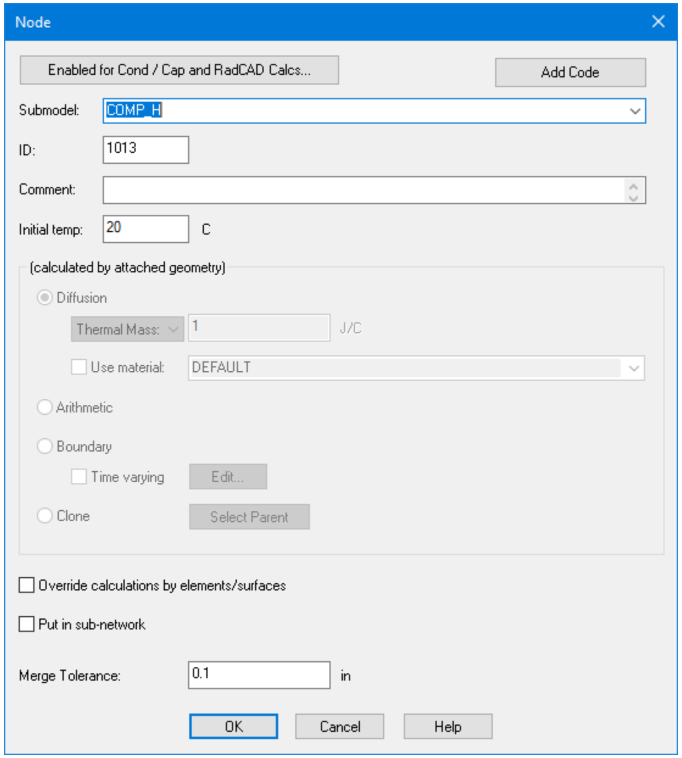
Adding Thermal Network Objects

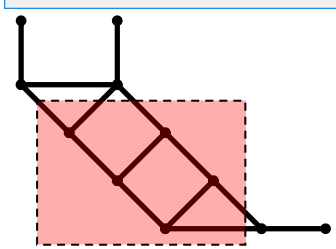




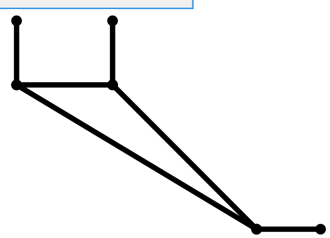
- Various non-geometric entities can be added
  - Node
  - Conductor
  - Contactor
  - Heat Load on Node
  - Heat Load on Surface
  - Heater
- As well as some geometric ones
  - Planar Finite element (FE) : Quad, Tri
  - Tetrahedron Finite Element
  - Wedge FE can be formed from extruded/ revolved Tri
  - Brick FE can be formed from extruded/ revolved Quad
  - Solid elements do not radiate…need to surface coat with zero thickness 2D elements to allow for radiation
- ***Node Merging is crucial to successful use of Edge Node or Finite Elements!!!***

Adding Thermal Network Objects: Node





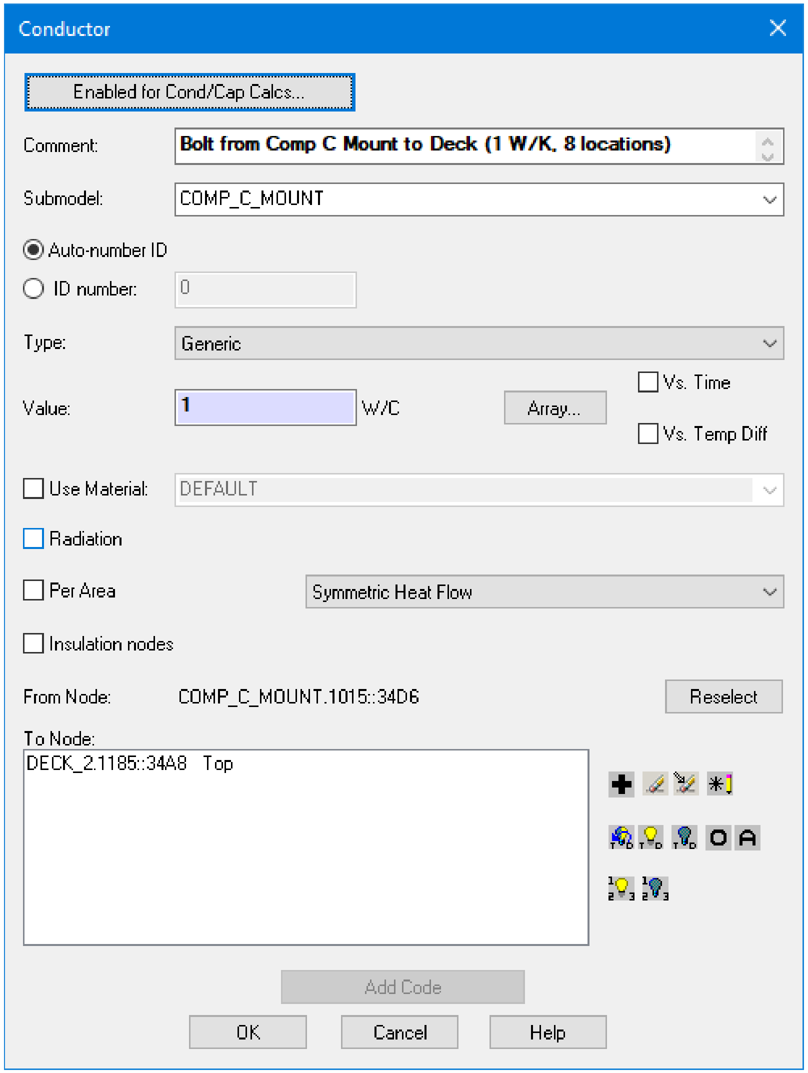
Original Network Subnetwork in Red



Solved Network

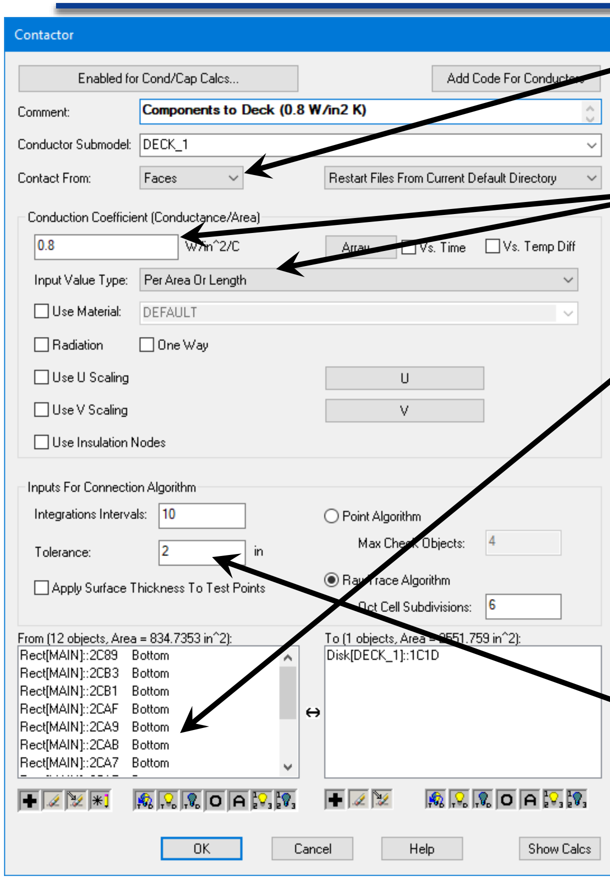
- Can Define…
  - Submodel, Initial Temp, ID
- Nodes assigned to a surface have their capacitance and type defined by the surface
  - User may over-ride this (e.g. make a node a boundary node)
  - Clone nodes have no properties and must have the base node type defined elsewhere.
  - Useful for making conductor connections is a separate location
- Subnetwork is a more advanced concept (see sketch at left)
  - Allows for interactions between a few discreet points to be solved only once in solver, and results mapped back later
  - Good for things like flexures, brackets

Adding Thermal Network Objects: Conductor

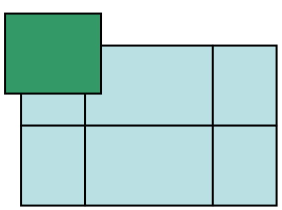


- Can Define…
  - Submodel, Conductance, ID
- Node-to-Node conductor only allows for two nodes to be selected at creation time
  - User can add or removed additional nodes later at Properties form
  - From Node may also be reselected
  - Note that Node *object* must be selected. A user may not specify the node by its identifier (e.g. SUBMODEL.ID)
- If Material is specified, then Value become multiplier (A/L)
- Can specify one-way or radiation type conductors (Default is linear)
- Per Area can be useful for heatpipes (Vapor to wall based on h)
- Ability to add/remove nodes, turn on/off visibility and node numbers of connected nodes

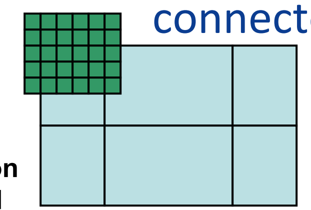
Adding Thermal Network Objects: Contactor



- Contactors find overlapping edge or area between two surfaces
  - Common Length/Area is then multiplied by user supplied factor. Absolute conductance also an option
  - Specify which face/edge to include in "from" surfaces
  - Allows for connections to still be made even with changes in break-down or nodalization of surfaces
- Two algorithms: point or ray trace
  - Edge only allows point algorithm
  - Ray Trace better for areas
- Generally better to not include thickness…
- Tolerance useful to ensure that improper connections are not made
  - Test points/sub-areas must be within tolerance to be connected (gap between surfaces/edges)

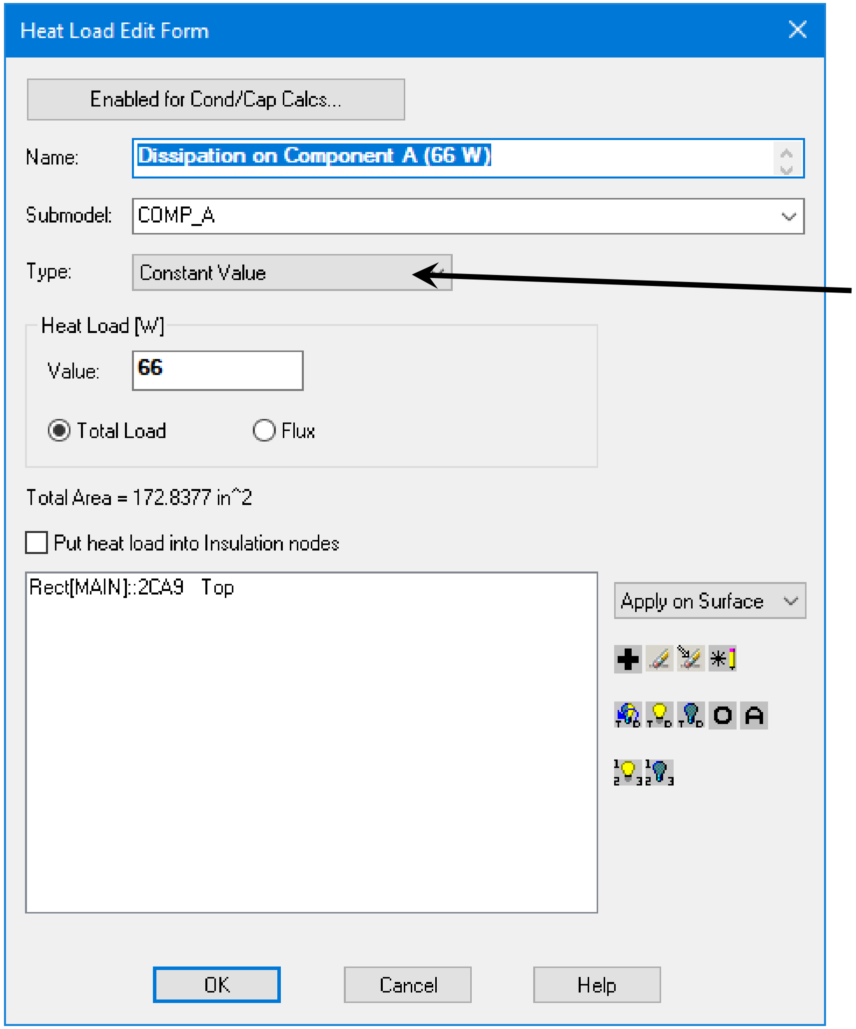


1. **Surface subdivided**
2. **Region containing sub areas determined**
3. **Sub area overlaps summed for each region**
4. **Summations multiplied by user factor**



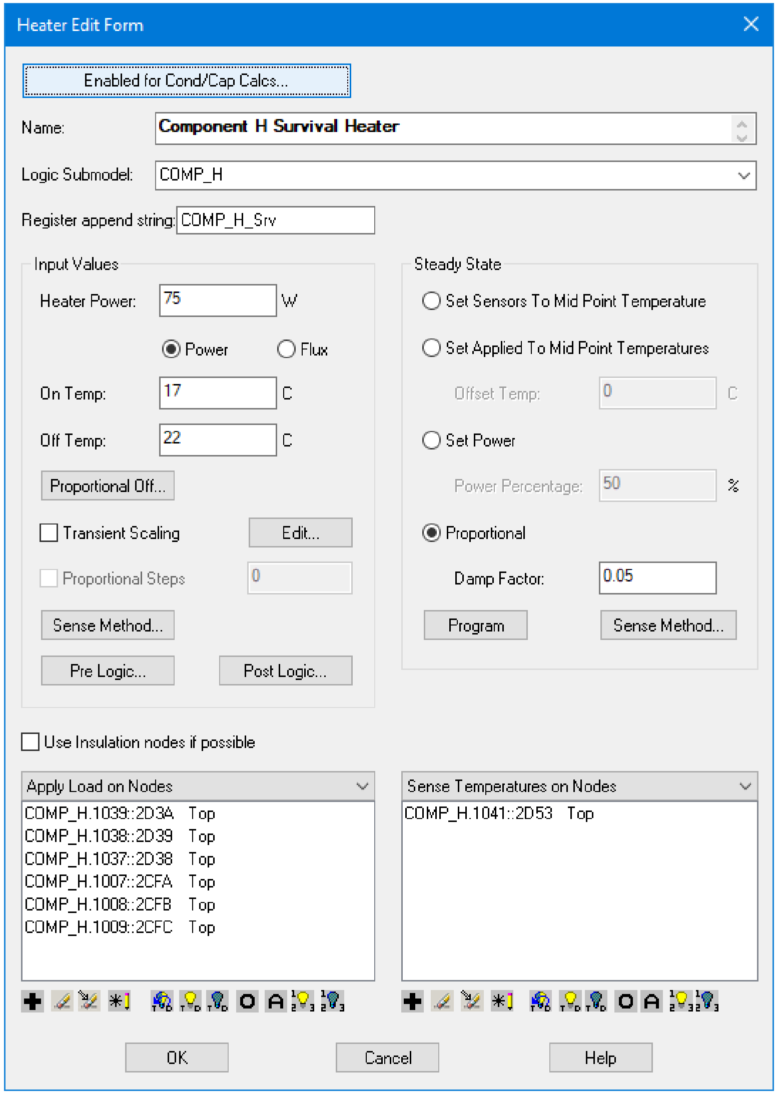
- Similar To/From Node control as Conductor (Visibility, Numbers, etc)

Adding Thermal Network Objects: Heat Load



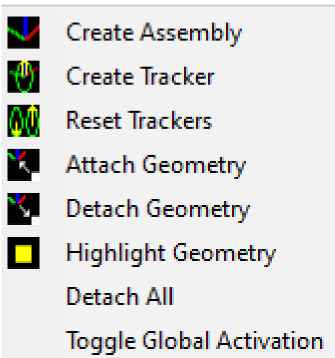
- Can Define…
  - Heat Load, Type, Submodel, Load/Flux, Side to apply heat
- May be added to a surface or a node
- Logic goes into VARIABLES0 section of specified submodel
- May be Time and/or Temperature dependent
  - Might be easier to handle Time or Temperature dependence via symbol manipulation in SINDA instead of through Thermal Desktop
- Whenever possible, output heat load as expression (allows greatest model flexibility)
- Similar Apply To Node control as Conductor (Visibility, Numbers, etc)

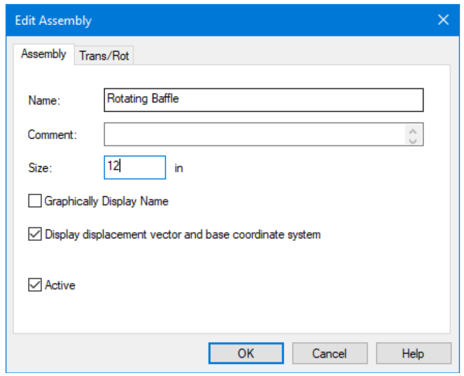
Adding Thermal Network Objects: Heater

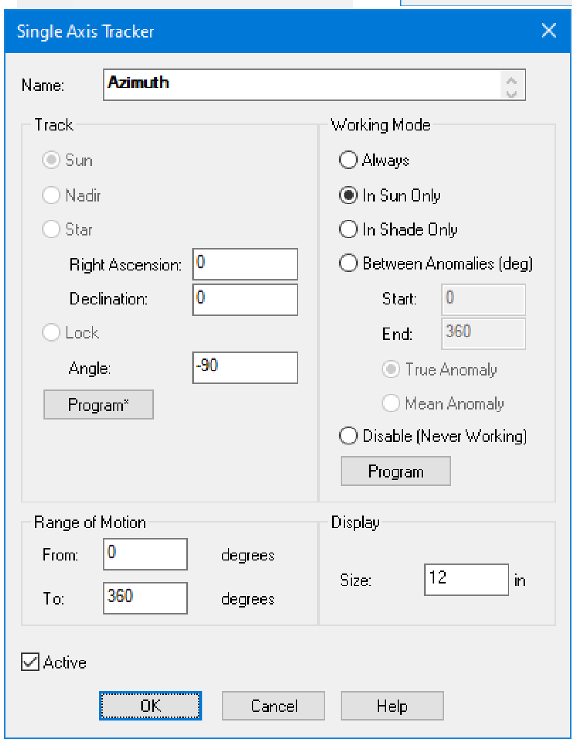


- Can Define…
  - Sensing Node/Surface(s), Application Node/Surface(s), Logic Submodel, Power, Type (Thermostatic, Prop), On/Off Setpoints, Steady state behavior
- Good practice to assign something meaningful to Register Append String
- Steady State behavior
  - Midpoint Temperature: holds as heater nodes regardless of power needed!! Debate as to whether to hold application location or sensing point. See TDHTR and TDREL functions in SINDA file
  - Percentage
  - Damped Proportional – recommended for most cases
- Sense Method
  - Average, Minimum, Maximum, User Specified
- If no Sense Temperature From defined, then Application node(s) is sensing location
  - A bit dangerous to use Set to Midpoint Temperature if Sensing Point and Application Point are different
- Heater can be converted to heatload by setting setpoints high
- Again, best to output Power and Setpoints as expressions when possible
- Similar To/From Node control as Conductor (Visibility, Numbers, etc)

Defining Trackers and Hierarchy

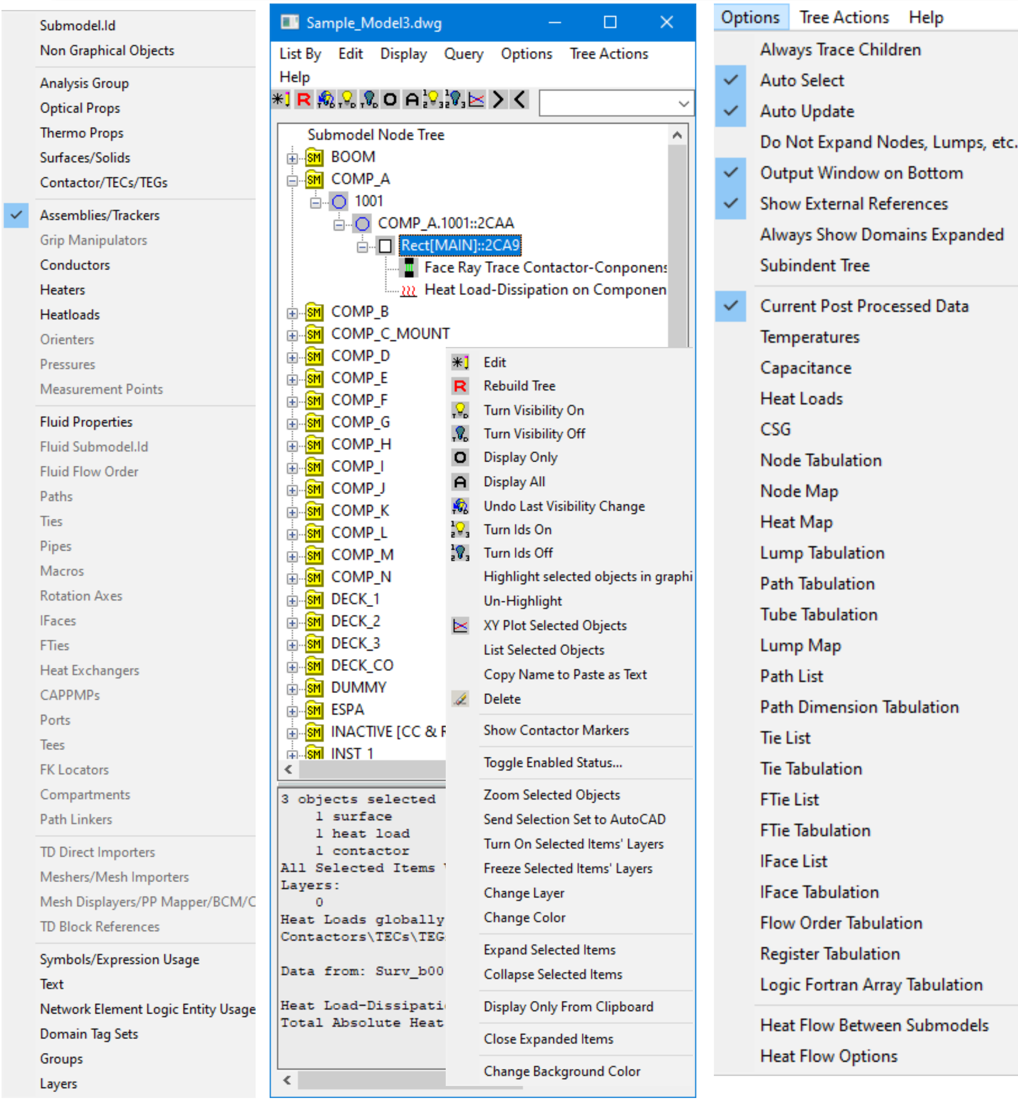






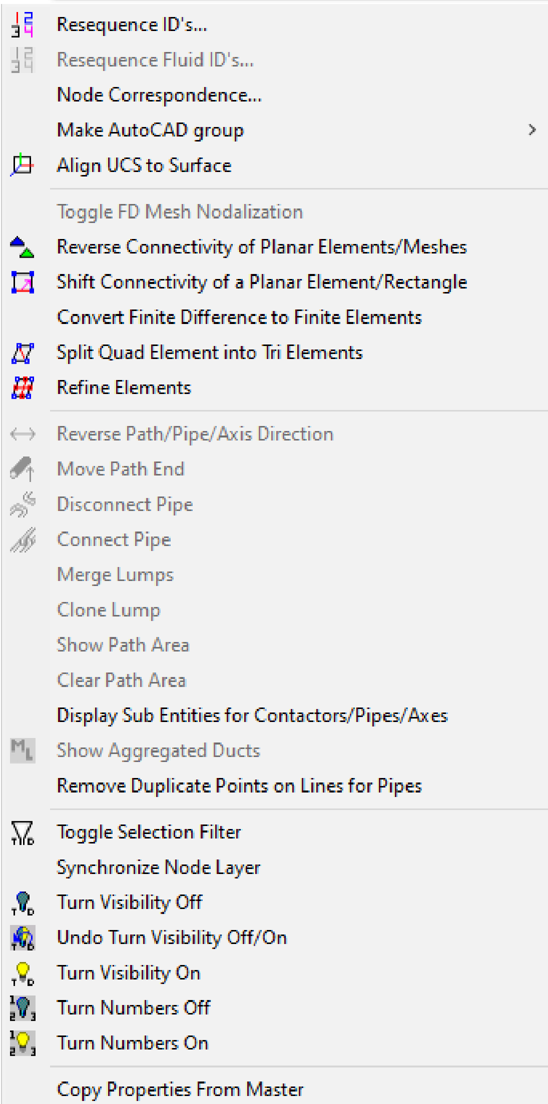
- Assemblies can be used to group together related geometry. *Geometric surfaces must be attached to assembly.*
  - If an Assembly is moved, all attached geometry moves with it (if it is active)
  - Assemblies may be attached to other assemblies
  - Assemblies my be dependent on symbols. Useful for allowing geometry to change based on configuration (stowed/deployed)
- Trackers are like assemblies, but rotate the attached geometry relative to some fixed point (e.g. the sun)
  - Trackers may also be nested
  - When trackers are active may be specified/programmed as well as what to track
  - Trackers may also be disabled in analysis
    - Disabled trackers are in their default state
  - Range of motion may also be specified
- Geometry attached to a tracker/assembly may not be attached to *another* tracker/assembly without first detaching it

Model Browser



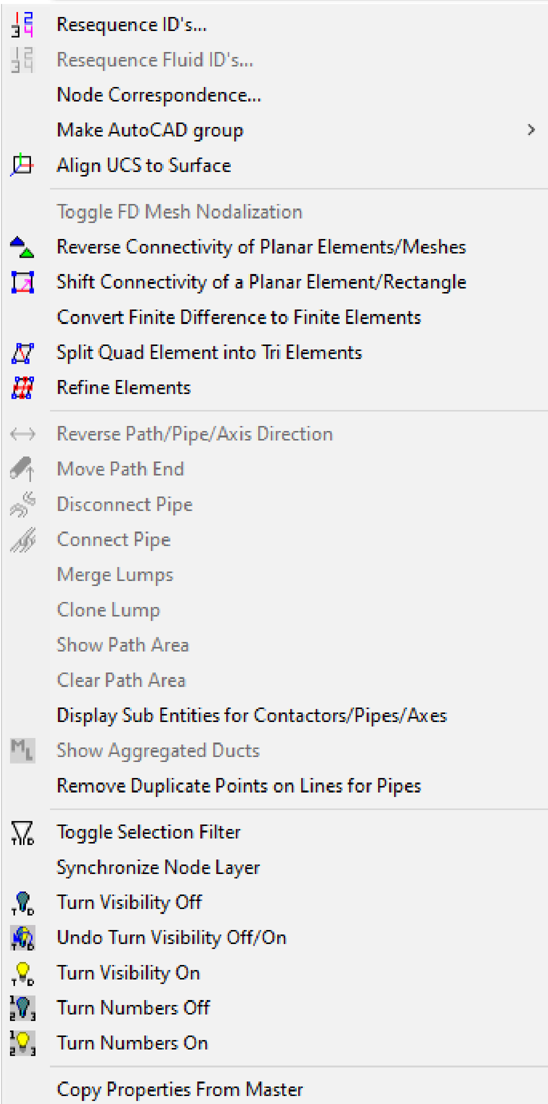
- The Model Browser is the best way to find what you are looking for in a model that is nearing completion or as you are building
- Numerous options for how to display the entity hierarchy of the model
  - Can see what nodes/surfaces are associated with Opt Properties, Heaters, Heatloads, Contactors, Symbols, Radiation Groups, etc.
  - May edit most selected objects in tree
  - Good idea to add comment to surfaces for better identification in the Model Browser
- May select nodes/surfaces in the tree and turn on/off visibility and node numbers
  - Multi-select allowed
  - O to show only selected, A to show All
- May edit many entities from Model Browser
- Options to Include child nodes when selecting parent or select surfaces/nodes in AutoCad window when selected in tree
  - Display results in section below tree
  - Heatflows (nodal, submodel to submodel)
- Many context menus accessed with right-click

Model Organization

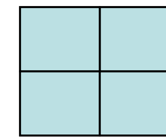


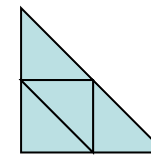
- While Desktop will generate the entire model for you, it is still a good idea to number regions for later identification
  - At a minimum, submodels should be used to organize the model hierarchy (change this by changing Submodel field for selected Nodes)
  - Within a submodel, Resequencing ID's is a good idea
    - 1xxx for component A, 2xxx for component b, 3xxx …
    - Easier to identify in text output later
- Node correspondence also available
  - Renumber nodes prior to export to SINDA
  - May merge radiation results into TMM nodes for model simplification
- Align UCS
  - Set drawing coordinate system to that of selected surface
- Convert FD to FE
  - One way operation. Cannot go back to FD from FE
  - Surface removed and elements created in its place
  - Caution: FD Solid Radiation not preserved

Model Organization (cont'd)



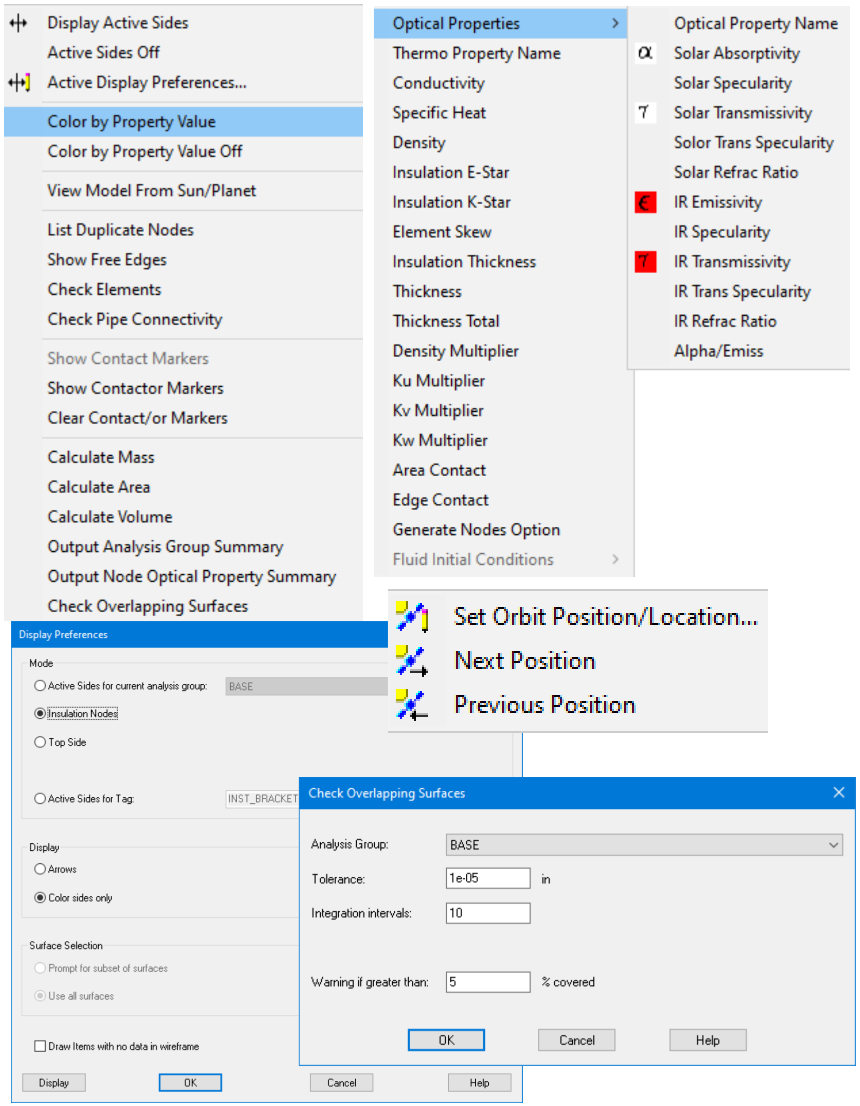
- Element updates
  - Reversing the connectivity of element changes the active side (1-2-3-4 to 4-3-2-1) changes +Z based on Right Hand Rule
  - Can split quad element into 2 tri elements
  - Can refine elements
    - Nodes added at midpoints for new elements
    - Quad to 4 quads, Tri to 4 tris





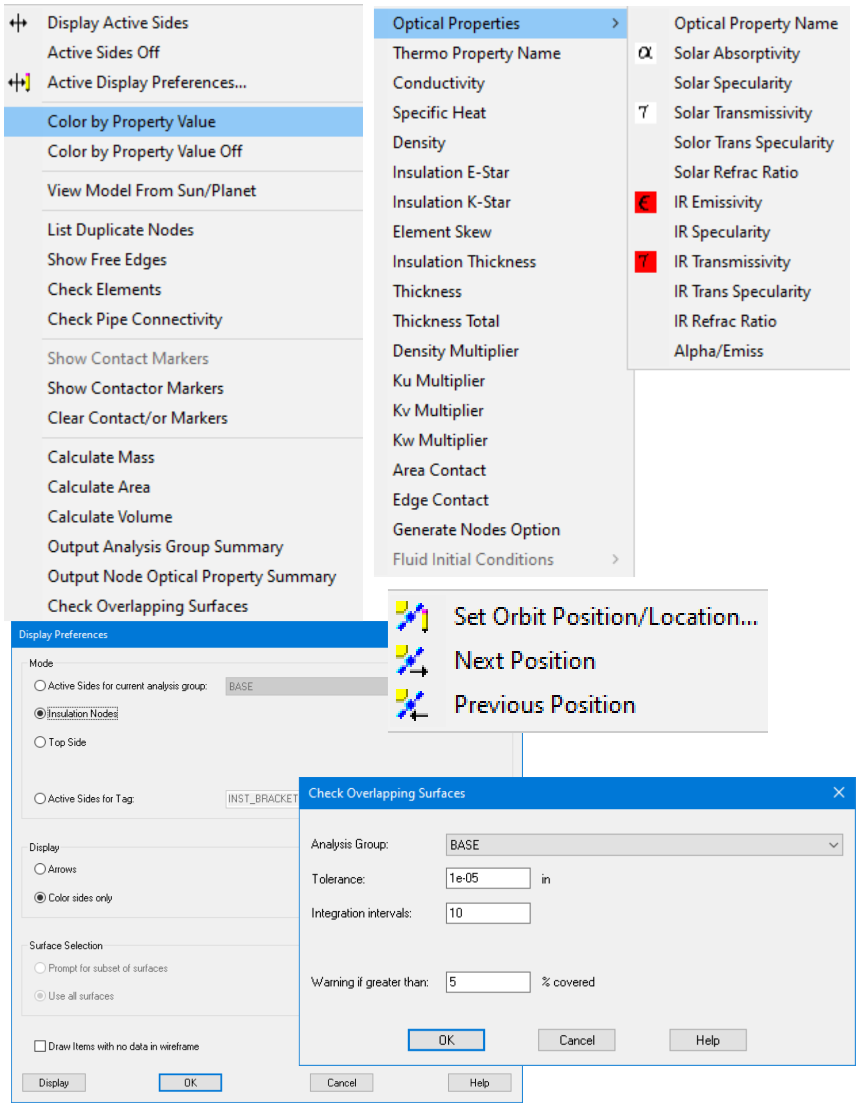
- Selection Filter
  - If OFF and more than one object type selected for edit, user is prompted for object(s) to edit
  - If ON, user is prompted even if only one type
- Control Visibility of selected surfaces
- Control Node Number display of selected surfaces
- Copy Properties From Master
  - Specify Master Object
  - Select object to copy Properties from Master to
    - Node Numbers, Subdivision, Names, and Dimensions/Locations are not altered
    - Best used for Material, Thickness, Opt Props, Radiation Active Sides, Double Sided Numbering

Model Checks



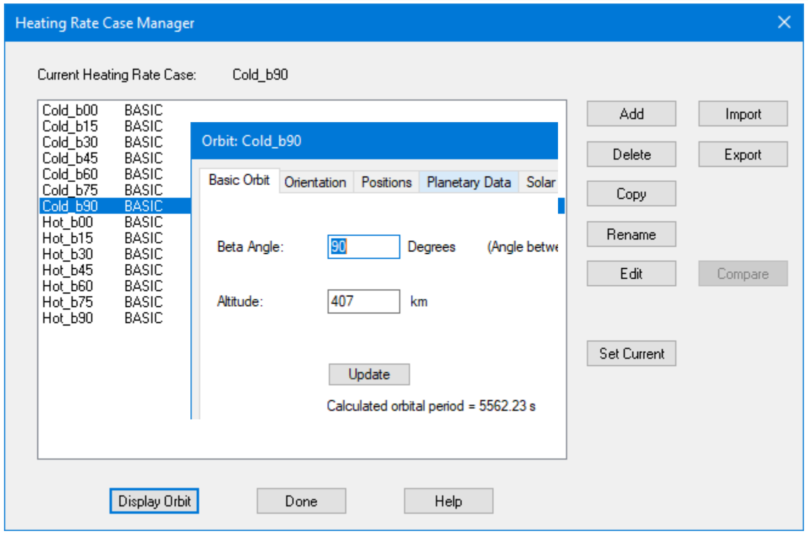
- Check Active Sides (for All Analysis Groups)
  - Active Sides may also show MLI, Top Side, or Domain Tag Set active side
  - May show as arrows or colors
    - Green: visible side is active
    - Lt. Blue: visible side is inactive
    - Yellow: both sides are active
    - Dk. Blue: both sides are inactive
    - Red: not in analysis group *(NOT THE SAME AS BOTH SIDES INACTIVE)*
- Color by
  - Optical Property Name or Value
  - Thermophysical Property Name or Value
  - Thickness
- View local model from Sun/Planet
  - Not the same as viewing in the orbit (no planet included)
  - User may select orbital position

Model Checks (cont'd)

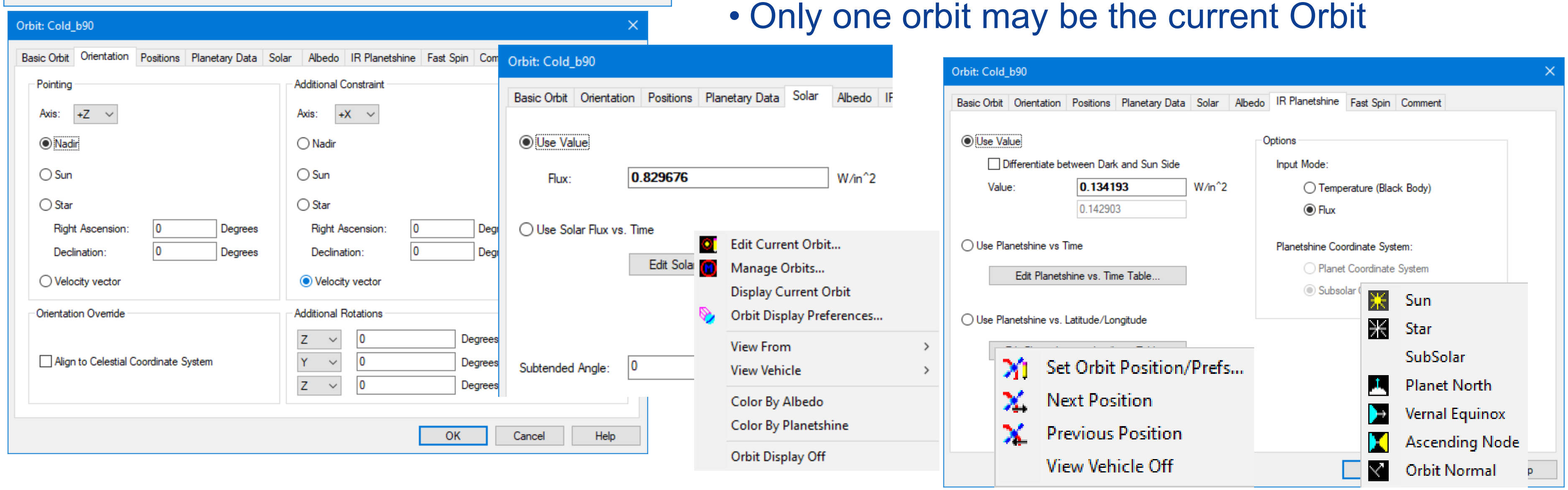


- List Duplicate Nodes
  - Good unless many locations for duplicates is intended
  - If only a few, then it is usually easy enough to keep track
- Show free edges
  - Used to show if FE edges were not connected via node merging
  - Should only be along outer edges
- Contact*or* markers (not Contact!)
  - Graphically displays locations of sub-areas/sub-lengths where contact is made
  - Excellent way to make sure correct edges/sides were selected
  - Best to clear before running again
- Calculate Mass
  - Submodel breakdown based on Area, t, ρ
- Output Analysis Group/Node Opt
  - Tabular output of Node/Group/Opt Orop
- Overlapping surfaces
  - Automatically checked by default in Radiation Task under Case Set Manager
  - Autocad Group made for surfaces that overlap (but does not include surfaces that are overlapped…). See log file (CheckOverlappingSurfaces.log) for details

Defining Orbits

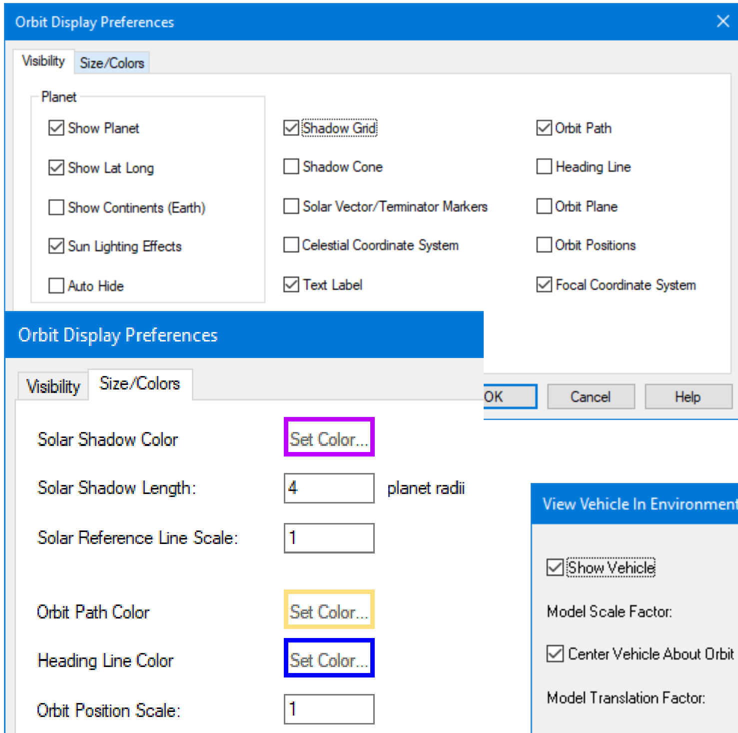


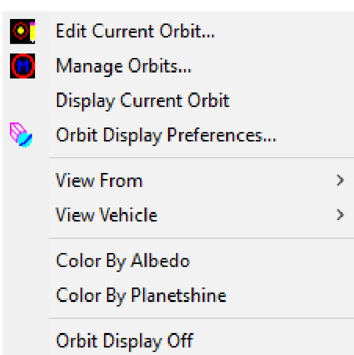
- Multiple Orbits Allowed
  - Orbit Parameters (Alt, Incl, Beta, RAAN…)
  - Orientation (Pointing, Rotations)
    - Can slew entire model by making rotation depend on hrMeanAnom (3*hrMeanAnom is 3 rev/orb)
  - Spin allows for fast spin of entire vehicle (not slew like in orientation) Specify # of orbit positions, terminators
  - Solar, Albedo, Planet IR (Can be Time or Lat/Long dependent (A,P only))
- Orbit is specified under Radiation HeatRate or Articulating Radks task
- Only one orbit may be the current Orbit

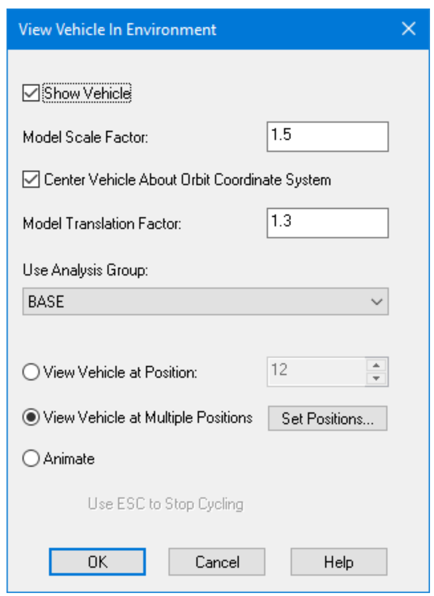


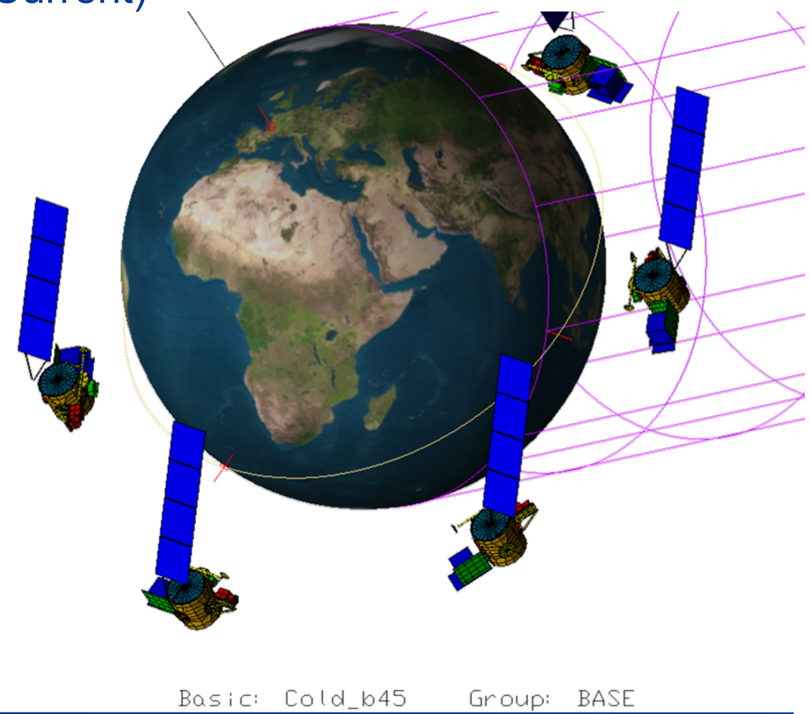
Visualizing Orbits

- Orbit Display Options
  - What to show (Grid, Planet, Positions, etc)
  - Sizes/Colors
- Vehicle Display Options (Size, Positions)
- View From Options (Sun, Star, etc)
- Only one orbit may be the Current Orbit
- Visible surfaces based on Selected Analysis Group (Defaults to Current)

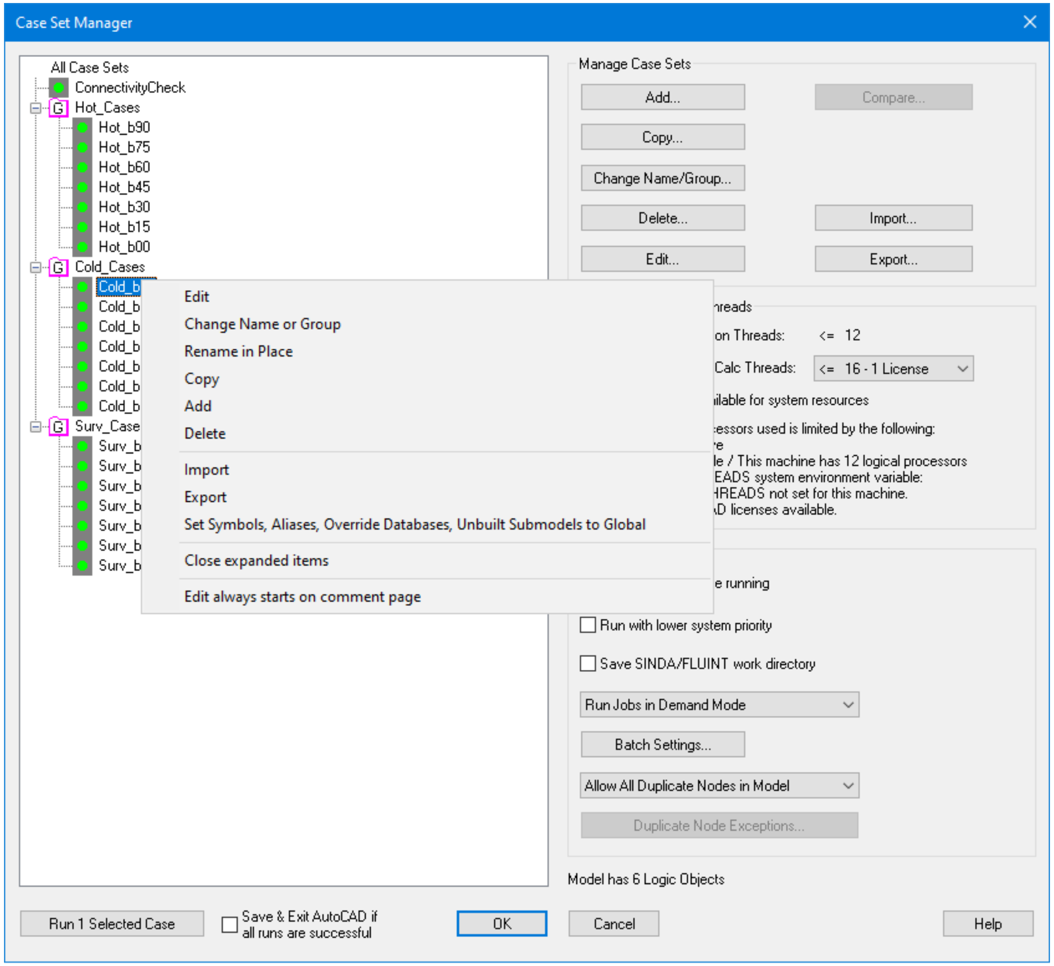








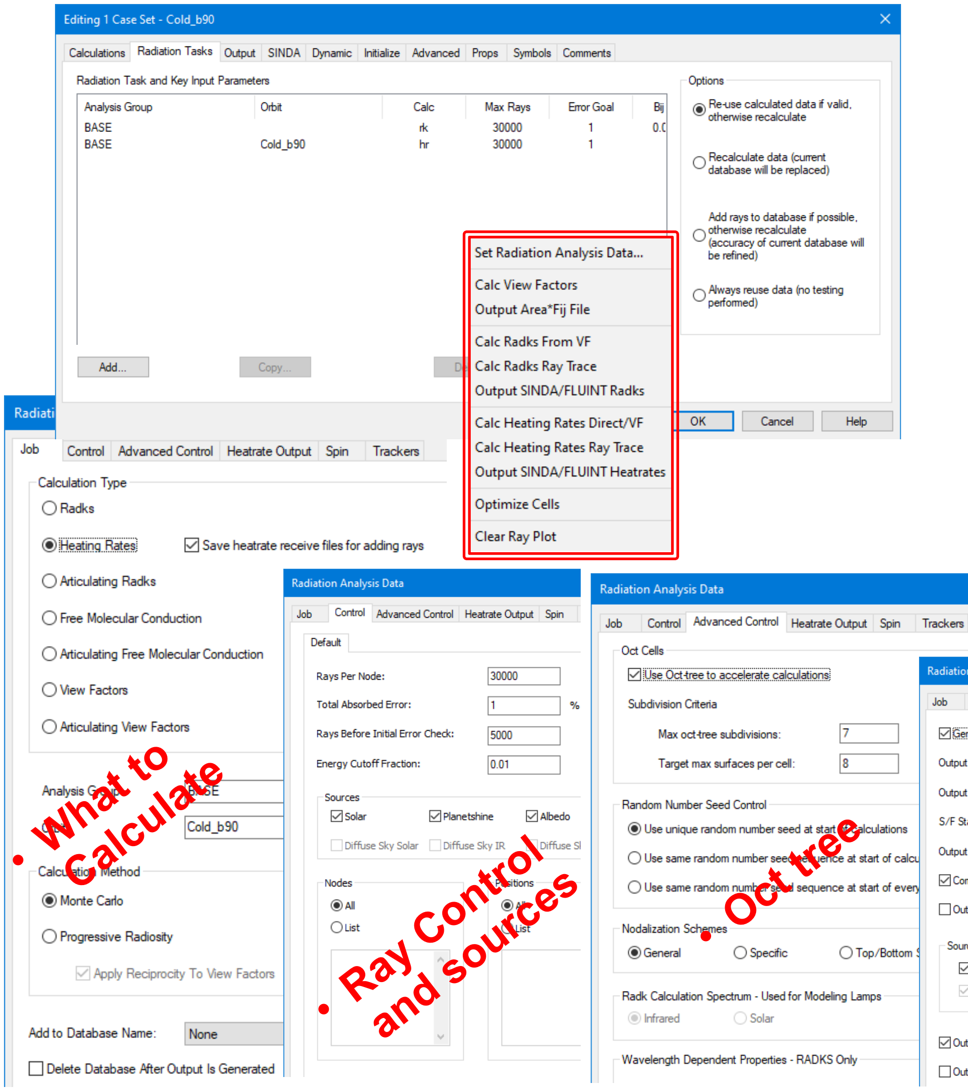
Case Set Manager

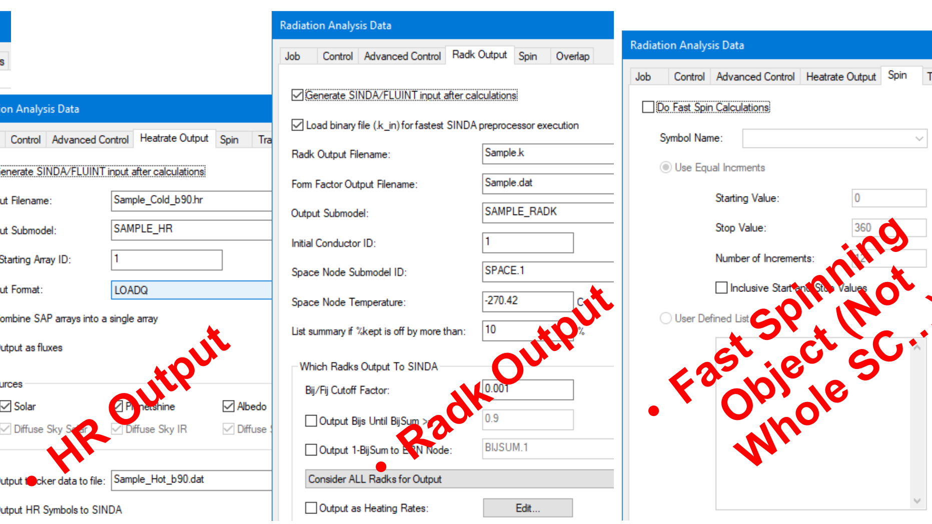


- Many Cases can be defined
- Cases include:
  - Radiation Tasks
  - TMM parameters (Time, Solution, Output)
  - SINDA logic
  - Symbol and Property Overrides
- Multiple cases may be selected to run sequentially
- Cases may be imported, exported, copied, deleted, etc
- Double Click or Edit to change properties
  - Multiple cases may be edited at one time
- Right Click Context Menu allows for temporary assignment of Overrides to model. Excellent way to verify CaseSEt is as intended

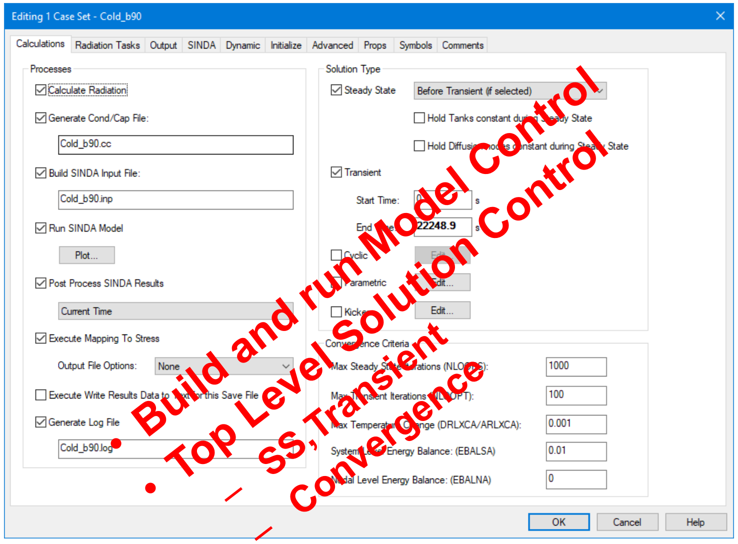
Case Set Manager (Radiation Tasks)

- Multiple radiation tasks may be defined
  - Radk, HR, Articulating Radks
- Desktop has some smarts to see if anything has changed that neccessitates a rerun of radiation tasks
  - Or the user can specify to use files that are already there
- Properties accessed by Double Clicking or Properties Button
- May also use Radiation Calculations menu item

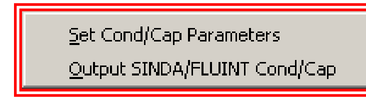


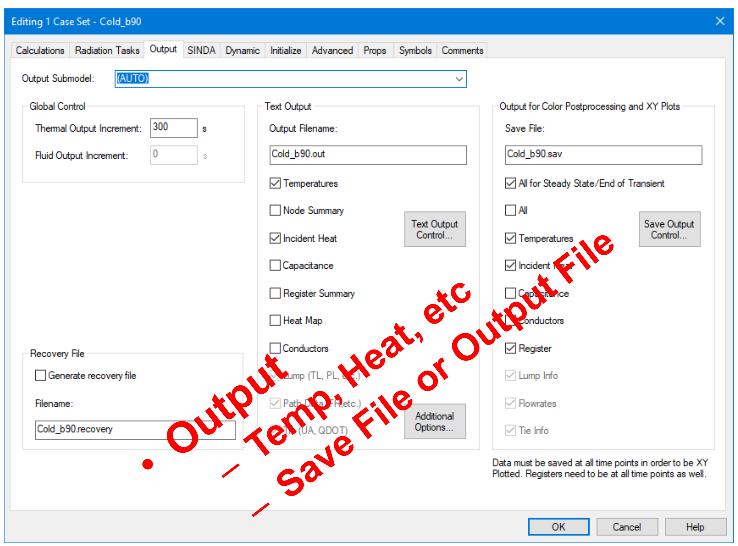


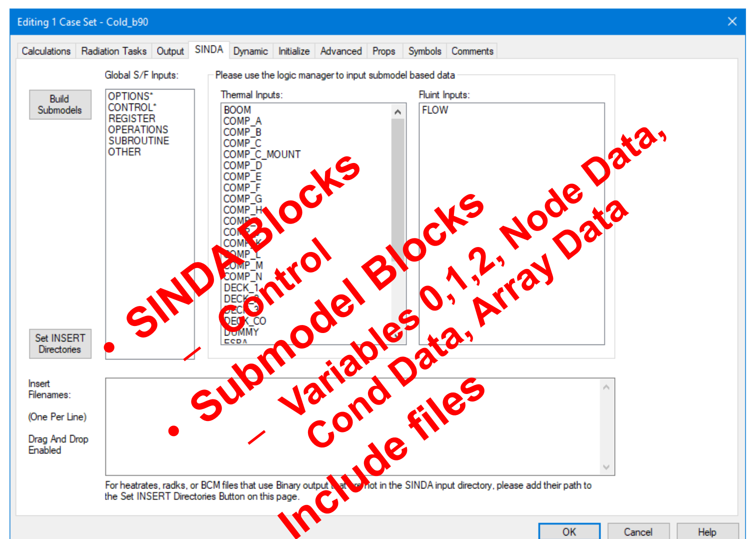
Case Set Manager (SINDA)



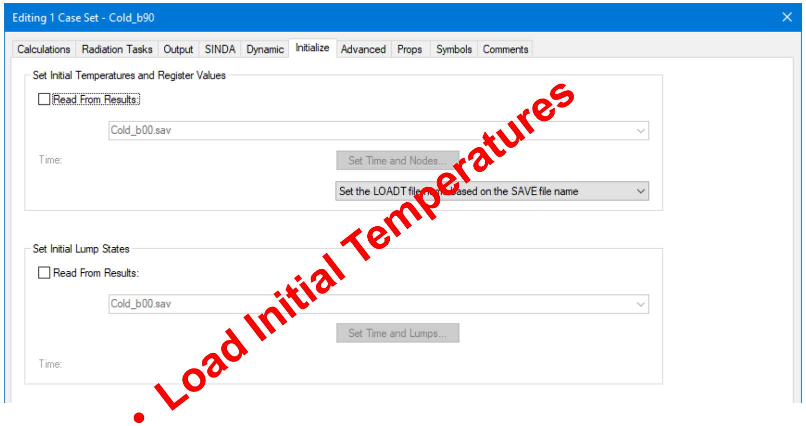
- S/F Calculations
  - Build/run/post process model
  - May build Cond/Cap using Conduction Calculation menu
- S/F Output
  - What and where to output
- SINDA
  - What submodels to build
  - Access to all Control variables
  - Allow symbols to pass through as SINDA registers
  - User generated SINDA logic, data, and parameters
  - Included Files



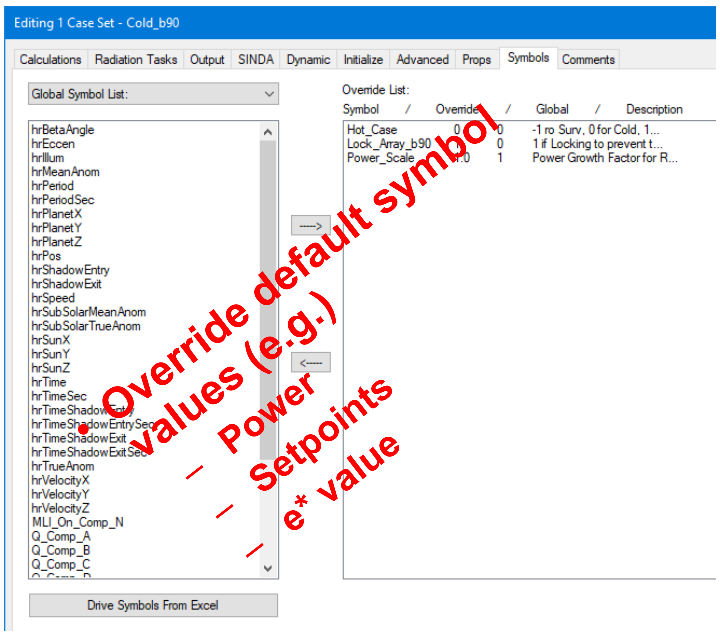


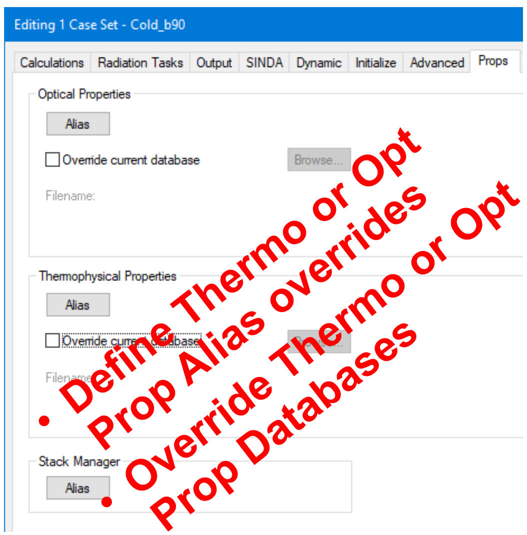


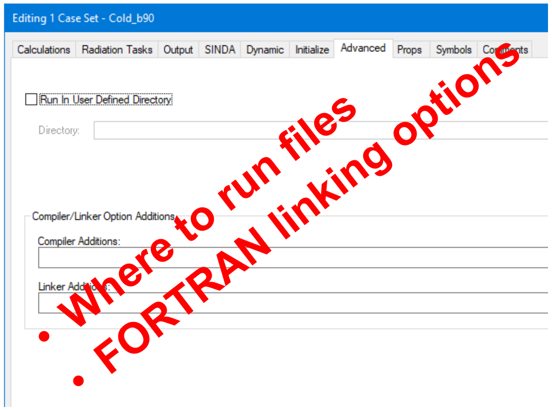
Case Set Manager (Overrides)



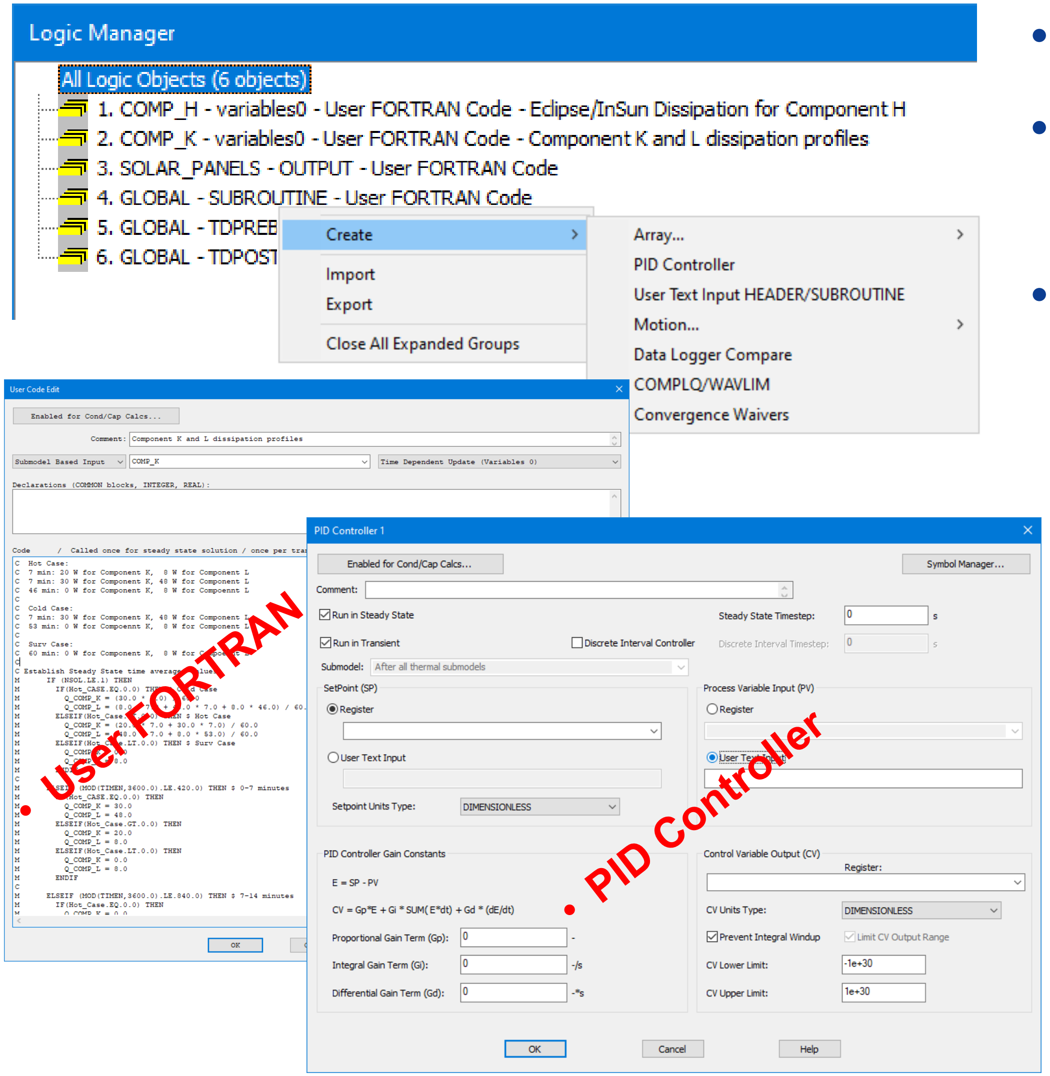
- Initialize
  - Set Initial Conditions
- Advanced
  - Where to store files for run
- Props
  - Override property aliases or databases
- Symbols
  - Override default symbol values
  - Great place to define power, setpoint, thickness, or any other uncertain parameter specific to case
- Comments tab also exists
  - Very useful for documentation that stays with model (change log, how to use, etc)



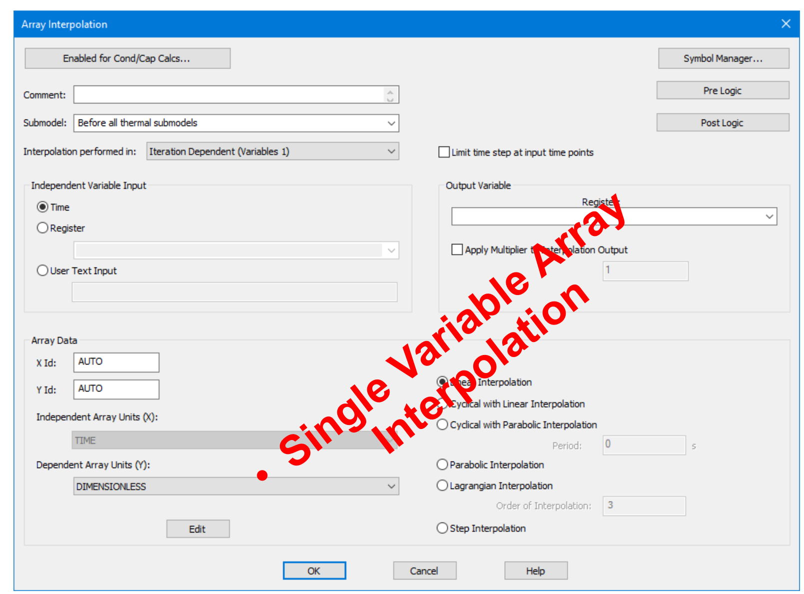




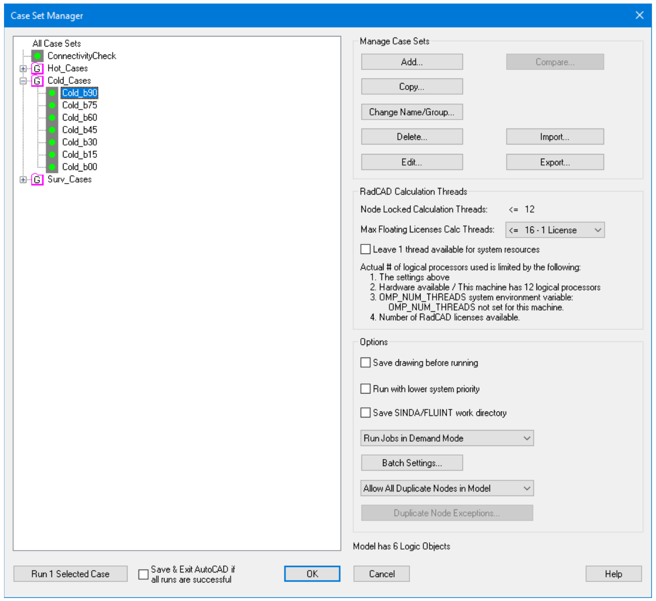
Logic Manager



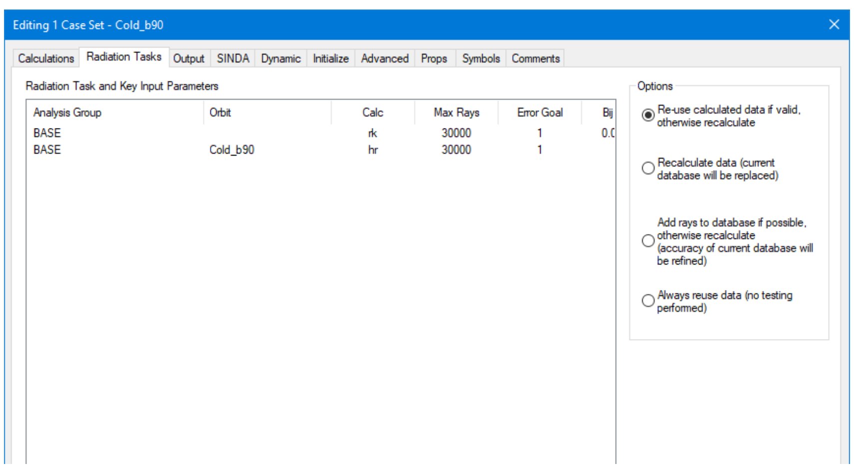
- Items in Logic Manager are included in all cases in Case Set Manager
- User may create
  - Interpolation (1 or 2 independent vars)
  - PID controller
  - SINDA logic blocks (VAR1, VAR0, etc)
- Submodel where logic is to be placed is specified as well as which submodel logic block
  - Additional blocks included for before BUILD, after BUILD, and after run

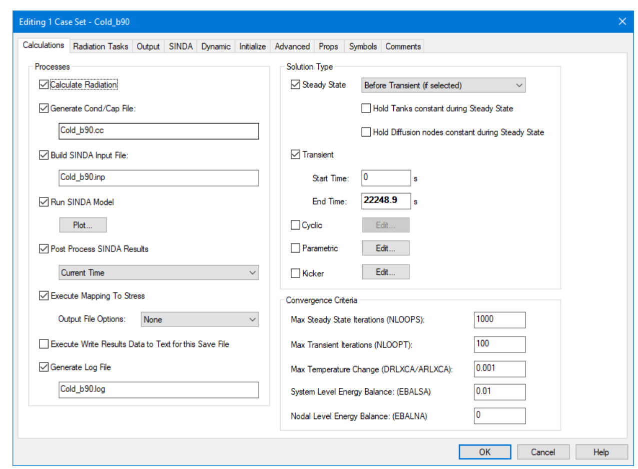


Running Case(s)



- Radiation tasks are run first (if not using old data)
  - Names of output files included as INSERT statements
- Conduction is generated next, followed by top level SINDA file
- SINDA is then called and the model is run
- Results are loaded from Sav file and color contour plot is displayed
- Radiation Tasks faster on Multi-Core machines
  - SINDA still a single threaded application
  - 2 Cases may run faster to only build SINDA and run external to Thermal Desktop in separate directories



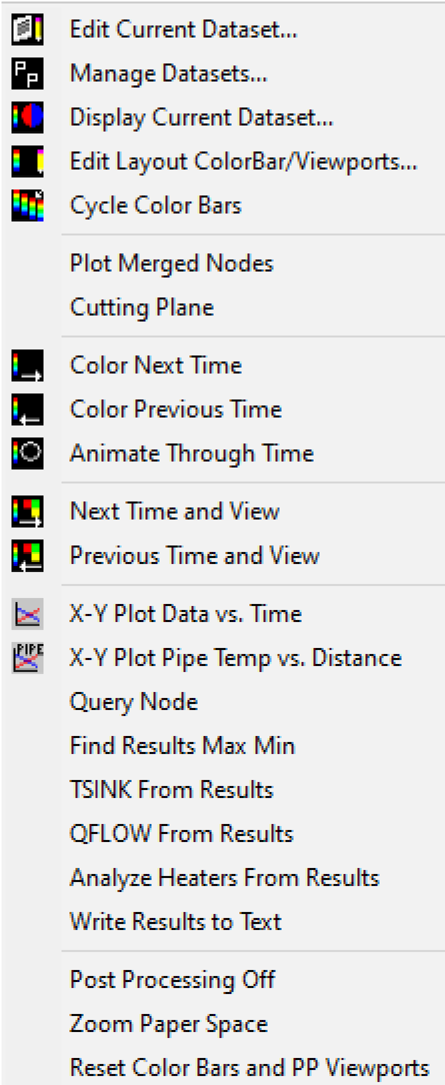


Files Created

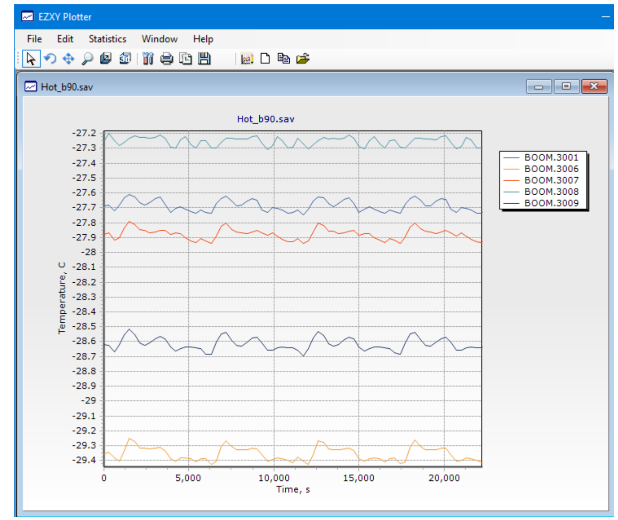
- Radiation
  - Calcualtion Folders : *rch/rck*
  - **Radk output: *.k, .k_in (and maybe .kl, .ka, and .kb for Articulating Radks)***
  - Radk info: *.xls (Nodal area, emis, Bij sum, Bij self, Bij inactive, Error)*
  - **HeatRate output: *.hra, .hrl, .hra_bin***
  - HeatRate info : *DirectIncidentSolarFlux, ReflectedAbsorbed, DirectAbsorbed, TotalAbsorbed, TotalAbsorbedPercentError, OPS (Node/Property/Surfaces relations), .ar (nodal areas)*
- SINDA Input Files
  - **CondCap – (Nodes, Conductors, Heatloads, Heaters, etc) : *.cc***
  - **SINDA input file : *.inp***
- SINDA Output Files
  - **Preprocessor output file : *pp.out***
  - Compiler log file : *Messages.log*
  - SINDA execution messages file : *Messages.txt*
  - **ASCII output file : *.out***
  - **Binary output file : *.sav (or CSR folder)***
  - Needed for HeatFlows (I think) : *.savpcs*
- General
  - Log file from ThermalDektop : *.log*
  - List of nodes/objects that are disabled : *Disabled.log*
  - Mass Calculations : *Mass_submodel.txt, mass_node.csv*
  - Node Duplicates : *DuplicateNodes.txt*
  - Overlapping Surface Check : *CheckOverlappingSurfaces.log*
  - ContactorData, MapData folders : *files for contactor and FEM mapping files*

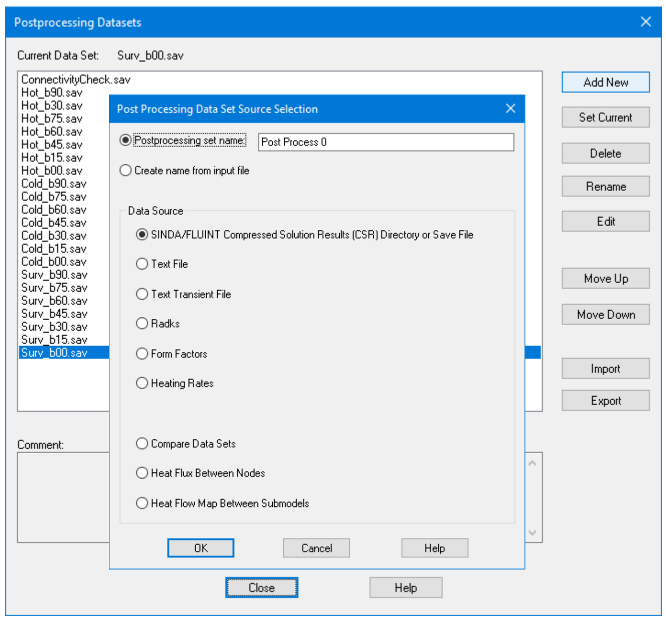
BOLD files are the most important

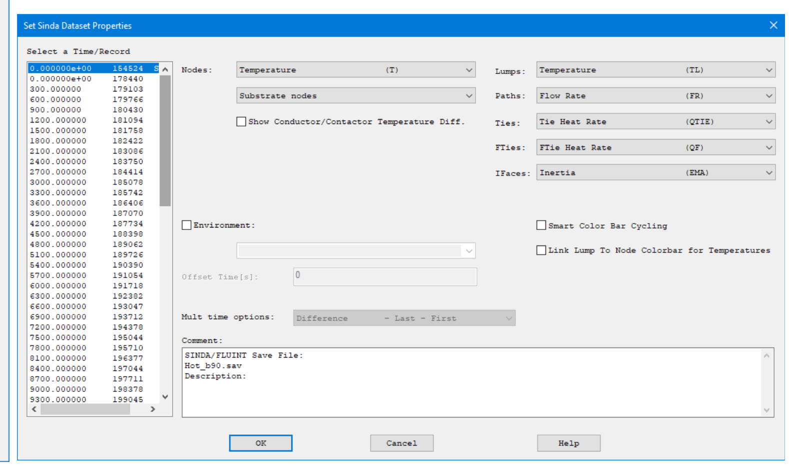
Post Processing



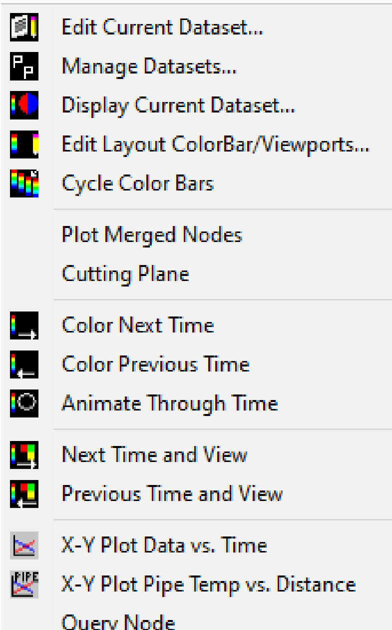
- As with multiple Case Sets, multiple Results sets may also be stored in Desktop, with only one current
- Numerous Data Types may be imported (sav, text, radk, heat rates, etc)
  - Timestep selected for Contour plots
  - Compare DataSet option
- EZ-XY Scatter plots for selected objects





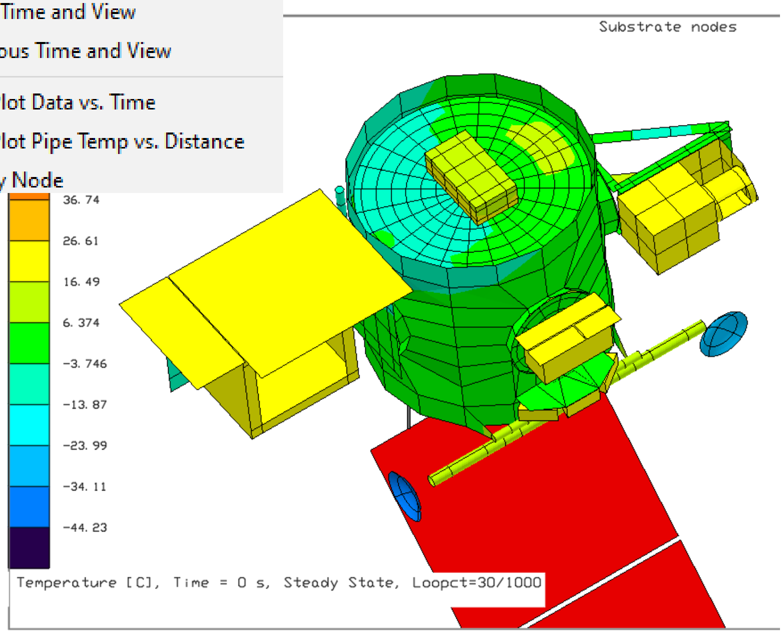


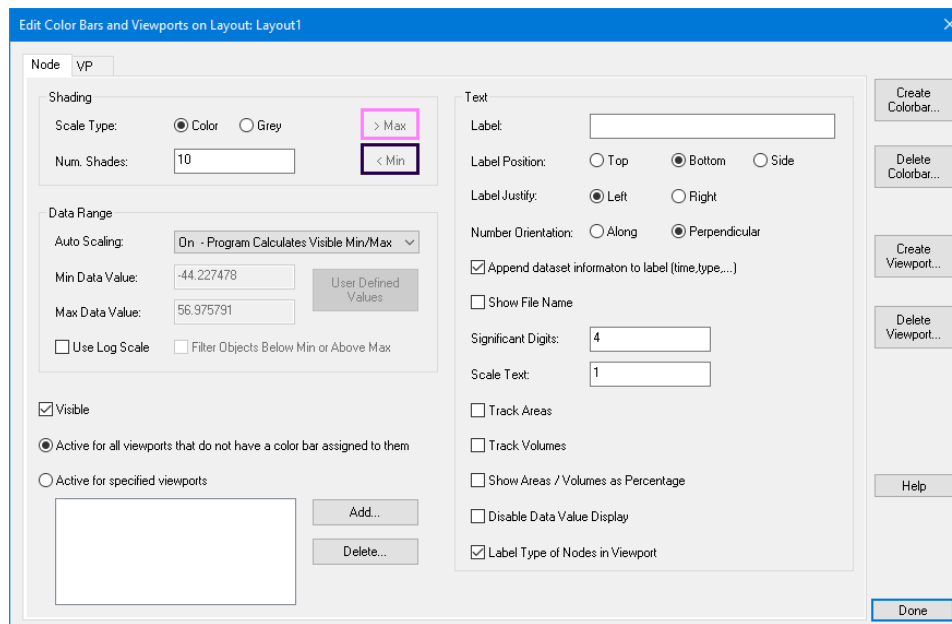
Post Processing



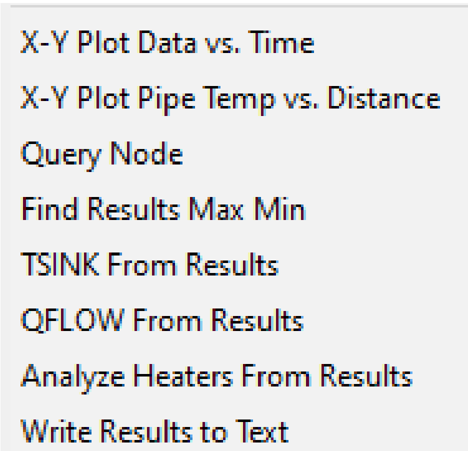
- Query Nodes for info on user selected nodes
- Color bar options (location, scale, divisions)
  - Note that user specified settings may still apply when showing α,ε
- Contour Animations or manually step through time
  - Can display from orbital viewpoint for selected timesteps as well (e.g. from Sun) – Note, this may only work well over first orbit (Time for subsequent orbits > period)

```
Command: _RcQueryNode
Select items for post processing data query or [MB/Do
Select items for post processing data query or [MB/Do
BOOM.1001   10.2753
BOOM.3006   -29.3559
BOOM.3001   -27.6865
BOOM.3007   -27.8778
BOOM.3008   -27.2539
BOOM.3009   -28.6173
Max BOOM.1001 10.2753
Min BOOM.3006 -29.3559
Avg   -21.7527
Total -130.516
Auto update model browser from command: RCQUERYNODE
```

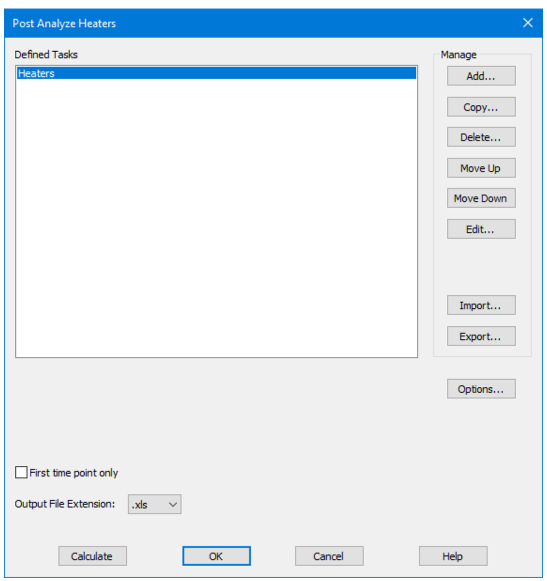


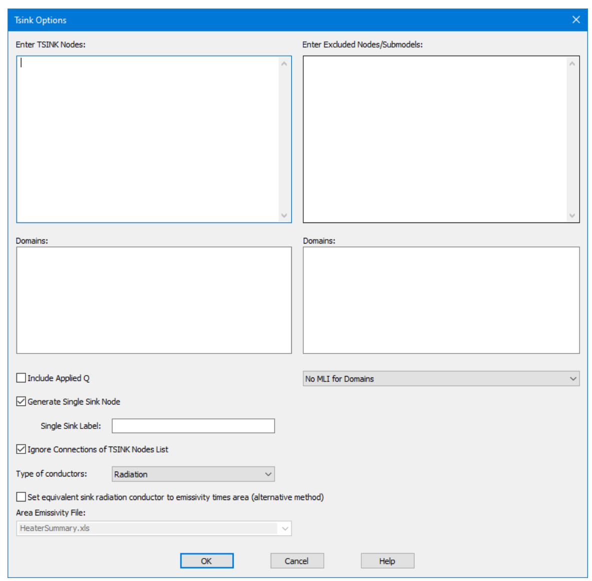


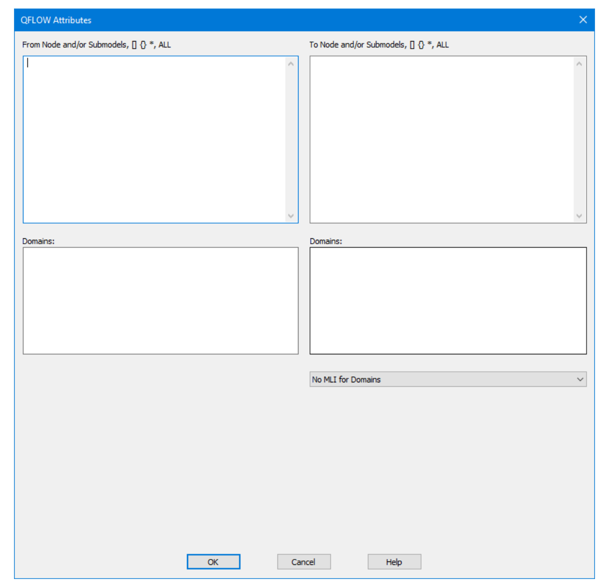
Post Processing



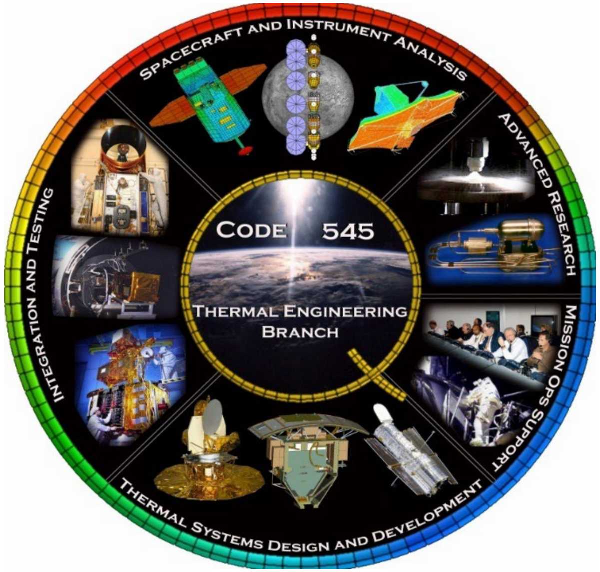
- Find Max Min
  - Allow search over multiple sav files for Min and Max; results listed at submodel level, integrated submodel, or node level
- TSINK from results
  - Specify regions of interest and regions of exclusion
- QFLOW from results – (May be easier to do in Model Browser)
  - Specify to and From regions
- Analyze Heaters from Results
- Write SAV file results to Text file for further manipulation







Process for Building Thermal Models



PURPOSE

This section is meant to be an overview of the model building process

It is based on typical techniques (Monte Carlo Ray Tracing for radiation exchange, Lumped Parameter, Finite Difference for thermal solution) used by the aerospace industry

This is not intended to be a "How to Use Thermal Desktop" section, but more of a "How to Build Thermal Models" section and the techniques will be demonstrated using the capabilities of Thermal Desktop. Other codes may or may not have similar capabilities…

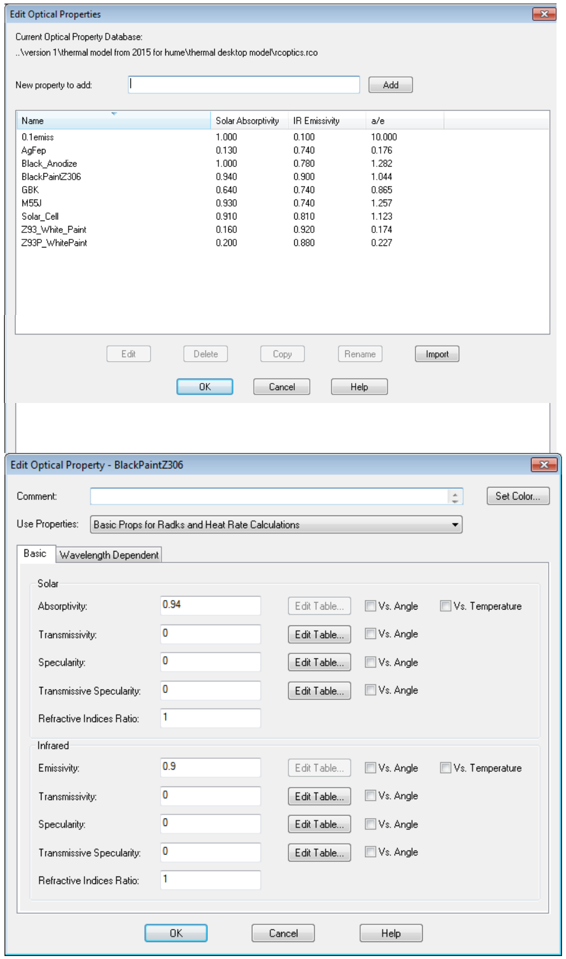
The General Model Building Process can be broken into four top level steps:

1. ***Build Model***
2. ***Check Model***
3. ***Execute Model***
4. ***Verify Results – This section will be covered near the end***

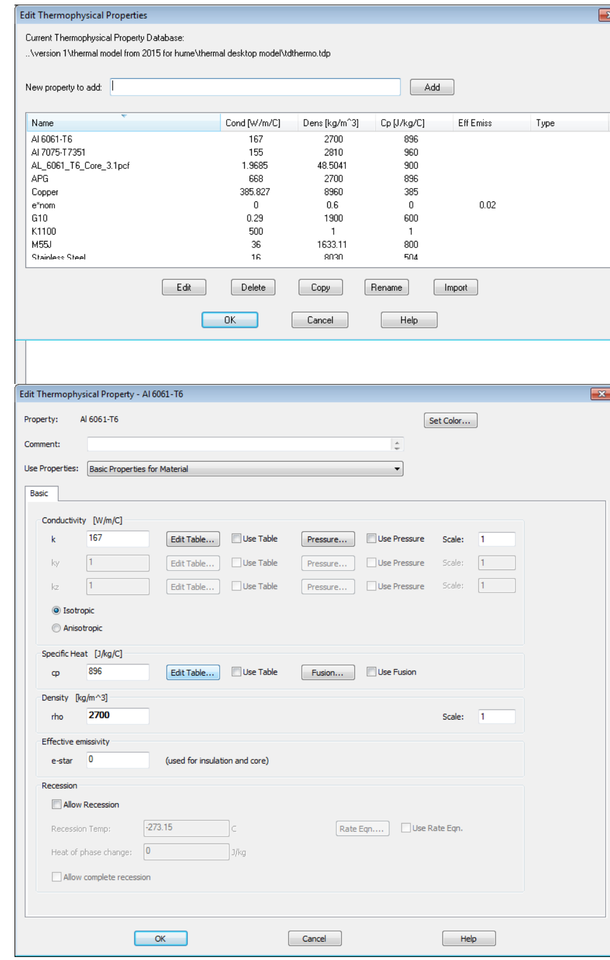
BUILD MODEL: General Model Building Process

1. **Define ThermoOptical and ThermoPhysical Properties**
2. **Get CAD from Mechanical Designer – STEP format preferred, make sure it is solids and not only surfaces**
   - *Process CAD into layers for controlling visibility*
3. **Build and define each CAD component**
   - *Geometric Surfaces, Nodes, Elements (Plate, Solid) – Add Comments to help*
   - *For Solid Element geometry, apply surface coat of zero thickness plates*
   - *Determine Node Subdivisions and Edge vs. Centroid Nodalization*
   - *Merge Nodes*
   - *Assign Node Numbers and Submodels*
   - *Assign Material and Thickness (Orienters for Solid Elements)*
   - *Assign Active Sides, Optical Properties, and Radiation Analysis Groups*
   - *Assign MLI*
4. **Determine Variable Parameters**
   - *Add Symbols (Thickness, Power, Multipliers, IF Values, etc) and assign to geometry if needed*
5. **Accommodate Moving Geometry**
   - *Define Assemblies and/or Trackers and Assign Geometry (Include nodes to move FE)*
6. **Establish Model Connectivity**
   - *Add Contactors between surfaces/edges at interfaces*
   - *Add Conductors between nodes/surfaces*
7. **Establish Boundary Conditions**
   - *Add and Define Heat Loads*
   - *Add and Define Heaters*

BUILD MODEL: Define ThermoPhysical and ThermoOptical Properties

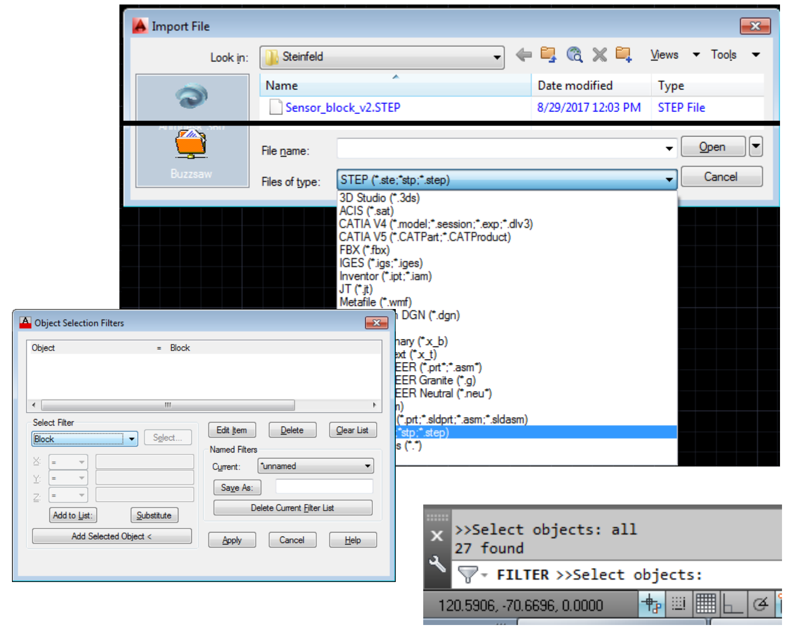


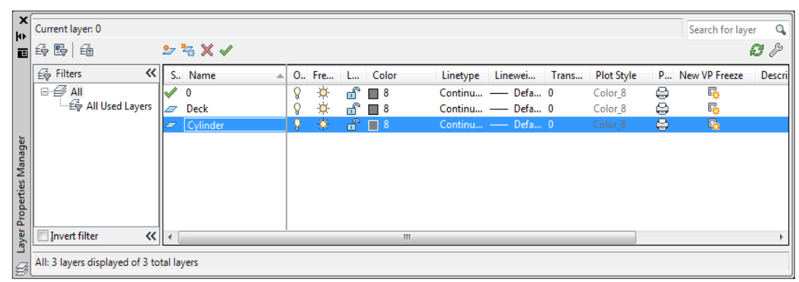
- Add ThermoOptical Properties:
  - ✓ Open/Create Property Database
  - ✓ Add Property and Specify Name
  - ✓ Define α and ε
  - ✓ Include comment/source
  - ✓ Repeat until all properties added…
  - ✓ Repeat for EOL or other cases
  - ✓ Can import from other models
  - ✓ Define any Property Aliases that may be used (e.g. Radiator_Coating). Use ALIAS_ as prefix
- Add ThermoPhysical Properties:
  - ✓ Open/Create Property Database
  - ✓ Add Property and Specify Name
  - ✓ Define ρ, C_p, and k (or k_x, k_y, k_z)
    - ✓ For MLI, define ε*
    - ✓ For PCM, define FUSION
  - ✓ Can import from other models
  - ✓ For MLI, recommend using ρ of 0.6 kg/m^3 and k and C_p of 0. For MLI assignments, define thickness for MLI as 1.0 m
    - ✓ This will allow mass of MLI to be estimated based on 0.6 kg/m^2
  - ✓ Define any Property Aliases that may be used (e.g. Panel_Material). Use ALIAS_ as prefix



BUILD MODEL: Importing CAD

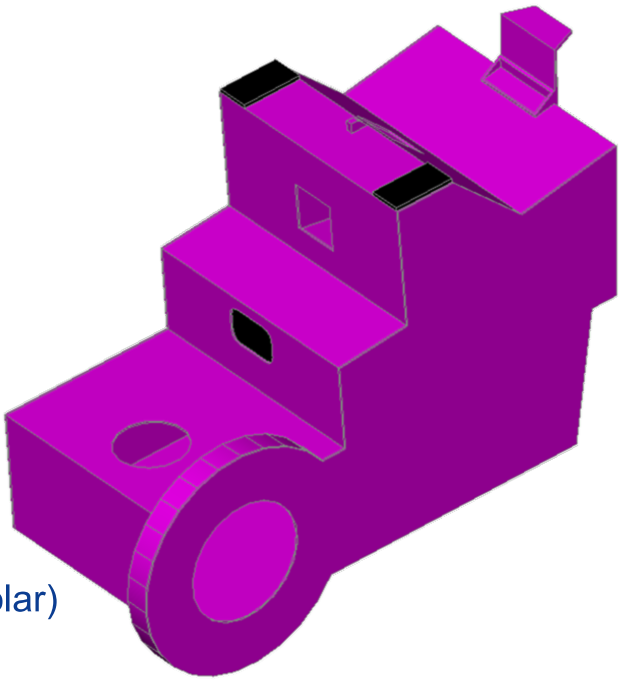
- To import a STEP file:
  - ✓ Type *import*, select file, and wait for notification in bottom right
  - ✓ Type *burst* and type ALL when selecting objects (this will explode *only* blocks into constituent parts). Repeat until no objects are found.
  - ✓ Alternatively, could use *explode* and *'filter* to only select blocks but more tedious
  - ✓ At this point, everything should be a 3DSOLID or non-block entity type
  - ✓ Create layers for all the components for the design (*layer*)
  - ✓ All Solids can now be placed on layers and have colors changed to help with visibility control
  - ✓ Solids can also be copied and sliced (*slice*) to take measurements and "see" what is inside
  - ✓ Can select properties for block and set transparency or control transparency by layer





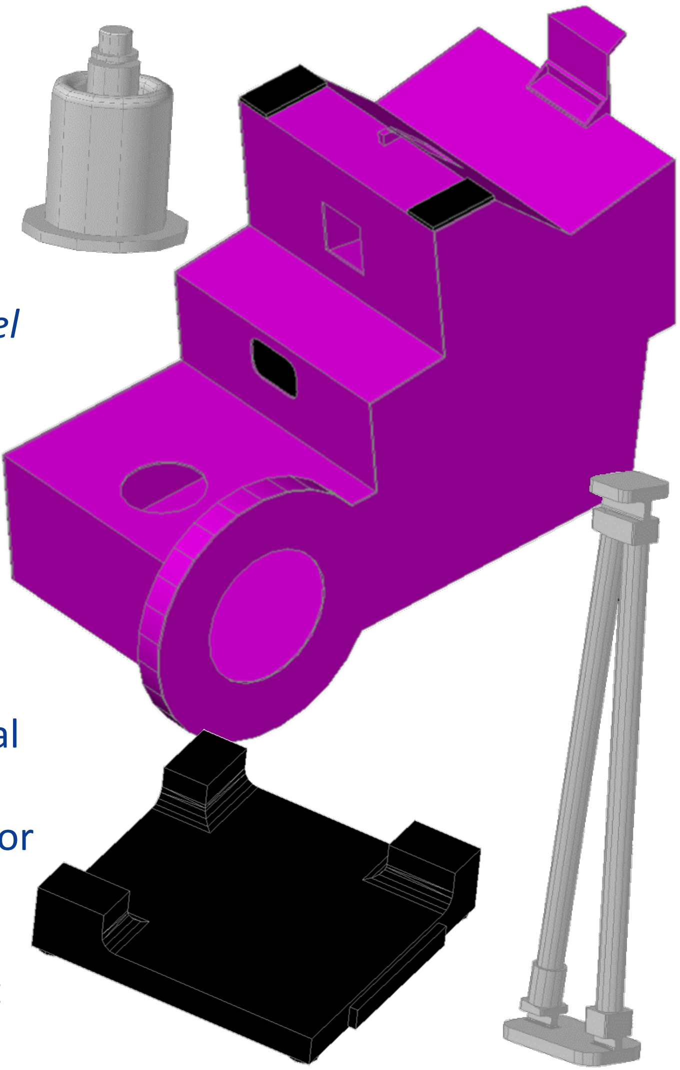
BUILD MODEL: Working with CAD

- **Many functions help with using CAD:**
  - ✓ ZOOM: enlarge and shrink image on screen
  - ✓ 3DFORBIT: rotate image in screen
  - ✓ OSNAP: allow user to select midpoint, endpoint, quadrant, intersection, node, center, perpendicular, nearest
  - ✓ UCS: define 2D working plane
  - ✓ PROPERTIES: change layer, color and linetype for selected objects
  - ✓ LAYER: allow visibility, color, linetype changes for specific layers
  - ✓ LINE: create line segment
  - ✓ DIVIDE: create N points along line segment
  - ✓ FILLET: connect two, non parallel lines
  - ✓ MOVE: translate geometry
  - ✓ COPY: copy geometry within drawing
  - ✓ ARRAYCLASSIC: create copies or an object as array (rectangular or polar)
  - ✓ ERASE: remove object from drawing database
  - ✓ DIST: measure distance and angle between two points
  - ✓ MEASUREGEOM: get volume of solid CAD
  - ✓ EXPLODE/BURST: convert assembly into selectable constituent parts
  - ✓ SLICE: cut solid objects by slicing plane
  - ✓ ROTATE3D: rotate geometry about specified axis by specified angle
  - ✓ MIRROR3D: create object that is mirror image of selected object about specified plane
  - ✓ 3DALIGN: align geometry based on 3 points in reference frame and destination frame
  - ✓ ORTHO: toggle if second point is orthogonal (along X or Y) to first point
  - ✓ FILTER: allow removal or inclusion of selection objects based on properties



BUILD MODEL: Building Geometry

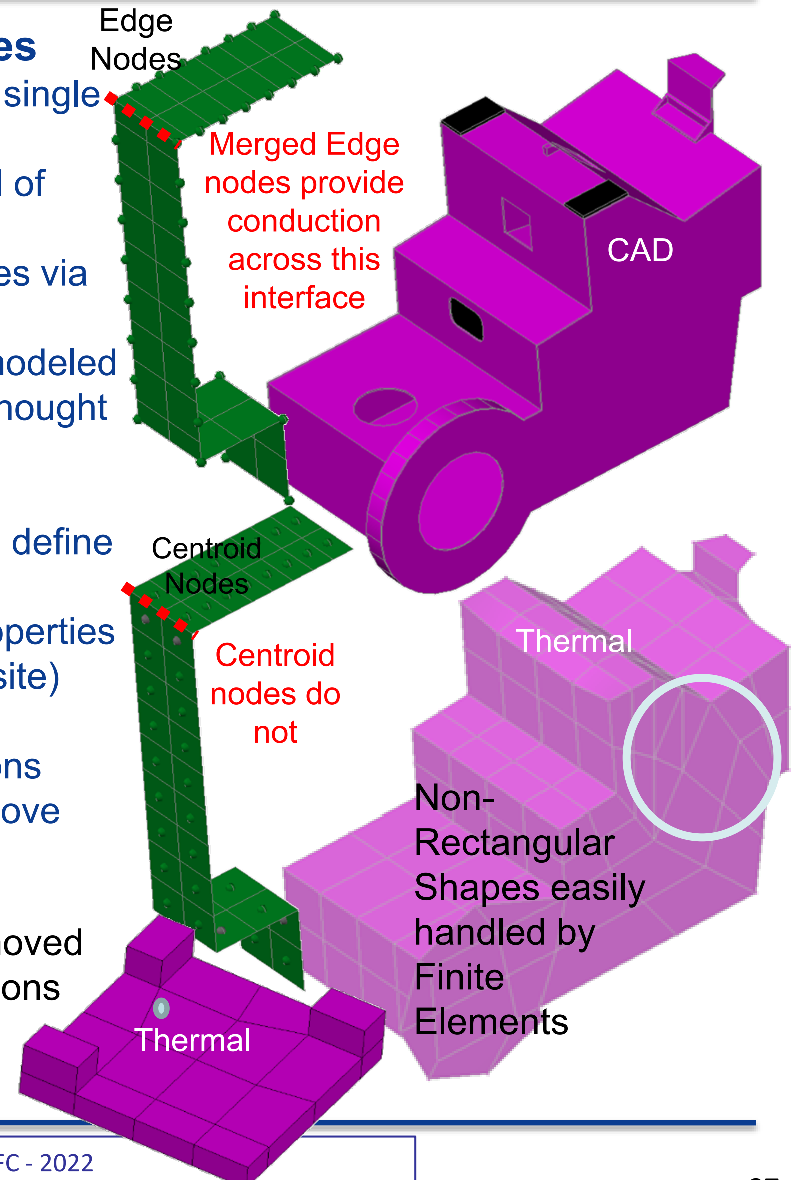
- Most models represent 3D CAD with 2D simplifications
- The representations include additional properties that make them thermal
  - ThermoOptical properties and radiatively active sides
  - ThermoPhysical properties and thickness
  - Node assignments and subdivision
  - *The node assignments form the basis for a network thermal model which is used to compute temperatures and heater powers. It is the link between the geometric and thermal models.*
- Some surface shapes follow naturally occurring geometry
  - Rectangle, Cylinder, Disk, Cone, Triangle, Sphere, Paraboloid, etc.
- Other 2D shapes do not conform to simple geometry
  - For these, the use of finite elements is recommended, which allows a user to more closely match CAD geometry
- Sometimes, it is necessary to model heat flows as three-dimensional
  - Solid Primitives (Box, Disc/Cylinder, Sphere, Cone) exist which provide the same properties as their 2D counterparts, but allow for a network based on 3D subdivision
  - For more complex shapes, 3D finite elements (Tetrahedron, Wedge, Brick, and Pyramid) are available. Note that these do not include radiation properties…

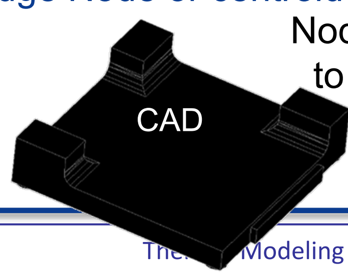


BUILD MODEL: Meshing Geometry (Nodal Discretization)

- **Use of Edge Nodes recommended over centroid nodes**
  - ✓ Centroid and edge nodes both account for conduction *within* a single surface
  - ✓ Edge nodes move nodes to edges of each sub-surface instead of centroid
  - ✓ Edge nodes that are merged allow conduction between surfaces via the shared edge nodes
  - ✓ Use of finite elements allows for more complex shapes to be modeled
  - ✓ Building elements by hand not as time consuming as may be thought
    - ➢ Define nodes first by placing user nodes at location on CAD geometry
    - ➢ Define element by selecting nodes in right hand rule order to define “top side”
  - ✓ Solid elements can also be defined and allow non-isotropic properties
    - ➢ Manually define by selecting nodes (one face then the opposite)
    - ➢ OR Revolve/Extrude Plate elements
  - ✓ Benefit of elements is ability to move nodes to desired locations
  - ✓ Elements cannot be moved…Elements defined by nodes…Move the nodes instead
  - ✓ Define surfaces as either edge Node or centroid node
  - ✓ Shell coat solid elements
  - ✓ **Merge nodes**
  - ✓ Assign rest of properties…

**** Extremely important!!!!***





BUILD MODEL: Working with Finite Elements

- Finite Elements are a powerful tool, but can be more difficult to work with than primitive surfaces
- Elements cannot be double sided and the node numbers are not able to be changed by the element. You can change the node number of the node itself and any elements that reference it will be updated, but you cannot change a node number from the element properties like you can for a primitive surface.
- Element shapes are defined by the nodes. Move a node to a different location and the shape changes.
- Material Orienter may be needed for solid element to define axes for anisotropic material property
- Options exist under Modeling Tools that are useful for working with elements
  - ***Refine Elements****: subdivides elements to convert a quad element into 4 quads based on the midpoint of each edge. Similarly this can be applied to triangular elements and creates 4 tri elements.*
  - ***Reverse Connectivity****: At times, the top side of an element is defined incorrectly. The connectivity can be reversed to switch the top and bottom sides.*
  - ***Shift Connectivity****: If using edge contactors with elements, it often assigns the first, second, etc. edge for making contact. The marker is displayed along the specified edge(s). However, instead of having to guess which edge is the first, the connectivity can be shifted to redefine the first edge. The marker will visible shift and this command should be repeated for each element until the markers line up along the desired location.*
  - ***Convert FD to FE****: Elements cannot be subdivided. A rectangle defined with edge nodes can be converted to elements (FD/FEM Network), but a 3x5 edge node breakdown become 8 individual elements.*
- Options also exist under FD/FEM Network that are useful for working with elements
  - ***Extrude/Revolve Planar Elements****: You can extrude a set of planar element along a line or revolve around an axis to define 3D elements. The number of elements along the extrusion can also be specified.*
  - ***Surface Coat****: Solid elements do not radiate and cannot be used with a contactor. Therefore, to use these features, a plate element of zero thickness is needed. Surface Coating created zero thickness plate elements on all free faces of a group of solid elements. Free faces are defined as faces that are not shared between elements.*

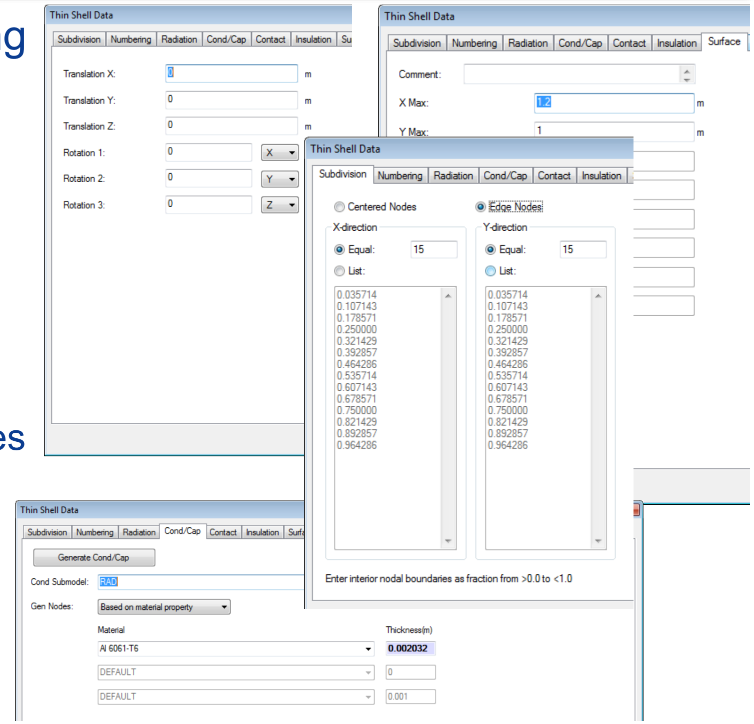
BUILD MODEL: Nodal Distribution (Model Fidelity)

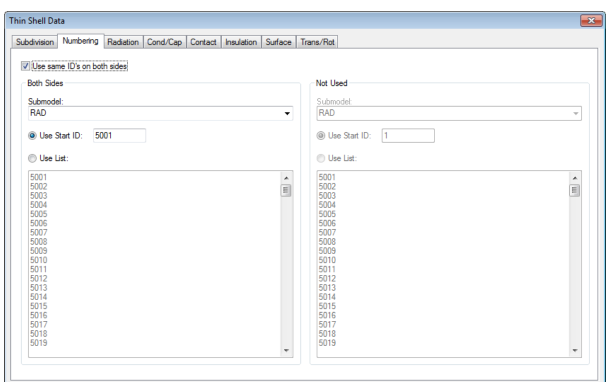
- Common question: **How many nodes should I use?**
- The answer depends on the design…
  - Thick, high conductivity material structure…fewer nodes
  - Thin OR low conductivity structure…some nodes
  - Thin AND low conductivity structure…many nodes
- Decide as an analyst your maximum allowable temperature between two nodes. (0.5 K, 2 K, 5 K? and where)
  - ✓ If two nodes are very close in temperature, they can be merged/eliminated
  - ✓ If two nodes are very far apart in temperature, you may need more in between
  - ✓ Do not force two nodes to be at nearly the same temperature with a very high coupling…merge instead
- If conduction is…
  - ✓ Mostly 1D, consider only discretizing in one direction (along strut, bar, or heatpipe)
  - ✓ Mostly 2D and planar (e.g. thin panel), you should discretize in two directions
  - ✓ Mostly 2D and non planar (cylinder, cone), consider uneven loading from environment or radiation when deciding on circumferential discretization
  - ✓ Mostly 3D, use solids or solid FE
- Often it also comes down to a matter of convenience for modeling conduction across an interface
- After you have run your model for the first time, investigate if the gradients warrant more or fewer nodes
  - Best to stay with surfaces and edge nodes if possible while determining this.
  - It is easy to renodalize a surface; it is far more difficult to remesh elements (unless you use the TD or other mesher)
  - Can Refine Mesh in TD which makes an FE Tri into 3 Tris and a Quad into 4 Quads…cannot go back though (other than using UNDO)

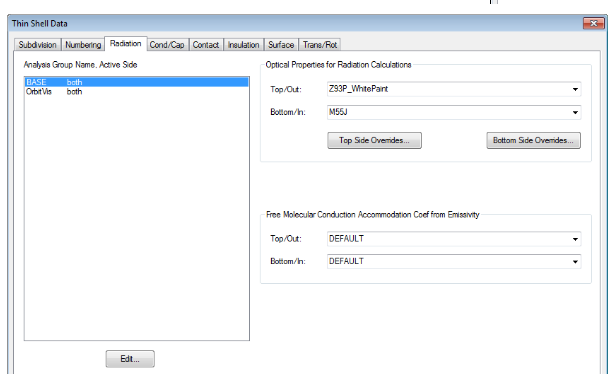
BUILD MODEL: Assigning Properties

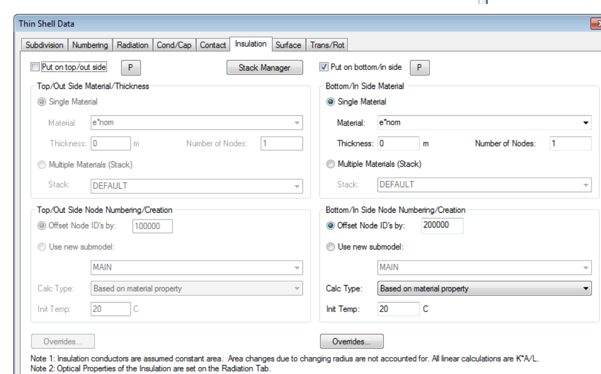
- Once surfaces/solids are geometrically defined, remaining properties need to be defined in Thermal Desktop
  - ✓ *Subdivision*: X,Y,Z breakdown and subdivision, Edge vs. Centroid*
  - ✓ *Numbering*: Submodel and Node Numbers (multiple submodels allowed)*
  - ✓ Radiation: Active Sides, ThermoOptical Properties, Overrides
    - ✓ Active sides defined for each Radiation Group
  - ✓ Cond/Cap: ThermoPhysical Material, Thickness
  - ✓ Insulation: Top and Bottom Sides, Insulation Material, Overrides
  - ✓ Contact: **Do Not Use**
  - ✓ *Surface*: Dimensions, Comment*
    - ✓ *Comment for Finite Elements (not Surface)*
  - ✓ *Trans/Rot: Additional rotations and translations*

** These properties not applicable to finite elements*



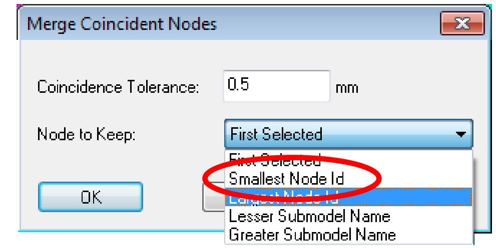


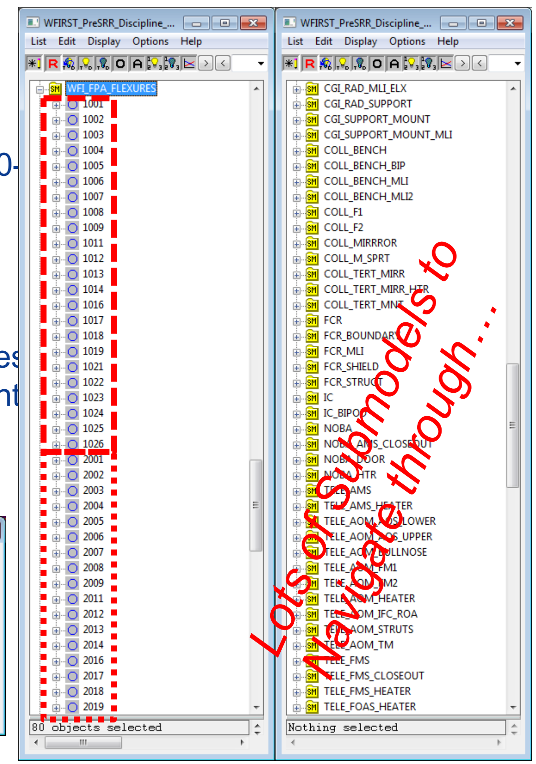


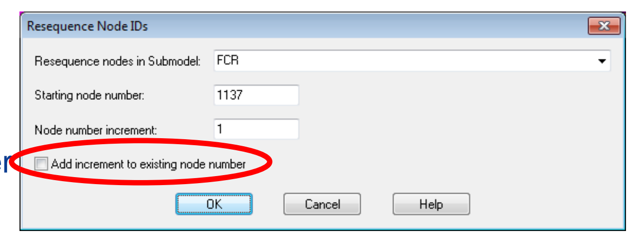


BUILD MODEL: Node Numbering

- Each node is an independent calculation point in SINDA
  - ✓ Use duplicate numbers if you intend for the temperatures to be the same (e.g. @ I/F)
- If node numbers are duplicated and nearly coincident, best to merge them into single node entity
- Use submodels for organization to identify component associations (e.g. Opt_Bench, Scan_Mir, FPA)
  - ✓ Too many submodels gets cumbersome in the ModelBrowser…don't go overboard
- But what node numbers should be used?
  - – Good practice to number nodes according to their location within an assembly
  - ✓ Best to keep MLI offset at 100000 for one side and 100000/200000 for both
  - ✓ Number first portion of a component in the 1xxx range, the next in 2xxx, and so on…
  - – Could start with 1xxxx range, but run out of "blocks" sooner before the 100000 range
  - – If you have more than 1000 nodes in a component either skip ranges (e.g. 1000-2999, 3000-3999, 4000-4999, etc) or use a block of 10000 for the entire component
- Why bother? Why not just renumber everything and let TD display the contours?
  - – Helpful for navigating model to have ranges…If 1000-1999 is the motor shaft, can turn its visibility on and off in the ModelBrowser by selecting that range
  - – Helpful for post processing…if all 100000 nodes are MLI, can likely ignore their temperatures. If 1000-1999 is the motor shaft, can apply limits to those nodes knowing what they represent
  - – Helpful for documentation…likely you will not be the last user of a model. The next person who picks it up and has to work with it will have an easier time learning the model if it is organized.
- To propagate numbering scheme in Z direction…
  - ✓ Renumber all nodes to 100000
  - ✓ Renumber first layer of nodes to 1000
  - ✓ Copy 1st layer of nodes in z over top of 2nd layer
  - ✓ Merge nodes and keep Smallest Node ID
  - ✓ Renumber with Add Increment checked
  - ✓ Repeat







A few minutes to organize node numbers can save many minutes during postprocessing

BUILD MODEL: Radiation Groups

- While ThermoOptical properties are specified for a surface, it does not necessarily mean that the surface participates in the radiative calculations
- Radiation Groups define collections of surfaces (i.e. enclosures). Only one Radiation Group is considered at a time during calculations
- For each Radiation Group, participation in calculations (activity) can be specified
- Multiple Radiation Groups can be used to define different configurations
- Radiation groups can be OR merged (If Top is active in on and Bottom is active in the other, OR merging makes Both sides active)
- They can also be copied to spawn a different configuration which may be used for changes in the geometry such as use of different instrument models or removal of a cover

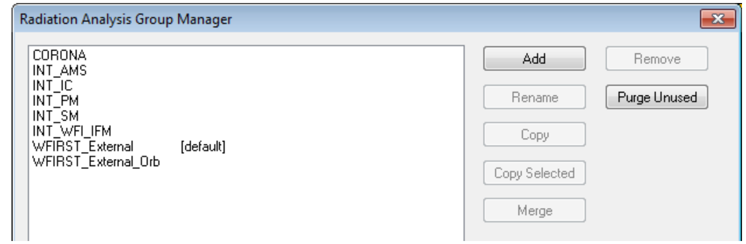
✓ Define the Radiation Groups before assigning activity

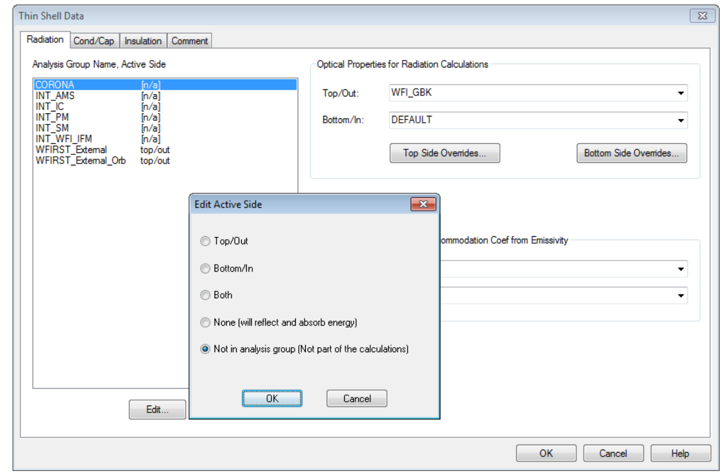
✓ For each surface, assign whether the top side, bottom side, both sides, or neither side participate in the solution. Additionally, Not in Group can be used for surfaces that should be excluded completely from calculations.

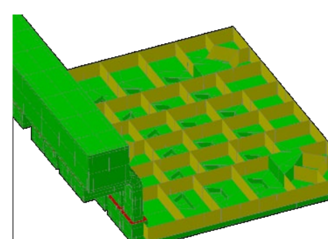
- Active None vs. Not in Group: Active none will still reflect and absorb energy; it just won't be tallied for the surface. Not in Group: surface is not even there.

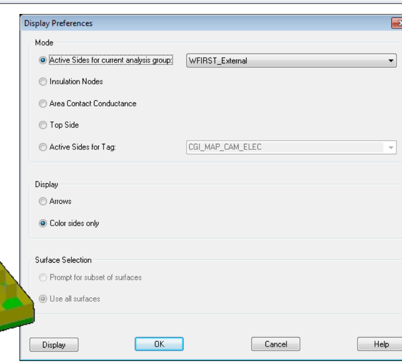
✓ Once activity is assigned, visualize using Model Checks-Display Active Sides. This will be for a specified Radiation Group. Colors indicate activity: Yellow ((Both Sides), Green (Visible Side), Light Blue (Not Visible Side), Dark Blue (Neither Side), Red (Not in Group).

✓ Be sure to check activity for every group that will be used in calculations…



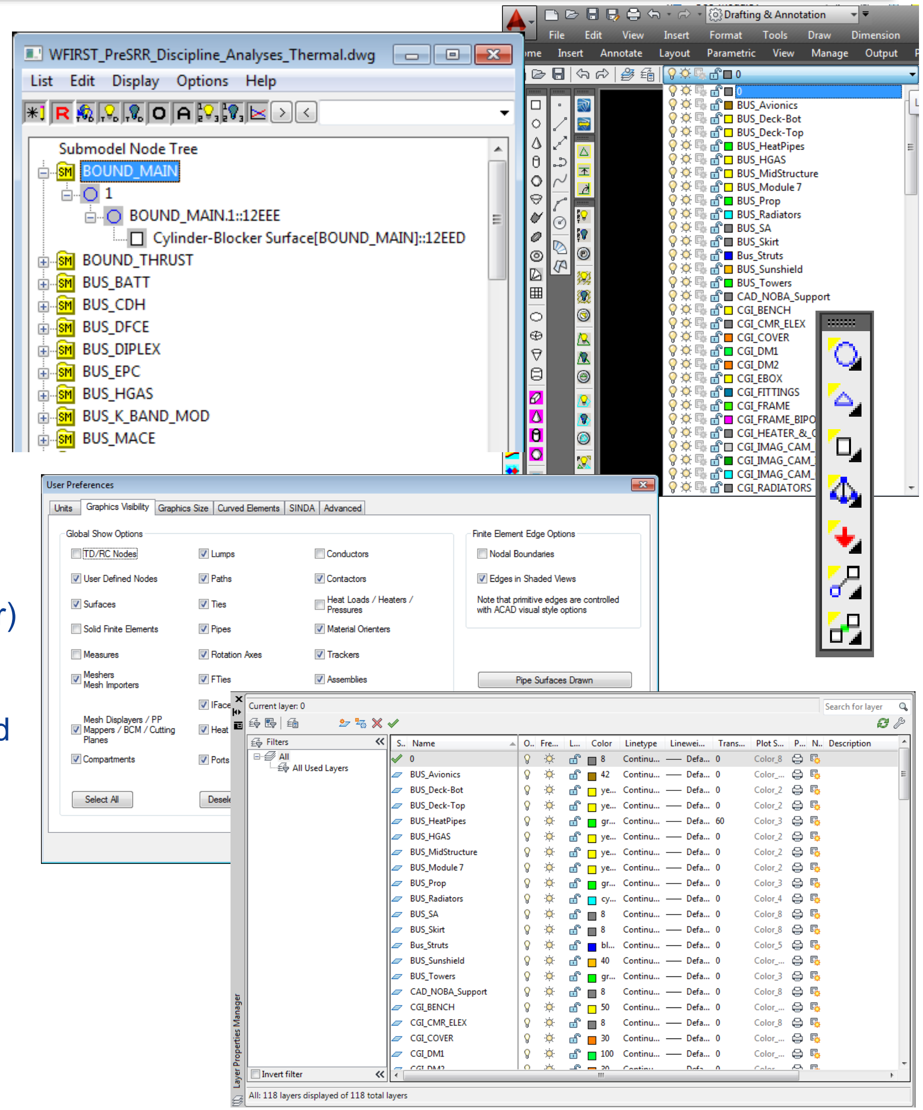






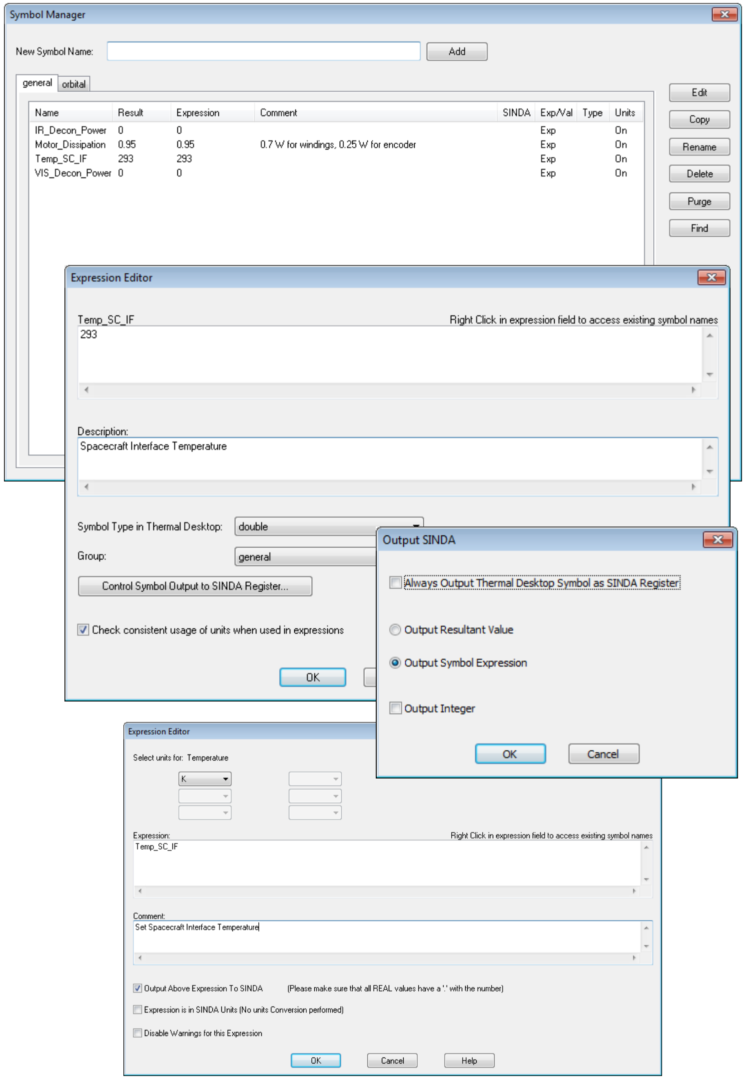
BUILD MODEL: Controlling Visibility

- Seldom is it useful to display the entire model. Most often, only one part of the model is being updated at a time
- Visibility control allow only parts of the model to be show. Three ways to control visibility: Layers, Global Entity Visibility, Object Level Visibility
- Layers: (layers command, native to AutoCAD)
  - If a layer is Off or Frozen (Blue Light Bulb or Snowflake) anything on that layer will not be displayed
  - Also can control from drop down along toolbar
  - HINT: If something isn't showing up in post processing, make sure the layer is not frozen in the VP too!
- Global Entity: (Thermal-Preferences)
  - If the visibility of a particular object type is unchecked, it will not be displayed
  - On/Off Toggles for common objects (Node, UserNode, Surface, Solid Element, Heat Load, Conductor, and Contactor) available on toolbar
- Object Level: (Blue and Yellow Lightbulb toolbar buttons)
  - Objects can be selected in Model Browser and visibility turned on or off (Blue and Yellow Lightbulb)
  - Objects can be selected from the screen and visibility turned off
  - Objects can be selected by group (even if currently off) and turned on
  - On/Off state can apply to solid CAD but hard to manage; use "all" for selection if needed
- *Best to use Layers for CAD objects. User choice for Thermal objects, as visibility can be controlled by Model Browser*



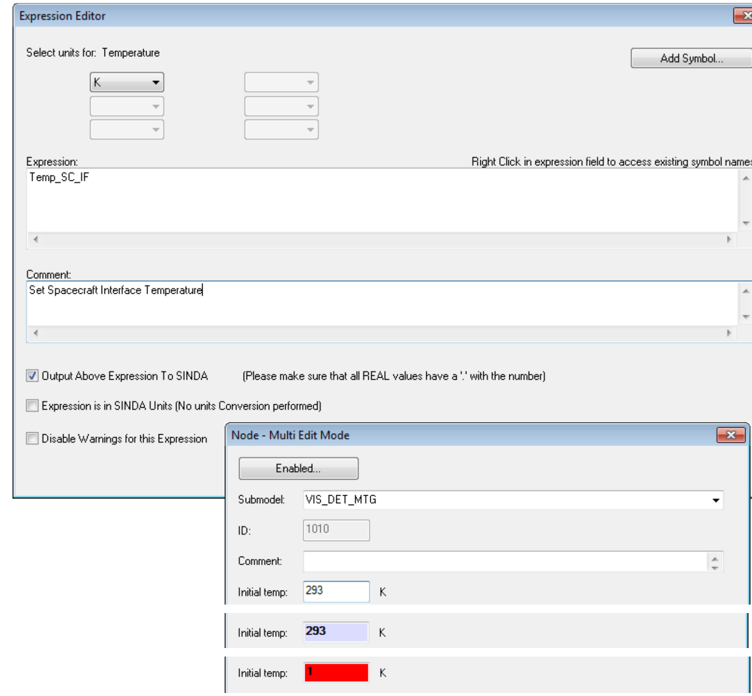
BUILD MODEL: Defining Variable Parameters

- Symbols can be used to control just about anything in Thermal Desktop
  - Dimensions (thickness, length, width, height)
  - State Flags (On/Off, Enabled/Disabled, Open/Close, Stow/Deploy)
  - Margin Factors (Heat Load Multiplier)
  - Other Values (Conductance, Heat Load, Setpoint, Rotations, etc)
- Symbols can
  - be single values (Integer or Real), expressions, or arrays of values
  - depend on other symbols
  - be overridden in the Case Set Manager for a particular analysis
- Symbols should
  - ✓ Be named intuitively. Use prefixes for thickness (thk_), heat (Q_), conductance (G_), Multipliers (Mult_), etc.
  - ✓ Be grouped together where logical (power, subsystem, etc)
  - ✓ Include a description or comment
- Symbols can be passed through to SINDA as REGISTERS
  - This is automatically done if the symbol is needed in SINDA
  - User can override to ensure symbol is passed as REGISTER if not detected by Thermal Desktop
- To use symbol, dbl click in textbox to access Expression Editor
- *Values can be set in Symbol Manager, Case Set Overrides, or user defined logic…be careful where you define things*



BUILD MODEL: Assigning Variable Parameters

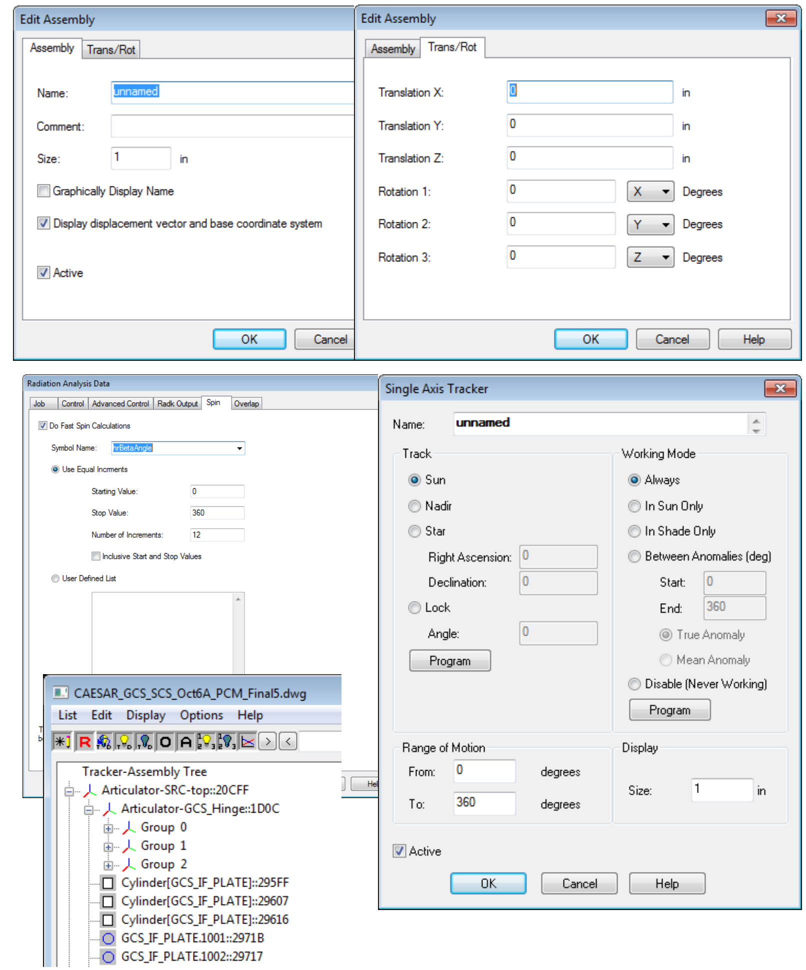
- Once a symbol is defined in Thermal Desktop, double click in just about any textbox to access Expression Editor to use it
  - Values dependent on expressions are indicated in **Bold** font
  - Values with a description are highlighted in light blue
  - Illegal or incalculable expressions default to 1 with a red background (importing a model without importing symbols)
- Unit conversion can be performed
- Can output expression to SINDA (as REGISTERs)
- Conditional statements (***Not Valid in SINDA FORTRAN block…***)
  - (Heater_ON == 1) ? 10 : 0
  - If Heater_ON is equal to 1, set expression to 10 otherwise 0
- *interp* function can be used to access elements of symbol arrays
  - interp(CaseArray,PowerArray,CaseFlag)
  - Look in CaseArray for Case Flag and return entry at same index in PowerArray – ***Not Valid in SINDA FORTRAN block…***
- Example symbol usages include:
  - *Power Dissipation (Heat Load or Heater)*
  - *Conductance Values or Convection Coefficient*
  - *PID Gains*
  - *Assembly Rotation Angles or Translations (Open/Close)*
  - *Thickness*
  - *Geometric Distances*
  - *Boundary Temperatures*
  - *Initial Temperatures*



  - *Setpoints*
  - *MLI Assignment*
  - *Number of Rays or Error for MCRT*
  - *Margin Factors/Multipliers*
  - *State Flag (On/Off)*
  - *Failure Flag (zero out value to disconnect or null)*
  - *Absorptivity/Emissivity/k/e*/etc*
  - *Etc…*

BUILD MODEL: Moving Geometry

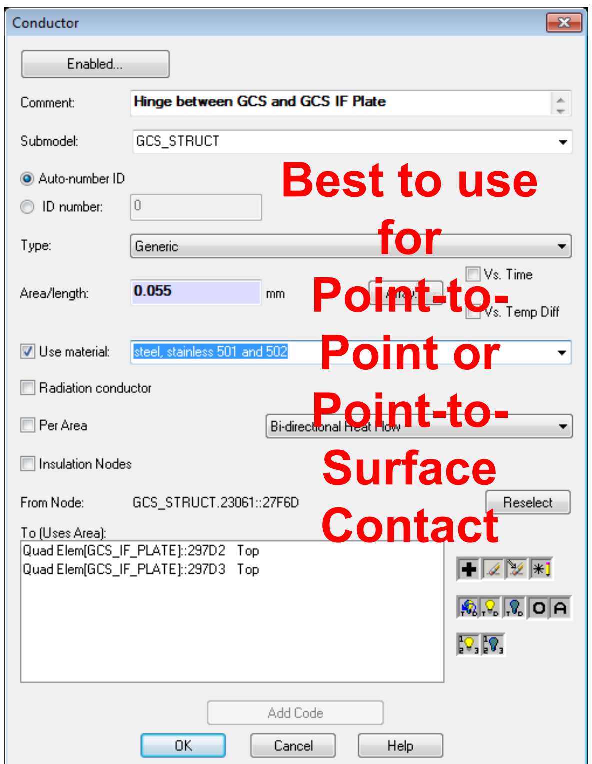
- Two methods exist in Thermal Desktop for handling moving geometry: ***Assemblies*** and ***Trackers***
  - For both, geometry must be attached
  - If using FE, nodes must be attached since nodes define elements
- Assemblies
  - Everything attached moves when assembly is moved
  - Be careful using AutoCAD move command; moving TD objects and the assembly makes everything attached to the assembly move twice.
  - Useful for moving geometry based on user needs (e.g. door opening, shield deploying, lid closing, etc). Could be orbit position dependent.
- Trackers
  - Similar to Assemblies, but point specified axis towards target by rotating about second specified axis. Can be nested, but gets complex quickly
  - Useful for simple solar arrays or more complex spacecraft pointing
  - Utilized by HR and Articulating Radk calculations
- Fast Spin (many times in one orbit position)
  - Fast spinning capability exists for radiation calculations for scan mirrors, rotating drums, etc.
  - Use an assembly, assign spinning geometry, and assign rotation angle to symbol variable
  - In Radk or HR Spin Tab, specify start angle, end angle and increments
  - Be sure to use 0 error and max number of rays per *position*
- Slow Spin/Slew (<< one time in orbit position)
  - Orbit Rotation or slew to multiple of *hrMeanAnom*



BUILD MODEL: Establishing Connectivity

- When nodes are not merged to establish connectivity between two surfaces, a user must add this manually. Two methods exist in Thermal Desktop for this: ***Conductors*** and ***Contactors***

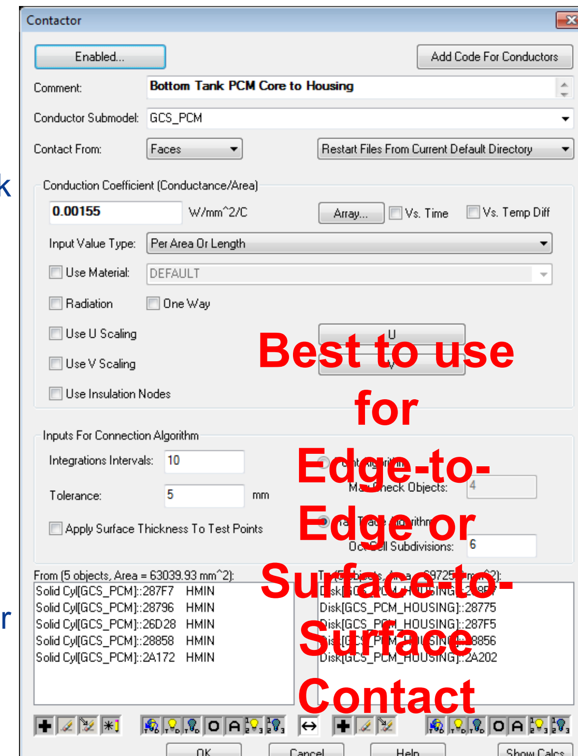
Conductor: Tie a specified node to a surface, multiple nodes, or a single node



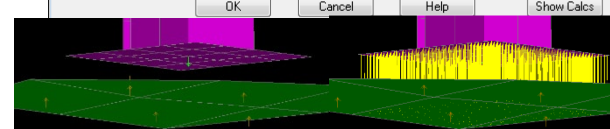
Best to use for Point-to-Point or Point-to-Surface Contact

- Both
  - Can be material dependent, time or temperature dependent or constant value
  - Can be to insulation nodes, one way, or radiative conductors
- Conductors
  - Remeshing/renodalization can break this link
  - Can be per Area, Cannot mix types (i.e. cannot add a surface to a node to node conductor)
- Contactors
  - Renodalization does not break this link, unless new elements are created
  - ✓ Must specify Edge or Face contactor type and edges/face sides to make contact
  - ✓ Use Ray Trace for Face Contact
  - ✓ Specify reasonable tolerance
  - ✓ Preference is to use Per Area or Length over absolute conductance
  - ✓ Often, best not to Apply Surface Thickness
  - Contact From displayed as Green Arrow, To faces as Yellow Arrows
  - Can visually display contact made (or missed)

Contactor: Tie nodes associated with two surfaces or two edges together

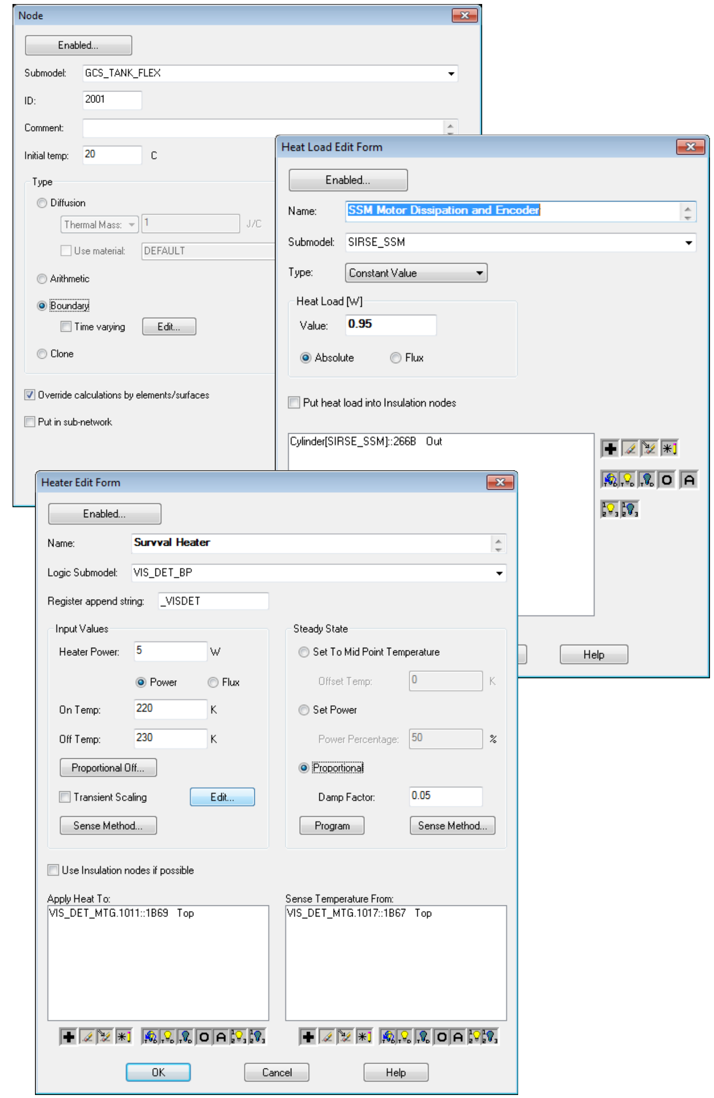


Best to use for Edge-to-Edge or Surface-to-Surface Contact



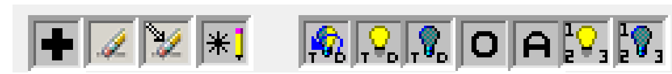
BUILD MODEL: Establishing Boundary Conditions

- **Boundary conditions must be applied to models. Typical boundary conditions for thermal analysis include: fixed temperatures and fixed heat loads.**
  – Boundary nodes defined by overriding the calculations by elements/surfaces. Need to also specify temperature.
- **Heat Loads**
  ✓ User specified application of heat to nodes, surfaces, or solids
  ✓ Can be total load or flux
  – Can be temperature dependent or time dependent
  ✓ Specify Value and assign appropriate submodel
- **Heaters**
  ✓ User specifies application of heat location and sensing location
  – Can be applied to nodes, surfaces, or solids
  ✓ Specify Available Power, On Temperature, and Off Temperature
  ✓ Specify control approach (thermostatic or proportional)
  ✓ Specify Steady State behavior (Proportional recommended)
  ✓ Specify Append string for symbol output recognition
    - TD creates power, total power, on time and #cycle registers using this string
  – Can specify sensing method (defaults to area weighted average temperature, but could be min temp or user defined)

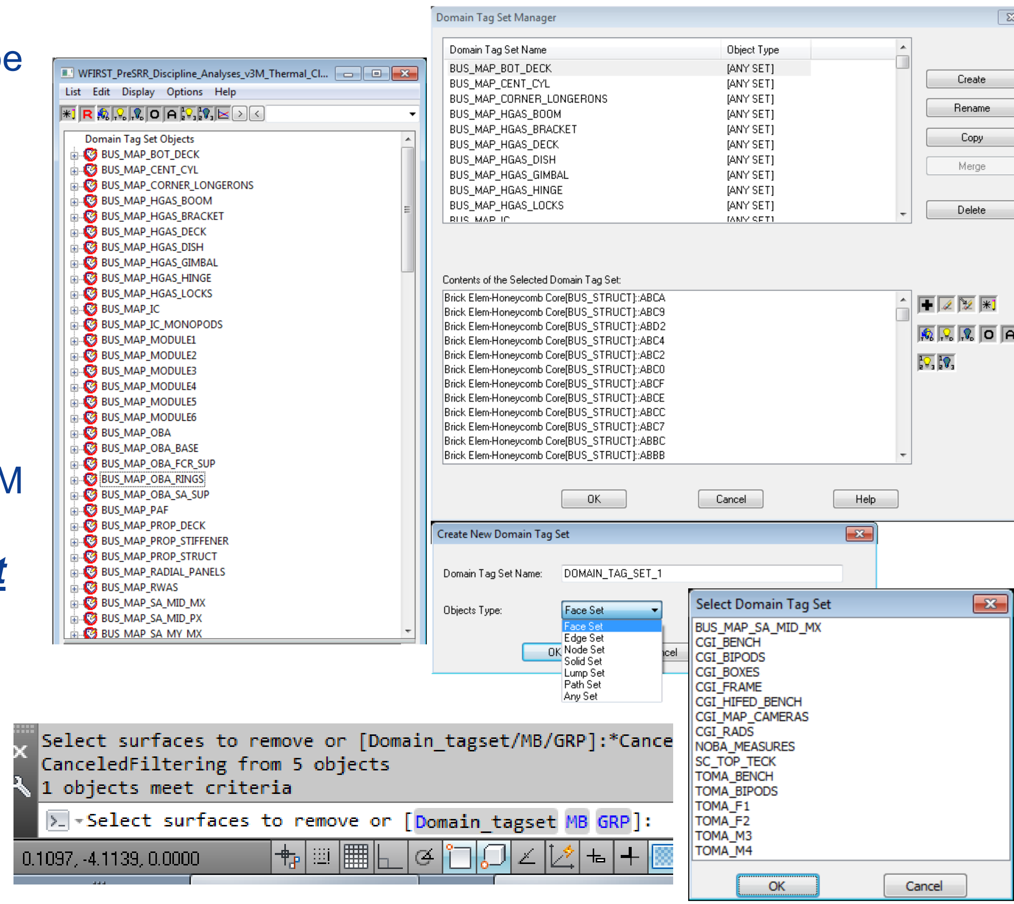


BUILD MODEL: Modifying Entities in Objects and Domain Tag Sets

- The entities referenced by Heat Loads, Heaters, Conductors, and Contactors can be modified from those references when originally created
  - Add (+), Remove, Remove Selected, Edit
  - Switch Visibility, Turn on Visibility, Turn Off Visibility, Show Only Selected (O) or All (A), Turn on, Turn off Node Numbers



- Domain Tag Sets are groups of objects that can be used in numerous ways
- Create the Domain Tag Set
  - ✓ Specify the Name
  - ✓ Specify the type of object it will contain (cannot change this later!!)
  - ✓ Add/remove objects using buttons described above
- Domain Tag Sets show up in Model Browser and are useful for controlling component On/Off visibility
- Used as Groups for mapping temperatures to FEM
- Domain Tag Sets may be used as target objects for Conductors, Heat Loads, Contactors ***even if it does not contain any objects yet***
- Useful for defining interfaces that may change on either side (dummy plate vs. SC deck)
- When prompted to select objects, type d to bring up list box. Only Domain Tag Sets of a valid type will be shown.



BUILD MODEL: Integrating Delivered Models

- Models delivered from other organization should have already been checked out. Delivered models (PARTs) and assembly models (ASSY) often need to be prepared for integration
  - ✓ *Remove boundary conditions from PART model that represent interfaces: boundary nodes, blocker surfaces. Keep conductors/contactors to boundary nodes to be able to connect to non-boundary interface. All that should remain is what needs to be imported into next higher assembly. Ensure that Radiation Group names are predefined or do not conflict with those that exist in the ASSY file. This should be a clean PART file for import*
  - ✓ *Remove previous instance of PART from the ASSY model if necessary or consider if better to keep old PART in ASSY and rename submodels with OLD_ prefix to have previous integration template. Save this as a clean ASSY file for integration*
  - ✓ *Ensure both PART and ASSY are using the same units*
  - ✓ *Use AutoCAD's INSERT command to insert PART file. Explode and place at correct location and orientation. Could use Copy/Paste if no Domain Tag Sets need to be preserved*
  - ✓ *If you need to move the entire inserted PART, be careful with any nodes/surfaces that may be attached to assembly. Moving everything will move the attached entities twice (relative to assy and then the assy itself)*
- ✓ Ensure that all PART submodels added have the same number of Nodes, Surfaces, Contactors, Heat Loads, etc. in the PART file and the new ASSY file
- ✓ Import Optical and Material Properties (Model Browser will show *Property Not Found* for undefined props)
- ✓ Import Symbols (Model Browser will show Symbol values as *[]* when referenced but not defined)
- ✓ Import Logic Object Manager objects and PART file node correspondence if needed
- ✓ Verify that Radiation Groups are correct. Can merge imported PART groups with existing ASSY groups
- ✓ Ensure Domain Tag Sets from PART are included in ASSY
- ✓ Determine if any symbol or property alias over rides from PART case set are necessary in ASSY case sets
- ✓ Reconnect contactors/conductors to ASSY side entities and remove any temporary PART boundary conditions. Merge any nodes that may be coincident between PART and ASSY prepared files
- ✓ Add Tab to Notes (Utilities-Notes-right Click Tabs) and paste text documentation (text only, no images, etc)

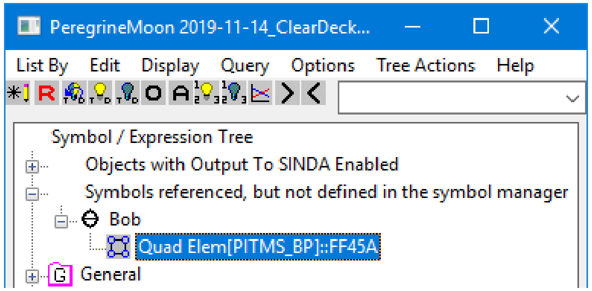
CHECK MODEL: General Model Checks

The following list can be used for your own model or models you received from others:

- ✓ **Ensure Material and Optical properties have correct values defined**
- ✓ **Ensure all surfaces have material and optical properties assigned**
- ✓ **Verify thicknesses** *(Model Checks-Color by Property Value-Thickness)*
- ✓ **Check Mass of Submodels** *(Model Checks-Calculate Mass)*
- ✓ **Check Free Edges of Elements** *(Model Checks-Show Free Edges)*
- ✓ **Check for Duplicate Nodes** *(Model Checks-List Duplicate Nodes)*
- ✓ **Check for Coincident Nodes** *(FD/FEM Network-Merge Coincident Nodes)*
- ✓ **Active Sides for Radiation Group** *(Model Checks-Display Active Sides)*
- ✓ **MLI Assignments** *(Model Checks-Active Display Preferences)*
- ✓ **Verify All Contactors make contact** *(Model Checks-Show Contactor Markers)*
- ✓ **Verify User Defined nodes are connected** *(Model Browser)*
- ✓ **Verify heater setpoints and Steady State behavior**
- ✓ **Verify proper behavior of assemblies/trackers** *(Orbit-Display Current Orbit)*
- ✓ **Verify proper associations of Symbols** *(Model Browser)*
- ✓ **Verify reasonable values for conductors, heaters, contactors, etc (units check and avoid large couplings)**

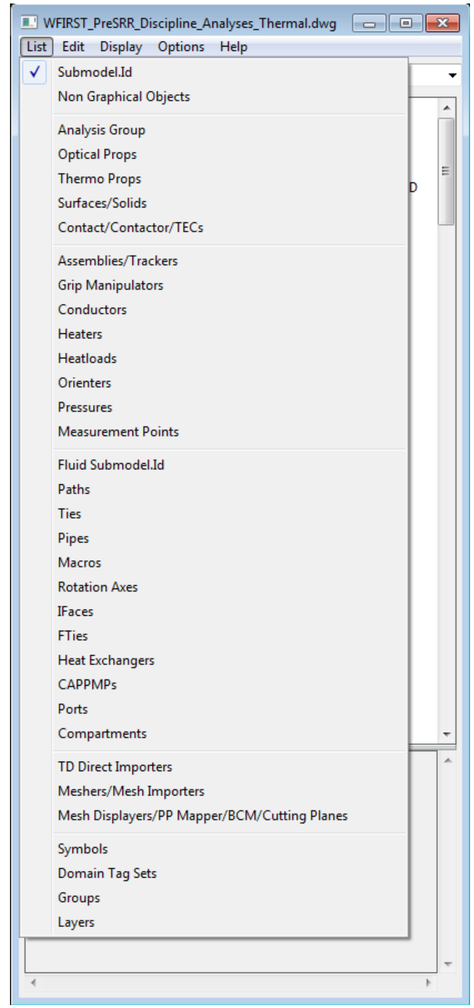
Submodel ->	Sub A	Sub …	Sub Z
Build Geometry	X	X	X
Shell Coat	X		
Nodal Subdivision	X		
Merge Nodes	X		
Submodel	X		
Node Numbering	X		
Thickness	X		
Material	X		
Material Orienters			
Active Sides			
Opt Props			
Insulation			
Comment			
Assembly/Tracker			
IF Contactors			
IF Conductors			
Heat loads			
Heaters			

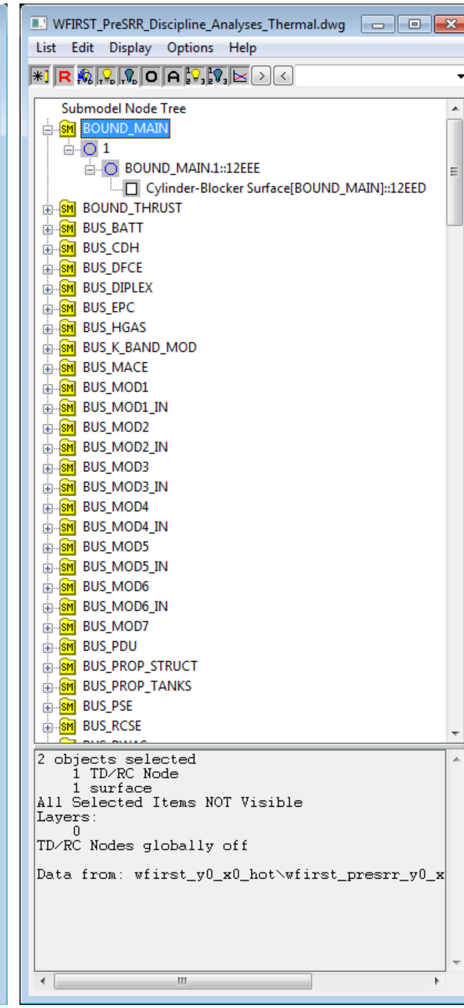
If building a lot of new geometry, it can be helpful to have a per submodel checklist to keep track of whether you have fully assigned everything needed



CHECK MODEL: Model Browser

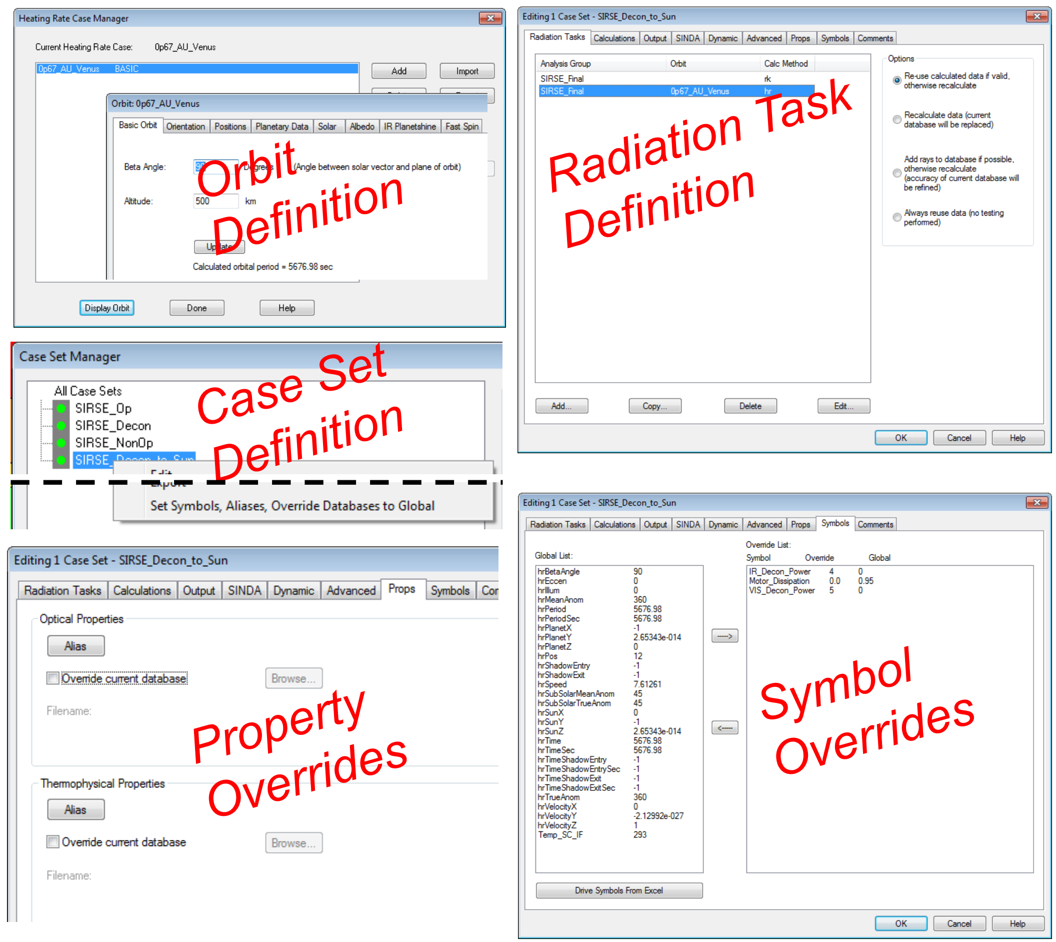
- The Model Browser is a powerful capability to view the relationships between all the Thermal Desktop objects and manipulate what is displayed in AutoCAD
  - *Many options for listing and controlling display*
  - *Shows relationship hierarchy (e.g. node-surface, Conductor-node)*
- Submodel: for model/node organization
- Non Graphical: Orbits, Properties, Case Sets
- Analysis Group for Active Side Visualization
- Optical and Thermo Props for property assignments
- Surfaces/Solids for Conduction Submodels
- Contact/Contactor for displaying associated surfaces
- Assemblies/Trackers for showing assembly hierarchy
- Conductors for user defined conductors
- Heaters/Heatloads for used defined dissipations
- Orienters for anisotropic conduction
- Fluid objects
- Meshers: TD Mesh definitions
- Mesh Displayers: FEM Mesh Mappers (STOP)
- Symbols for showing object dependencies
- Groups/Domain Tag Sets for user defined collections
- Layers for AutoCAD layers for visibility control





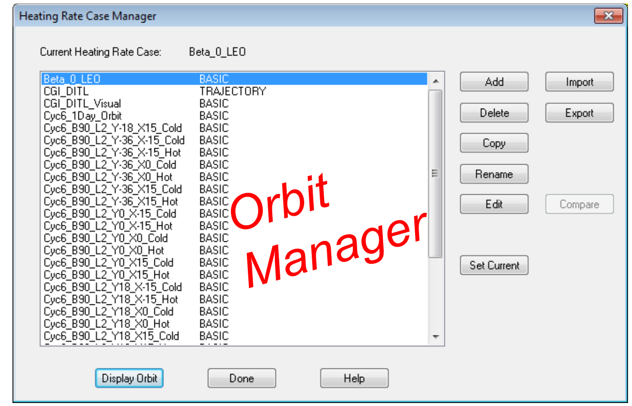
EXECUTE MODEL: Defining Analysis Cases

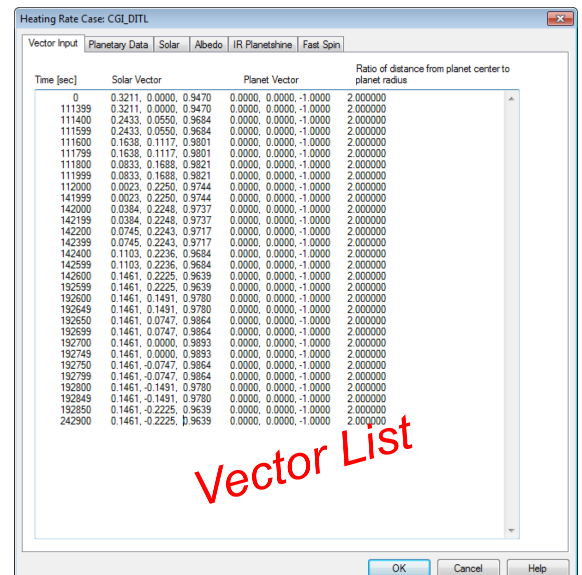
- ✓ Define Orbits
- ✓ Define Case Set name and File Names
- ✓ Define Radiation Tasks (Radiation Groups, Radk, HR, Articulating Radks, Orbits)
- ✓ Symbol Overrides
- ✓ Property Overrides
- ✓ Operations Block (TIMEND, Steady, Transient, etc)
- ✓ Verify Control Cases
- ✓ Define Outputs and Intervals
- ✓ Add Comments to document Case Set
- ✓ Temporarily Set all Symbols, Aliases, and Property DataBases to case set and verify model is as intended. Restore back when done verifying…
- ✓ Run Case(s)
- ✓ Post Process Results

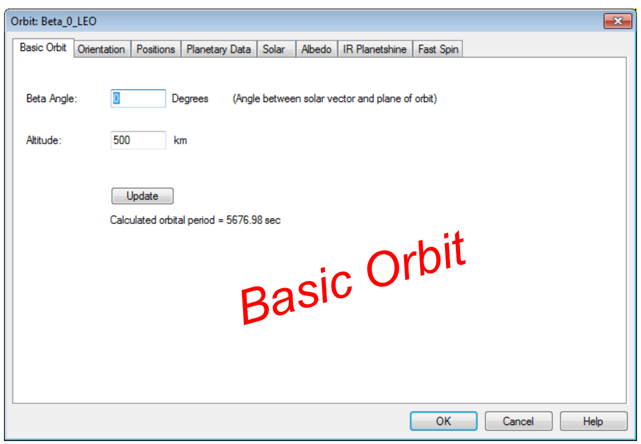


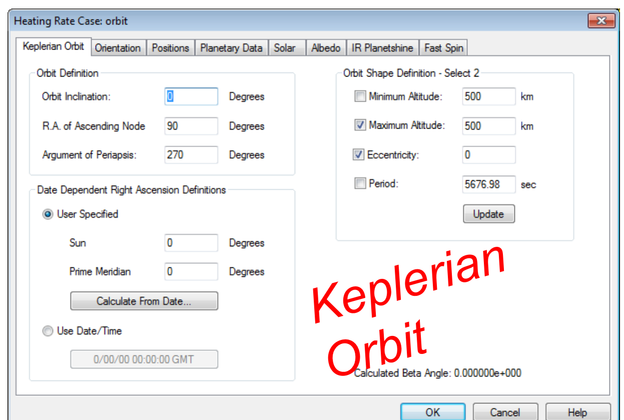
EXECUTE MODEL: Defining Orbits

- Orbits are defined to specify the variability of heating based on orbiting a celestial object (Planet, Sun, Moon, etc)
- Sources typically include: Direct Solar, Planetary Heating, and Albedo (Solar Reflections off Planet)
- Orbits can be specified by Altitude, Planetary Data (Size, Mass, Temperature, Fluxes, etc) or by Vector List (Good for small body orbits (comet, asteroid)). Orbits internally decompose to vector list for calculations
  - *Can use Beta = 90 with no Earth/Albedo for Lagrange orbits*
- Orbit Manager allows you to define multiple orbits and specify the current orbit for visualization
- Orbits are referenced in Radiation Tasks in the Case Set Manager
- Orbits can be imported from and exported to other models



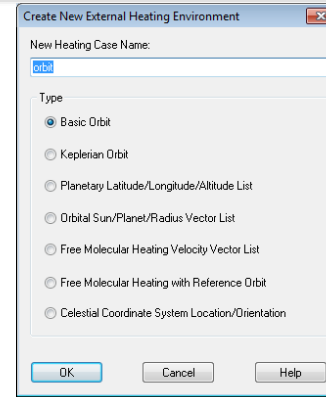


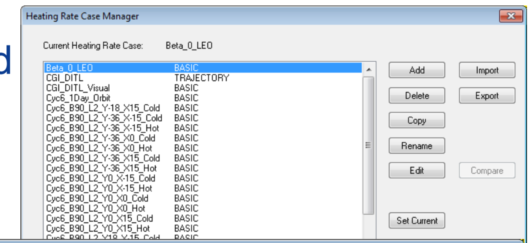


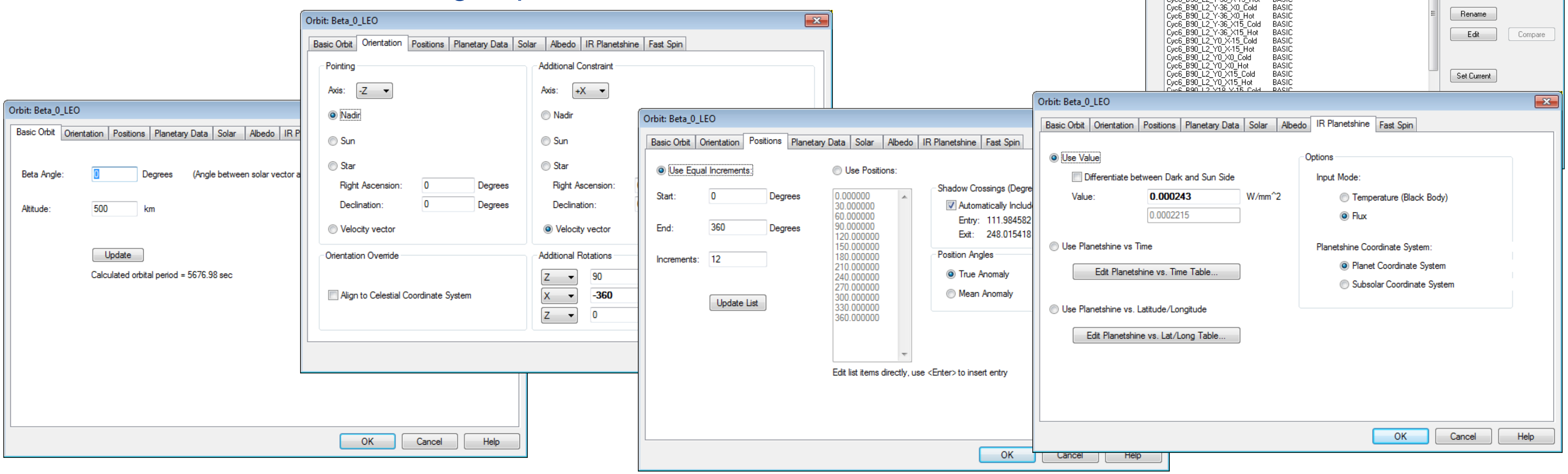


EXECUTE MODEL: Defining Orbits

- ✓ Open Orbit Manager and select Add…
- ✓ Define Type of orbit and Specify Name
- ✓ Enter Basic orbit information (e.g. Alt., Beta Angle, Incl., RAAN, Vector List, etc)
- ✓ Specify orientation of SC wrt celestial objects (e.g. Sun/Nadir/Zenith Pointing)
- ✓ Specify Euler Rotations to orient SC. Can use hrMeanAnom to do slow spin…
- ✓ Define number of orbit positions (default is 12: 15 positions with beginning and end + Eclipse Entry and Exit)
- ✓ Specify environmental heating (Solar, Albedo, IR Planetshine). Albedo and Planetshine can be Lat/Long dependent

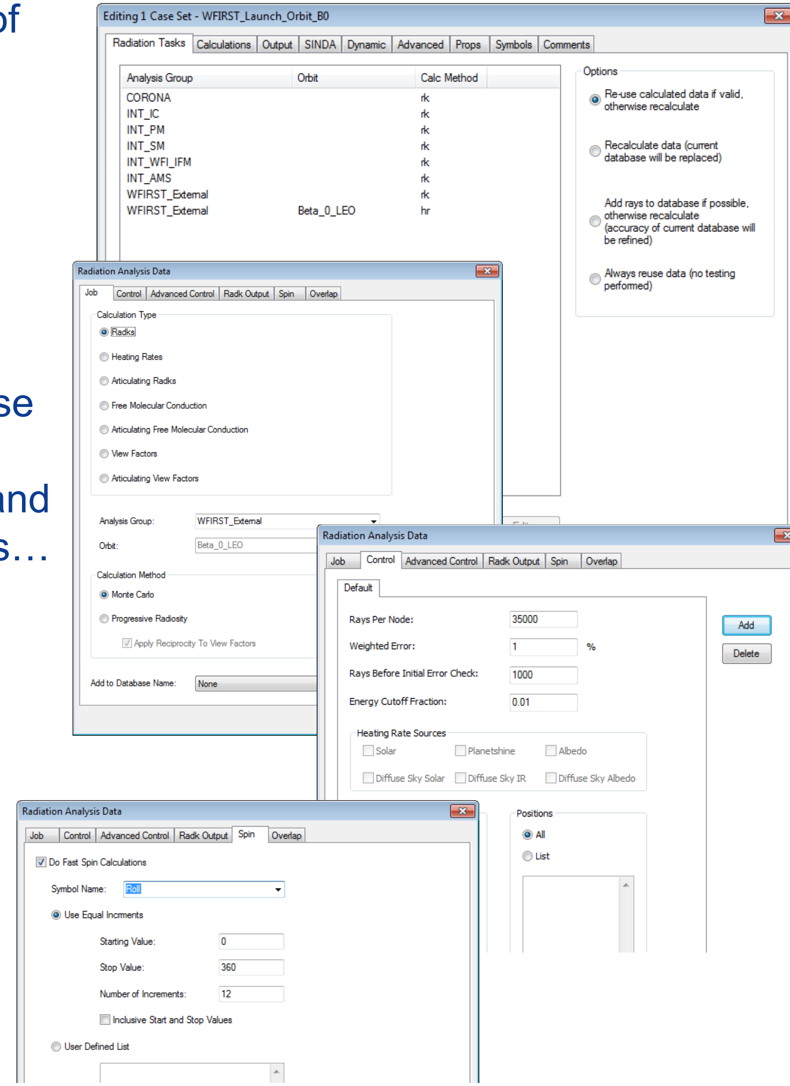






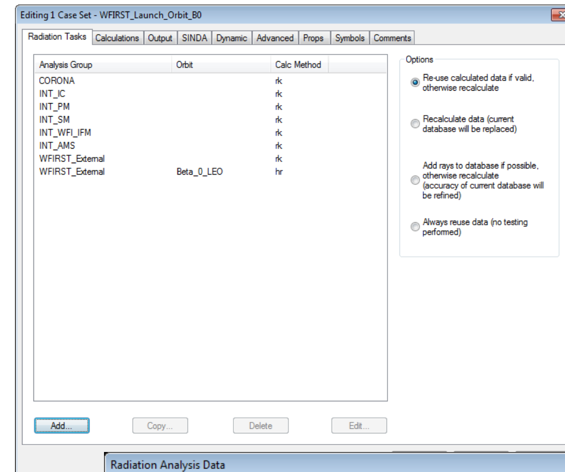
EXECUTE MODEL: Defining Radiation Tasks

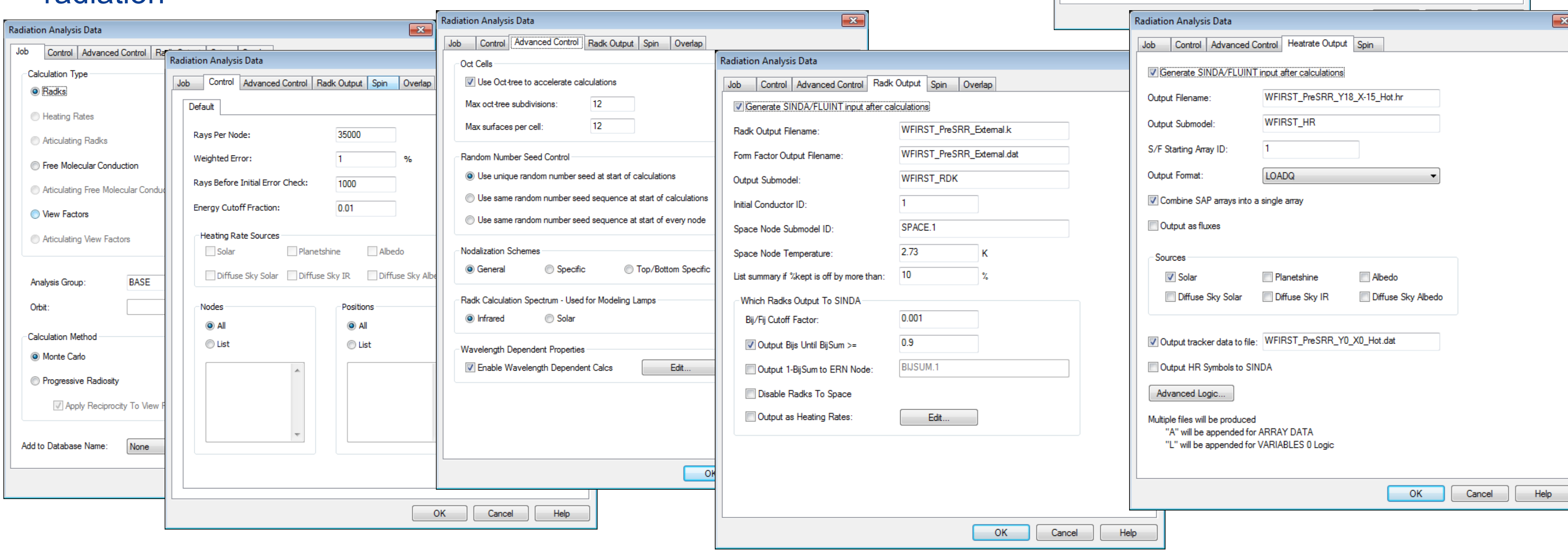
- Radiation Tasks are a subset of a Case Set. The results of these Tasks serve as inputs to the thermal model and generally include internal radiation within an enclosure, external radiation, and environmental heating. Radiation can be for moving geometry.
- Each radiation task generates a file that is included along with the Cond/Cap calcs to form the SINDA model for temperature solution
- Results from Radiation Tasks may be used by multiple case sets. TD is generally fairly smart at evaluating if anything has changed since the last time a radiation task was run and determining if the data is still valid. User can over ride this…
- Each radiation task has its own control parameters including Max Rays to fire, Acceptable Error Criteria, Output File and Submodel, Oct Cell Subdivision, etc
- Fast Spin Capability also exists to vary a spin angle multiple times within one orbit position. Averaging these results (Radk/HR) together over multiple spin positions in one orbit position can simulate a fast spinning object (Scan Mirror, Reflector, etc). To effectively utilize this, an assembly should be created with the spinning geometry attached and the rotation based on the symbol representing the spin angle. **Use 0 Error!**



EXECUTE MODEL: Defining Radiation Tasks

- ✓ Open Case Set, Go to *Radiation Tasks* Tab, Click Add…
- ✓ Define Type of Task (Radk, Heating Rates), Analysis Group, and Orbit if needed
- ✓ *Add to Database Name* if you don't want to overwrite similar
  - – Name of Database is RadGroup-Orbit-PropFile.rch or RadGroup-PropFile.rck
- ✓ Specify Max Rays, Error, and Energy Cutoff under *Control*
- ✓ Specify Levels and Objects on *Advanced Control* (Objects > 8)
- ✓ Set output filename and submodel…***DO NOT USE DEFAULTS!!***
- ✓ Set Bij Cutoff and Sum and disable Radks to space if internal radiation

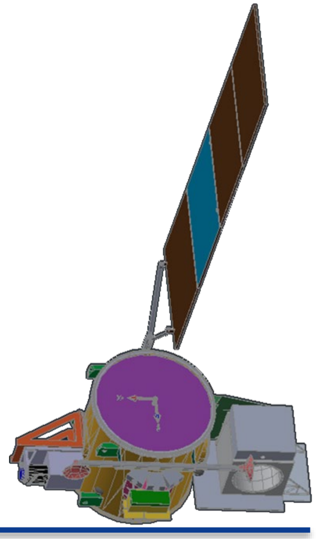




Building a Thermal Desktop Model

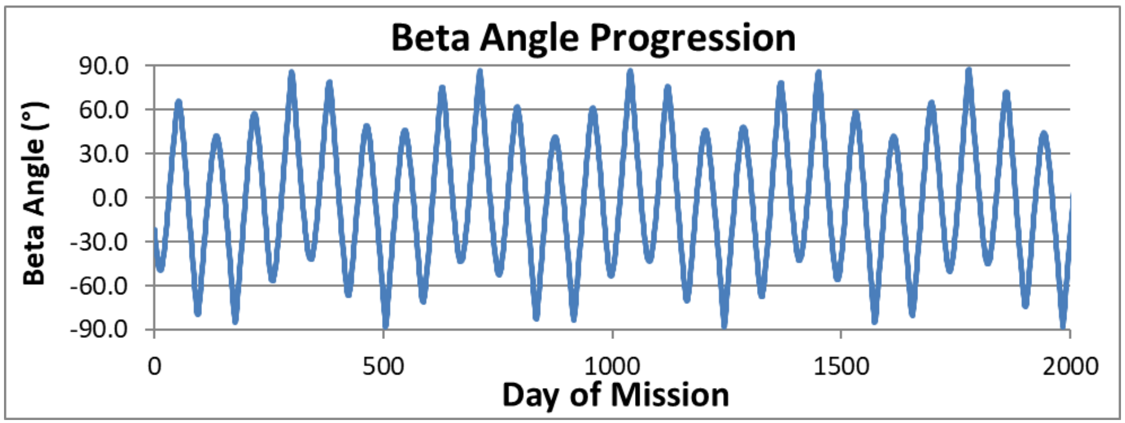
Building a Model from Scratch

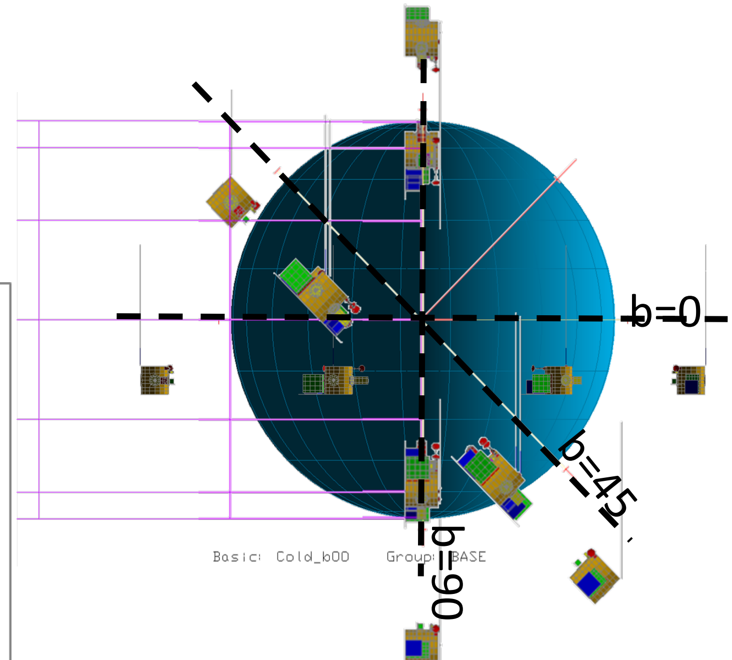
- A systems engineer (who was once a Thermal Engineer) asked if you could help them out on a proposal.
  - They know their orbit parameters and launch date
  - They have identified the Current Best Estimates for Dissipations
  - They have identified the Operating and Non-Operating Limits for their components
  - They have a CAD layout of their proposed design and know the materials they plan to use
  - They have concluded that there are no gradient or stability requirements
- They are hoping you could quickly build them a Thermal Desktop model and provide inputs as to whether their radiator size is adequate as well inputs for heater services
- Their design is built around using a propulsive ESPA Grande, which allows them to ride share with another spacecraft. As such, the dimensions are constrained by the size of the ESPA structure
- Three instruments are supported on the ESPA ports, all nadir (+Z facing). The instruments are isolated and the spacecraft is not responsible for their thermal control
- The spacecraft flies along the X axis, but performs a yaw flip at each Beta = 0 crossing



Orbit Info

- Launch Date: 7/25/28
- Altitude: 407 km
- Inclination: 65°
- Mission Duration: 3 years





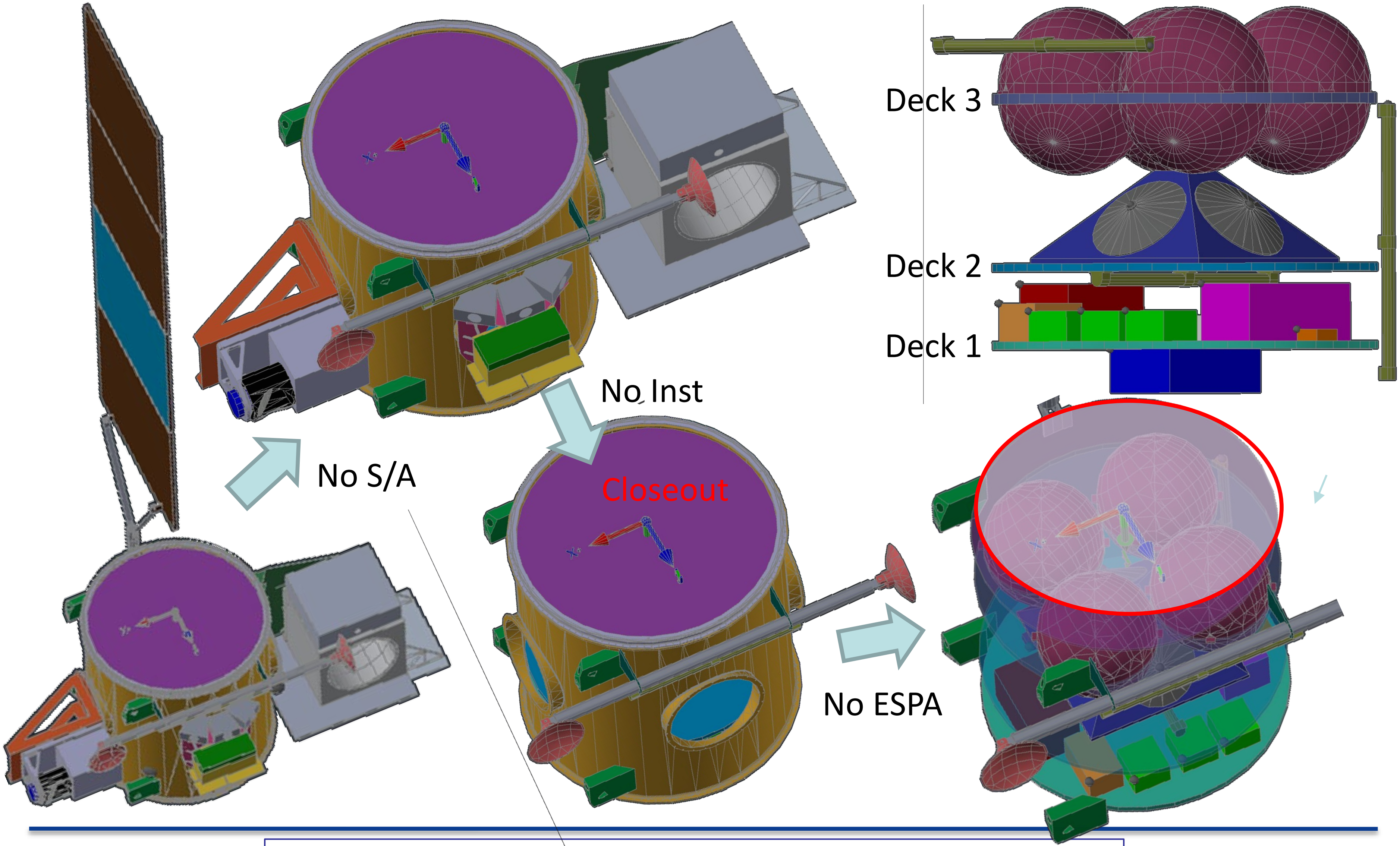
- So, their range of Beta Angles varies from -90° to +90°
- Their spacecraft coordinate system has +Z pointing nadir and Ram along the X axis
- Fortunately, their Systems engineer knew that a Yaw Flip (rotation about the Y axis of 180°) allows the beta angle to range from 0° to 90°, and this is their design
  - This also keeps the +Y side of their spacecraft from ever having direct solar illumination…good place for a radiator

Dissipation and Limit Info

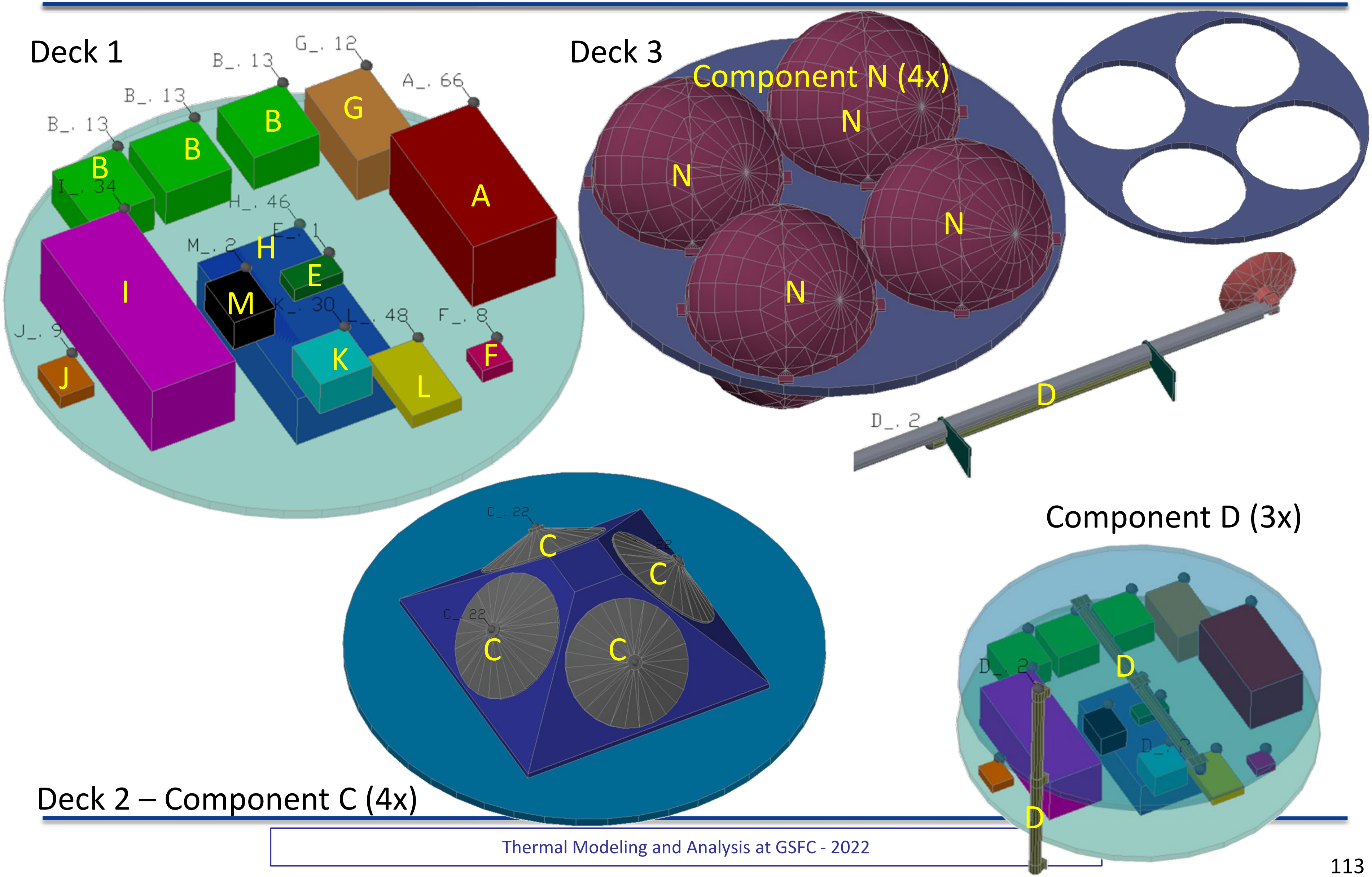
- Component H dissipates 46 W while in eclipse and 22 W while in sun, unless the orbit is full sun, for which the dissipation is 0 W
- Component K dissipates 30 W for 7 minutes out of every hour. For the remaining 53 minutes, it might dissipate 20 W for 7 of the 53 minutes, otherwise it dissipates 0 W.
- Component L dissipates 48 W for 7 minutes out of every hour. For the remaining 53 minutes, it dissipates 8 W. Assume the dissipations for K and L are synchronous.

Component	SurvLow (°C)	OpLow (°C)	OpHigh (°C)	SurvHigh (°C)	CBE Power (W)
A	-20	-10	40	50	66
B1, B2, B3	-20	-10	40	50	13
C	-20	-10	40	50	22
D1, D2, D3	-44	-34	71	81	2
E	-20	-10	40	50	1
F	-20	-10	40	50	8
G	-20	-10	40	50	12
H	0	10	30	40	46 Ecl/22 Sun/0 Full Sun
I	-20	-10	40	50	34
J	-20	-10	40	50	9
K	-20	-10	40	50	30 / 20 / 0
L	-20	-10	40	50	48 / 8
M	-20	-10	40	50	2
N	2	5	25	30	10 W Heaters 7/14 Top+Bot

CAD Layout



CAD Layout

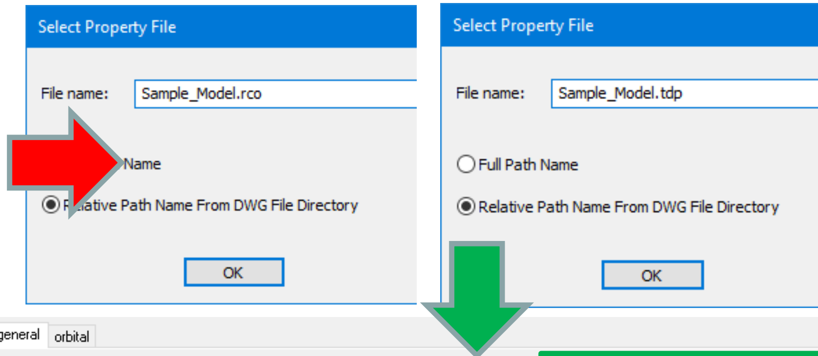

Deck 1
B_. 13
G_. 12
A_. 66
B_. 13
B_. 13
B
B
B
G
A
I_. 34
H_. 46
F_. 1
M_. 2
H
E
I
M
K_. 30
L_. 48
F_. 8
J_. 9
K
L
F
J
Deck 3
Component N (4x)
N
N
N
N
D_. 2
D
Deck 2 – Component C (4x)
C_. 22
C
C
C_. 22
C
C
Component D (3x)
D_. 2
D
D

Assumptions

- ESPA, Decks, Brackets, Avionics Box Walls and Baseplates are all Aluminum. The only non aluminum component is N, which is Titanium
- Boxes reject their dissipations through baseplate conduction and radiation off the sides should be neglected
  - The means only the baseplate footprint needs to be modeled
- The ESPA structure itself is covered with Multi-Layer Insulation (assume GBK Outer Layer and 0.05 e*)
- +Y Face (Deck 1) will be used as radiator using White Paint. Most avionics mount to –Y side of the +Y panel using a high conductivity interface filler (assume 0.8 W/in^2 K), except Component H, which is thermally isolated, blanketed, and rejects heat out the box top.
- The Truncated Pyramid on Deck 2 is also aluminum and is used to conduct heat to the ESPA structure. Components C dissipation can be applied directly to the pyramid. Assume the pyramid bolts to Deck 2 with 8 bolts (Assume 2 W/K/bolt)
- Decks 1, 2, 3 and the Closeout are both bolted with 24 bolts (Assume 2 W/K/bolt) to the ESPA structure around the circumference
- Component N is thermally isolated from Deck 3 (Assume 4 locations at 0.1 W each) around the midplane circumference where it mounts to the deck.
- Files needed to get started in zip archive embedded in this slide

Sample_Model_Files.zip

Model Preparation: Symbols and Properties



general | orbital

Name	Result	Expression	Comment
Hot_Case	0	0	-1 ro Surv, 0 for Cold, 1 for Hot
Lock_Array_b...	0	0	1 if Locking to prevent tracking during Beta
Power_Scale	1	1	Power Growth Factor for Radiator Sizing
Q_Comp_A	66	66	
Q_Comp_B	13	13	
Q_Comp_C	22	22	
Q_Comp_D	2	2	
Q_Comp_E	1	1	
Q_Comp_F	8	8	
Q_Comp_G	12	12	
Q_Comp_H	46	46	Component H dissipates 46 W while in eclip
Q_Comp_I	34	34	
Q_Comp_J	9	9	
Q_Comp_K	30	30	Component K dissipates 30 W for 7 minutes
Q_Comp_L	48	48	Component L dissipates 48 W for 7 minutes
Q_Comp_M	2	2	

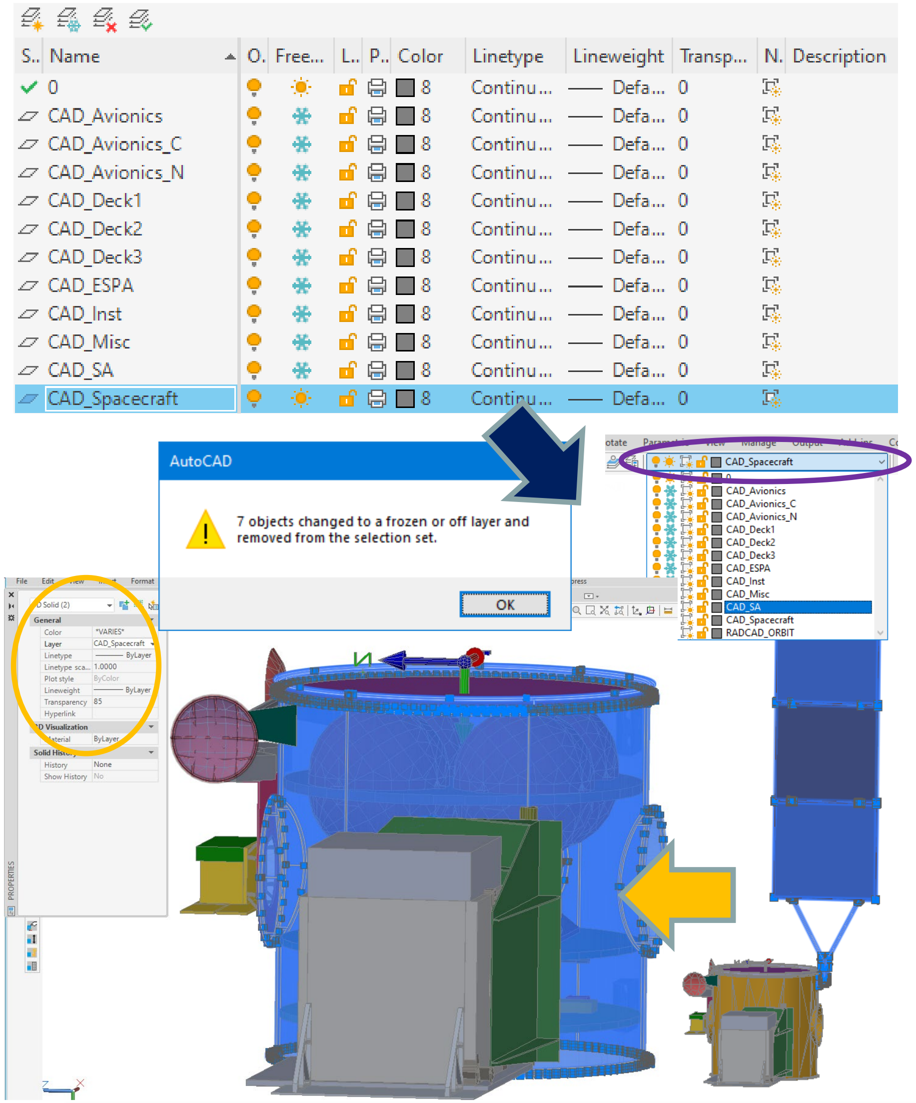
Name	Result	Expressior
Q_Comp_A	66	66
Q_Comp_B	13	13
Q_Comp_C	22	22
Q_Comp_D	2	2
Q_Comp_E	1	1
Q_Comp_F	8	8
Q_Comp_G	12	12
Q_Comp_H	46	46
Q_Comp_I	34	34
Q_Comp_J	9	9
Q_Comp_K	30	30
Q_Comp_L	48	48
Q_Comp_M	2	2

- Open a new Thermal Desktop model and save it as Sample_Model.dwg. Create a new optics file (*Optical Properties-Open/Create Property DB*) and save it as Sample_Model.rco. Do the same for the material property file (*Thermophysical Properties-Open/Create Property DB*) and name it Sample_Model.tdp
- Add Symbols for each Component Dissipation (rcEditSymbol) as shown to left. These will be used to assign dissipations to the various components
- Add a new symbol named Hot_Case:
  - -1 will indicate a survival case
  - 0 will indicates a cold case
  - 1 will indicate a hot case
- Add a new symbol named Power_Scale
  - All dissipations should be multiplied by this value to allow for a power growth factor for sizing radiators in hot cases
- Add a new symbol named Lock_Array_b90
  - This will be used later to handle the solar array tracking for Beta 90
- Define the optical properties as shown to the left
- Define the material properties as shown below
- Type in Units; set to Inches. Do this for TD too. Then type CLASSICINSERT to insert Sample_CAD.dwg to 0,0,0.

Name	Solar Absorptivity	IR Emissivity	a/e
NoRad	1.000	1.000	1.000
SC_BlackAnodize	0.730	0.820	0.890
SC_BlanketExt	0.600	0.800	0.750
SC_SilverTeflon	0.090	0.850	0.106
SC_SolarArray	0.710	0.820	0.866
SC_WhitePaint	0.110	0.890	0.124

Name	Cond [W/in/K]	Dens [kg/in^3]	Cp [J/kg/K]	Eff Emiss	Type
SC_Aluminum	4.2418	0.0442451	896		
SC_MLI_05	0	9.83224e-06	0	0.05	
SC_Titanium	0.1905	0.0721031	540		

Preparing the CAD



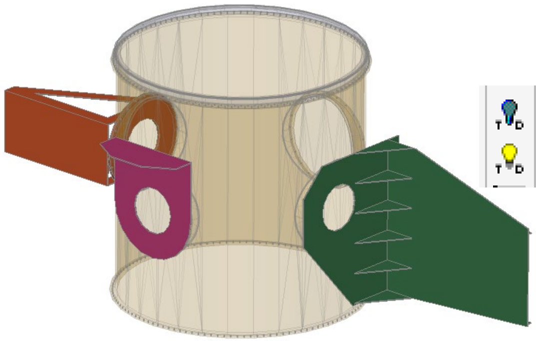
- LAYERs are a good way to organize a model and prepare CAD for use in building the thermal model. There is only one layer that is current; this layer is where any new entities are added. Good practice to add CAD_ prefix in front of layers for CAD geometry which groups them all together alphabetically.
- Begin by adding new layers to divide the model into useful groupings. Add all the layers as shown to the left with the Snowflake icon
- Visibility of layers can be controlled by either turning the layer on or off (Lightbulb) or Freeze/Thawing the layer (Snowflake/Sun). Nominally, freeze/thaw is better than On/Off, but both are still available for legacy reasons. Make all the newly created layers frozen. Can also adjust transparency for all objects on particular layer
- Objects can be selected at the main interface and their layer changed by selecting the new layer in the toolbar dropdown. Changing an objects layer to one that is frozen, will impact that objects visibility. This is a good way to start with larger, easier selectable objects and move them to frozen layers and work your way deeper into the CAD geometry
- Could also change the layer using PROPERTIES command. This also allows transparency to be set at an object level

Preparing the CAD

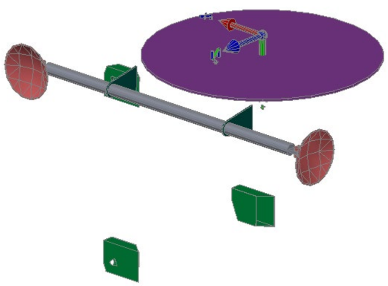
- Define the layers in the model as shown on this slide
- Use tdPurgeBlocks command to remove unused layers, blocks, groups, etc
- The rcVisOff command may be used to turn visibility off for CAD when the object can be selected. However, to turn on visibility for an object not currently visible for selection, might need to use <u>ALL</u> option or Groups at Select Objects prompt

CAD_SA

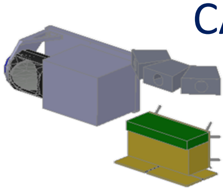
CAD_ESPA



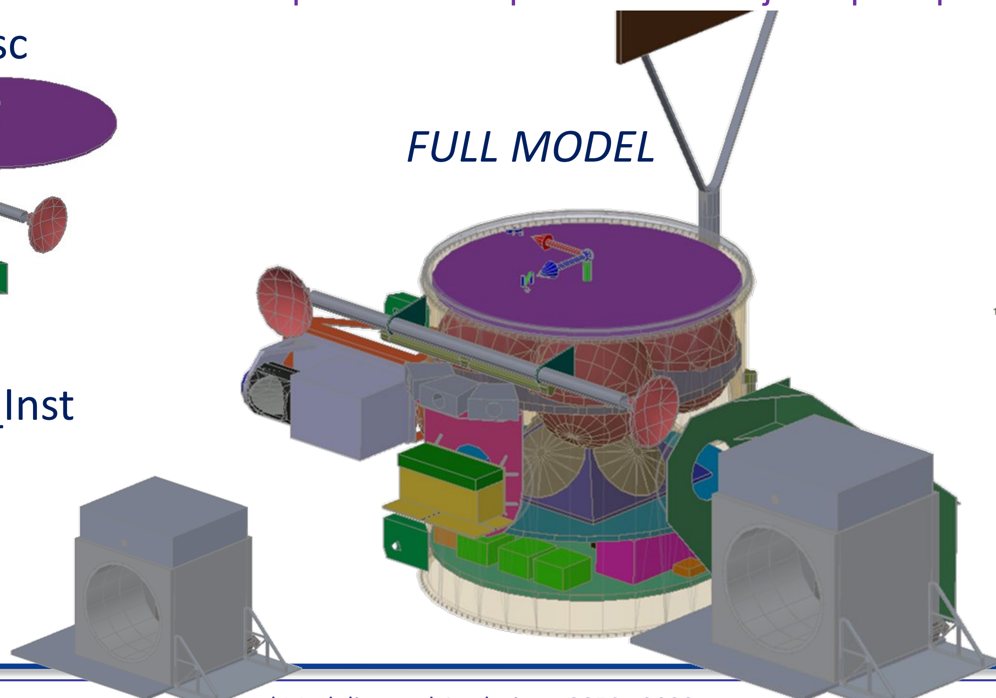
CAD_Misc



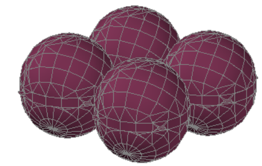
CAD_Inst



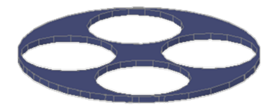
FULL MODEL



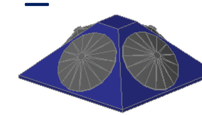
CAD_Avionics_N



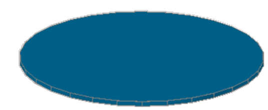
CAD_Deck3



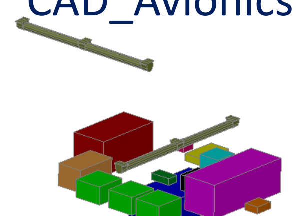
CAD_Avionics_C



CAD_Deck2

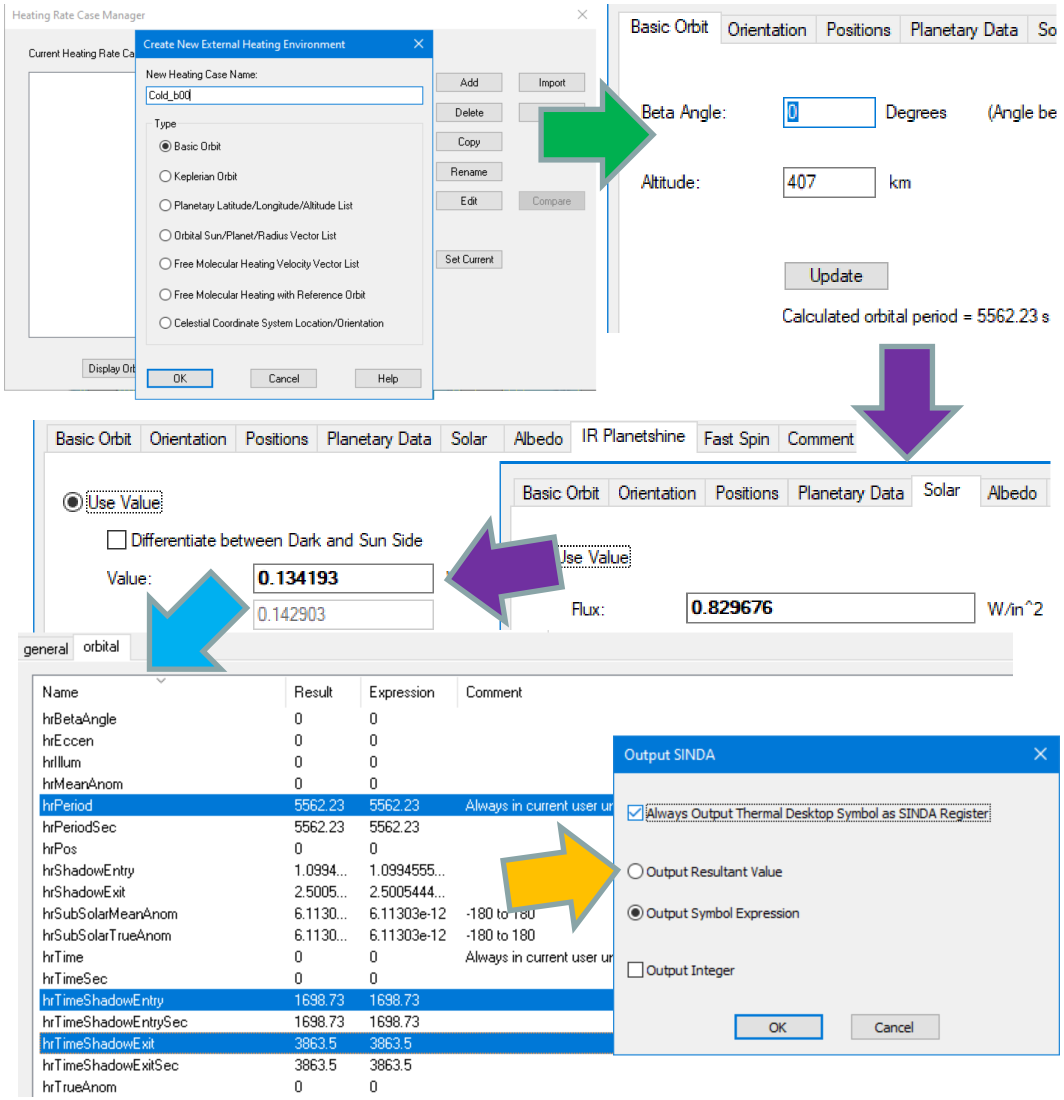


CAD_Avionics



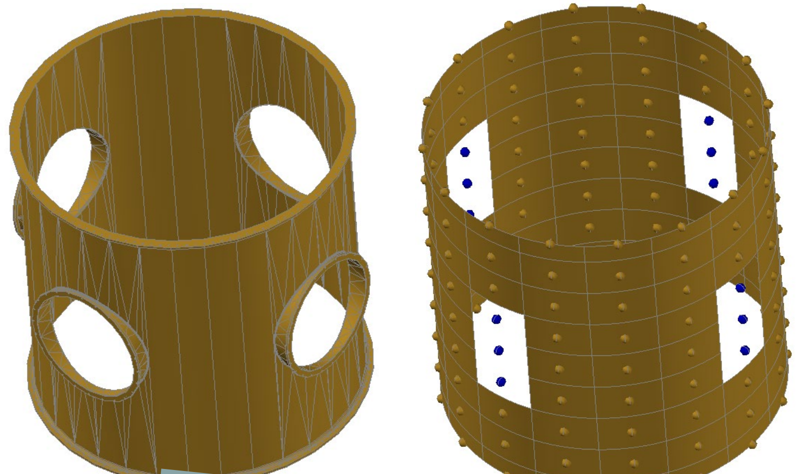
CAD_Deck1

Creating Orbits

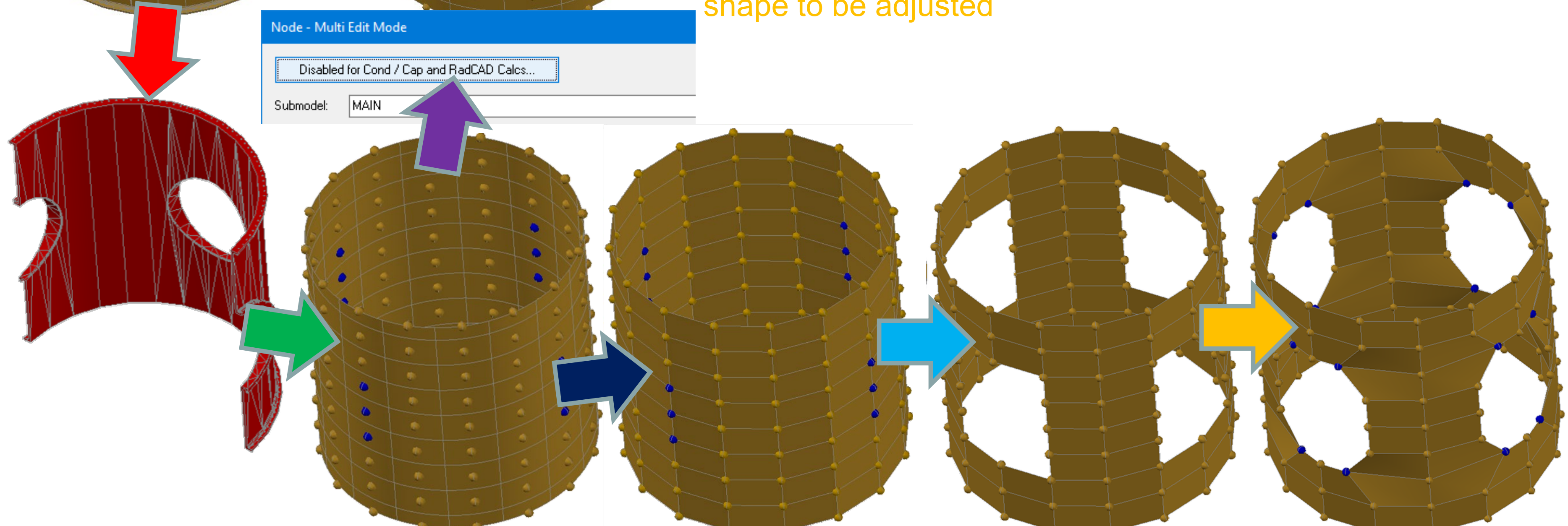


- Since Beta angle varies between 0 and 90, multiple orbits should be investigated. For this evaluation, increments of 15° are probably good enough. Open the Orbit Manager (rcManageOrbits)
- Add a new orbit named Cold_b00 as a Basic orbit, which needs only Beta Angle and Altitude for trajectory information. Enter Beta Angle=0 and Altitude=407 km. Note that the orbital period is also calculated.
- Since this is a cold case, it is advisable to enter the cold biased environments. This could also be done with Symbol overrides in CaseSets, but this might force recalculation of the HeatRates when all that changed are the fluxes. Best to have different orbits
    - Use 1286 W/m^2 (Cold) and 1420 (Hot) for Solar
    - Use 0.25 (Cold) and 0.35 (Hot) for Albedo
    - Use 208 W/m2 (Cold) and 265 (Hot) for Planet IR
- Accept the default of 12 positions, +Z (Nadir), and +X (Velocity). Note that timesteps are created just before/after eclipse entry and exit. Create Hot and Cold cases for 0,15,30,45,60,75,90 beta angles
- Edit the symbols (rcSymbol) and take note of all the symbols now created on the orbital tab. Select the indicated ones and edit.
- Click *Control Symbol Output to SINDA* button and select *Always Output Thermal Desktop Symbol as SINDA Register*. This will be used later…

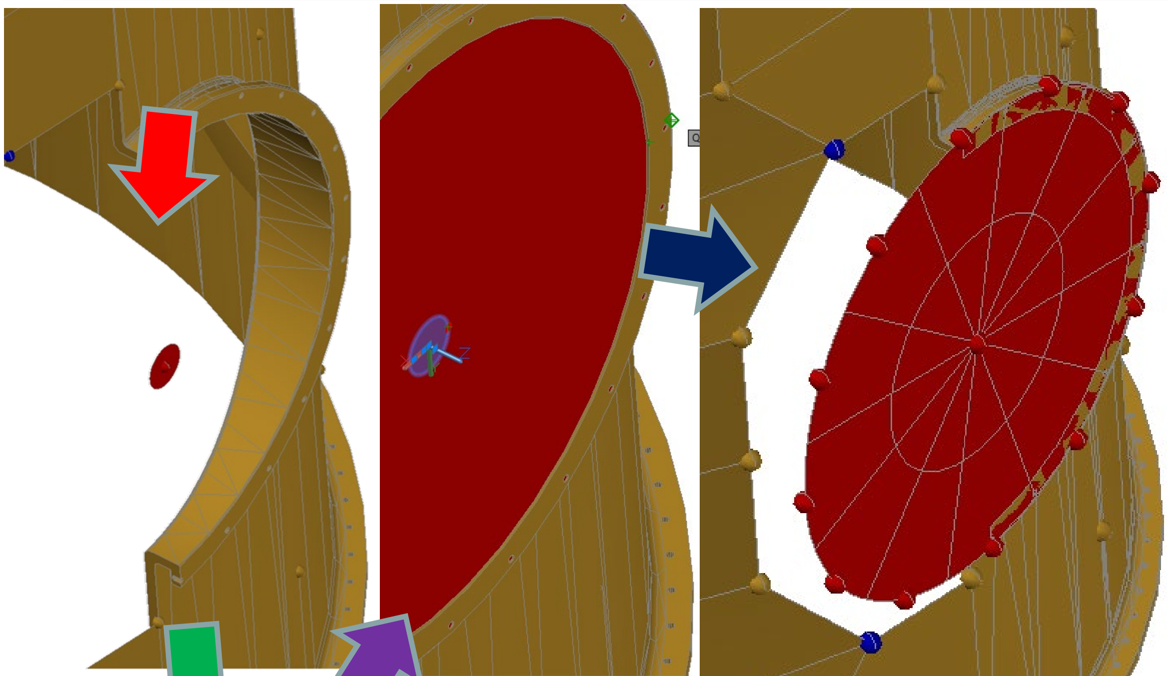
ESPA Main Structure

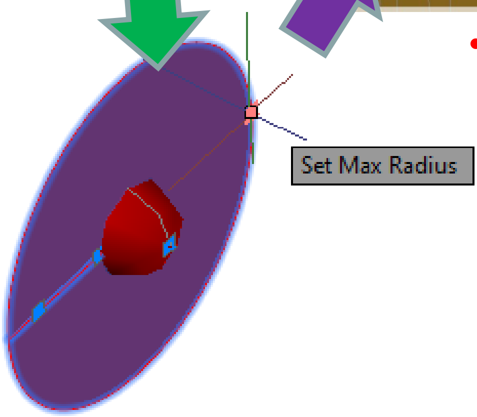


- SLICE solid along YZ plane and select point about axis to yield a Half ESPA.
- Use Points to define cylinder, nodalized as Edge Nodes (17 x 9)
- Select nodes indicated in Blue and Edit as a group. Make them disabled to effectively make an opening in a surface
- Alternately, you could Convert to FE (*Thermal-Modeling Tools-Convert Finite Difference to Finite Element*) but this is a one way operation
- Elements can be deleted to make cutouts
- Moving Nodes shown in Blue (Bottom Right) allows for element shape to be adjusted



ESPA Ports Structure

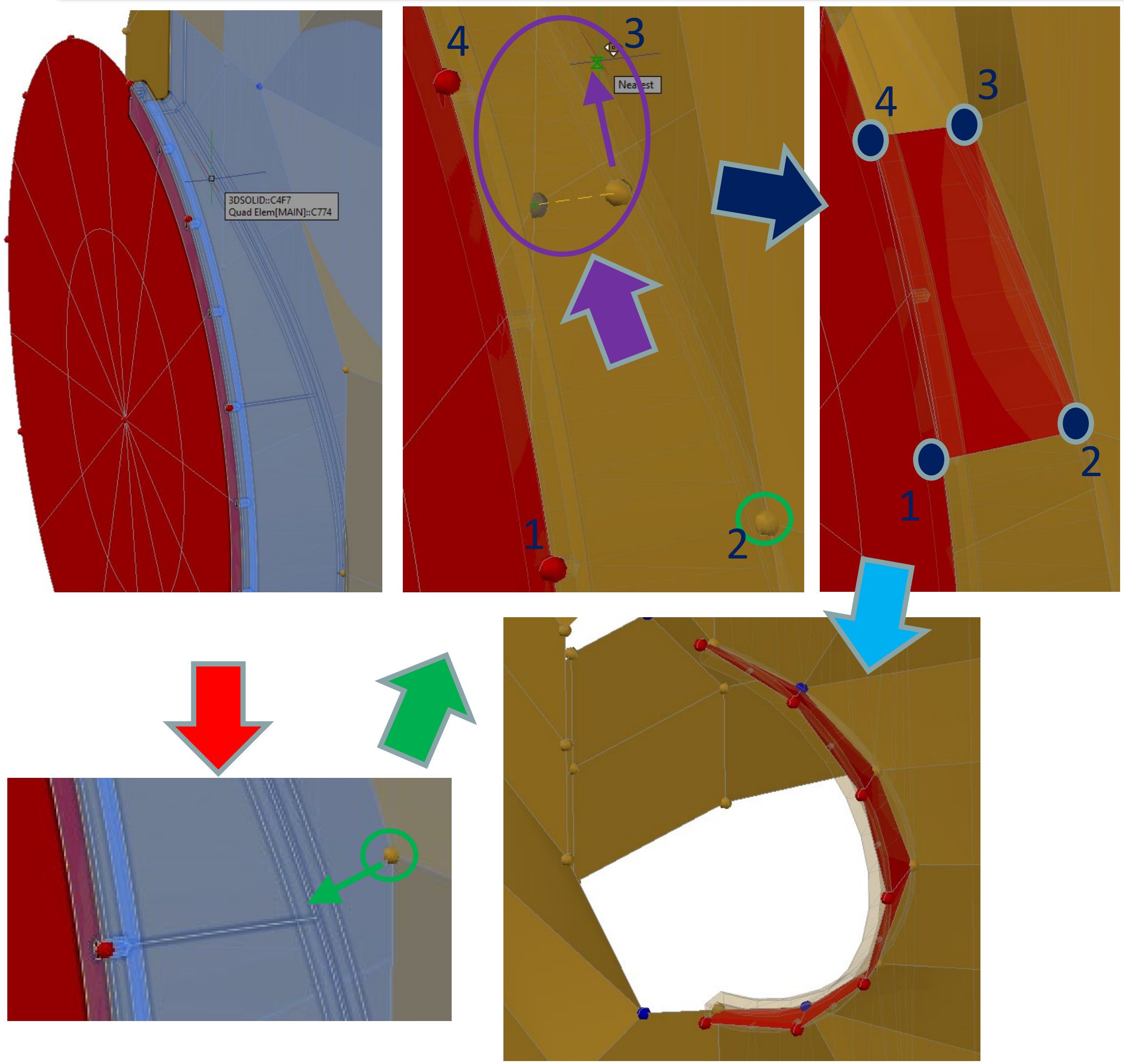




- Using the port on the –Z side, create a disc using the center point and a second point along –Z to define the axis. Accept the remaining default parameters. Note that this helps keep the axes aligned with the User Coordinate System (UCS).

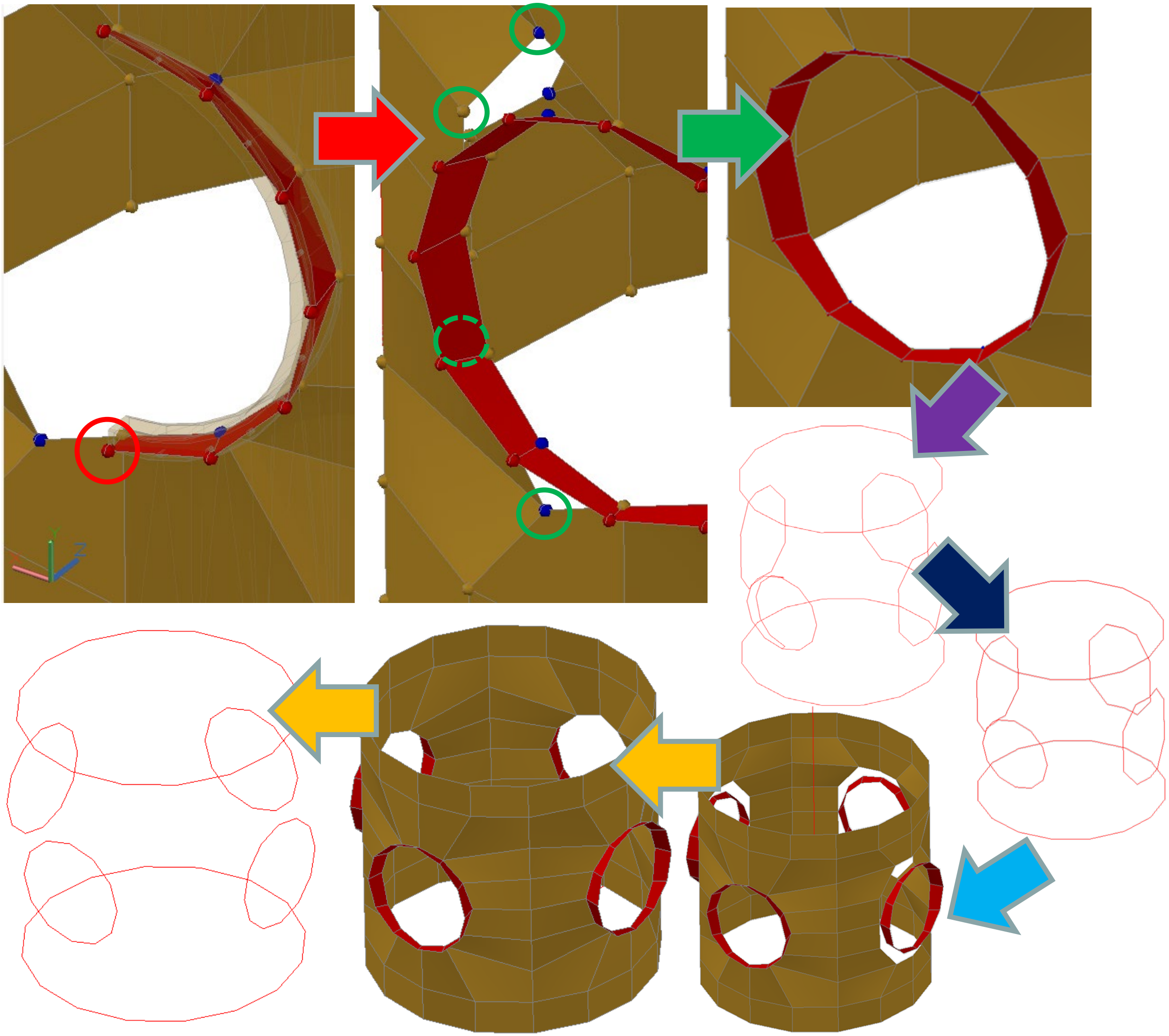
- Selecting the disc will display the grips (small blue squares). Each grip allows some geometrical parameter of the surface to be modified. The right-most grip controls the outer diameter. Note that if a geometric parameter (e.g. Max Radius) is defined as an expression, grip manipulation is not allowed.
- Select this grip and drag it towards the right. Select QUAD to snap to a quadrant on a curve and pick the point shown to the right with a green diamond indicating the quad location. This has now sized the diameter to that of the port. Other useful Object Snaps include: End, Mid, Int, Cen
- Nodalizing the disc with 9 angular/ 2 radial divisions will create nodes that can be then used to generate elements for the port walls.

ESPA Ports Structure Walls



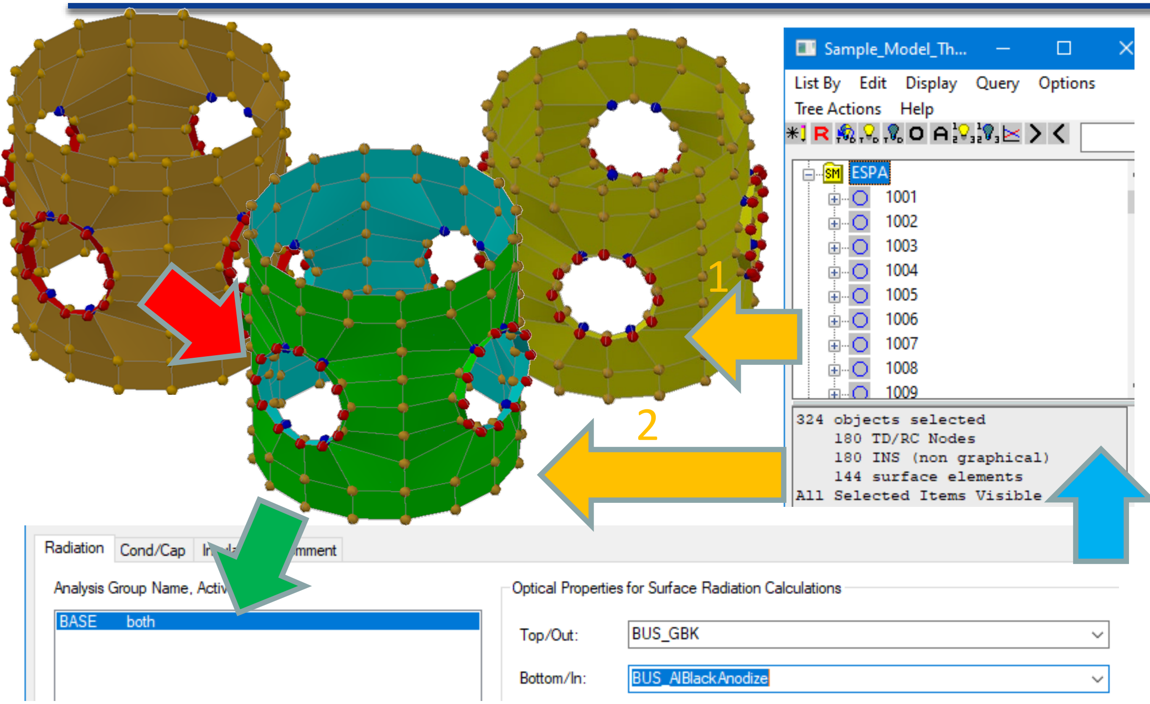
- Recall that unlike surfaces, where the node locations are defined by the surface, finite element shapes are defined by their nodes. Therefore, to alter the shape of an element, it is the node that must be MOVEd.
- This node moved to align with CAD
- This node is in the process of being moved to align with CAD using NEAR snap point
- 4 nodes used to create element using 1-2-3-4 order, following right hand rule to make top side of element facing outward
- Continue to move nodes and make elements for half of the port (on –X side of Y axis) – should be 6 elements when complete *(Note for top and bottom nodes just move them an approximate distance along the Y axis to align with CAD)*

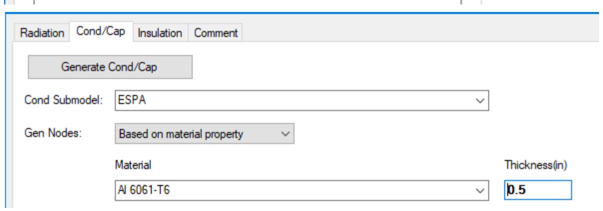
ESPA Ports Structure Walls

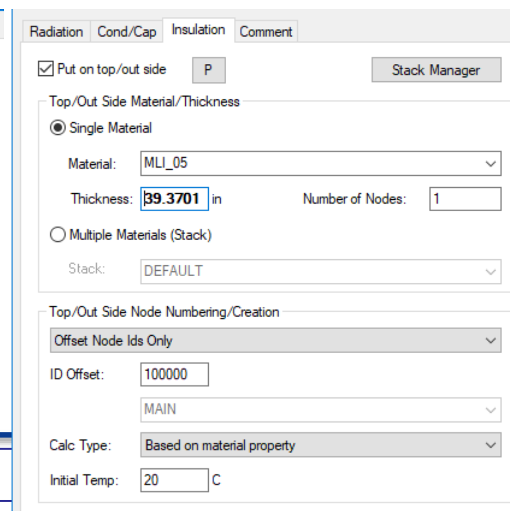


- Use the MIRROR3D command to mirror the 6 elements about the YZ plane using the lower point indicated as the point on the plane
- Move the indicated points to the new node locations so that the gaps are closed out. Hint, the one in the dashed circle may be easier to select using a Selection Window in Wireframe display style
- Perform a free edge check using *Modeling Tools-Show Free Edges*
- Select *FD/FEM Network-Merge Coincident Nodes* and select all nodes. Use tolerance of 0.1 inches and run the free edge check again.
- Rotate UCS such that Z is aligned with cylinder axis. Then use ARRAYCLASSIC command to create a Polar array of 4 objects filling 360° to create the other ports. Note that this command is always about the current Z axis.
- Remember to move and merge nodes and re-check free edges

Making the ESPA Geometry into Thermal





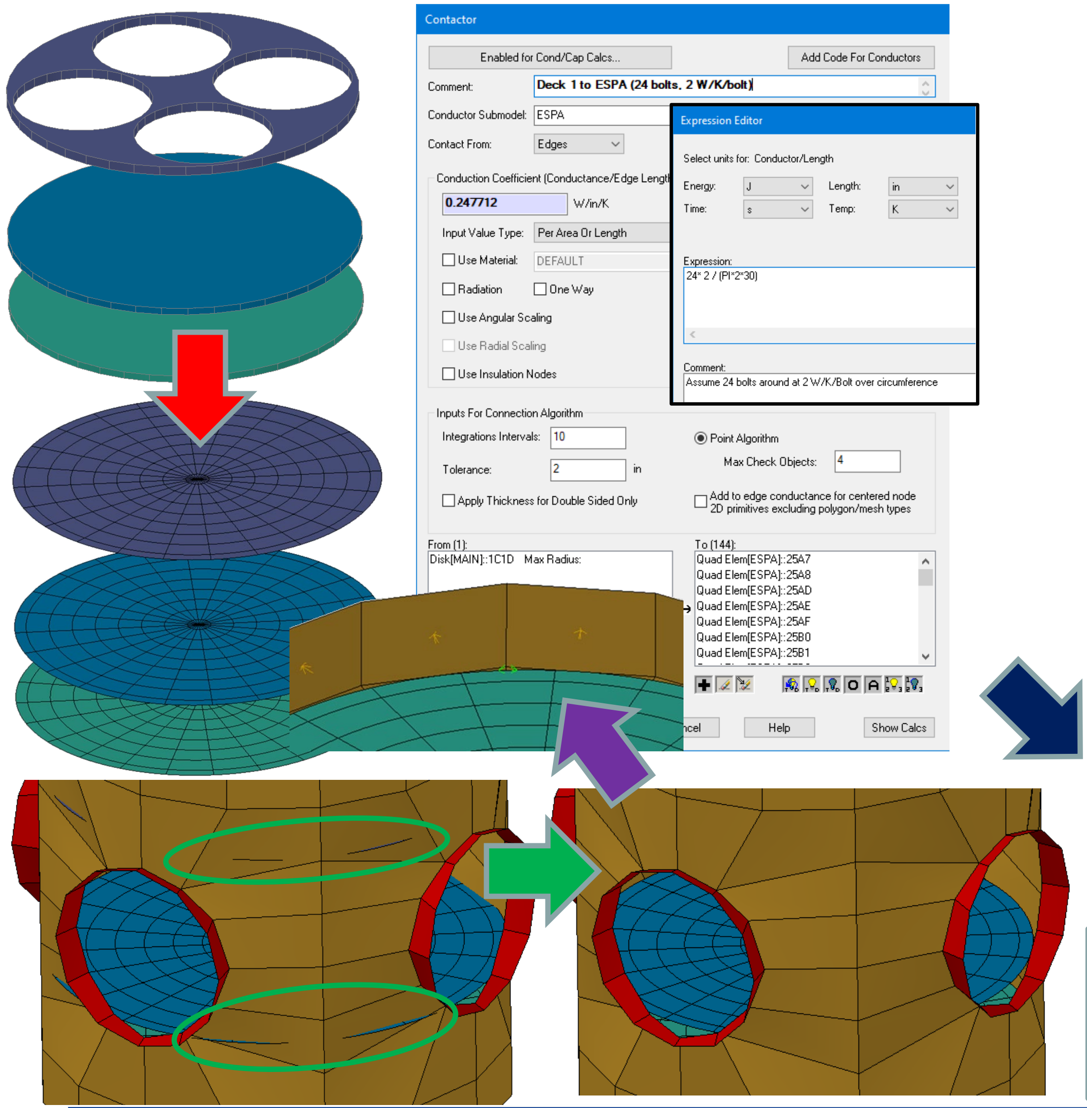


- *Model Checks-Display Sides Preferences:* select Top Side and Display. If any are incorrect, you can reverse the connectivity using *Modeling Tools-Reverse Connectivity of Planar Elements/ Meshes.* If there are many, might be easier to use *FD/FEM Network-Synchronize Element Normals*
- Edit all Surfaces and set the following:
  - RADIATION: Active: Both
  - RADIATION: Top/Out: BUS_GBK
  - RADIATION: Bottom/In: BUS_AlBlackAnodize
  - COND/CAP: Cond Submodel: ESPA
  - COND/CAP: Material: Al 6061-T6
  - COND/CAP: Thickness: 0.5 in (as expression)
  - INSULATION: Put on Top/Out Side (checked)
  - INSULATION: Material: MLI_05
  - INSULATION: Thickness: 1 m (as expression) *
  - INSUALTION: ID Offset: 100000 **
- Edit all nodes and set Submodel to ESPA
- *Modeling Tools-Resequence ID's…* and select all nodes. Use starting number of 1001 and offset of 1
- Check the ModelBrowser to see that the # of nodes, surfaces, and insulation match.
- Check that both sides are Active (1) and MLI is applied to only the outside (2). Use editActivePrefs

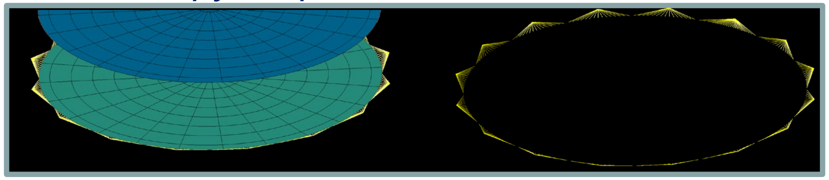
* MLI_05 property has a mass of 0.6 kg/m3. Making the thickness 1 m allows MLI mass to be estimated (*Model Checks-Calculate Mass*)

** Using Offset of 100000 for Top and 200000 for bottom allows MLI nodes to be easily identified in SINDA output using ID < 99999

Decks 1-2

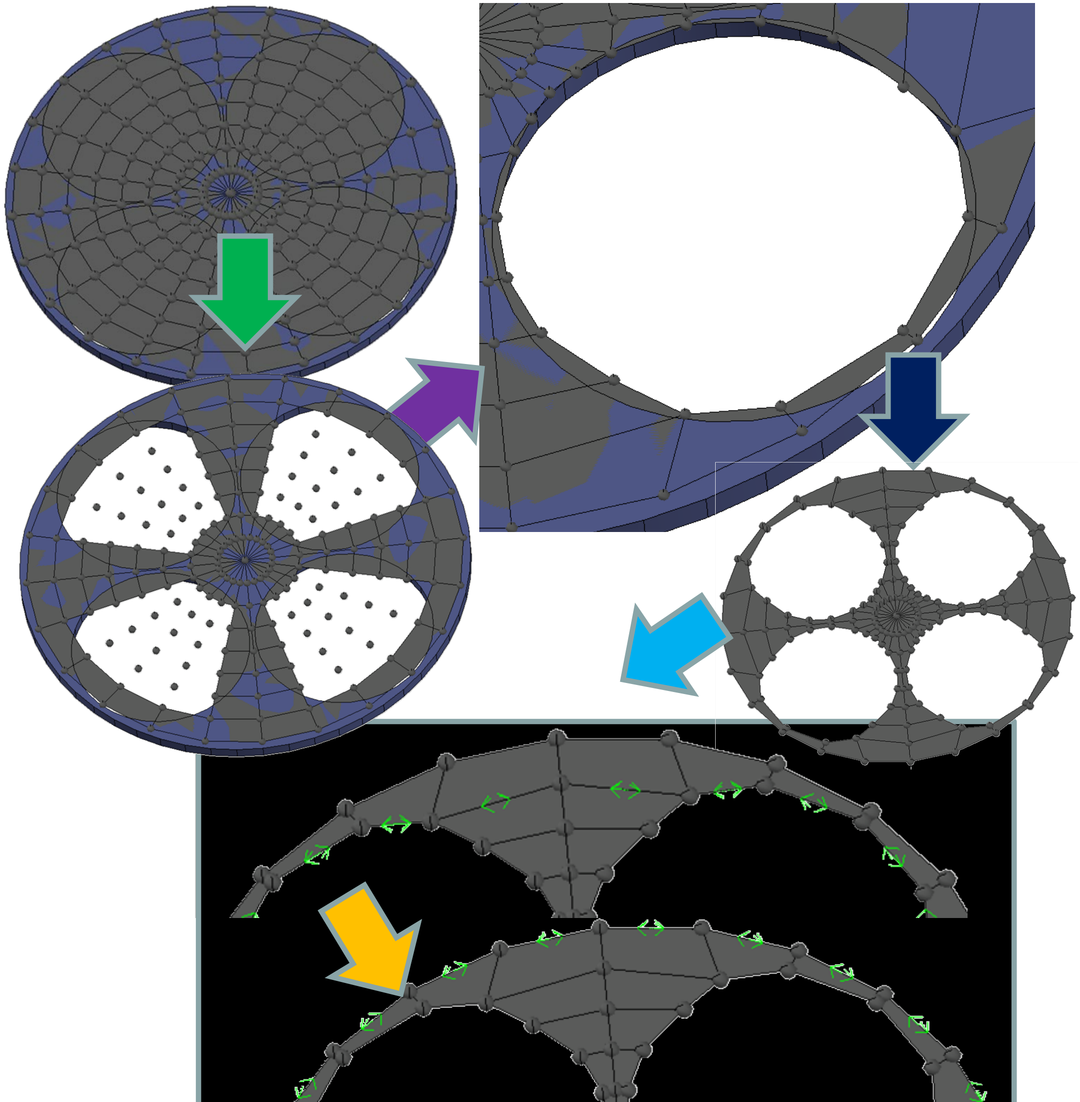


- Make disk for each deck with an Edge node definition, 25 nodes Angular, and 9 Radial. Since the decks will either act
- After turning the visibility of the ESPA on, it can clearly be seen that some of the deck protrude through the ESPA walls. Shrink the radius a bit to ensure the internal deck does not protrude through and view space (R=28.5")
- Now that the disks are not protruding through, they can be contacted to the ESPA walls using a contactor from Deck 1 to all ESPA surfaces
  - Use Edges with an expression value of 24 * 2 / (PI * 30) and Per Area or Length
  - Ensure only Max Radius is used
  - Specify a reasonable tolerance (2")
  - Point algorithm is the only one available for Edge Contactor. Ray Trace generally preferred for Face type contactor
  - Add Description in Comment field
- Click Show Calcs to verify contact made. Repeat for Deck 2 Only. Hint: Use *Modeling Tools-Copy Properties from Master*



Surface Visibility: On Off

Deck 3



- Deck 3 has large cutouts to accommodate. Begin by converting the FD disk to FE
- Delete elements inside cutouts and remove orphaned User Nodes
- Move nodes to reshape elements. Easy to move from Endpoint to Midpoint to shrink elements and ensure nodes stay in plane. Can also move to Nearest if edge exists to snap to. Third option is to ensure the UCS XY aligns with the desired plane and move freely.
- Repeat for each quadrant OR Delete Quadrant elements and use ARRAYCLASSIC (Don't forget to merge nodes)
- Now make an edge contactor from all outer edge elements to the ESPA surfaces selecting the First Edge for all From Surfaces
- Use *Modeling Tools-Shift Connectivity of Planar Element/Rectangle* to select each element until the First edge is on the outer edge to make contact to the ESPA ring. Note that this is often much easier than trying to determine if the First, Second, Third, or Fourth edge should be selected in an edge contactor. Be sure to define the rest of the contactor.

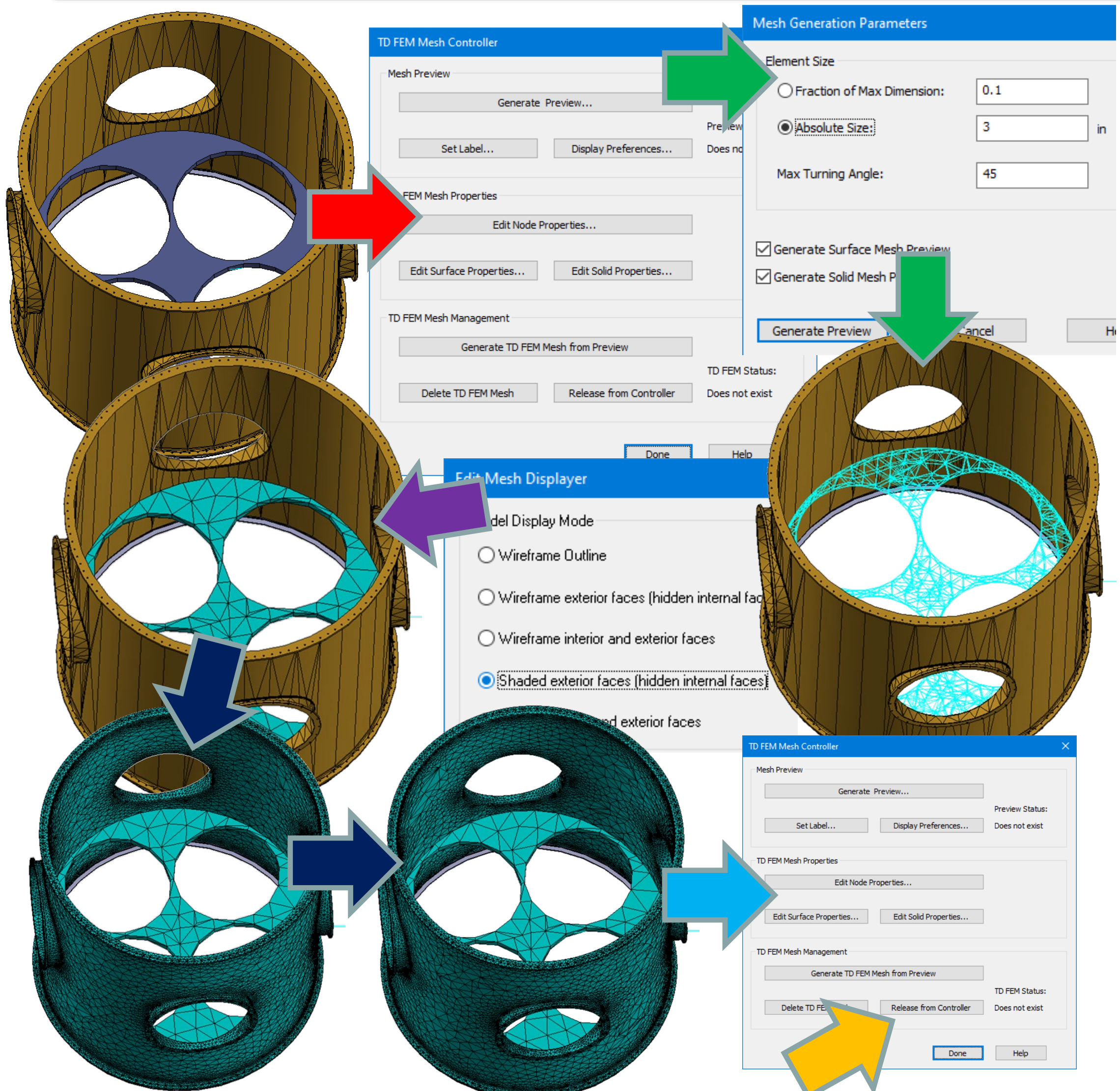
Deck Thermal Property Assignments

- Deck 1:
  - NUMBERING: Submodel: DECK_1
  - NUMBERING: Starting ID: 1001
  - RADIATION: Active: Both
  - RADIATION: Top/Out: SC_WhitePaint
  - RADIATION: Bottom/In: SC_BlackAnodize
  - COND/CAP: Cond Submodel: DECK_1
  - COND/CAP: Material: SC_Aluminum
  - COND/CAP: Thickness: 0.040*2 (as expression with comment Honeycomb, 2x 40 mil facesheet)
- Deck 2:
  - NUMBERING: Submodel: DECK_2
  - NUMBERING: Starting ID: 1001
  - RADIATION: Active: Both
  - RADIATION: Top/Out: SC_BlackAnodize
  - RADIATION: Bottom/In: SC_BlackAnodize
  - COND/CAP: Cond Submodel: DECK_1
  - COND/CAP: Material: SC_Aluminum
  - COND/CAP: Thickness: 0.040*2 (as expression with comment Honeycomb, 2x 40 mil facesheet)

Note: setting a starting node number of 1001 (or something other than 1) allows quick identification of surfaces whose node assignments have been addressed

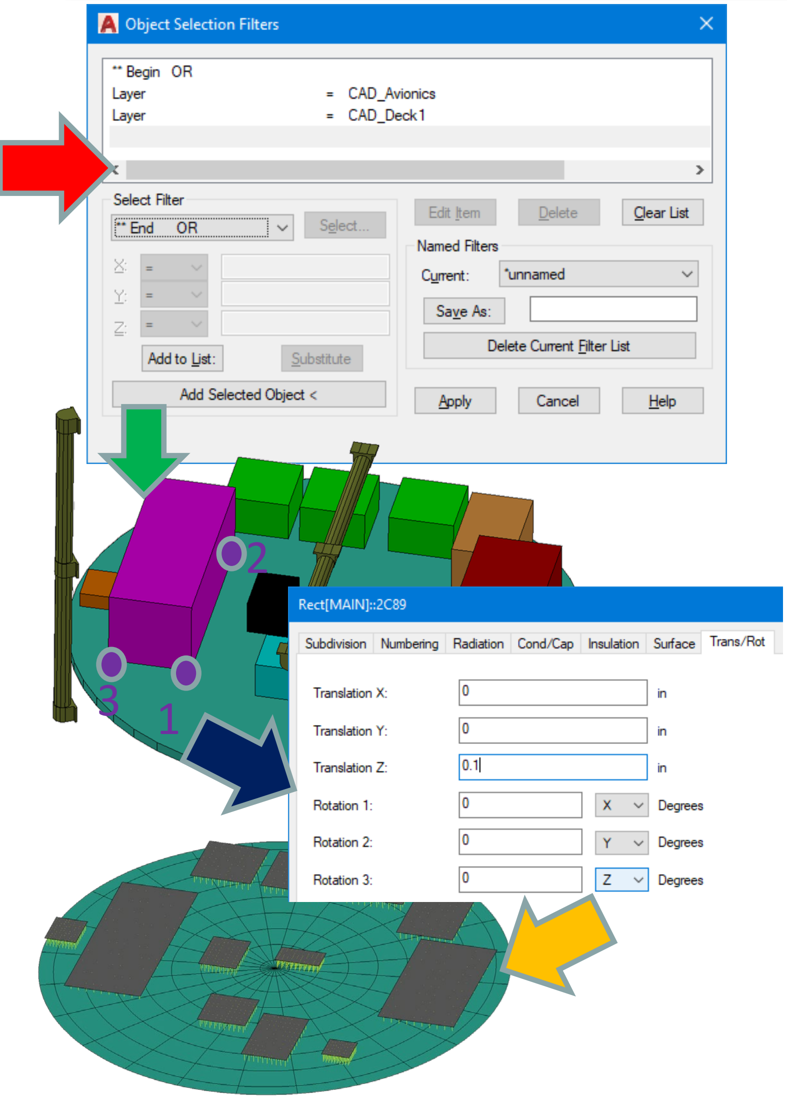
- Deck 3: Since this surface was broken out into elements, the node numbers cannot be assigned via the element definition. Instead, the nodes must be selected and edited. At this point, all of the nodes should be in the MAIN submodel, making it relatively easy to select them from the model browser for modification. Set the submodel to DECK_3
- Deck 3: edit all the elements associated with the DECK_3 nodes and set as follows:
  - RADIATION: Active: Both
  - RADIATION: Top/Out: SC_BlackAnodize
  - RADIATION: Bottom/In: SC_BlackAnodize
  - COND/CAP: Cond Submodel: DECK_3
  - COND/CAP: Material: SC_Aluminum
  - COND/CAP: Thickness: 0.040*2 (as expression with comment Honeycomb, 2x 40 mil facesheet)
  - INSULATION: Put on Bot/In Side (checked)
  - INSULATION: Material: MLI_05
  - INSULATION: Thickness: 1 m (as expression)
  - INSUALTION: ID Offset: 100000
- Now need to number the nodes for DECK_3. Select *Modeling Tools-Resequence ID's*.
  - Starting node number: 1001
  - Increment: 1

Quick detour into Mesh Controllers



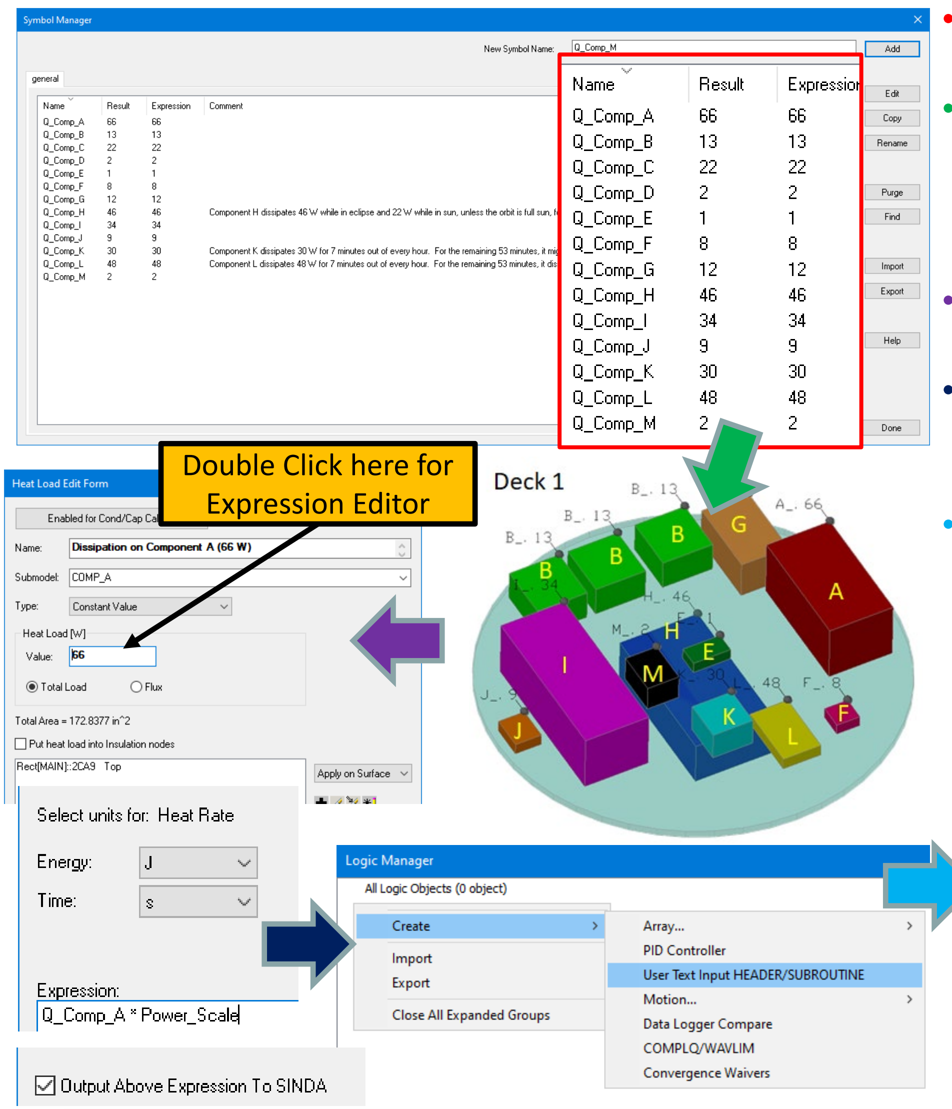
- Mesh controllers are offered in Thermal Desktop, but generally, the CAD needs to be very well conditioned to be able to use them effectively. Turn on only the CAD on the ESPA and Deck_3 layers, but turn off the visibility for the instrument brackets
- Begin with creating a Mesh Controller for Deck_3 (*tdMesh*). Select the blue part and click *Generate Preview.* The options for mesh control are fairly limited and based on a characteristic length. Enter 3 inches and click *Generate Preview*.
- The Wireframe view can be difficult to interpret. Select *Display Preferences* and select Shaded Exterior Faces. Change the Absolute size to 5 and note differences in the generated mesh.
- Go through the same exercise with the ESPA using both 3 and 5 and note how much denser this mesh is. With all the holes, chamfers and fillets, this part is not suitable for a reasonable thermal mesh.
- If a mesh is good enough, the preview can be assigned to the Mesh Controller after defining the properties.
- If the mesh will no longer change, it can also be released from the controller. For this case, the mesh controllers should be deleted

Components - Deck 1

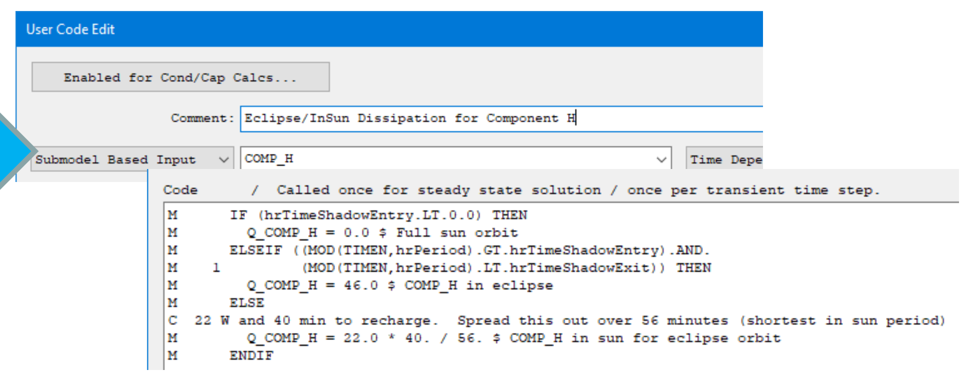


- Begin by turning all objects visibility off (rcVisOff All). Then turn on layer visibility only for only 0, CAD_Deck1, and CAD_Avionics. Lastly, turn visibility on for only objects on the CAD layers of interest (rcVisOn 'filter): Note the ' invokes the filter command transparently (i.e. within another active command)
- Within the Filter command, first add a Begin OR to the list, then add Layer=CAD_Deck1, next add Layer=CAD_Avioncs, lastly add an End OR. Select Apply and type in All at the Select Objects: and hit return again.
- Only the Boxes and Deck1 should now be visible. Turn the visibility on for the DECK1 submodel through the ModelBrowser. Type in rcRectangle and Click points 1,2, and 3 shown to left. This makes a rectangle for the base of Component I, with the bottom side facing the deck.
- On the Trans/Rot tab, update the Translation Z to 0.1 in. This will "move" the origin of the rectangle up by 0.1 in along the surface Z axis. This ensures a small gap will exist between two surfaces that may be coplanar in the CAD.
- Make a contactor between the rectangle baseplate (Bottom) and the deck disk (Component I to Deck, Faces, Submodel: DECK_1, Per Area or Length, 0.8 W/in^2 K, Ray Trace, Uncheck Apply Thickness, Tolerance:1). Show Calcs. If no contact is made, increase Tol=2.
- Repeat for all other components. Could opt to include all Components in single From field in Contactor to Deck. Trade of many contactors/flexibility to update each vs. single location

Components - Deck 1 Heat Loads (Part 1)



- Add Symbols for each Component Dissipation (rcEditSymbol) as shown to left
- Add a heatload to the surface representing Component A (rcHeatload). Set the following values:
  - DESC: Dissipation for Component A (66 W)
  - SUBMODEL: COMP_A
  - VALUE: Q_Comp_A * Power_Scale and output as expression in Editor
- Repeat for Components B,E-G,I-M assigning appropriate names for each HeatLoad
- Using the LogicManager (rcLogic), create a new Logic Object by Right clicking-Create-User Text Input HEADER/SUBROUTINE.
- Define this as Submodel Based Input, Variables 0 and enter the Comment and Logics as shown below. Recall the orbital symbols were output as SINDA registers and can now be referenced in the logic code. Note that the values will change with the orbit used in a CaseSet



User Code Edit

Enabled for Cond/Cap Calcs...

Comment: Component K and L dissipation profiles

Submodel Based Input | COMP_K | Time Depend

Code / Called once for steady state solution / once per trans:

```
C  Hot Case:
C  7 min: 20 W for Component K,  8 W for Component L
C  7 min: 30 W for Component K, 48 W for Component L
C  46 min: 0 W for Component K,  8 W for Compoennt L
C
C  Cold Case:
C  7 min: 30 W for Component K, 48 W for Component L
C  53 min: 0 W for Compoennt K,  8 W for Component L
C
C  Surv Case:
C  60 min: 0 W for Component K,  8 W for Compoennt L
C
C Establish Steady State time averaged values
M      IF (NSOL.LE.1) THEN
M        IF(Hot_CASE.EQ.0.0) THEN $ Cold Case
M          Q_COMP_K = (30.0 * 7.0) / 60.0
M          Q_COMP_L = (8.0 * 7.0 + 48.0 * 7.0 + 8.0 * 46.0) / 60.0
M        ELSEIF(Hot_Case.GT.0.0) THEN $ Hot Case
M          Q_COMP_K = (20.0 * 7.0 + 30.0 * 7.0) / 60.0
M          Q_COMP_L = (48.0 * 7.0 + 8.0 * 53.0) / 60.0
M        ELSEIF(Hot_Case.LT.0.0) THEN $ Surv Case
M          Q_COMP_K = 0.0
M          Q_COMP_L = 8.0
M        ENDIF
M      ELSEIF (MOD(TIMEN,3600.0).LE.420.0) THEN $ 0-7 minutes
M        IF(Hot_CASE.EQ.0.0) THEN
M          Q_COMP_K = 30.0
M          Q_COMP_L = 48.0
M        ELSEIF(Hot_Case.GT.0.0) THEN
M          Q_COMP_K = 20.0
M          Q_COMP_L = 8.0
M        ELSEIF(Hot_Case.LT.0.0) THEN
M          Q_COMP_K = 0.0
M          Q_COMP_L = 8.0
M        ENDIF
C
M      ELSEIF (MOD(TIMEN,3600.0).LE.840.0) THEN $ 7-14 minutes
M        IF(Hot_Case.EQ.0.0) THEN
M          Q_COMP_K = 0.0
M          Q_COMP_L = 48.0
M        ELSEIF(Hot_Case.GT.0.0) THEN
M          Q_COMP_K = 30.0
M          Q_COMP_L = 48.0
M        ELSEIF(Hot_Case.LT.0.0) THEN
M          Q_COMP_K = 0.0
M          Q_COMP_L = 8.0
M       ENDIF
C
M      ELSE $ 14-60 minutes
M        Q_COMP_KK = 0.0
M        Q_COMP_LL = 8.0
F        CALL NOTFOUND
M      ENDIF
```

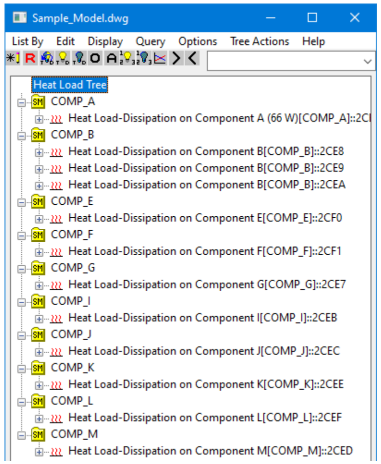
Block 1

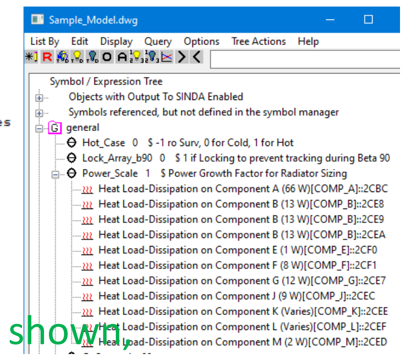
Block 2

Block 3

Block 4

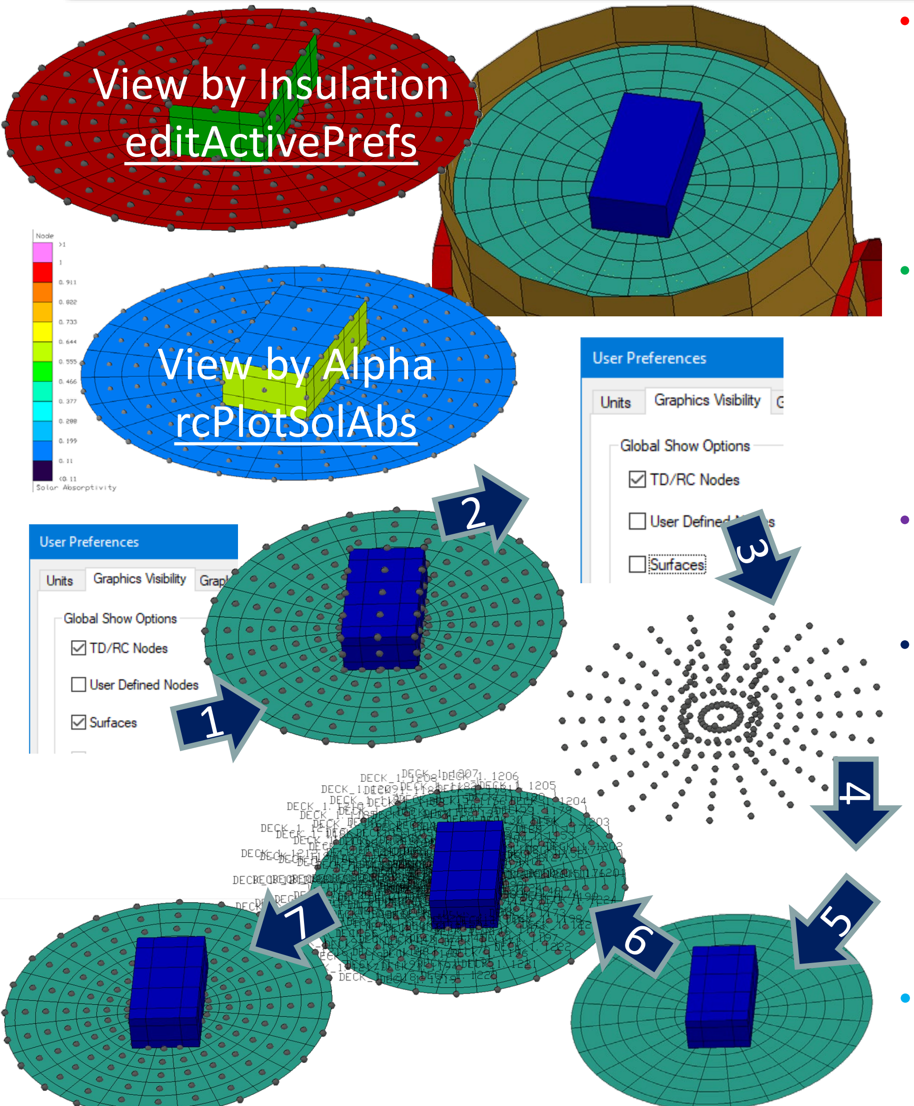
Block 5 – Type as shown, deliberate errors introduced





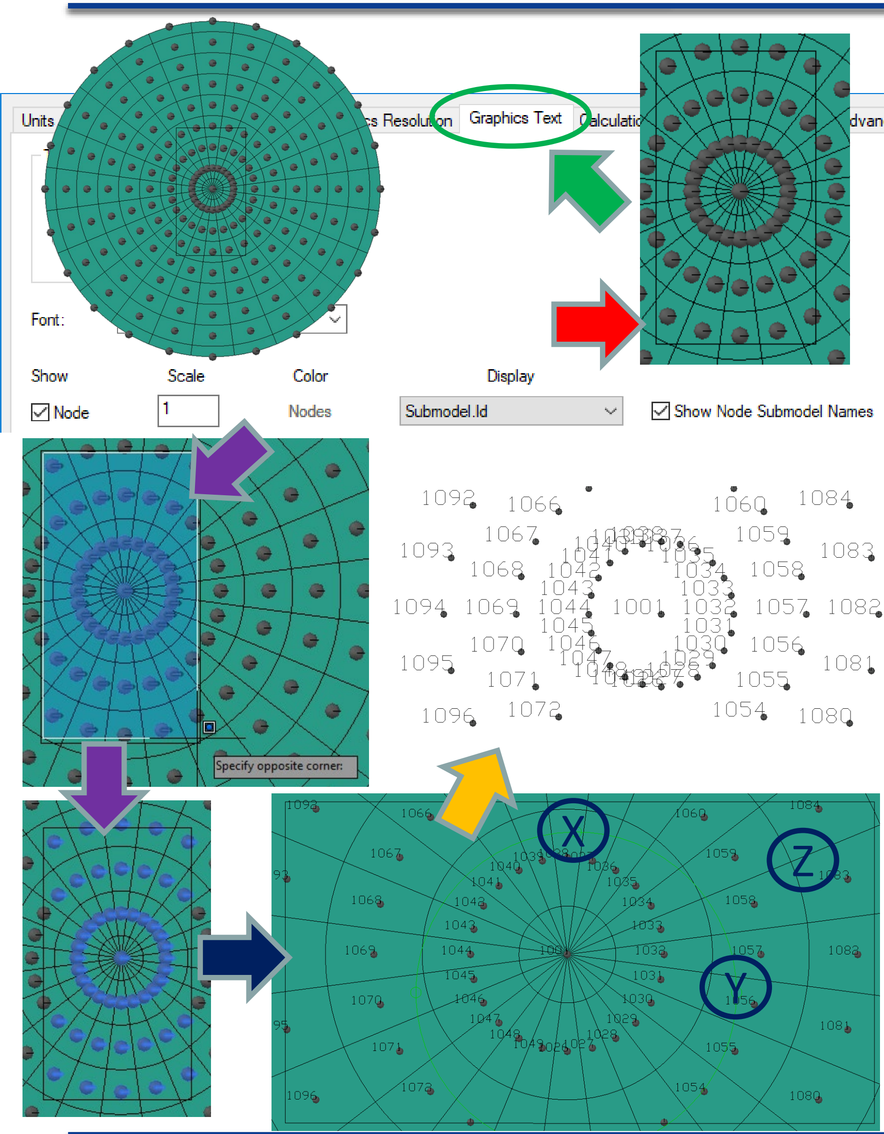
- While Component H's dissipation depended on eclipse, Component K and L have a fixed cyclic behavior (60 min period) independent of orbital position.
- Create another Logic Object as shown to the left. This one controls the dissipation assigned for Components K and L for steady state and transient analyses
  - First block is preceded by "C" and are comments
  - Second block preceded by "M" checks to see if solution is in steady state (NSOL<=1) or transient (NSOL>1). If steady, then the time averaged values are assigned
  - Third block assigns transient dissipations for 0-7 minutes using the current time in SINDA (TIMEN) and the MOD function
  - Fourth block assigns transient dissipations for 7-14 minutes
  - Fifth block assigns dissipations for 14+ minutes or survival
- The MOD function returns the remainder after dividing argument 1 by argument 2. So, using the current time and the period returns the local time in a cycle. While this may be able to be accomplished using a time dependent heat load, the logic offers greater flexibility.
- Assigning all the Heatloads as expressions allows the values to be updated during the SINDA run and applied as assigned. If not output as expressions, then the resultant (constant) value would have been assigned.
- Use the ModelBrowser (rcModelBrowser) to visualize by Heatload and Symbol Usage to check that heatloads are defined and properly reference Power_Scale

Component H (Part 1)

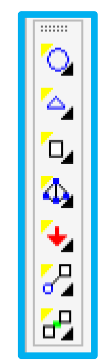


- Component H is a different from the other components mounted to Deck 1, in that it is isolated from the deck and uses its own top panel to radiate to space with the other walls and baseplate being blanketed. For now, assume the isolation is perfect, but it will block the deck's view to space. In this area, it is desirable to put insulation and change the properties.
- Build the 6 rectangles to represent the box. Make it 0.2" thick, Aluminum for all sides. Add MLI to all but the +Y face with SC_BlanketExt facing outward and NoRad facing inwards. Make only outward facing surfaces active. Nodalize it 3x3x5 edge nodes with 5 being along the long edge. Be sure to assign the COMP_H submodel and use a starting ID of 1001.
- To adjust the MLI on DECK_1, the Override features will be used to assign MLI and different properties to specified nodes. But first those nodes need to be identified…
- It is desired to turn on just the nodes associated with the deck. The following steps allows this to be done:
  1. Use rcPreferences to globally display only surfaces and nodes
  2. In ModelBrowser, Display Only DECK_1 and COMP_H
  3. Disable Global Surface visibility
  4. Manually turn off visibility for all nodes shown (rcVisOff)
  5. Enable Global Surface visibility again.
  6. Turn on Node IDs (rcNumbersOn) and select DECK_1 disk
  7. Turn off Node IDs (rcNumbersOff) and select DECK_1 disk
- Turning node numbers on automatically will display nodes as well if the global visibility is on

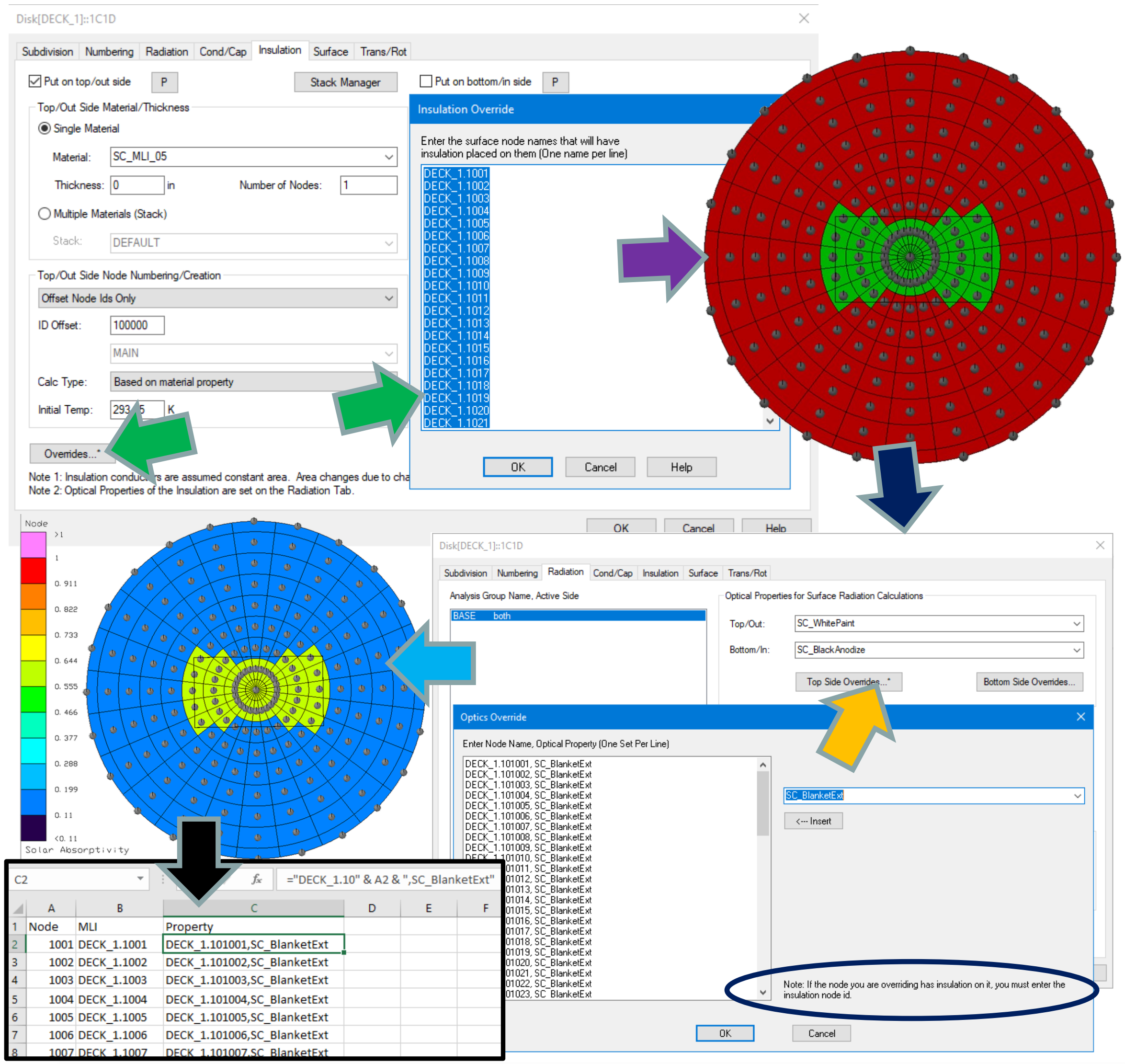
Component H (Part 2)



- Now that only the deck, Component H, and the Deck nodes are visible, it is time to find the nodes to over ride. Type in –VPOINT and enter 0,1,0 to view the model from the +Y axis. Turn off the visibility of the top and bottom panels of COMP_H.
- Any nodes completely enclosed within the rectangular sides should have their node numbers turned on. But including the submodel may get a bit cumbersome. Turn off the submodel display (rcPreferences-*Graphics Text Tab-Show Node Submodel Name-Uncheck*)
- Turn on node numbers (rcNumbersOn) and then type in "w" to indicate a window selection, which will only select objects completely enclosed in the window. For symmetry, manually select the remaining nodes indicated to the left.
- Rotating about the screen axis (3dforbit) makes it easier to see. Clicking and dragging outside the green circle rotates the view about the screen Z axis. Clicking and dragging about the +/-90 locations rotates about the screen X, clicking and dragging about the 0 and 180 locations rotates about the screen Y axis. Clicking inside the large green circle allows free rotation of the model
- Turning off surface visibility (rcPreferences or use the rcToggleSurfaceVis command). There are toggles for Nodes, User Nodes, Surfaces, Solid FE, HeatLoads / Heaters, Conductors and Contactors.
- The range of nodes is now easily identified as 1001-1049, 1054-1060,1066-1072,1080-1084,1092-1096

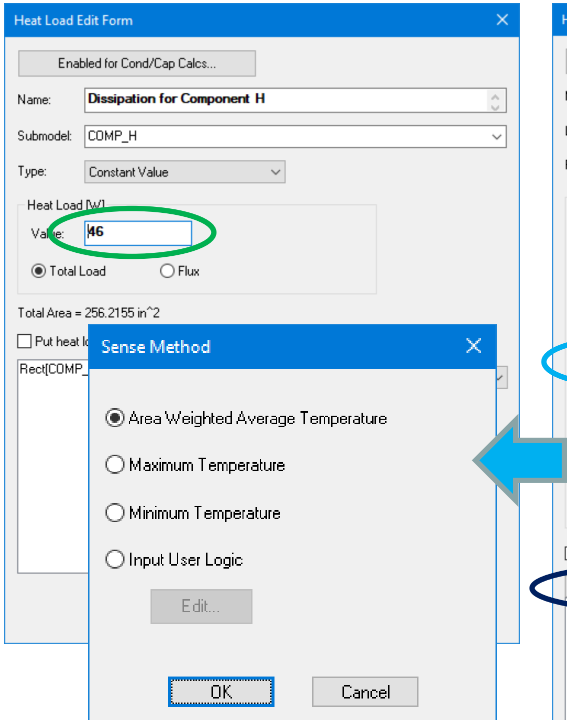


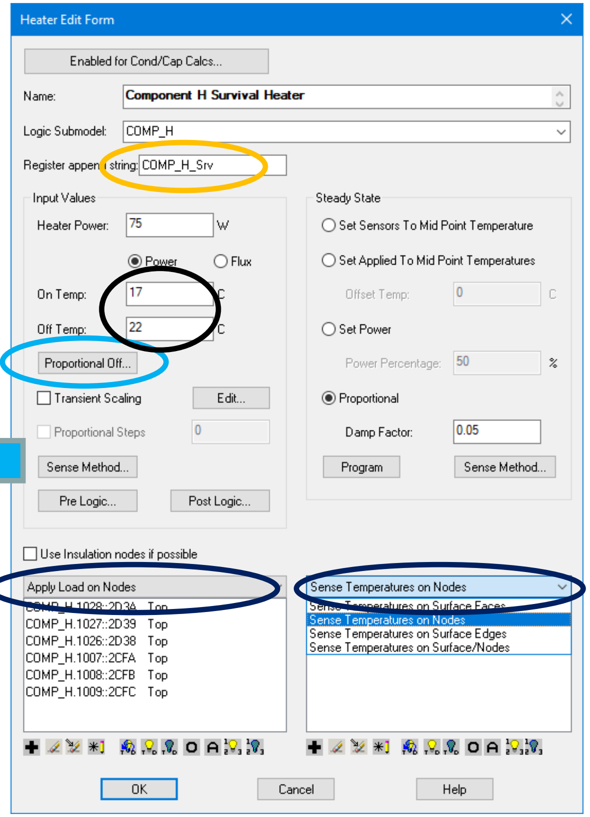
Component H (Part 3)



- Now that only the nodes are known to modify, edit the DECK1 disk and add MLI to Top Side with values as shown to left.
- Click the Overrides button and enter each node number where MLI is to be added including the submodel
- Display MLI assignments (editActivePrefs) and note that only the nodes indicated show MLI (green). All other nodes do not have MLI (red).
- Edit the disk again and move to the Radiation Tab and click the Top Side Overrides button. Add a similar list of Node number with submodel followed by a comma and then the name of the Optical property to use instead. Note that if you are applying the property to insulation, you MUST specify the MLI node number and not the underlying surface node number.
- Visualize by Solar Absorptivity to ensure correct property assignments.
- Note: If overrides are applied, it is usually indicated with a * in the button caption
- Making the lists of these overrides is more easily done in a spreadsheet and then copied and pasted into Thermal Desktop

Component H (Part 4)



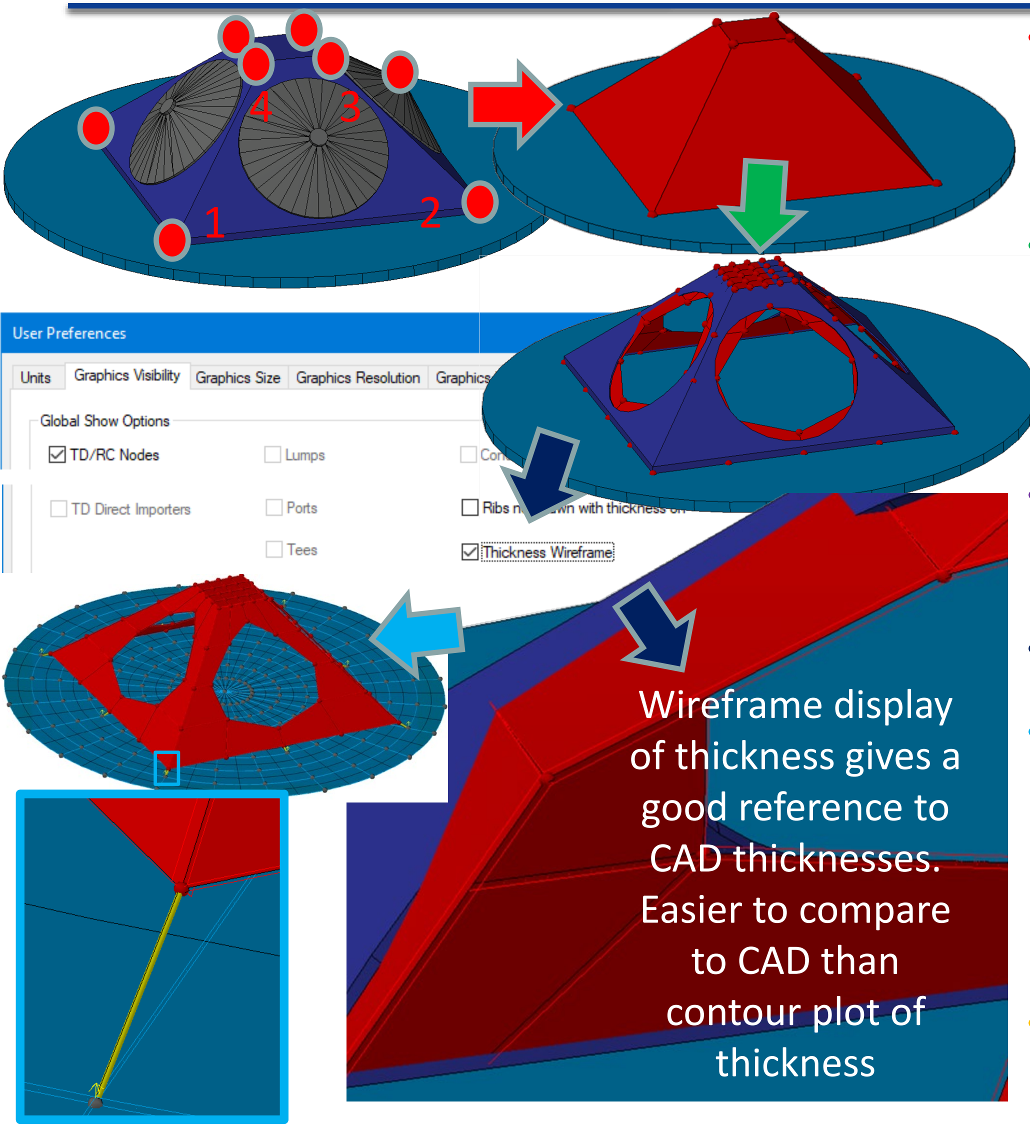


Note: Steady state can be tricky. Set to midpoint utilizes boundary nodes and might cool. Set Power implies knowing a priori the best value to use. Damped Proportional is generally preferred, but can lead to unstable SS performance

Note: using symbols and outputting as expressions for the On and Off points allows for the greatest flexibility. During a run, setpoints can be set very high to fail on or very low to fail off.

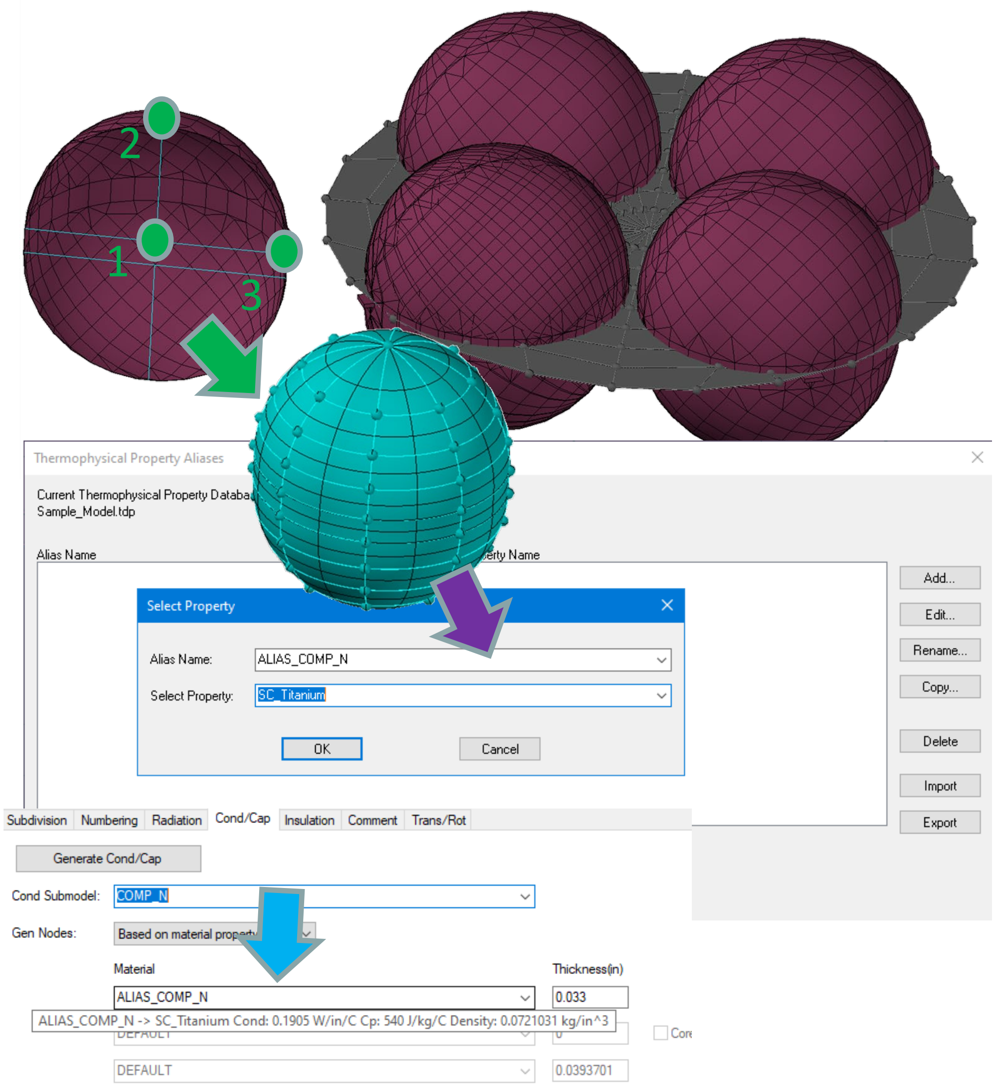
- Now that only DECK_1 is updated, the remainder of Component H can be defined.
- Apply a heatload (rcHeatload) to the baseplate surface. Assign Q_Comp_H and output as expression..
- Next apply a heater (rcHeaterNode) to the 3 central nodes on each box side but assign the sensing node as the central baseplate node. Note that the toolbar button is for rcHeater, which expects surfaces to be selected.
- The heater object can apply either to nodes or surfaces, but not both. Similarly, it can sense from either nodes or surfaces. Selecting the dropdown above the Apply listbox and Sense listbox allows the user to specify this but will clear any existing entries.
- Sense method can be used to specify a sensing other than average. Similarly, proportional control may also be specified
- The register append string should be unique and meaningful. A variety of SINDA registers are created for average power, on time, available power, number of cycles, etc.
- If a flight heater has a fixed power mode, it is easy to turn a heater into a heat load by setting the On and Off points very high (e.g. ON 999, OFF 1000), which forces it always on

Component C/Pyramid - Deck 2



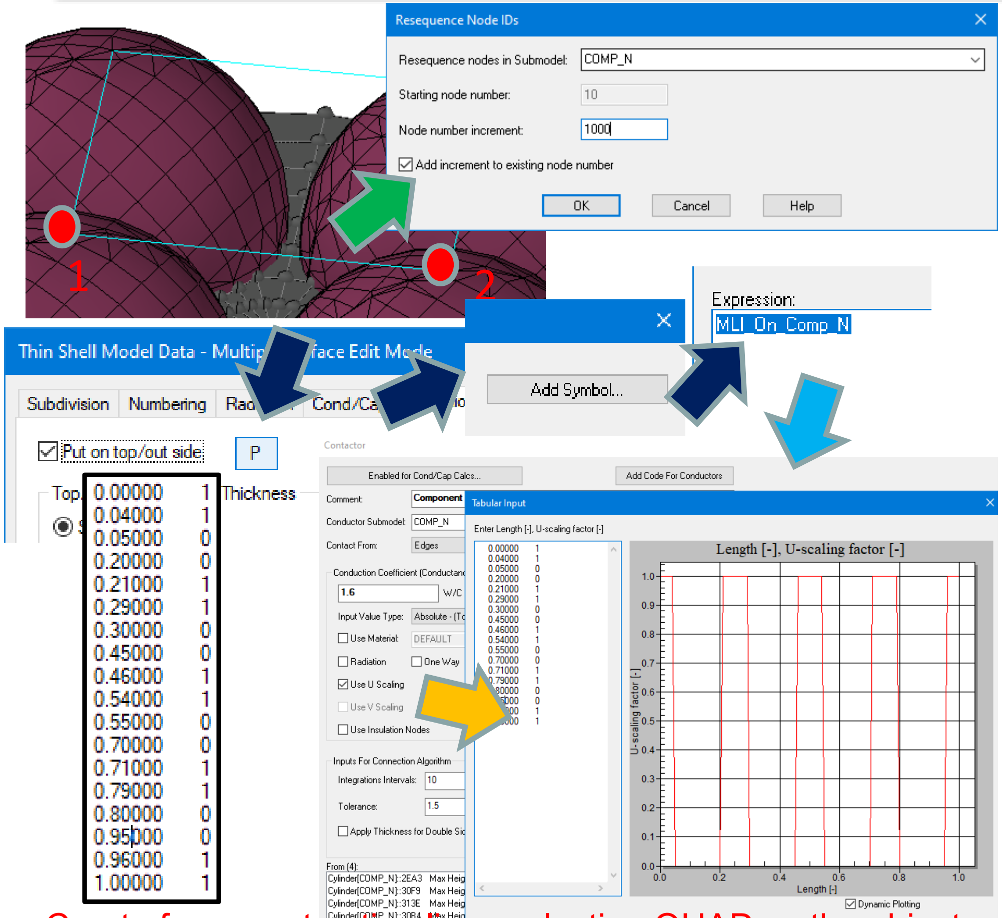
- Begin by displaying only the CAD objects on Deck 2 and Components_C. Add user nodes (rcNode) at each of the vertices as indicated. Create 4 elements with the normal facing out to represent each side of the truncated pyramid and one more for the top.
- Use Refine Elements (rcRefineElements) to subdivide a single quad FE into 4 Fes (or 1 Tri into 4 Tris). Perform this twice to get a 44 mesh on each side. Then delete the center 4 elements from each side as well as the central node. Lastly, move each node on the inside to the midpoint of its corresponding element
- Merge all nodes, renumber starting with 1001, and update the elements:
  - RADIATION: Active: Both, BlackAnodize both sides
  - COND/CAP: Aluminum, 0.1575 in thick, COMP_C_MOUNT
- Check thickness against the CAD. (rcPreferences-Graphics Visibility-Wireframe Thickness
- Add a conductor (rcConductor) form each pyramid corner to the closest node on the Deck_2 disk as well as at the midpoints. This will represent bolt locations from the mount to the deck of 1 W/K per location. Note that it is possible to add additional To nodes after the initial conductor creation. Can also make a node to Surface conductor (rcConductorMultiNode)
- Add 4 heatloads to all the internal circular edges and assign with a value of Q_Comp_C as an expression. (Don't forget about the * Power_Scale)

Component N - Deck 3 (Part 1)



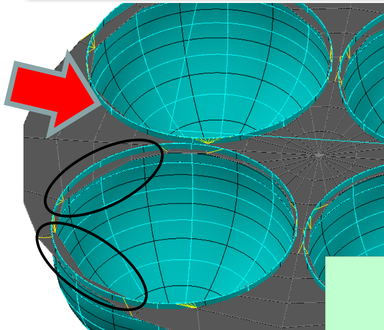
- Begin by displaying only the TD objects for Deck_3 and the CAD objects for Component_N. Copy the bottom left component from 0,0,0 to 0,0,30. When moving something a relative distance, it may be easier to simply enter the points instead of clicking on CAD points. Slice the copy in the YZ plane using the center as a point on the slicing plane. Lastly, create some construction lines as shown.
- Create a sphere (rcSphere) clicking on Points 1,2, and 3 as indicated. Accept the defaults for the rest. Set the Min Height to 0 to make it a hemisphere. Create a second sphere for the bottom and a cylinder for the mid band. Nodalize the spheres 9 angular x 5 height and the midband 9 angular and 2 height. Merge all the nodes and renumber starting with 1001.
- Create an alias for a thermophysical property (editThermoAliasInfo). Aliases are placeholders for properties that can be overridden in a CaseSet when a property is not yet established. Name them beginning with ALIAS_ to identify it as not a standard property.
- Update the three surfaces:
  - RADIATOIN: OUT=BlanketExt, IN=NoRad ACTIVE=Out
  - COND/CAP: Thick=0.033, Material=ALIAS_COMP_N, Submodel=COMP_N
  - INSULATION: Top, MLI_05, OFFSET=100000
- HINT: Hover over the material to display a tool tip with values or alias references

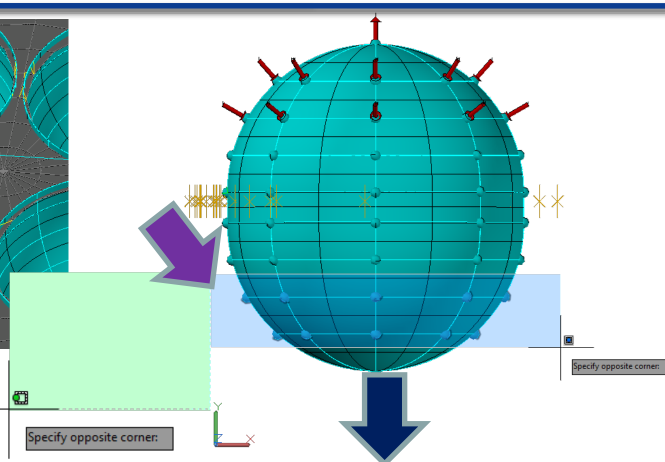
Component N - Deck 3 (Part 2)



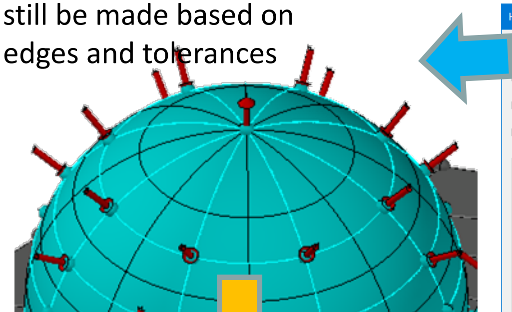
- Create four construction lines selecting QUAD as the object snap point at the top of each dome. Turn off the CAD for Component N and the TD objects for DECK_2. Copy the three Comp N surfaces only from point 1 to point 2. Merge the new nodes if needed.
- Select the three new surface and re-sequence the node IDs (ReseqNodeInfo). Make sure the box is checked to *Add Increment to existing node number* and set it to 1000. Click OK and the second version of Component N will be in the 2000 range. Repeat for the other 2 versions of component N (3xxx and 4xxx). HINT: this could be done by copying both versions and setting the increment to 2000
- Now move all 12 surfaces from 0,0,0 to 0,0,-30 to re-place them back at the original position. It may be easier to build portions of the model some distance off to the side and move it into position when complete.
- Edit all 12 surfaces and go to the Insulation Tab. Click the P button beside the Top/Out side checkbox. Click *Add Symbol* with a new name of MLI_On_Comp_N and a value of 1. Enter the new symbol name in the expression editor. A 1 indicates that this surface should have MLI; whereas, a zero would indicate no MLI. This symbol could be overridden in a CaseSet to explore these two options.
- Add an edge contactor from the 4 central cylinders to the entire Deck 3. Make it an edge contactor using the Max Height for each cylinder, with a tolerance of 0.5 and a magnitude of 1.6 (4 * 4 * 0.1) and absolute contact
- Use the U scaling to only make contact near 0°, 90°, 180°, and 270°

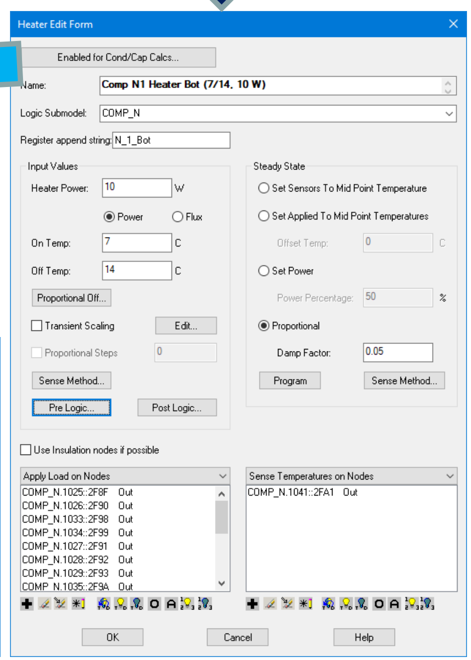
Component N - Deck 3 (Part 3)

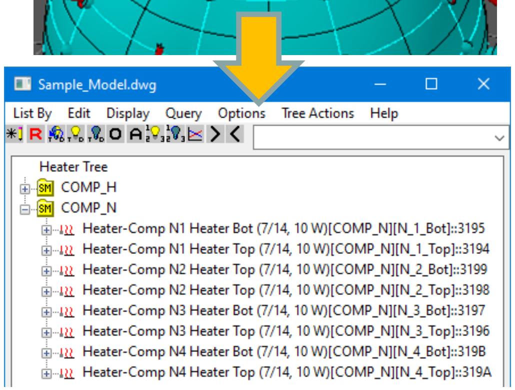




NOTE: Even though the surfaces “cut” through each other, contact will still be made based on edges and tolerances

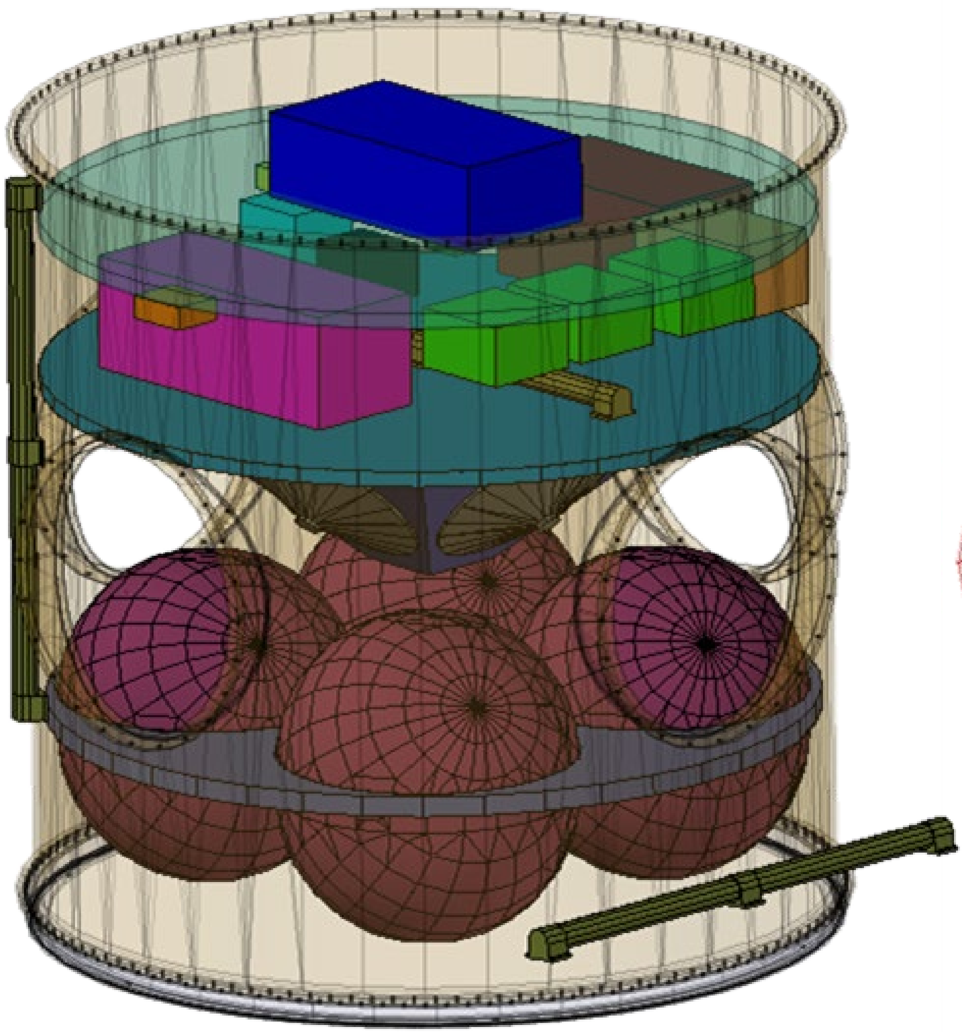


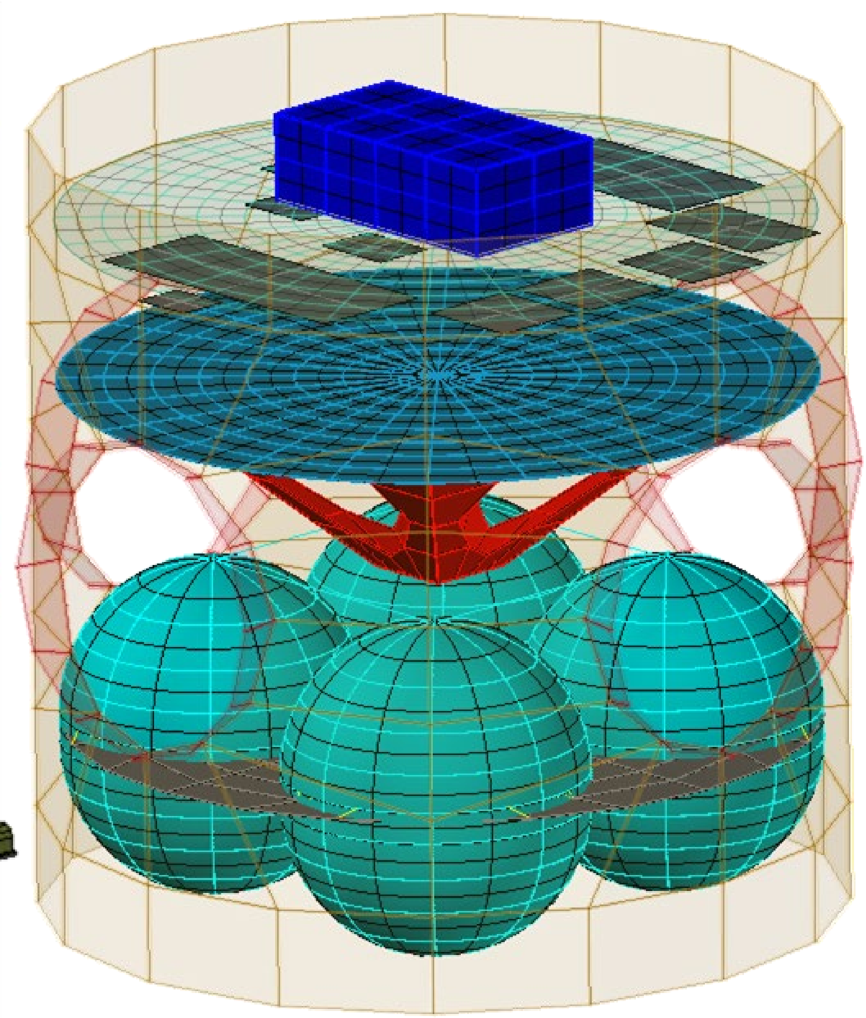




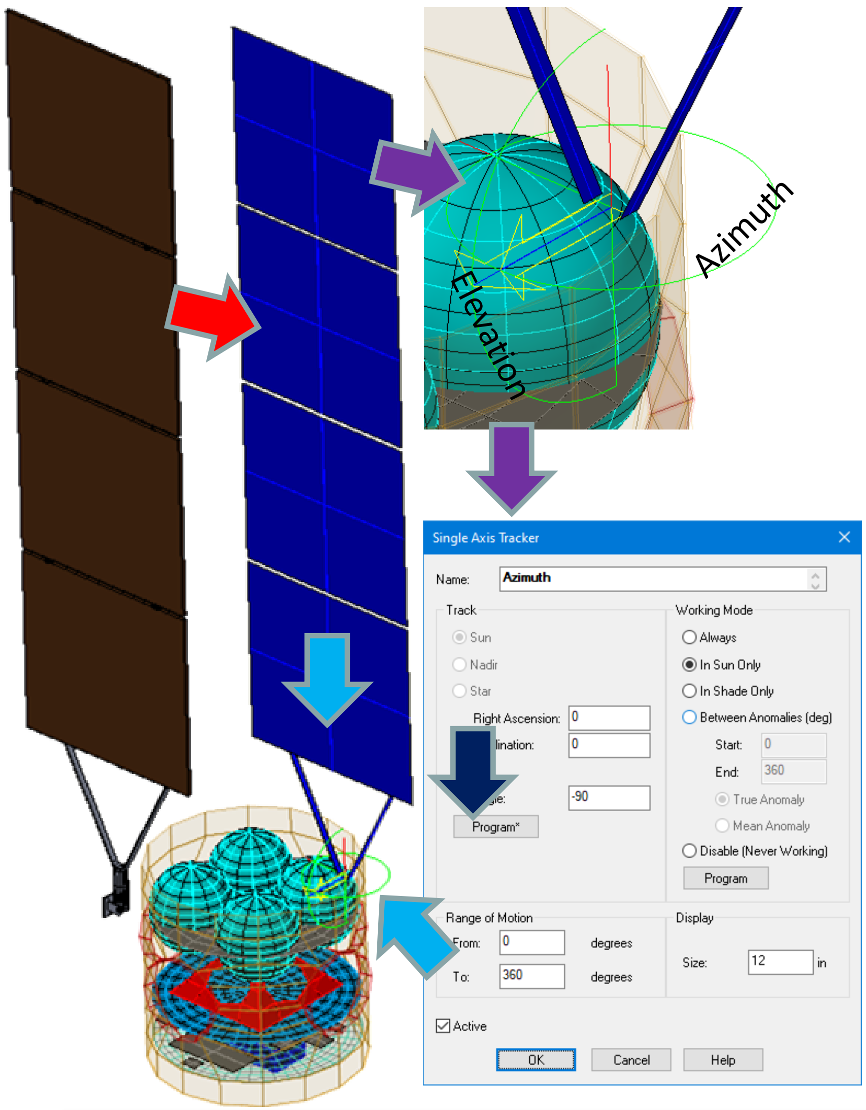
- Showing the contactor markers highlights how connections are made at only the quadrant points on Component N. This “masking” could be used with a box baseplate to only make contact along the midpoints and/or at the corners if desired.
- Turning on only Comp_N nodes 1000-1999 shows a single instance. Setting the viewpoint (-VPOINT) to 0,0,1 allows for easy selection of the bands of nodes to select for the application of a heater.
- Create a new heater on nodes (rcHeaterNode). Clicking the top left location of the blue Window and dragging to the right selects via a Window (where only objects contained completely within the boundary are selected). If dragging instead to the left, it becomes a Crossing, where any objects crossing the boundary are selected). Uses can specify W or C (or even WP: Window Polygon or CP: Crossing Polygon) if desired.
- Select the two latitudes of nodes for Heat Applied and the Pole for Sensing. Apply the values as shown to the left, adjusting for the Comp_N # and Location.
- Note that the applied heat locations are indicated with arrows pointing in towards the surface, whereas the sensing location is indicated by an arrow pointing away from the surface
- Listing by Heater in the ModelBrowser shows enough information to identify each unique heater by the Description or Register String

Model So Far



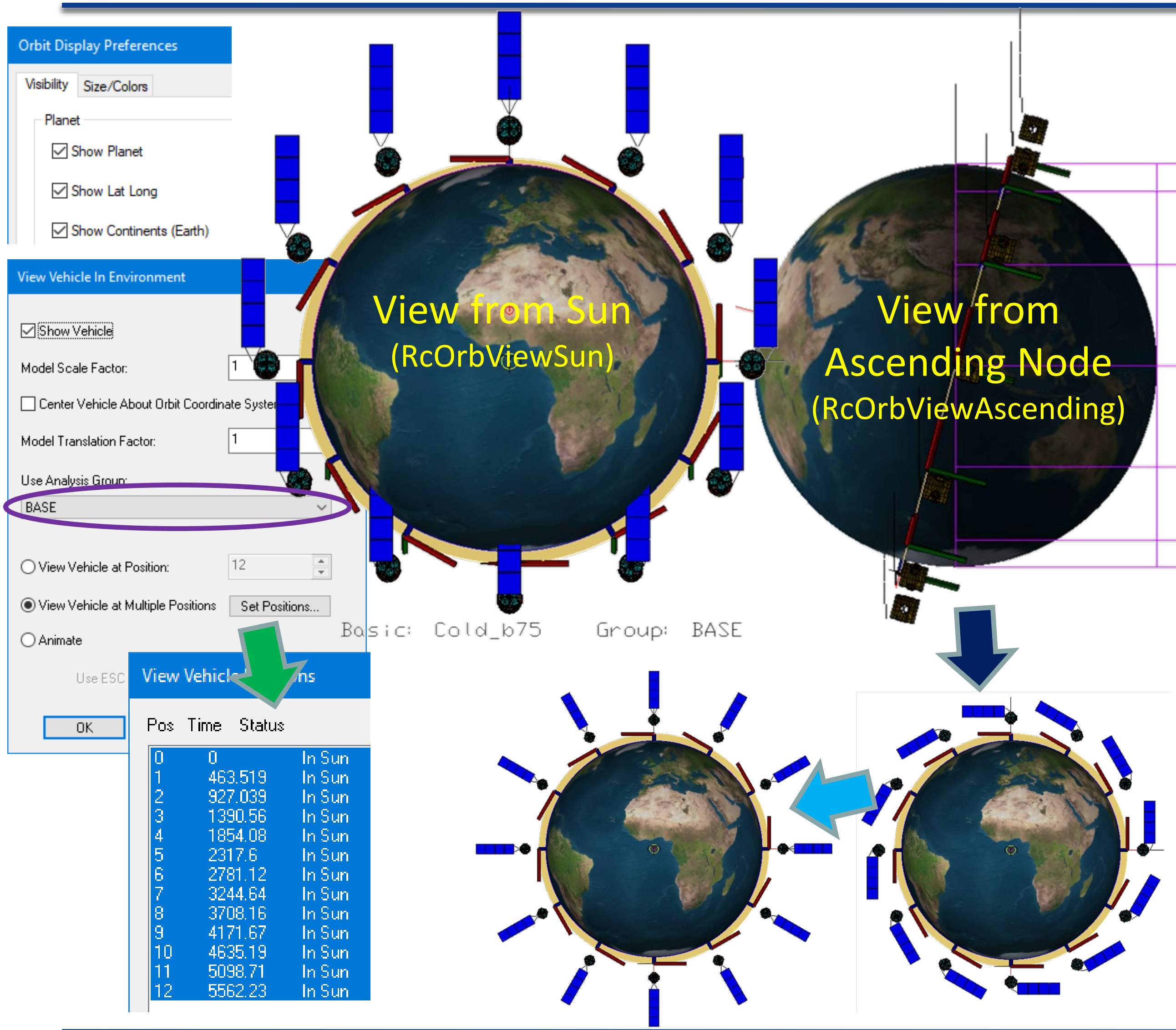


Solar Array



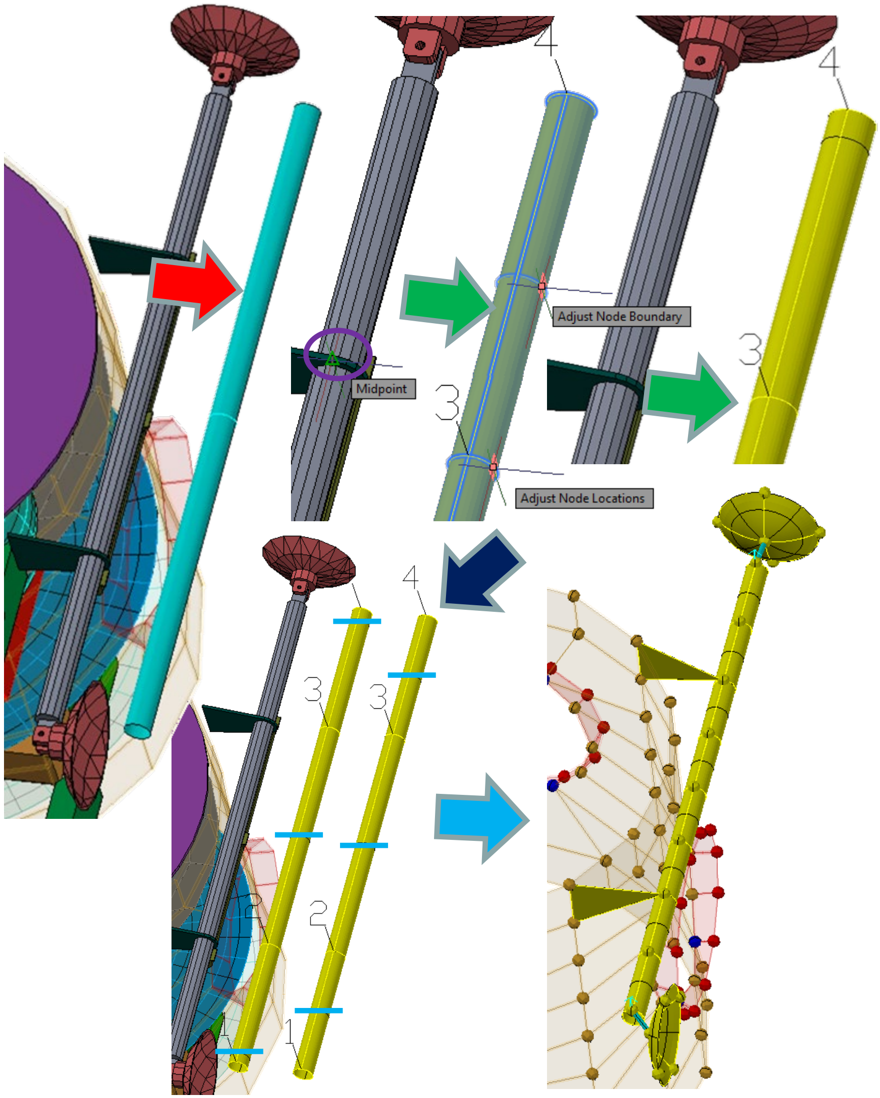
- Create 4 rectangles for the solar panels: Nodes 1001-1004, Submodel: SOLAR_PANELS, Active: BothTop:SC_SolarCells, Bot: SC_WhitePaint,, Material: SC_Aluminum, Thick: 0.08. The yoke surfaces are likely negligible but are visually displayed
- This gimbal assembly has a 2 motor, Azimuth/Elevation scheme to allow the array to track the sun over 0-90° beta angles and over all orbital positions. To model this, Trackers will be used. Trackers as similar to Assemblies, except that a tracker allows rotation about its local Z axis and points to the specified target. In this case, the trackers will be nested, with the Elevation tracker moving with the rotation of the Azimuth tracker.
- Create a tracker (rcTracker) named Azimuth at the center location of the CAD gimbal, oriented as shown (full 360°) and with properties as shown to left (might need to use Rotate3d command). Create a second tracker at the same location named Elevation (180°). Note that the yellow arrow indicates the vector to be pointed to the target. The Elevation tracker should have a range from -90 to +90.
- For the Azimuth tracker, click the Track Program button and enter "(Lock_Array_b90 == 0) ? 0 : 3". This format is essentially a (BoolTest) ? TrueValue : FalseValue and can be uses as an IF type statement. In this case, the tracker is locked at -90 when Lock_Array_b90 is not zero. Set Elevation to always track Sun.
- To have any effect, Trackers (and Assemblies) need to have objects attached. Attach the Elevation tracker to the Azimuth tracker (*rcArticAttach*). The attach the panels to the Elevation.

Solar Array – Visualizing the Orbit



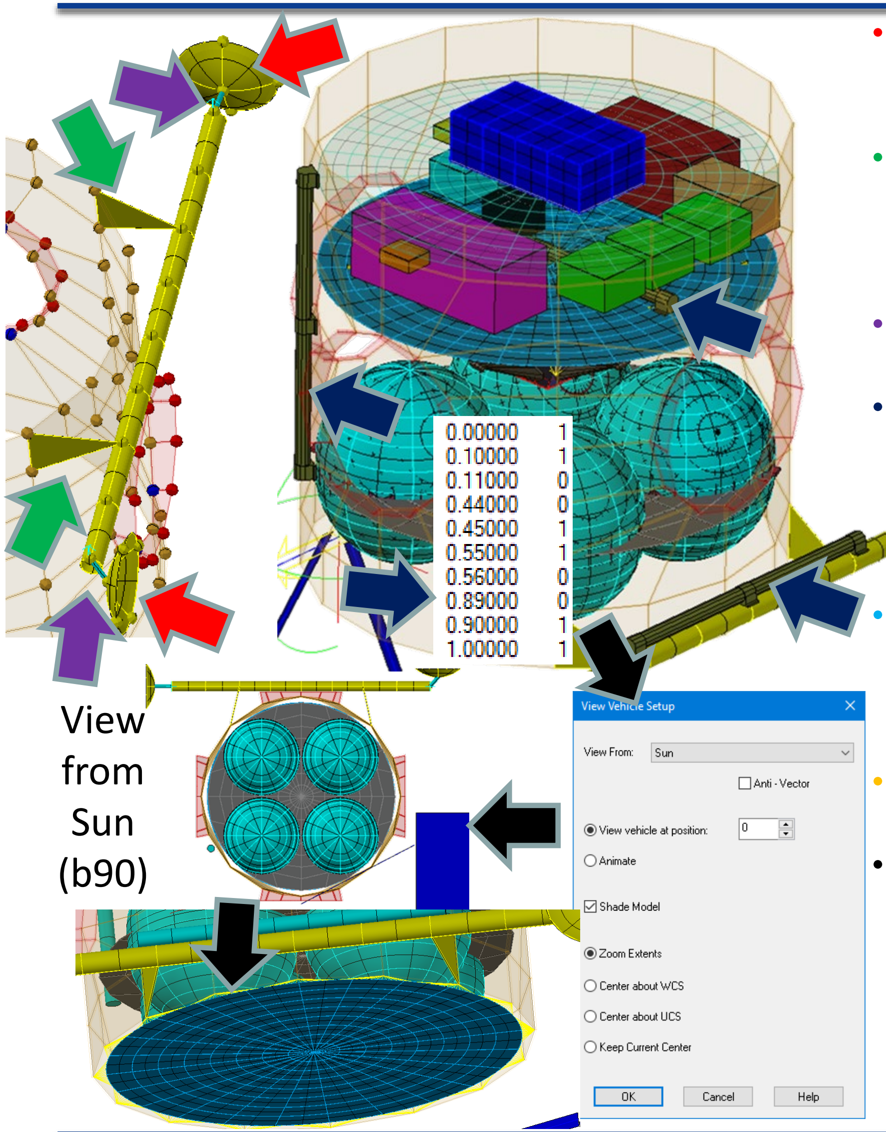
- Set the Cold b75 orbit as current (rcManageOrbits) and click Display Orbit.
- The display options might need to be updated (editViewVehicle). Do not center about Orbit CS, ensure Show Vehicle is checked, and View at all orbit positions
- NOTE: In older versions of Thermal Desktop, the active Radiation Analysis Group determined what was shown in orbit display. Newer versions allow the user to specify which Group to show
- Set the Current Orbit to Cold_b90 and visualize viewing from the sun. Note that the Azimuth tracker can meet the pointing criteria at any angle, but the default is likely not how it would fly.
- Edit the symbols and change the Lock_Array_b90 to 1. Visualize the orbit from the sun again and note how the array orientation is more in line with how it would fly. Change the Lock_Array_b90 back to 0.

HGA Boom



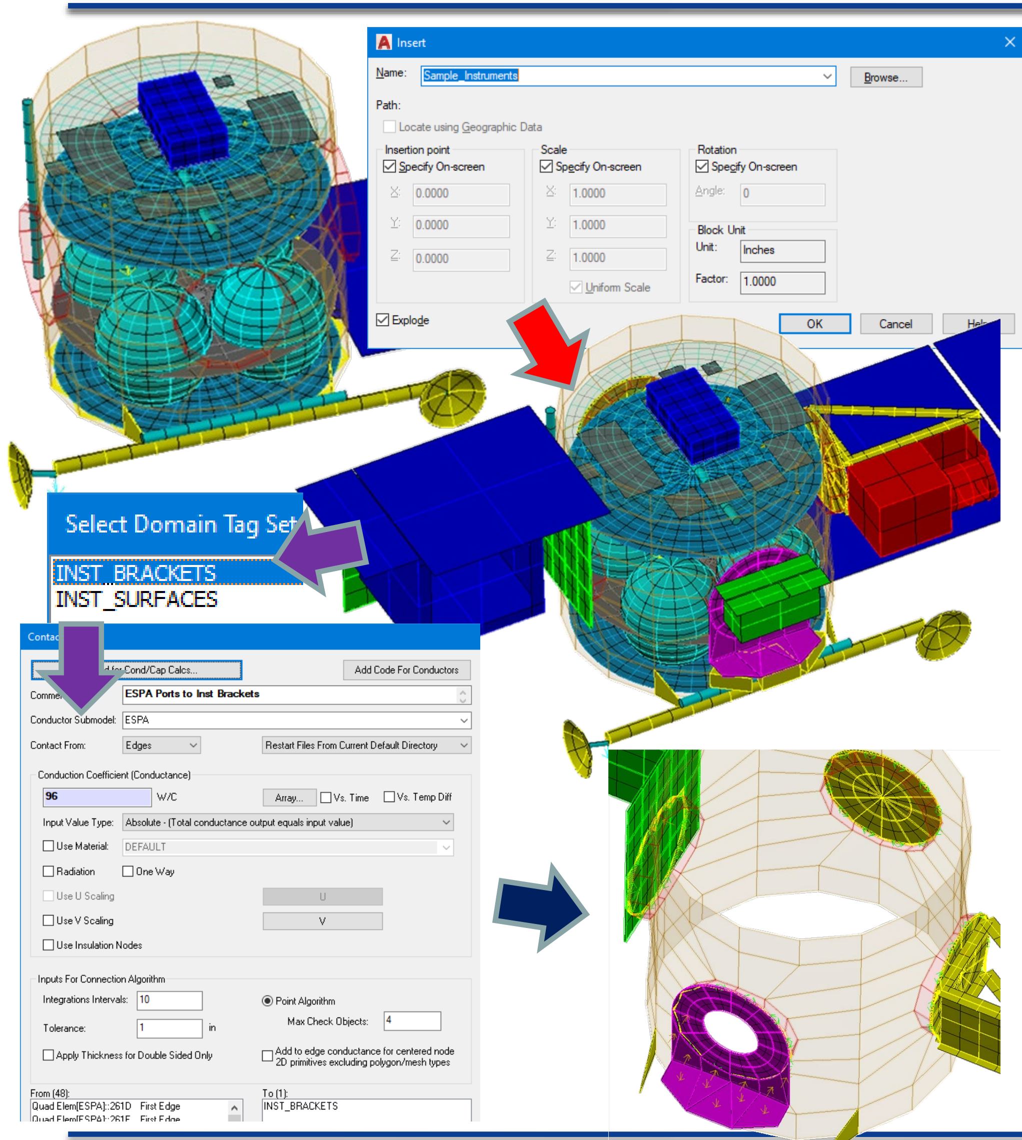
- Create a cylinder representing the boom over the entire length. Nodalize it 1 angular x 4 height using edge nodes Move this cylinder 10" in Z and display node numbers.
- Type in rcGripsBoundary and select the cylinder. Note that the grips now show nodal locations and boundaries. These can be dragged and moved to align with selected points. Click the node location at 3 and drag to indicated midpoint. It now aligns with the bracket, but the nodal boundary has also shifted. NOTE: sometimes it is not possible to specify nodal boundaries that position the nodes where desired.
- Typing rcGripsKeyPoint and selecting the cylinder alters the number and meaning of the grips once more, with the 3 grips indicating the Point 1, 2, 3, etc locations when the object was first created. Type rcGripsParameter and select the object to return to the default grip behavior.
- Even if the nodes can be placed where desired, the nodal sub-areas may not be desirable. Since node locations are always at the midpoint between boundaries, the associated areas may not be optimum. Making separate surfaces allows for better control over the surface sub-areas with varying lengths for the different surfaces. Define the three cylinders end-to-end and merge nodes and delete the original cylinder.
- Nodalize the ends 1x3 and the center 1x5. Active:Out, Out:BlanketExt, In: NoRad, Material::Aluminum, Thick: 0.075, Submodel:BOOM, MLI: Outside, MLI_05, Offset 100000. Also create the end paraboloids, triangular bracket elements and conductors between boom end and dishes.

HGA Boom, Component D, and Closeout



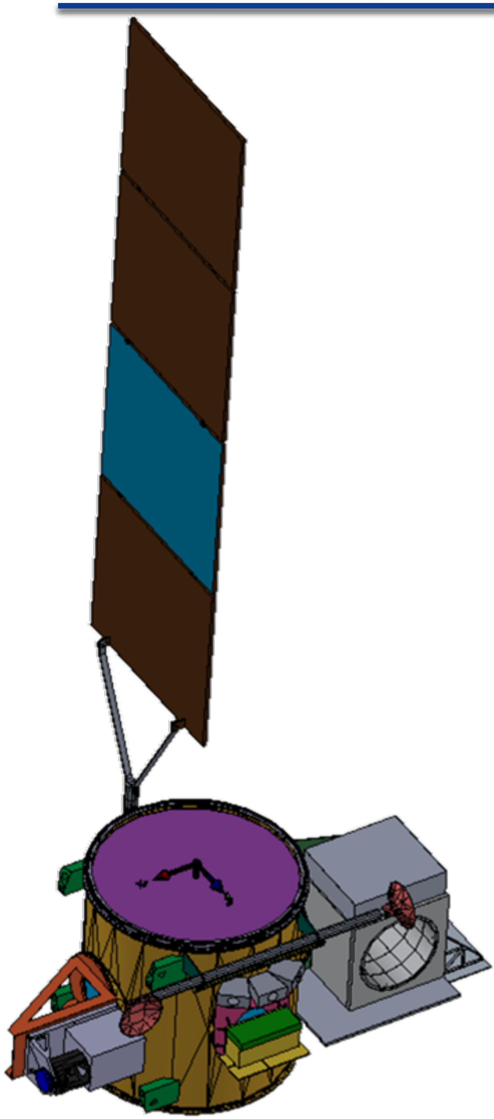
- Dishes: Edge Nodes: 5 angular, 2 height, Nodes:2001+ and 3001+, Active Both, Top/Bot: WhitePaint, Submodel: BOOM, Material:Aluminum, Thick: 0.1
- Brackets: Tri element using 2 ESPA nodes and 1 BOOM node, Active:Both, Top/Bot: BlanketExt, Submodel: BOOM, Material: Aluminum, Thick: 0.2, Insulation: Top/Bot, MLI_05, TopOffset 100000, BotOffset:200000
- Dish Conductors: From Center of Paraboloid to end of Boom. Desc: Dish to Boom End, Submodel: BOOM, Value 0.02 W/K
- Add 3 solid cylinders to represent Component D (3 instances) aligned with X,Y, and Z axes.. Submodel: COMP_D, 1 angular x 1 radial x 9 height; Nodes: 1000+, 2000+, 3000+, Active: HMIN, HMAX, RMAX, Opticals:BlanketExt, Material: Aluminum, MLI on HMIN,HMAX,RMAX, MLI_05, Offset:100000.
- Make a Contactor from all three solid cylinders to the ESPA surfaces, the BOOM surfaces, and the DECK_2 surfaces. Set this as an Absolute Conductance (Total=3*3*2) from the RMAX face with a V scaling as shown to the left. (Show Calcs !)
- Add a single HeatLoad (rcSolidHeatLoad) to all 3 cylinders with a value of Q_Comp_D * Power_Scale and output as expression
- Based on a view from the sun (editOrbitViewFrom), a closeout panel is needed to keep Component N from getting too hot, especially in a Beta 90 orbit. Add a closeout disk below Comp. N and couple it with an edge contactor to the ESPA. Submodel: DECK_CO, Node:1001+, Active: Both, BlackAnodize facing In, SilverTeflon facing out, Material: Aluminum, Thick: 0.04, MLI on outward side, Offset: 100000

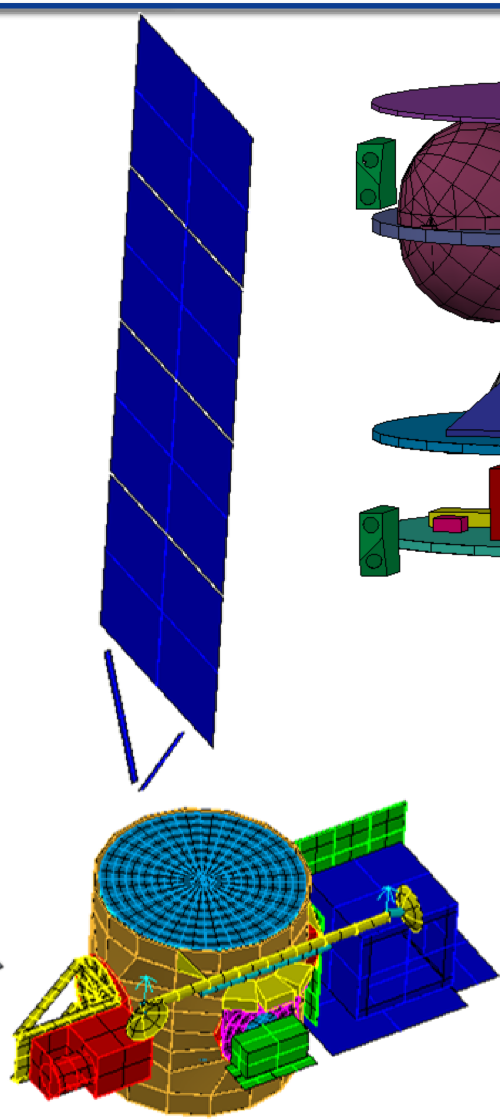
Instruments

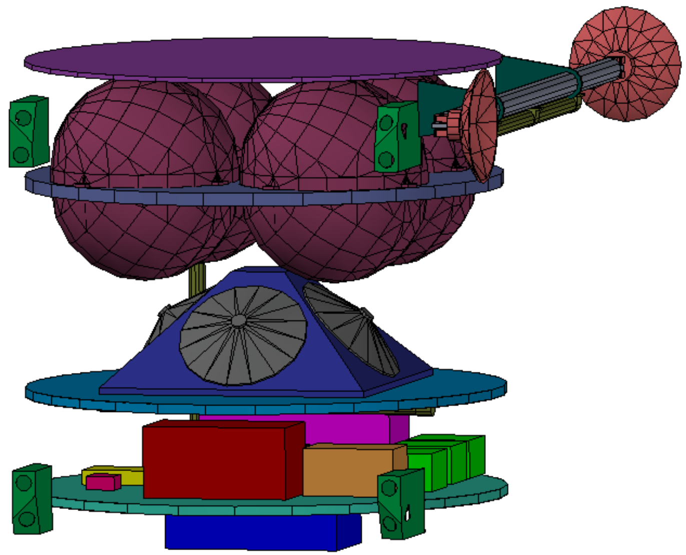


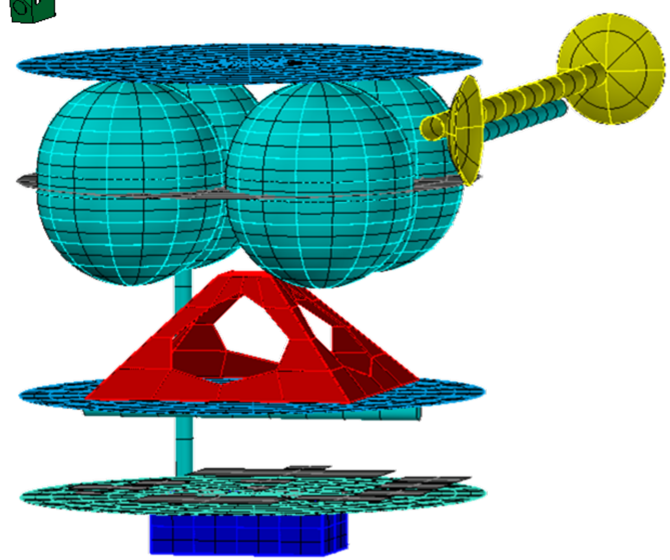
- Often, subsystem models are delivered and require integration with a higher assembly model. The preferred method is to use the AutoCAD INSERT (CLASSICINSERT) command rather than to copy and paste. The reason for this is to preserve Domain Tag Sets, which have no object representation to be pasted. Type in CLASSICINSERT and select the Sample_Instruments.dwg file. Check all the Specify On Screen boxes and Explode and Insert to 0,0,0.
- Note that this model came with Domain Tag Sets, which includes the surfaces that will be bolted to the ports. Domain Tag Sets are similar to AutoCAD groups, but are preserved during model insertion.
- Create an edge contactor From the port edge wall surfaces and when selecting the To, select "D" to list the DomainTag Sets and select INST_BRACKETS. Adjust all of the From surfaces using Shift Connectivity to define the proper first edge.
- Assign a value of 12 * 4 * 2 (12 bolts over 4 ports at 2 W/K/bolt). Show the calculations to make sure the contactor is correct.
- Lastly, the Instrument optical and material properties need to be imported. Edit the Optical Props and click the Import button and select Sample_Instruments.rco and select all properties. Do the same for the Materials with the tdp file. Note that LogicObjects and Symbols may also need to be imported

(Nearly) Completed Model

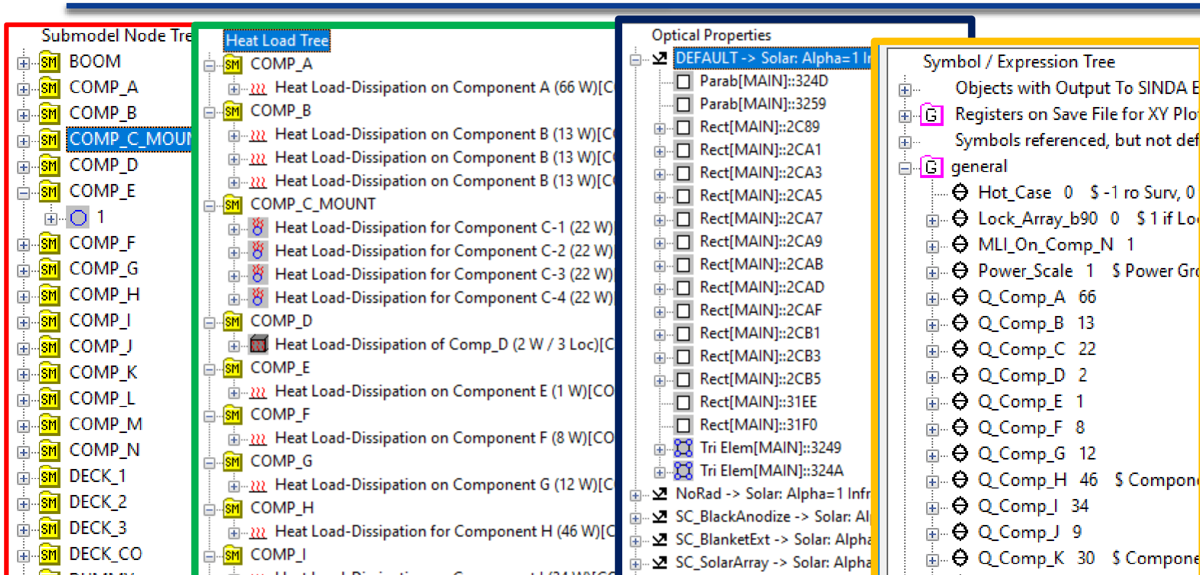


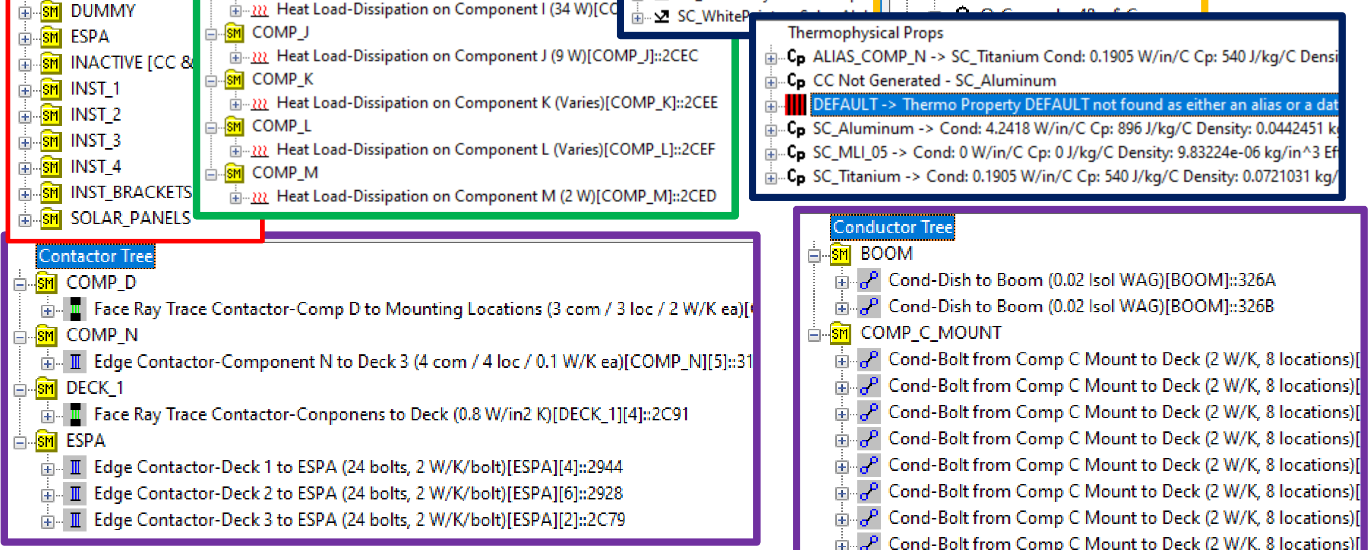




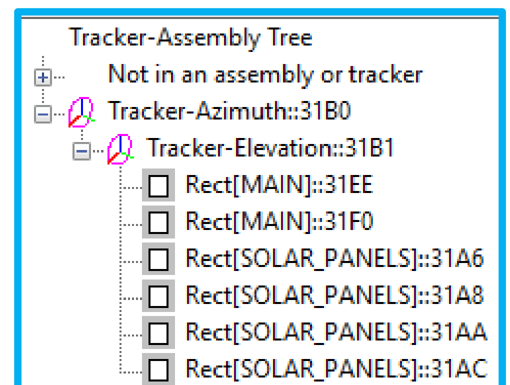


Model Checks

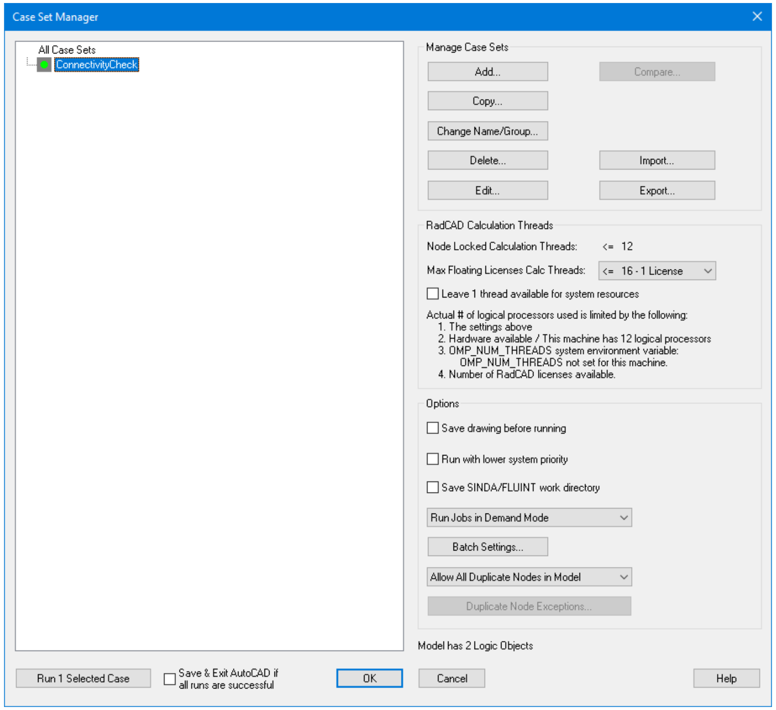


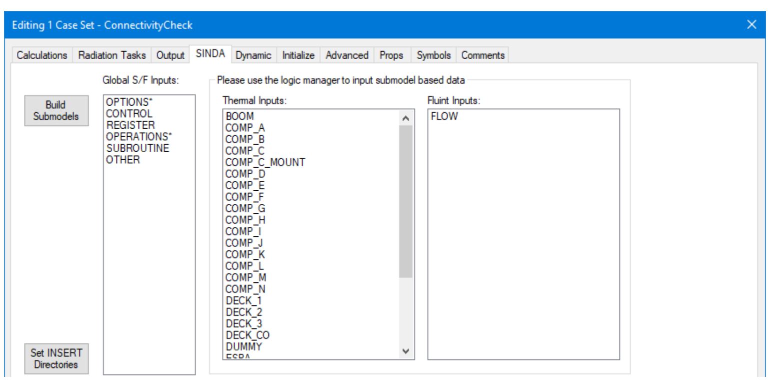


- Visualize the model in the Model Browser by Submodel.ID looking for any Node 1's that should be renumbered. Also, a good time to check for Duplicate Nodes
- Check Heatloads. Note that the naming assigned helps to identify the values
- Check Connectivity via Contactors and Conductors. For portions of the model, it can be useful to visualize contactor markers and turn off all surface visibility and inspect where contact is made.
  - Check that all optical and Material properties are assigned
    - Check that Assemblies and Trackers are properly defined
    - Check symbol usage against expected object references



CaseSets



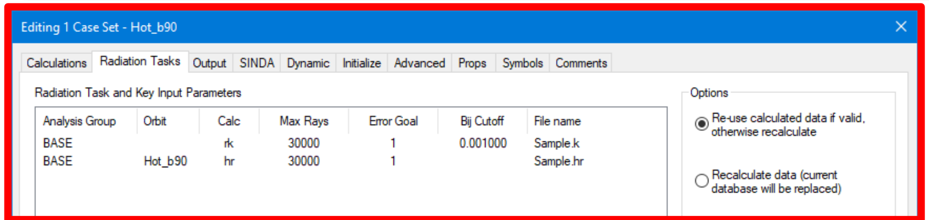


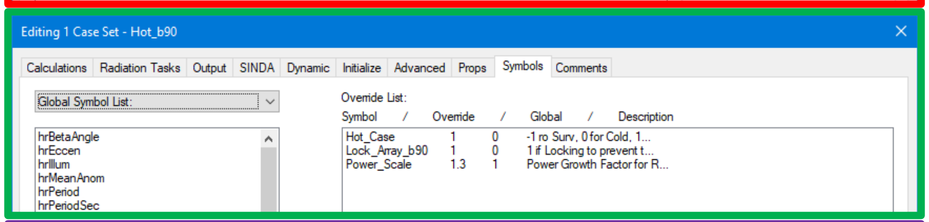
- Create a CaseSet in the Case Set Manger named ConnectivityCheck (rcCaseSet). Uncheck *Calculate Radiation*. Select the SINDA Tab and click the OPERATIONS ListBox entry. Before the code, enter:

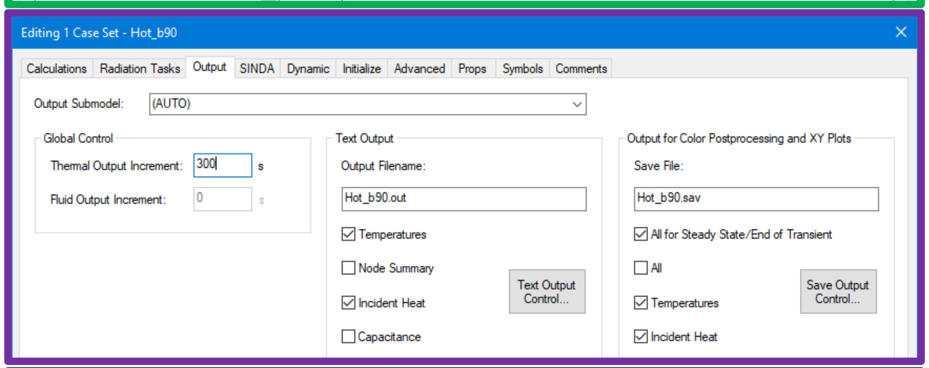
```
M       CALL HTRMOD('SOLAR_PANELS','ALL')
M       CALL HTRNOD('ESPA',1001)
```

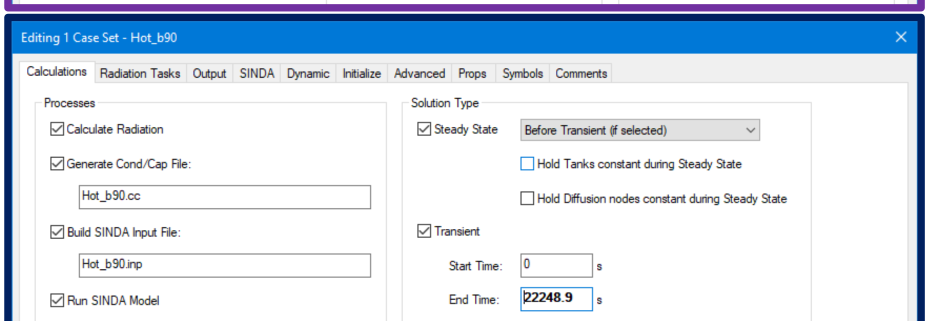
  These lines hold the unconnected SOLAR_PANELS submodel as a boundary and node 1001 in the ESPA submodel also as a boundary. This is a connectivity check to make sure everything intended is connected to a boundary node.
- The Pre-Processor should fail. Two deliberate errors were introduced. Edit the pp.out file and search for ERROR. The first error is Hot_Case not being defined as a Register. This can be reconciled in the Symbol Manager. The second is Q_COMP_KK and Q_COMP_LL not being found. These were typos; correct in the LogicObject and rerun.
- Now it makes it past the Preprocessor but fails during compile. Edit the messages*.txt file. Most often there is an Unresolved External (i.e. illegal function name) or undefined variable in FORTRAN. Remove the "F   CALL NOTFOUND" line and rerun
- Now, it preprocesses and compiles, bur fails during the processor. Edit the .out file and search for ABNORM. Most likely, there are nodes that are unconnected to a boundary. In this case, all of COMP_H is unconnected. Add a HTRMOD line to OPERATIONS.
- Finally, a last rerun is successful and connectivity throughout the model has been established. Note that the results are likely gibberish, but the lowest node temperature should be 20°C as defined by the ESPA node default initial condition

CaseSets



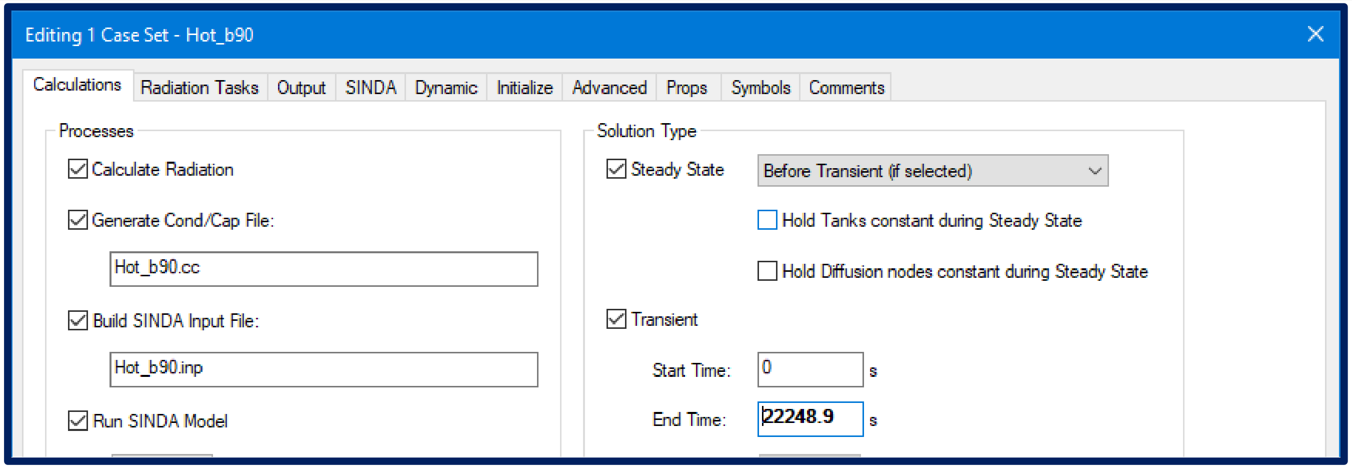






- Create a new CaseSet called Hot_b90. Add a Radiation Task: Radks, Rays: 35000 rays, Error: 1%, Cutoff: 0.01, Filename: Sample.k, Submodel: SAMPLE_RADK. Add a second Radiation Task: Heating Rates, Orbit: Hot_b90, Rays: 35000 rays, Error: 1%, Cutoff: 0.01, Filename: Sample_Hot_b90.hr, Submodel: SAMPLE_HR. It is wise to name files with unique names so as not to always overwrite. For example, Radks for BOL and EOL could be used with multiple orbits. The Radk files may include BOL or EOL in the name, while the HR files might include Hot/Cold (or BOL/EOL) along with the orbit name. NEVER ACCEPT THE DEFAULT VALUES! (C13E1.k provides absolutely no information about the file…)
- On the Symbols Tab, add Hot_Case=1, Lock_Array_b90=1, and Power_Scale = 1.3. Symbol overrides are the best method to define differences between simulation cases.
- On the Output tab, select 300 s for the output interval and output T and Q to both OUT and SAV files
- On the Calculation Tab, select both Steady State and Transient and enter hrPeriod*4 for the End Time
- After running this model, display only COMP* submodels and show the results for Steady State. Ideally, they should all be between 40 and -10.

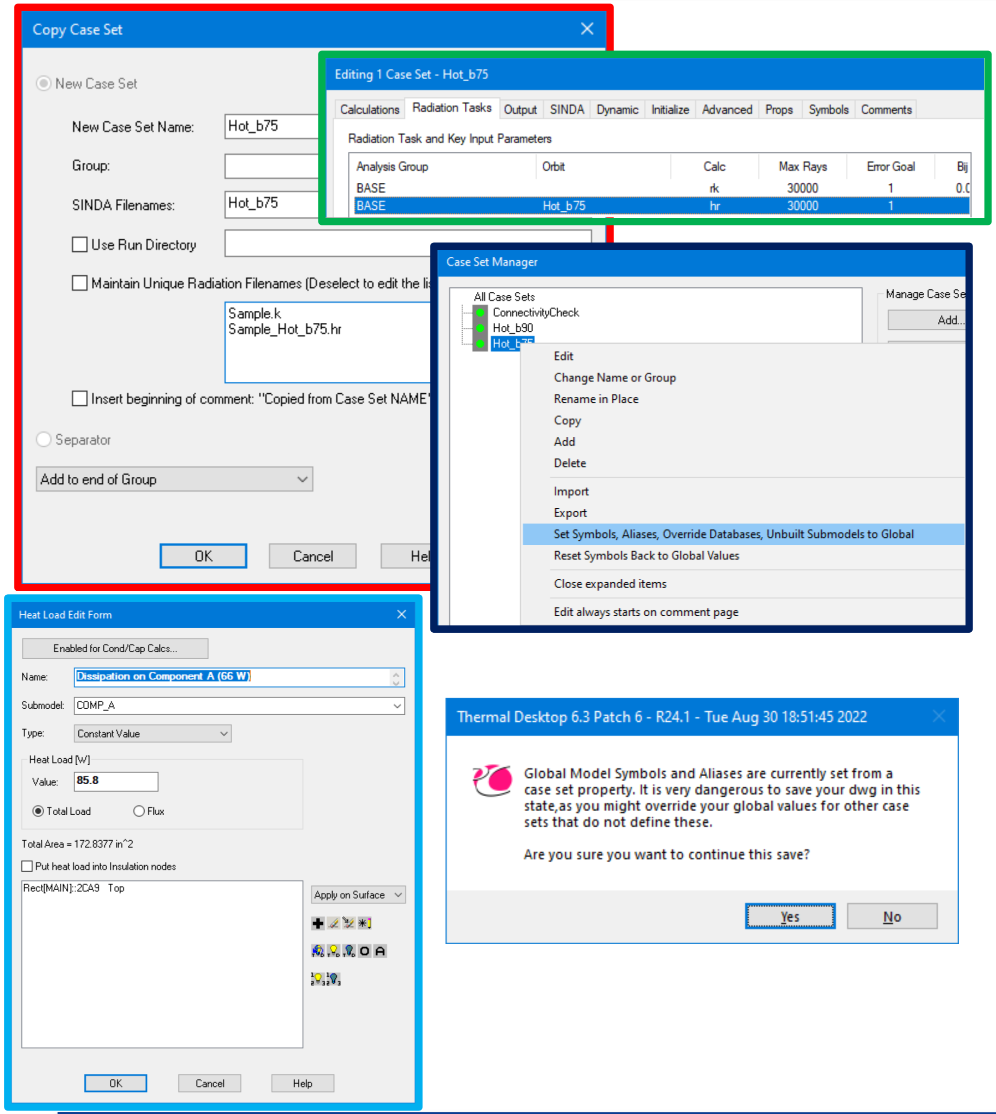
A brief Diversion: Steady State vs. Transient



- Analysts need to decide if the predictions they get from a steady state solution are adequate for presentation quality outputs
- In general, the recommendation is to not rely only on steady state predictions, but rather to use steady state to determine better initial conditions for transient to achieve *quasi-steady state*.
- Steady state uses orbit average values for environments and hence may not capture the hottest or coldest points in an orbit
- Furthermore, heater behavior in steady state seeks to find a value between on and off and cannot predict behavior such as rapid cycling or insufficient heat to reach the off temperature
- Lastly, if there are any stability requirements, it is impossible to demonstrate compliance using only steady state analyses

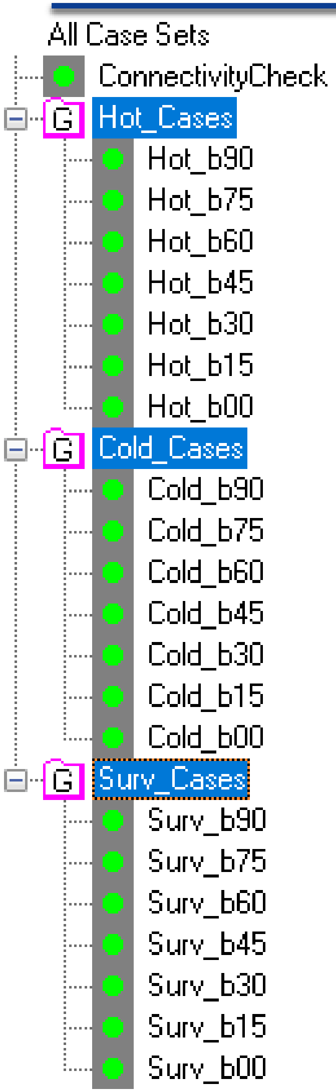
Recommendation: use steady state only analyses very early for rudimentary sizing and faster runs, but switch to SS + TR as soon a reasonable for better predictions

CaseSets

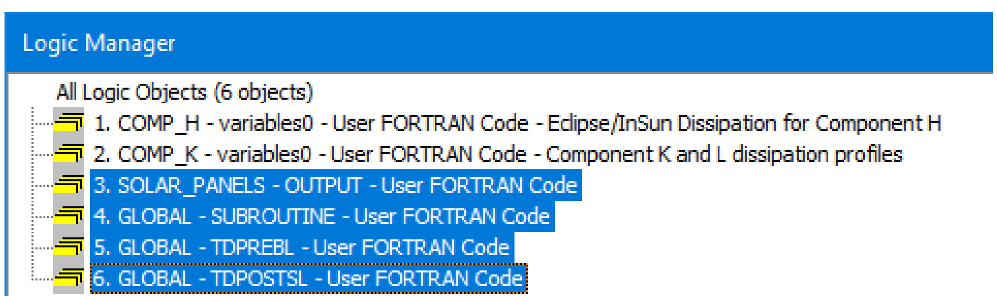


- Copy the Hot_b90 case and rename to Hot_b75. Uncheck the box to Maintain Unique Radiation Filenames and change Sample_Hot_b90.hr to Sample_hot_b75.hr
- On the Radiation Tasks Tab, double click the Heating Rates task and change the Orbit to Hot_b75
- On the Symbols tab, change the Locak_Array_b90 back to zero, as this is not a Beta 90 case and the tracker should be unlocked.
- At the Case Set Manager, select the Hot_b75 case and right click. Then select the *Set Symbols…to Global*. This is a very powerful feature to verify a CaseSet has everything as intended. This will actively change all Symbols (etc) to the override values throughout the model. Be careful not to Save the model in this State unless you want all the symbols to acquire the override values. The next command below allows a Reset back to the original symbol values. A warning is displayed during Autosave if the Global values have been overridden.
- Edit the Component A dissipation notice that the power dissipation is now 85.8 (66 * 1.3) due to the Power_Scale symbol.
- Go back to the Case Set Manager, select a case and right click to "Reset Symbols back to Global Values"
- It is generally easier to copy new CaseSets and modify them from an existing, functional case.

Sample Model Conclusion (For Now)



- The Hot, Cold, and Survival cases were all generated. The Cold Cases were copied from the corresponding Hot Case, changing the HR filename and orbit. The cold cases were then all selected and the symbols updated for cold.
- It turned out to be easier to export the Cold Case Sets, import them again as duplicates, and then rename them to Surv and update the Symbols, since the HR files were the same as Cold.
- Each group of 7 were assigned to a CaseSet group for better organization.
- Some specialized logic should be added to output the nodal temperatures of interest for Components A-N. This logic takes advantage of user developed subroutines to find the Max and Min temperatures in a specified submodel, with options for also narrowing it by node range within the submodel. Furthermore, another routine allows averaging over a range. Import the LogicObjects from the Sample_Instruments.dwg. Also, edit the hrBetaAngle symbol and ensure it is output as a SINDA register.
- Run all 21 cases and for each case a Hot/Cold/Surv_b##.txt file will be created with relevant output every 300 s. These files will be used later to support the Data Analysis portion.



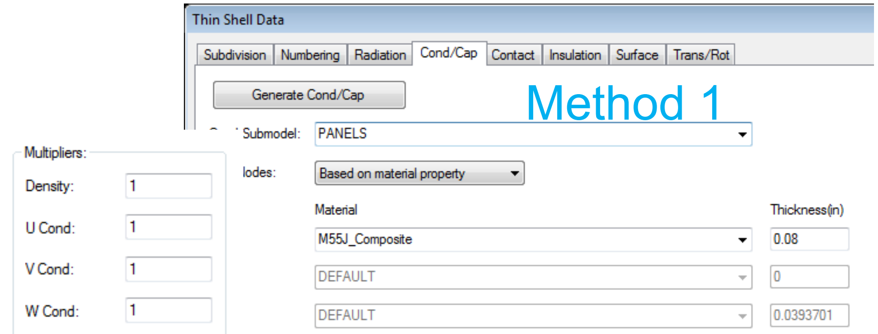
Modeling Specific Component Types

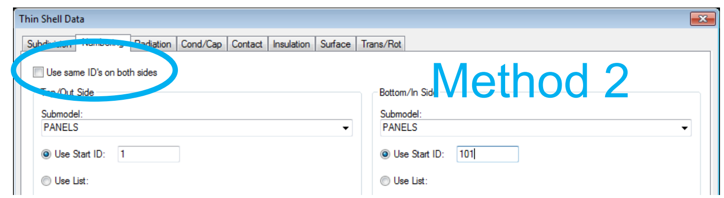
BUILD MODEL: Simulating Thermal Hardware

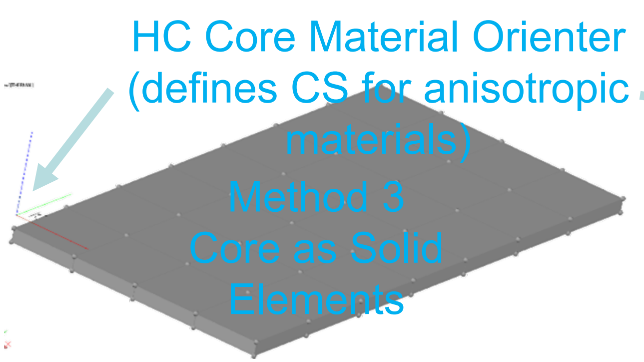
- Every analyst will develop their own modeling style (i.e. how physical components will be represented in the analysis model)
- The following slides include suggestions on methods to model:
  - Honeycomb Panels
  - Insulation
  - Thermistors and Thermocouples
  - Constant Conductance Heat Pipe
  - Variable Conductance Heat Pipe
  - Controller (most often for a heater)
  - ThermoElectric or CryoCooler
  - Phase Change Material
  - Louvers

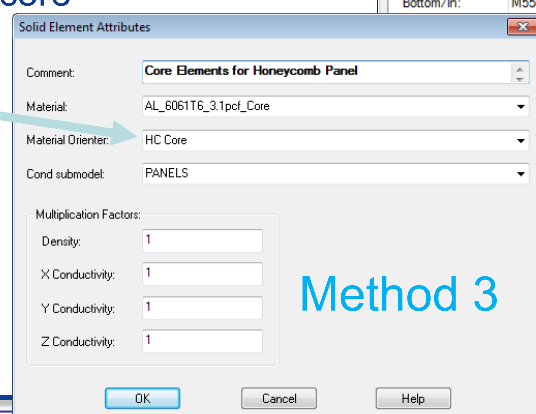
BUILD MODEL: Honeycomb Panel Modeling

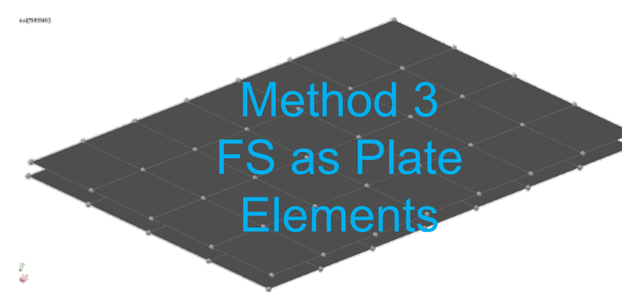
- **Three ways exist to model Honeycomb Panels**
  1. Neglect Core: use a single sided surface with 2x the facesheet thickness and the material of the facesheet
  2. Include Core through Conductivity: use a double sided surface with thickness and material specified for facesheet, core, facesheet. Note: the in-plane spreading effect of the core material is not included using this method…
  3. Include full core effect: model the core as a solid element and the facesheets as plate elements using the same nodes as the top/bottom for each facesheet. If applying non-isotropic properties, must define material orienter for solids.
- TIP: you can use the U Cond/V Cond multipliers increase the conductivity as a function of the core material to account for in-plane spreading by the core





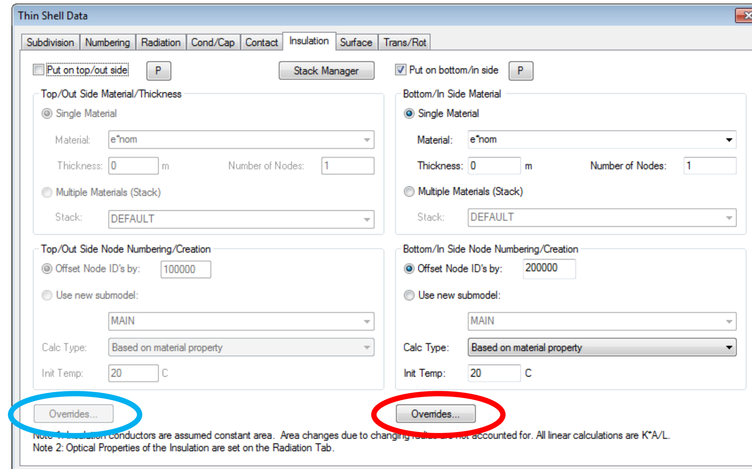


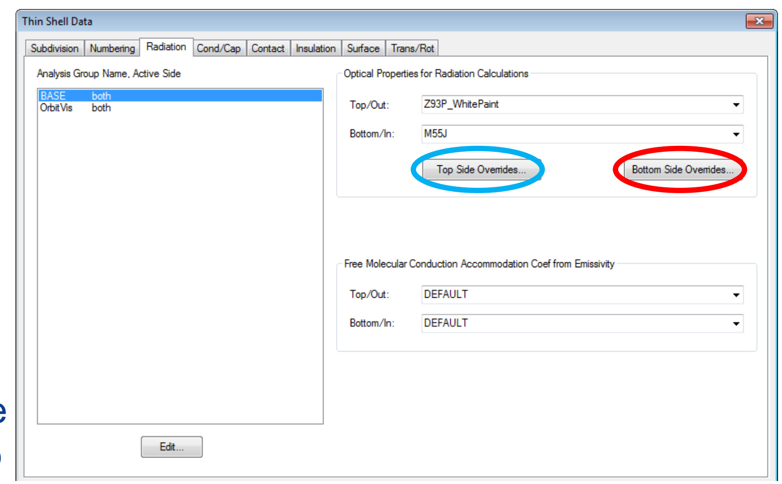




BUILD MODEL: Insulation Modeling

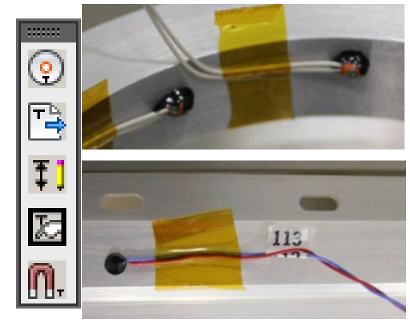
- Insulation may be conductive or MLI (i.e. radiative)
  - ✓ Define insulation material
  - ✓ If material has conductivity <> 0, k and thickness will be used
  - ✓ If material has ε* defined, radiative will be used (thickness ignored)
  - ✓ Assigning kg/m3 of 0.6 and applying a thickness of 1 m will allow MLI mass to be estimated by model
  - – Possible to have both…
- Use intuitive node numbering
  - ✓ Keep all non MLI nodes below 100000
  - ✓ Use Node Offset of 100000 (200000 if both sides have insulation)
  - – When post processing, any nodes above 100000 are MLI temps and can likely be disregarded
- If overrides are used (i.e. applying MLI to only some of the nodes), make sure to adjust the optical property overrides as well
  - ✓ Opt Prop overrides will need to reference MLI node number, not underlying surface
- MLI can be “programmed” to either be enabled or disabled using the P button at the top
- For “tented” MLI (offset by large gap from surface), best not to use low ε optical property for inner or outer layer – doubles isolation to have low ε and low ε* in series



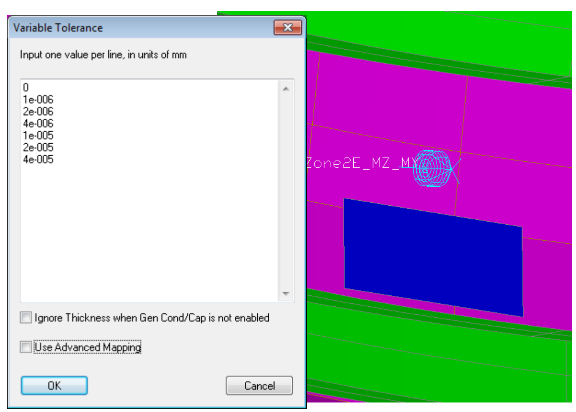


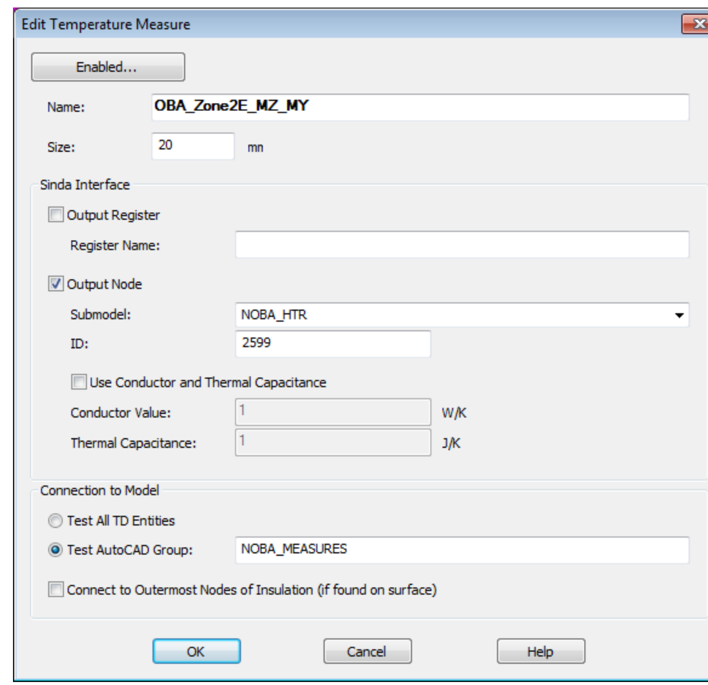
BUILD MODEL: Thermocouple and Thermistor Modeling

- Thermocouples, Thermistors, and other measurement devices are the tangible data readings visible to the thermal engineer based on the actual hardware
- Thermal Desktop has an object type called a Measure, which represents a spatial location for a temperature reading.
- Measures are displayed graphically with their Name and show the mapped location. Can be moved to the exact location desired by the user



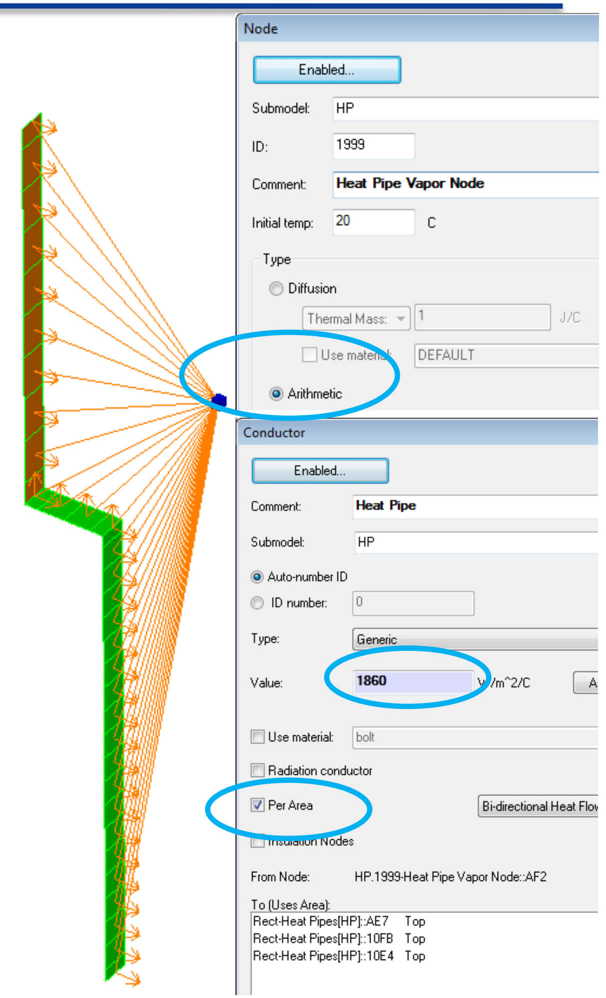
✓Define the Measure and place it correctly in space
✓Set the size large enough to be visible when displayed
✓Assign a meaningful name (e.g. Telemetry mnemonic)
✓Determine if you wish to output measure value to an existing Register or a Boundary Node
✓Good idea to Create groups to ensure Measure maps to reasonable entity
✓Set tolerance
✓Execute mapping
✓Snap Measures to Mapped Entity
– Can copy measures from One drawing to another
– Can import from text file





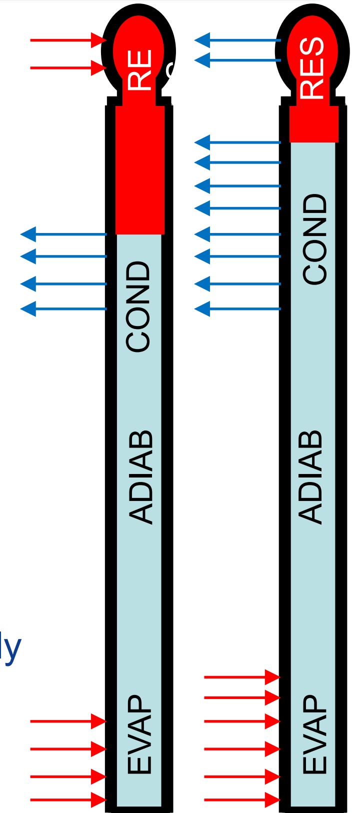
BUILD MODEL: Constant Conductance Heatpipe (CCHP) Modeling

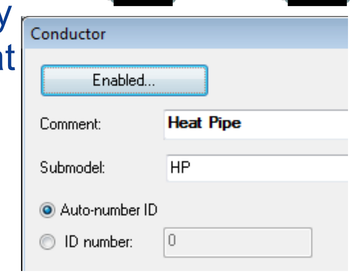
- **Heatpipes are commonly used devices in thermal designs**
  - They operate on an evaporation/condensation cycle that allows heat to be transported over large distances with a minimal temperature gradient
  - Heat enters the evaporator end of the pipe and vaporizes the working fluid
  - The vapor is driven to the condenser end of the pipe by a small pressure gradient
  - The vapor changes to liquid in the condenser end releasing the heat
  - The liquid is pumped to the evaporator by capillary forces through a wick structure
  - Unfortunately, the capillary forces that drive the fluid are very small compared to gravity which places constraints on HPs during ground testing
  - Device can operate in reverse depending on loads…
- **HEATPIPE exists in FloCAD, but a simpler way exists to model a heatpipe that does not require a FloCAD license**
  - ✓ Define surfaces based on length/shape of pipe
  - ✓ Set as edge nodes 1 x Z breakdown, assign nodes to increment from one end to the other (e.g. x001, x002, x003…) and merge at interfaces
  - ✓ Set width of surface to width of interface flange
  - ✓ Set Material to Aluminum (Assuming aluminum walls)
  - ✓ Set Thickness to effective thickness (HP Cross Sectional Area / Width)
  - ✓ Create Arithmetic node to represent vapor node and set to (x999)
  - ✓ Create a Node-to-Surface conductor from Vapor node to all HP surfaces
  - ✓ Check the Per Area option and set conductance value to h per linear inch / Flange Width (3.0 W/lin in. K typical for ammonia HP)
  - ✓ Add contactor from evaporator/condenser sections to structure/radiator



BUILD MODEL: Variable Conductance Heatpipe (VCHP) Modeling

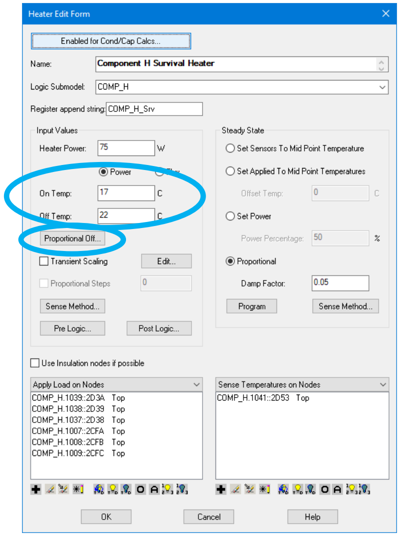
- A Variable Conductance Heat Pipe (VCHP) is an extension of a CCHP
  - It includes a reservoir at the condenser end filled with Non-Condensable Gas (NCG)
  - This reservoirs is cold biased and includes a heater to maintain its temperature
  - When the reservoir heats, the NCG expands and partially fills the condenser section, blocking condensation and reducing the effective condenser length
  - When the reservoir cools, the NCG contracts and recedes toward the cold reservoir, increasing the effective condenser length
  - By heating and cooling of the reservoir, the evaporator end of the VCHP can be controlled to a given temperature under a variety of source and sink values
- HEATPIPE routine in SINDA can accommodate modeling a VCHP, but requires additional inputs: [Vapor Node, Array of Vap-to-Wall Cond Numbers, Array of Segment Lengths, Volume, HP Diameter, h_{cond}, h_{evap}, Res Volume, Res Temp, NCG charge, Working Fluid and NCG IDs]
  - ✓ Define a heatpipe as previously specified and ensure that the nodes increase incrementally from the Evap to the Condensor. Ensure the vapor node is arithmetic!
  - ✓ When defining the Vapor to Wall Conductors, be sure to specify the ID and not Auto-number
  - ✓ This ID will be a fixed value set by the user to be specified in the Vapor-to-Wall Cond Array
  - ✓ Be sure to add a geometric representation of the reservoir and its radiator and heater: heat should be applied at the Res Radiator and sensing at the evaporator
  - ✓ Define Arrays with Segment Lengths and Cond Nos in the Logic Manager
  - ✓ Define a VARIABLES 1 block in logic manager with a call to HEATPIPE and add in the appropriate variable and argument values
  - ✓ Refer to SINDA/FLUINT manual for specifics…*be careful with initial conditions*

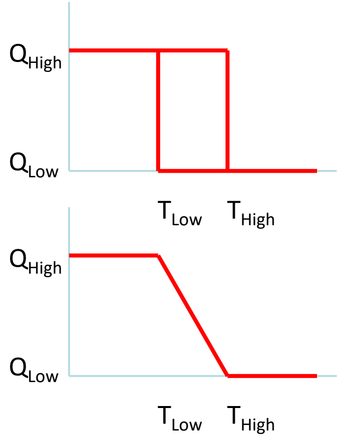


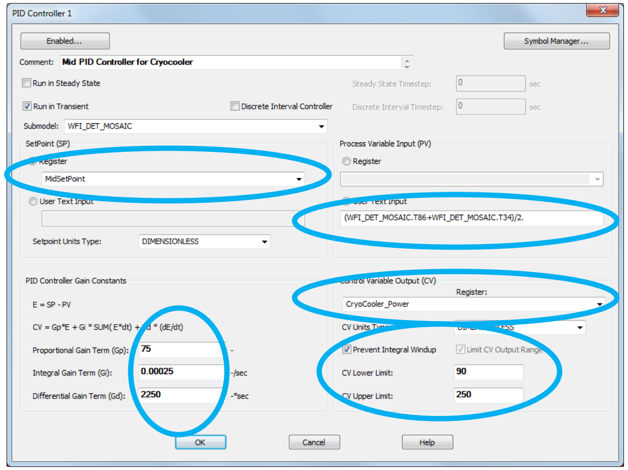


BUILD MODEL: Controller (Heater or Cooler) Modeling

- **More thermal models are asking for controller modeling: input value is mathematically evaluated to predicts output value of controller (e.g. Input: Temperature, Output: Heater Power)**
- **Common example is thermostatic or proportional heater:**
  - ✓ Define On and Off Temperatures and Max power to apply
  - ✓ Define Location to Apply Heat and Location to Sense Temperature
  - ✓ Use Proportional for Steady State behavior
  - ✓ Determine if Proportional is needed for transient
    - ✓ Often needed for tighter stability
    - ✓ ***Widen*** Range between Toff and Ton for better control
- **For a full PID controller, use a PID controller under Logic Manager**
  - ✓ Define Proportional, Integral, and Derivative Gains
  - ✓ Define Setpoint: what you are trying to achieve
  - ✓ Define Sensing Variable: what it actually is
  - ✓ Define Control Variable what you are actually changing (e.g. power)
    - ✓ Done as a Register to be used elsewhere in TD
  - ✓ Define upper/lower bounds on Ctrl. Variable (e.g. max/min power)
    - ✓ Generally a good idea to *Prevent Integral Windup*
  - ✓ Apply output variable to some TD object, usually a heat load
    - ✓ Make sure to output as expression so that changes in process variable as SINDA runs are applied correctly







BUILD MODEL: TEC or Cryocooler Modeling

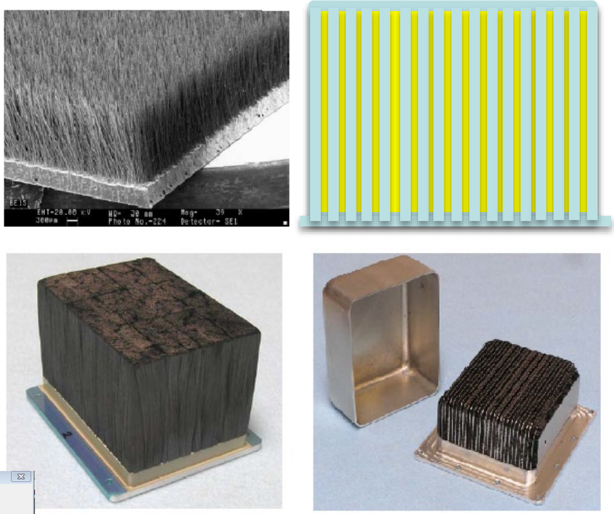
- **ThermoElectric Coolers utilize the Peltier principal to induce a temperature gradient between two junctions when a current is applied to provide cooling. Typically about 6% efficient**
- **Performance based on 4 related variables specific to device. Knowing 3 can allow 4th to be calculated**
  - Hot Side Temperature (can be retrieved from model)
  - Cold Side Temperature (usually the goal or setpoint)
  - Cold Side Load (can be calculated from model)
  - Hot Side Load (including input power, current, or voltage), often the independent control variable
- **User must decide how to model controller…**
  - Assume it can achieve control temperature and set cooling point to boundary temperature
  - OR Model controller with feedback and apply negative cooling load (see PID controller slide)
- **If assuming controller can achieve temperature…**
  - ✓ Need to extract heat removed from boundary cold side (QFLOW, QFLOWSET or HNQCAL)
  - ✓ Retrieve Hot Side Temperature from model
  - ✓ Determine cooling DT from T_{hot} - T_{cold}
  - ✓ Look up power needed for TEC to remove Q heat, and achieve DT for current T_{hot}
  - ✓ Apply power to Hot Side node
  - ✓ Routine developed for WFC3 to characterize TEC performance curves by 4th order polynomial
  - ✓ Could also use TRIVARIATE array with enough data points…
- **If constant power TEC…**
  - ✓ Determine heat removed based on input power, DT, and T_{hot}
  - ✓ Apply a negative heat load at cold side node
- **Hot Side and Cold side usually not coupled unless TEC is off…**

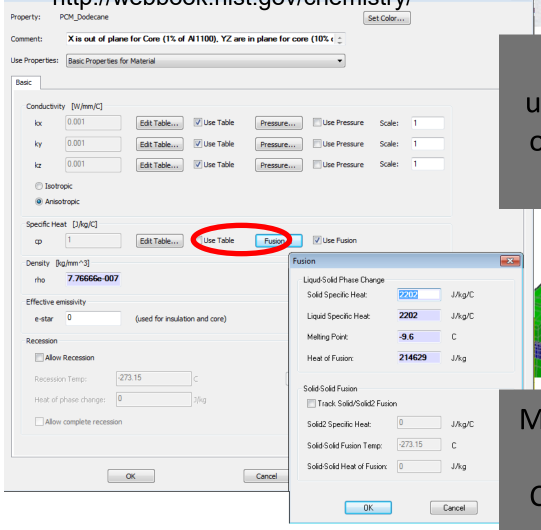
BUILD MODEL: Phase Change Material (PCM) Modeling

- **Phase change happens over a constant temperature**
  - PCMs (paraffins) are generally very poor conductors of heat
  - PCM assembly is generally a hermetically sealed aluminum housing to which a core for uniform spreading is bonded
  - Gaps in between core is filled with PCM
  - Core may contact multiple faces for improved through conductivity (aka vias)
  - Melting point is material specific. Paraffins typically used with Solid to Liquid phase change
- **Thermal Desktop includes capabilities to model PCMs via FUSION function**
  - ✓Define conductivity of material as the core (typically 10% or AL1100 for plane of fins, and 1% or AL1100 for out of plane)
  - ✓Define Specific Heat of material as FUSION
  - ✓Specify Melt Pointing and Heat of Fusion
  - ✓Majority of mass of Core+PCM is in the paraffin for density

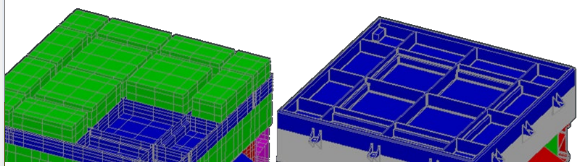
Alkane	Formula	MP (°C)	h_{FUSION} (kJ/kg)
Decane	$C_{10}H_{22}$	-29.6	202.3
n-Undecane	$C_{11}H_{24}$	-25.6	142.9
n-Dodecane	$C_{12}H_{26}$	-9.6	214.6
n-Tridecane	$C_{13}H_{28}$	-5.4	155.6
n-Tetradecane	$C_{14}H_{30}$	5.9	224.2
n-Pentadecane	$C_{15}H_{32}$	10.0	161.9
n-Hexadecane	$C_{16}H_{34}$	18.2	226.9
n-Heptadecane	$C_{17}H_{36}$	22.0	165.4
n-Octadecane	$C_{18}H_{38}$	28.2	240.0

Data from http://webbook.nist.gov/chemistry/



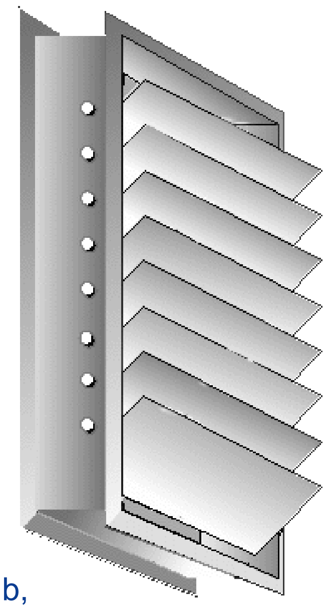


Core material helps to spread heat uniformly throughout assembly. Void areas of core are filled with PCM. PCM generally stays in contact with core



Model Housing as 2D Rectangles, PCM+Core as Solid with PCM Material assigned. Contactor between Core/Housing for bond

BUILD MODEL: Louver Modeling



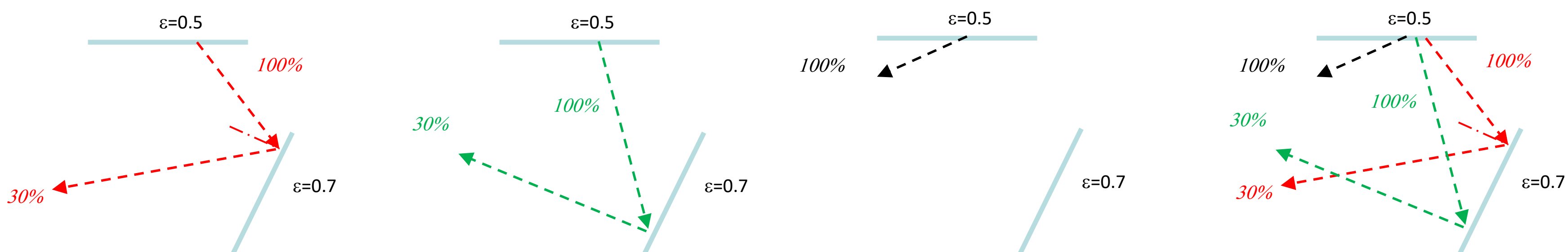
- Louvers are devices which passively activate as a function of temperature. They are often used to reduce the view of a radiator to space when they get colder and increase the view as they get warmer to conserve heater power.
- They function when a bi-metallic spring changes its length and spring constant as a function of temperature, rotating a set of highly reflective, parallel blades
- These blades change the view factor from an underlying radiator its environment
- Since Radiation couplings are generally for a fixed geometry and properties, the typical Monte Carlo Ray Trace does not account for this variability nor does it know the temps…
- To model this, it is necessary to modify primarily the Radk to Space and environmental loads in the thermal model. Radiation Model should be run with fully open properties…
- Account for louver in SINDA model (to first order at least)…
  - ✓ In OPERATIONS, loop from 1 to NGTOT for each Louver Node. For each *Index*…
    - ✓ CALL CONDAT(GETGMOD(*Index*),GETGNUM(*Index*),JTEST,KTEST,ZTEST)
    - ✓ Check if NDNAM(ITEST)=SPACE, NDINT(ITEST)=SpaceNode, NDNAM(JTEST)=Louver Sub, NDINT(JTEST)=Louver Node
    - ✓ Check if NDNAM(JTEST)=SPACE, NDINT(JTEST)=SpaceNode, NDNAM(ITEST)=Louver Sub, NDINT(ITEST)=Louver Node
    - ✓ If either condition is met, then flag *Index* as a Louver G and store Cond number for later reference
    - ✓ Store Fully Open conductor value for this G for future modification
  - ✓ In VARIABLES1, for each Louver node (G Index and Fully open conductance should have been determined and stored)
    - ✓ Determine the scale factor for partial closure based on temperature and Louver Performance curves: SCL (1 to >0)
    - ✓ Set G(Louver Index) = G Fully Open * SCL
    - ✓ Adjust Heat Load by SCL (If Nadir pointing, Tot Env - Eclipse Env = UV Env, Eclipse Env = IR Env, adjust based on α,ε)

Radiation Calculations

Radiation Calculations

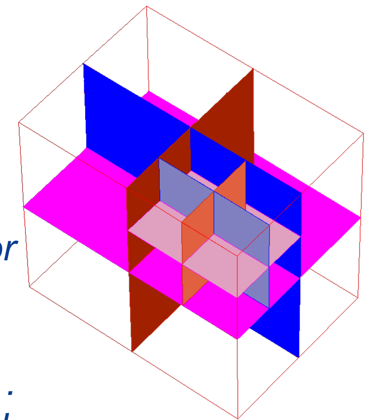
- **Monte Carlo Ray Trace**
  - *Begin with Surface 1 and find a random point and direction to "fire" the ray with 100% energy*
  - *Determine if the ray intersects any surface*
  - *Oct Cells subdivide model to reduce number of intersection tests*
  - *If ray intersects a surface, determine how much energy is absorbed by the surface (α,ε)*
  - *If some energy remains, determine if the energy is reflected specularly (Angle of Incidence = Angle of Reflection) or diffusely (Random direction)*
  - *Continue propagating ray until energy is below extinction threshold at which point it is either completely absorbed or completely reflected*
  - *Once extinguished, select new random point and direction for next ray*
  - *Once finished with Surface 1, move to next surface and repeat until all surfaces have been computed*

- **Example**
  - *The Red ray leaves Surface 1 with 100% energy and strikes surface 2. 70% of the energy is absorbed by Surface 2 and the remaining 30% of the original energy reflects specularly and finds its way to space*
  - *The Green ray leaves surface 1 with 100% energy and strikes surface 2. 70% of the energy is absorbed by Surface 2 and the remaining 30% of the original energy reflects diffusely and finds its way to space*
  - *The Black ray leaves Surface 1 with 100% energy and goes to Space*
  - *The Resulting Bij terms would be:*
    - *Bij (1 to Space) = (0.3 + 0.3 + 1.0) / 3 = 0.5333*
    - *Bij (1 to 2) = (0.7 + 0.7 + 0) / 3 = 0.4667*



Radiation Calculations

- **Oct Cells**
  - *Find bounding box around entire model*
  - *Divide by midplanes in all three directions to form 8 smaller boxes*
  - *Determine #surfaces in each of the 8 cells*
  - *If #surfaces is > specified, then subdivide cell again. Continue until max #surfaces or max #subdivisions is reached*
  - *Consider 8 surfaces meeting at point (i.e. ribs). Lower #surfaces can never be met…*
  - *Used to minimize intersection tests as each ray is computed*



- **How Many Rays?**
  - *Typically start with 35000. (2.65/1.76% error for 0.1/0.2 B_{ij})*
  - *Is the run time tolerable?*
  - *Spending time calculating small couplings that are eliminated?*
  - *Does it affect temperatures? By how much?*
  - *In the end it comes down to run time vs. accuracy…*

- **Error Calculations**
  - *Statistically, the error can be estimated with 90% confidence by:*

$$Error_{ij} = 1.65\sqrt{\frac{1-B_{ij}}{N_{rays}\,B_{ij}}} \times 100$$

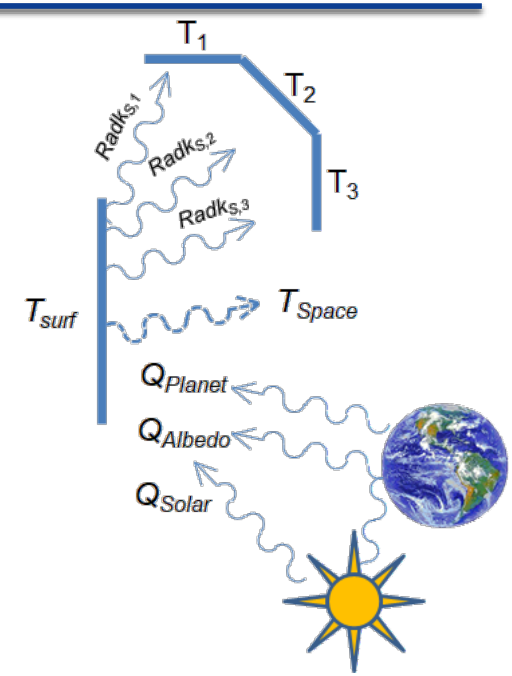
  - *Weighting the Error for every node yields a single term per surface to determine if sufficient rays have been fired for that surface*

$$Weighted\ Error_i = \frac{\sum_{j=0}^{n}(B_{ij} * Error_{ij})}{\sum_{j=0}^{n}(B_{ij})}$$

- **How do B_{ij}'s become Radks?**
  - *$Radk_{ij} = A_i\varepsilon_i B_{ij}$*
  - *By reciprocity though, $A_i\varepsilon_i B_{ij} = A_j\varepsilon_j B_{ji}$*
  - *So, should B_{ij}, B_{ji} or both be used?*
  - *Thermal Desktop chooses a weighted average based on the $Error_{ij}^2$ and $Error_{ji}^2$*
  - *So $Radk_{ij} = Radk_{ji} =$*
    $\{A_i\varepsilon_i B_{ij}/Err_{ij}^2 + A_j\varepsilon_j B_{ji}/Err_{ji}^2\} / \{1/Err_{ij}^2 + 1/Err_{ji}^2\}$
  - *With this approach, greater weighting is given to larger B values, as the error is smaller for larger B*
  - *However, Radks are output **only** for $B_{ij}\neq 0$ **and** $B_{ji}\neq 0$*

Some thoughts about Radiative modeling…

- **Two ways to think about radiation modeling:**
  - *Emitted* and *Absorbed* are handled independently: Heat energy leaves Node/Surface i based on T_i , A_i , and ε_i as an emitted Q_{OUT}. In parallel, heat is added as sources to Node/Surface i having come from Node/Surface j's with a magnitudes based on T_j , A_j , and ε_j and proportions based on the B_{ij} as absorbed Q_{IN}'s
  - Emitted and Absorbed are handled as a *net exchange*: Net Heat is exchanged between Node/Surface i and Node/Surface j based on T_i , A_i , ε_i , T_j , A_j , ε_j , B_{ij} and B_{ji} where the direction of the heat flow is dependent on temperature.
  - Most radiation codes utilize the latter approach to minimize the computational overhead
- **Environmental heat rates are the *absorbed* load on a surface (aka a backload) and do not represent a net exchange with the environment (hence the reason the Environmental loads are always > 0)**
  - The portion of the view factor representing the view to the celestial source (e.g. Solar, Planet, etc) is included in the Radk to space
  - Therefore, the net exchange is the combination of the Environmental load and an increased view to space (which may include view to the celestial sources)
- **Radks handle the <u>net</u> radiative heat *exchange between* Node/Surface i and Node/Surface j.**
  - Extra calculations are needed if you wish to determine only the absorbed heat on a Surface/Node i from Surface/Node j – (see Backload below)
- **Backloads and Equivalent Sink Temperatures are other useful techniques**
  - Backloads are the absorbed energy on a surface from all surrounding surfaces. If applying a backload, then the radk to space is increased for where the surrounding surfaces reside. This is essentially how the Environmental Heat Rates are handled.
  - Equivalent Sink Temperatures represent the effective Temperature "seen" by a surface if its entire field of view was replaced by a black body surface. This is often used to simulate the environment during testing. Note that in a test, the panels often have an e less than one, so the Equivalent Sink Temperature needs to be adjusted based on T_{surf}

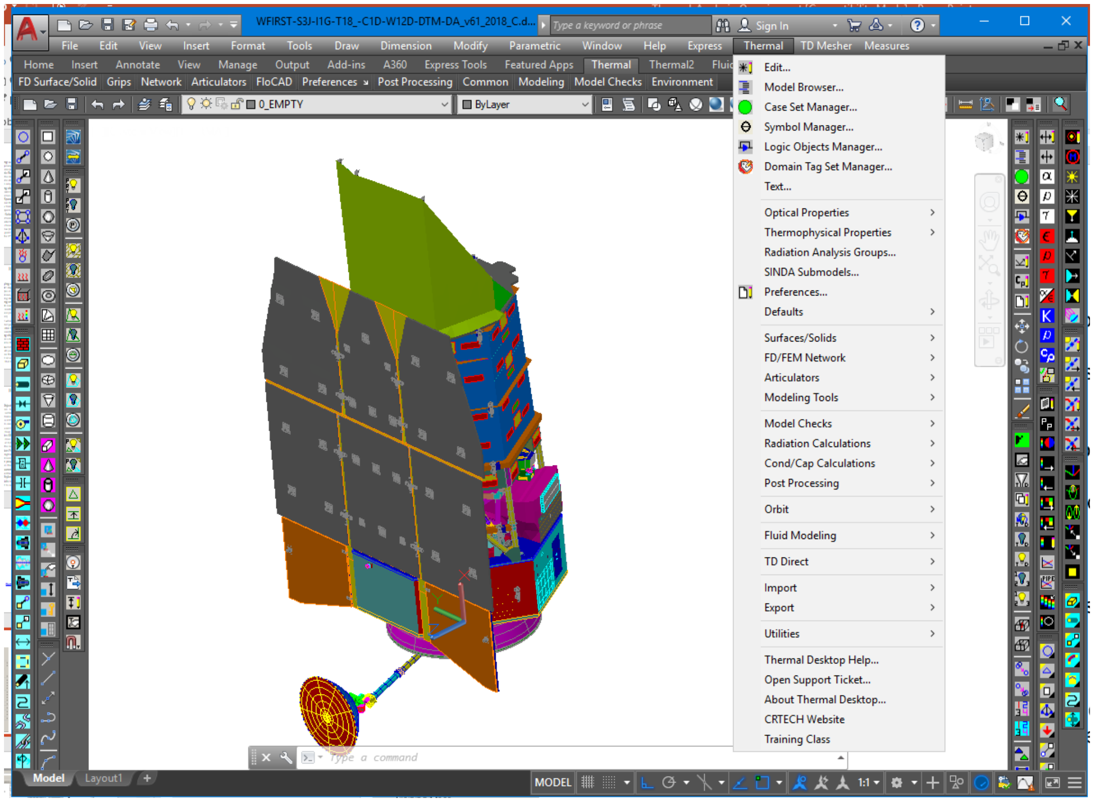


$$Q_{BL,i} = \sigma \sum_{j}^{1,N} Radk_{ij} T_j^4$$

$$T_{i,Sink}^4 = \frac{\sigma \sum_{j}^{1,N} Radk_{ij} T_j^4 + Q_{env}}{\sigma \sum_{j}^{1,N} Radk_{ij}}$$

$$Radk_{i,Sink} = \sum_{j}^{1,N} Radk_{ij}$$

After the Radiation Computations…



While Thermal Desktop does most of the heavy-lifting with generating the SINDA/FLUINT file, a good analyst knows how to check that the code is generating reasonable inputs…trust, but verify!

- After the radiation computations are complete, the absorbed orbital heat loads and radiation couplings are merged with other data to form the thermal model
- The format used is SINDA/FLUINT, which treats each calculation point as a node
- Nodes are connected via conductors, which may be linear: $Q=GL*(T_a-T_b)$ or radiative: $Q=\sigma*GR*(T_a^4-T_b^4)$
- Nodes may have capacitance and/or heat sources applied (heat dissipation, heater)
- At least one node must be a boundary (usually deep space)
- User logic governs the behavior of the model to change boundary conditions as a function of time or temperature and control the accuracy of the solution
- Results of interest are typically nodal temperatures and heat loads as a function of time

SINDA / FLUINT

Basic SINDA Outline

- What is SINDA
- The SINDA Deck
- Program Flow
- SINDA Syntax
- SINDA Model Example
- Running SINDA

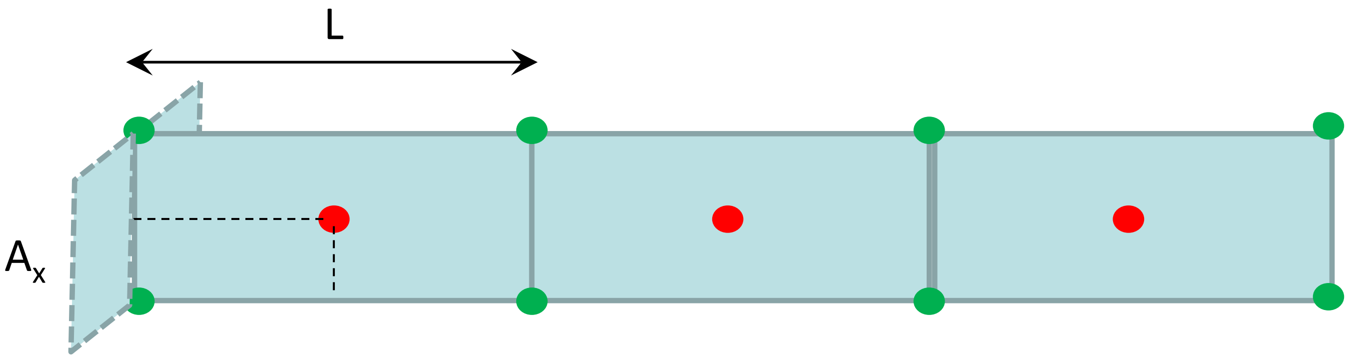
What is SINDA?

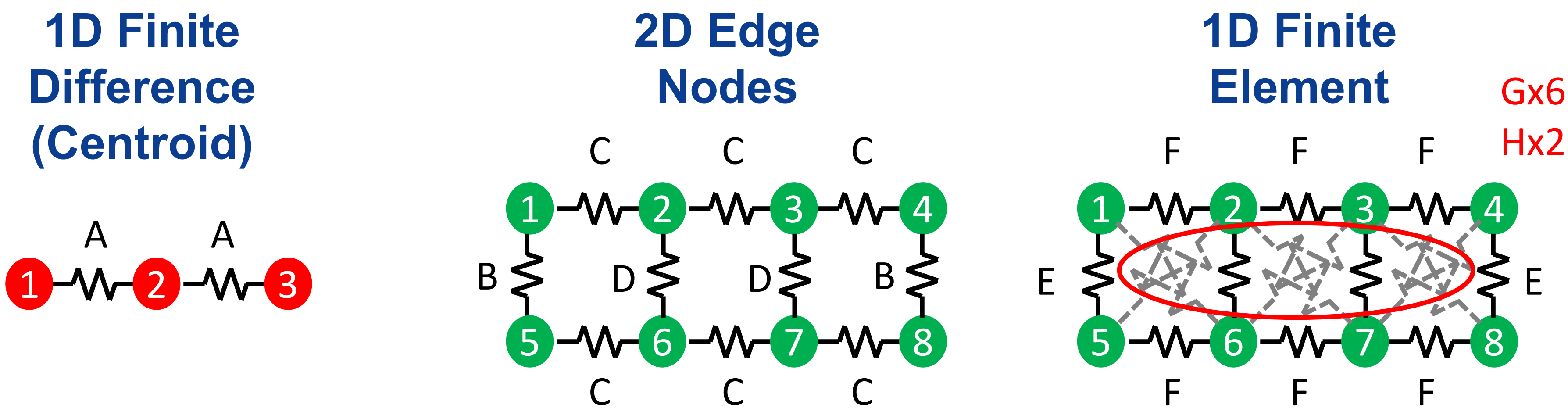
- SINDA: Systems Improved Numerical Difference Analyzer
- Many flavors exist:
  - Cullimore and Ring: Sinda/Fluint (commercial code GSFC uses most)
  - Government SINDA: SINDA85 (“free” but no longer developed)
  - SpaceDesign: Sinda/Fluint (commercial version included with TSS)
  - SINDA/G: Gaski SINDA (commercial code from MSC, formerly NAI, no longer supported)
- SINDA includes:
  - A pre-processor for reading the SINDA input deck and storing the data in intermediate files used by the solver
  - A set of library functions compiled in FORTRAN and linked to the executable
- The SINDA deck is a text file containing the information necessary to solve for temperatures, based on heat, conductance, and capacitance inputs
- As such, SINDA requires a FORTRAN compiler to function
- ***In the end, SINDA models are really just compiled FORTRAN programs***

A quick digression…

- SINDA is not *necessarily* a Finite Difference solver
- In fact, it is a generalized equation solver which could be used for other 1st order analysis types if we could input the terms appropriately for the matrix.
- In the end it is simply solving [G]*[T] = [Q] where G, T, and Q could be any physical terms related by linear equations
- SINDA has strengths over other equation solvers (e.g. MATLAB,NASTRAN) in that was developed to support many of the things that thermal engineers need through a library of functions included in the compiled executable
- Finite difference is a numerical formulation that estimates the physics behavior of heat flow in a simplified 1D manner after Taylor Series expansion and elimination of negligible terms
- Reference any basic heat transfer book for the derivation of the Finite Difference method
- The finite element method may also be used within the constructs allowed by SINDA but the “conductors” no longer directly represent 1D heat flows

Nodal representation of a 2D bar…





Where $k = L = A_x = 1$:

A=1, B=0.5, C=0.5, D=1, E= 0.16667, F= 0.16667, G=0.33333, H=0.33333

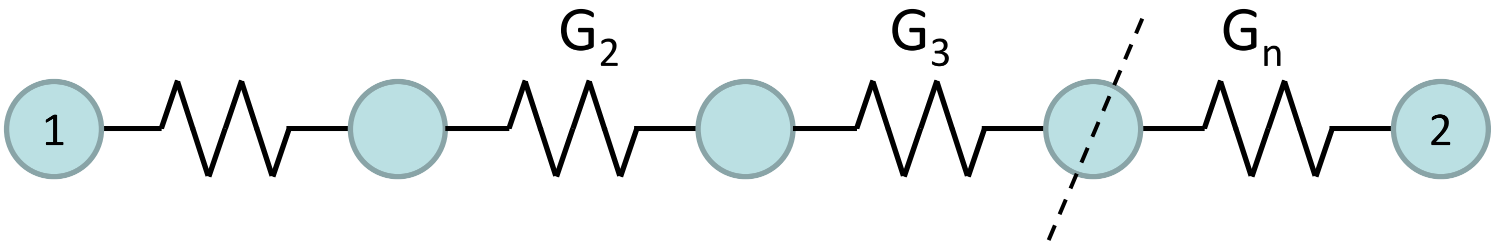
Where $k = A_x = 1$ and $L = 2$:

A=0.5, B=1, C=0.25, D=2, E=0.58333, F= *-0.16667*, G=0.41667, H=1.16667

Condensing Conductors…

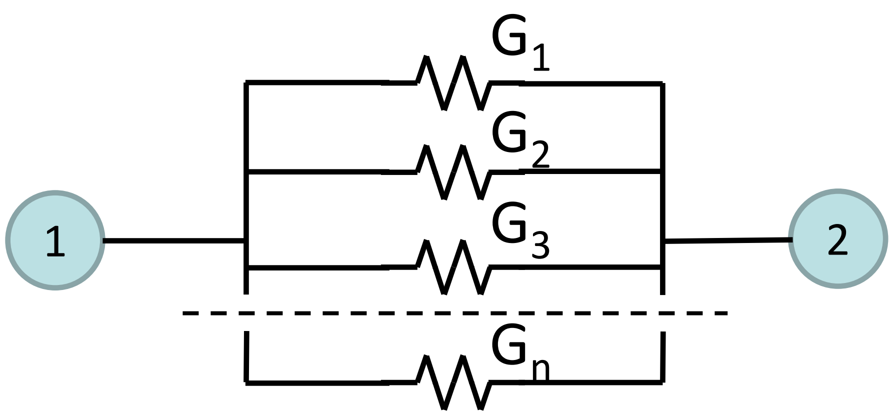
Conductors in series

$$1/G_{1\text{-}2} = 1/G_1 + 1/G_2 + 1/G_3 + \ldots + 1/G_n$$



Conductors in parallel

$$G_{1\text{-}2} = G_1 + G_2 + G_3 + \ldots + G_n$$



What is the SINDA Deck?

- The SINDA Deck is a text file with relevant sections delineated to define the inputs to a thermal model
- Sections are identified by the keyword HEADER. These HEADERs define one of two types of blocks:
  - Data Blocks: Node, Conductor, Source, User, Control, Array, Carray, Register
  - Logic Blocks: Operations, Variables 0, Variables 1, Variables 2, Output Calls, Subroutines
- Data blocks are processed and the data contained therein is generally stored in binary files that are referenced by the compiled thermal model executable during run time
- Logic blocks are translated into FORTRAN specific constructs and used to compile the executable representing the thermal model
- The SINDA deck is pre-processed to generate an executable program which solves for the requested thermal data

How is the SINDA Deck processed?

- The pre-processor first reads through the SINDA deck and reorganizes it into the expected, logical flow and creates the combined.inp file
  - INSERT statements are expanded
  - Duplicate HEADERs are combined into a single one
- The pre-processor then reads through the combined.inp file, checks for any syntax errors, and writes the binary data files needed for the run
  - Illegal references to undefined Nodes, Arrays, or Registers
  - Duplicate Node, Conductor, Array numbers
- When the pre-processor is passed, the resulting FORTRAN is checked for validity by the compiler
  - Unterminated IF blocks, Illegal FORTRAN syntax, unknown function calls
- Once the compiler generates and runs the executable, SINDA will make sure the model can be solved
  - Contains at least one boundary node and everything is connected (directly or indirectly) to a boundary
- The executable produces the output .sav (binary) and .out (ASCII) files with the user requested results for further post-processing

What is the basic syntax?

- Regardless of the current HEADER, the following syntax is valid
  - C in first column - everything on this line is "skipped" by SINDA processor. It's only a comment.
  - A "$" is used for in line comments at the end of a line of SINDA code. Everything after the $ is a comment. (Use "!" if an F type statement)
- Syntax specific to Logic Blocks
  - F in first column of a logic block denotes user generated FORTRAN
  - M in first column of a logic block denotes user generated MORTRAN
    - MORTRAN is an extension of FORTRAN where the user may reference SINDA specific elements and the pre-processor will convert them to their FORTRAN counterparts – more on this later…
  - Any character (besides 0) in column 6 of a logic block indicates that this line is a continuation of the previous line
  - All non HEADER, non-directive input should be past the 6th column.
- Syntax specific to Node, Conductor, and Source Data Blocks
  - FAC cards can be used to make units consistent. Just remember to have a FAC card that reset other data back to original units.
    - FAC cards are reset at the start of the HEADER block

How is the SINDA Deck structured?

- The following HEADERs are available:
  - **HEADER OPTIONS DATA** — Titles, I/O file names, options
  - **HEADER NODE DATA,smn** — Node descriptions
  - **HEADER CONDUCTOR DATA,smn** — Conductor descriptions
  - **HEADER SOURCE DATA,smn** — Nodal heat source descriptions
  - **HEADER CONTROL DATA,global or smn** — Execution control "constants"-all smn's
  - **HEADER REGISTER DATA** — User variables, all smns
  - **HEADER USER DATA,global** — User variables, all smns
  - **HEADER USER DATA,smn** — User variables (numbers) one smn
  - **HEADER ARRAY DATA,smn** — User arrays
  - **HEADER CARRAY DATA,smn** — User character arrays (strings)
  - *HEADER OPERATIONS DATA* — Analysis sequence (main driver)
  - *HEADER OUTPUT CALLS,smn* — Output operations and logic
  - *HEADER VARIABLES 0,smn* — User logic: time-dependence
  - *HEADER VARIABLES 1,smn* — User logic: temperature-dependence
  - *HEADER VARIABLES 2,smn* — User logic: wrap-up
  - *HEADER SUBROUTINE DATA* — Additional user written subroutines

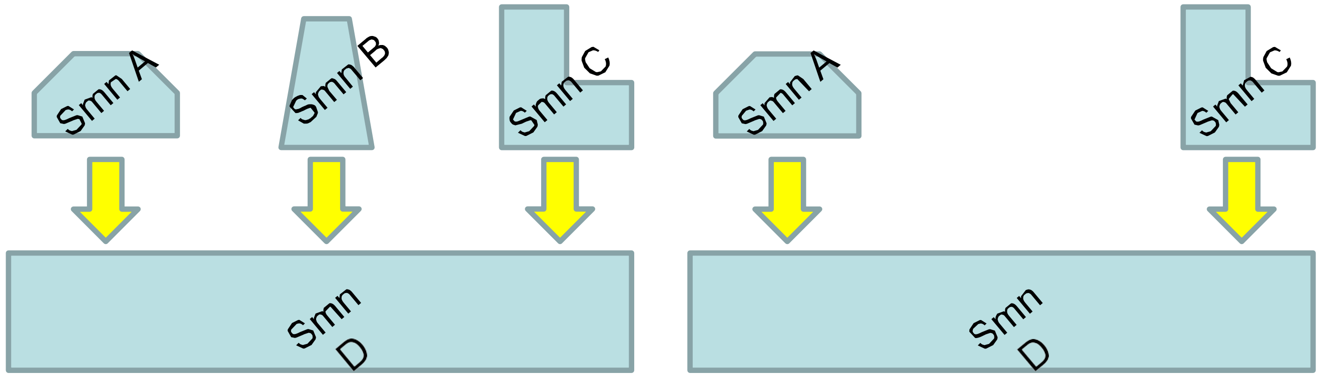
- **Bold** indicates Data block, *Italics* are Logic Blocks, Red covered in Basic SINDA, all others covered in Intermediate SINDA, Headers with ",smn" are related to a specific submodel

Executable Program Flow

- **HEADER OPERATIONS DATA** is the entry point. This is where everything starts during execution…
- Solution routines run through various logics block in the following order…
  - **VARIABLES 0** – process "Time" dependent logic – (one pass per timestep)
  - **VARIABLES 1** – process "Temperature" dependent logic
  - **Internally solve for temperatures**
    - If Converged, proceed to **VARIABLES 2**
    - If not Converged, LOOPCT < MaxLoops, and Steady Solution: go back through **VARIABLES 1**
    - If not Converged, LOOPCT < MaxLoops,NVARB1 = 0 and Transient Solution: go back through **Internally Solve for Temperatures**
    - If not Converged, LOOPCT < MaxLoops,NVARB1 = 1 and Transient Solution: go back through **VARIABLES 1**
    - If not Converged and LOOPCT >= MaxLoops, proceed to **VARIABLES 2**
  - **VARIABLES 2** – process post-solution data (e.g. heater on time)
  - If Transient, proceed next to timestep and go back to **VARIABLES 1**
  - If Steady or Transient and completed last timestep, return to **OPERATIONS**
  - *At the start, every OUTPUT increments, and before returning to OPERATIONS, go through OUTPUT CALLS, execute those instructions, and return*

What is a Submodel?

- Submodels are a way of logically grouping things together
- Names limited to 32 characters and may include Alphanumerics as well as an underscore (but may not begin with underscore)
- Allows node/conductor/array/numbering to not need to be predetermined for subsystems
  - Software that does not include submodels must be careful to avoid two organizations using the same ranges or conflicts can arise
- Included via a BUILD command which defines which submodels are solved
  - Submodels that are connected but not built are treated as boundary conditions
  - Submodels may not contain only logic
  - Can have multiple BUILDs



A few more things before diving into the HEADERs…

- INSERT/INCLUDE allows text from other files to be brought into model
  - INSERT is better as it can be nested
  - INCLUDE is allowed but has some issues with changing HEADERs therein
- PSTOP/PSTART – stops/starts writing line to pre-processor output
  - Useful for reducing pre-processor file size
- FSTOP/FSTART – changes default line type (FORTRAN/MORTRAN)
  - Doing nothing, MORTRAN is assumed
- DEFMOD – Default Model – allows local reference to SINDA created variables
  - For each node (#), there is a T#,Q,# and C# variable created for Temperature, Heat, and Capacitance respectively
  - For each conductor (#),there is a G# created for conductance
  - For each array (#), there is an A# and an NA# created for REAL array entries and INTEGER array entries (Co-located in memory using FORTRAN EQUIVALENCE)
  - For each user data (#),there is an XK# and a K# for REAL variables and INTEGER variables
  - Referencing any of these in an M-type statement needs an implied submodel; DEFMOD defines what this submodel is
  - Logic blocks with a submodel assignment imply that submodel as the DEFMOD

The First HEADER: OPTIONS DATA

- This HEADER must come first
  - Any others may appear in any order, but a good logical flow is all NODE HEADERS followed by all CONDUCTOR,ARRAY,VARIABLES 0,1,2 and ending with OUTPUT CALLS and SUBROUTINES
  - OPERATIONS is best immediately after OPTIONS DATA or just before first VARIABLES 0
- The block includes keywords to set up the model including
  - TITLE : description of problem
  - OUTPUT : name of ASCII output file
  - SAVE : name of Binary .sav file
  - USER1,USER2 – pre-specified user files
  - NAMES8 : allows NODE/CONDUCTOR numbers > 1E6
  - DOUBLEPRECISION: turns on DOUBLE PRECISION solution (64 bit)
  - MIXARRAY – allows both INTEGER and REAL entries in arrays

```
HEADER OPTIONS
  OUTPUT = CB90_Normal_Ops.out
  SAVE = CB90_Normal_Ops.sav
  DOUBLEPRECISION
  MLINE = 100000 $ Limits headers output
  MIXARRAY
  NAMES8
  TITLE GPM_Obs_CDR_v6.1c_2010.dwg - CB90_Normal_Ops
```

HEADER NODE DATA, submodel

- Basic Format
  - ***N#, Tinit, Capacitance***
  - If N# > 0 and Capacitance > 0,then node is DIFFUSION: has mass
  - If N# > 0 and Capacitance < 0,then node is ARITHMETIC: massless
  - If N# < 0 and Capacitance > 0,then node is BOUNDARY: infinite mass
  - If N# < 0 and Capacitance < 0,then node is HEATER: infinite mass
  - HEATER and BOUNDARY nodes are the same to the network, but SINDA provides some routines to retrieve information from HEATER nodes. Therefore, always use HEATER nodes instead of BOUNDARY nodes
  - Capacitance is Mass * Specific Heat
- Temperatures for nodes are calculated by SINDA. Models must have at least one boundary/heater node.
- Some special options exist for generating multiple nodes or temperature dependent capacitances…this will be covered later

```
HEADER NODE DATA,MYSUBMODEL
   101,70.0,2.0 $ Diffusion Node
   102,70.0,-2.0 $ Arithmetic
  -103,70.0,2.0 $ Boundary
  -104,70.0,-2.0 $ Heater
```

HEADER CONDUCTOR DATA, submodel

- Basic Format
  - ***G#, Ni, Nj, Conductance***
  - If G# > 0 then conductor is linear: $Q = G * (T_i - T_j)$
  - If G# < 0,then conductor is radiative: $Q = SIGMA * G * (T_i^4 - T_j^4)$
  - If Ni and Nj are in the same submodel as defined by the HEADER, then only the node numbers need be input
  - If Ni or Nj are not in the same submodel as defined by the HEADER, then the full node qualification must be specified SUBMODEL.N
- Conductances are used by SINDA to calculate heat flows between nodes which will satisfy the energy balance requirements
- Some special options exits for generating multiple conductor or temperature dependent conductors…this will be covered later

```
HEADER CONDUCTOR DATA, MYSUBMODEL
    101,101,102,10.0  $ Linear, within submodel
    102,102,OTHERSUB.1,10.0 $ Linear, between submodels
   -103,101,SPACE.99999,0.01 $ Radiative, between submodels
    104,101,103,-2.0  $ Linear (note negative!), likely FE generated
    105,101,-FLUID.1,10.0 $ Linear, one way (modeling fluid flow)
```

Conductance Terms

- Conductance couplings are linear and typically equal to $k\, A_x / L$
- Interface couplings are linear and typically equal to $A_{footprint}\, h_{I/F}$
- Convection couplings are linear and typically equal to $A_{surface}\, h_{conv}$
- Radiation couplings are non-linear and typically equal to $\sigma\, A_{Surface}\, \varepsilon\, \mathcal{F}$
- Where:

  k - thermal conductivity

  A_x - cross sectional area

  L - path length

  $A_{footprint}$ - Contact Area

  $A_{surface}$ - Surface Area

  $h_{I/F}$ - Interface conductance

  h_{conv} - convection coefficient

  σ - stephan-boltzman constant

  ε – IR emissivity

  $\mathcal{F}$ - (Script F) view factor – often computed by ray trace computer codes

HEADER SOURCE DATA, submodel

LEGACY INFO

- Basic Format
  - ***N#, Source***
  - Q values for each node are initialized to SOURCE data values at the beginning of VARIABLES 0
  - Numerous functions may over write these Q values if you are not careful
  - ***Recommend to avoid using SOURCE DATA, apply heat loads in VARIABLES0 or VARIABLES1 instead***
- Sources are used by SINDA to drive heat flows between nodes and towards a boundary to satisfy the energy balance requirements
- Some special options exits for generating multiple sources or temperature dependent capacitances…this will be covered later

```
HEADER SOURCE DATA,MYSUBMODEL
101,70.0 $ 70 W heat load
```

HEADER CONTROL DATA, submodel

- Basic Format
  - ***ControlVariable* = *Value***
- HEADER CONTROL DATA,GLOBAL which sets submodel independent variables as the default for all submodels dependent versions
  - SIGMA: Stefan-Boltzman Constant – *default = 1.0*
  - ABSZRO: offset to absolute zero (subtracted from actual temps) – *default =0*
  - NLOOPS: maximum number of iterations for steady state solutions
  - TIMEO: start problem time
  - TIMEND: End problem time
- Other *ControlVariables* may be specific to a submodel, including:
  - NLOOPT: maximum number of iterations for transient solutions
  - DRLXCA/ARLXCA: maximum allowable temperature difference between iterations for Diffusion/Arithmetic nodes
  - DTMPCA/ATMPCA: maximum allowable temperature difference between timesteps for Diffusion/Arithmetic nodes
  - EBALSA: maximum allowable system energy balance (%)
  - EBALNA: maximum allowable nodal energy balance (W) – Steady State only
  - FBEBALA: maximum allowable nodal energy balance (W) – Transient only
  - MATMET: control solution approach (Iterative, Matrix, AMG-CG)
  - DTIMEI: zero allows SINDA to calculate timestep, otherwise dt = DTIMEI
  - OUTPUT: frequency of going to OUTPUT CALLS in Transient

HEADER OPERATIONS DATA

- **This is where the program begins…**

- First thing that needs to be done is to BUILD the model by specifying which submodels should be included in the solution
  - BUILD ALL will include everything
  - Otherwise: BUILD ModelName,Submodel1,Submodel2,…SubmodelN

- Next, open any additional files for output and initialize values if needed

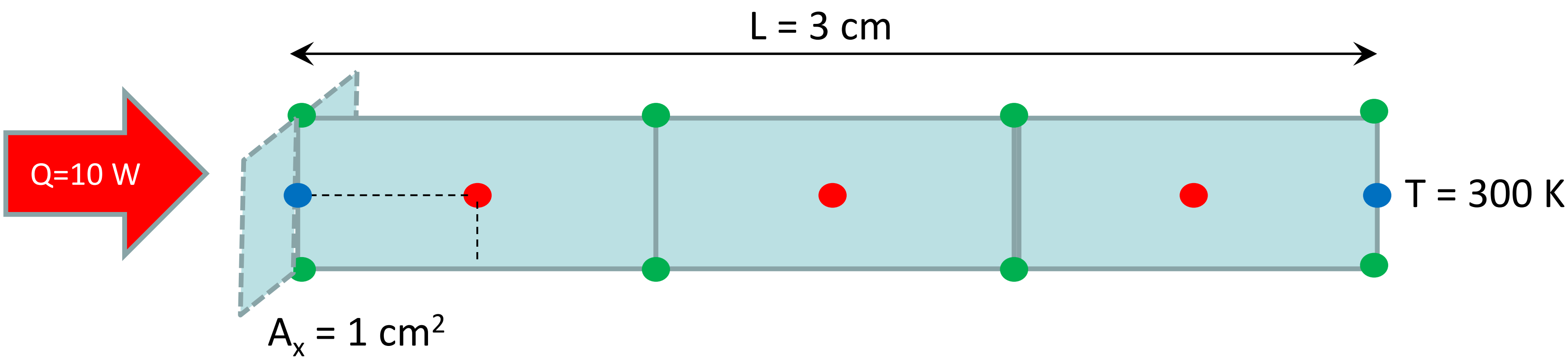
- Then, statements need to be included to request a solution
  - STDSTL or STEADY – calls routine to solve for steady state temperatures (mass is neglected)
  - FWDBCK or TRANSIENT – calls routine to solve for transient temperatures starting from TIMEO and ending at TIMEND

- Lastly, write any output to files needed at the end of the run
  - Don"t forget to close any additional files if opened earlier

HEADER OUTPUT CALLS, submodel

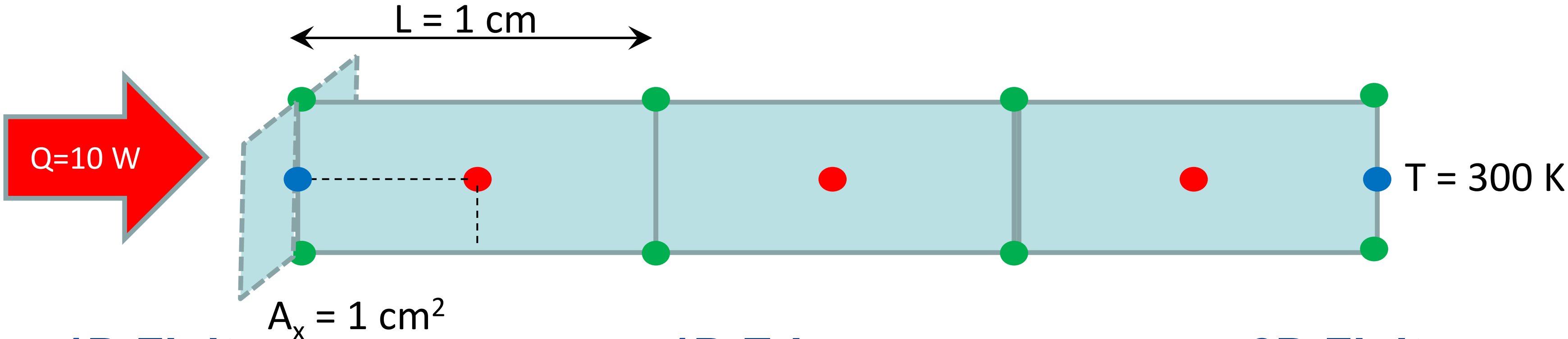
- Numerous canned routine for output
- Output to ASCII file:
  - CALL TPRINT(SubmodelName) – Outputs Temperatures
    - Submodel name must be in ‘ ‘, may use ‘ALL’
  - CALL QPRINT(SubmodelName) – Output Heat Loads
  - CALL CPRINT(SubmodelName); Output Nodal Capacitances
  - CALL GPRINT(SubmodelName); Output Conductances (File size may get large)
  - CALL HNQPNT(SubmodelName); Output heat added or removed from heater node to maintain temperature – no identifying header – DO NOT USE
  - Instead use CALL HNQCAL(SubmodelName) followed by QPRINT
  - CALL NODMAP* – Heat map for specified node (File size may get large)
  - CALL QMAP* – Heat map for all nodes (File size may get huge)
  - CALL SUBMAP – Heat flow between submodels (File size may get large)
- Output to Binary Save File:
  - Call SAVE(*args*,0) where *args* is „ALL" or a string combination of T, Q, G, C, N, or R for Temperatures, Heat, Conductances, Capacitances, Control Constants, or Registers respectively
  - Might go to CSR folder instead of a sav file if USECSR has been called

SINDA Model Example

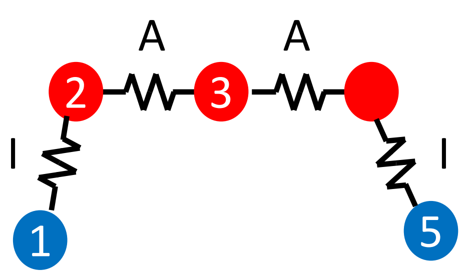


- As our first example let's take a solid bar with a constant cross-sectional area of 1 sq cm and a length of 3 cm made of a material whose thermal conductivity is 1 W/cm K. One end of the bar is held to 300 K, while 10 W of heat is applied to the opposite end. It is desired to know the temperature distribution along the bar…
- Consider both the Edge Node and Centroid node approach
  - Note for the centroid approach two nodes need to be added for the edges
- The closed form solution shows a linear distribution from 330 K to 300 K
  - Green nodes would be 330 K, 320 K, 310 K, and 300 K from left to right
  - Blue nodes would be 330 K left and 300 K right
  - Red nodes should be 325 K, 315 K, and 305 K from left to right

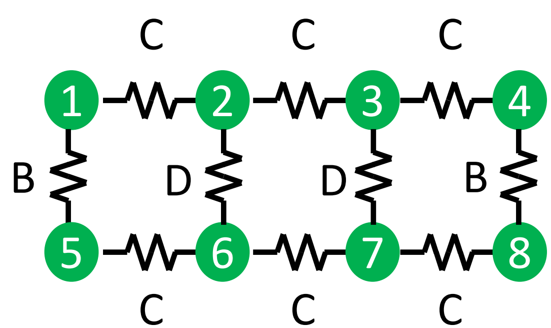
SINDA Model Example



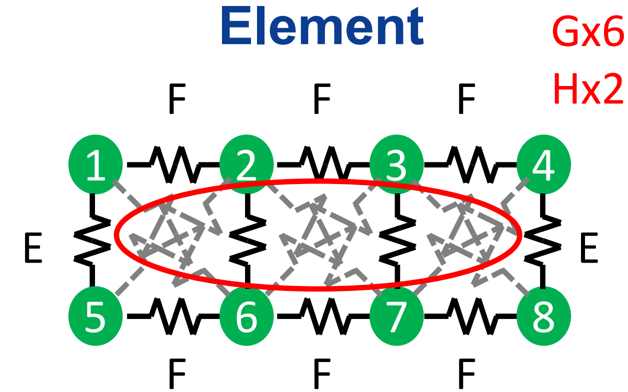
1D Finite Difference



1D Edge Nodes



2D Finite Element



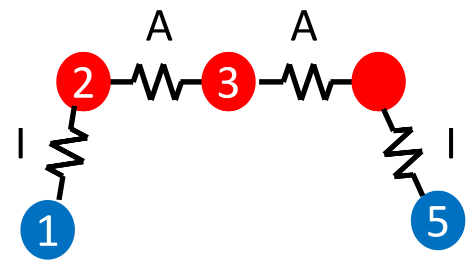
Where k = L = A_x = 1:

A=1, B=0.5, C=0.5, D=1, E= 0.16667, F= 0.16667, G=0.33333, H=0.33333, I = 2

The SINDA deck is shown on the next slide

SINDA Model Example – Input Deck (Centroid)

```
HEADER OPTIONS DATA
TITLE CENTROID CASE
     OUTPUT = CENTROID.out
     MODEL = CENTROID
HEADER CONTROL DATA, GLOBAL
     ABSZRO = 0.0
     ARLXCA = 0.001
     DRLXCA = 0.001
     NLOOPS = 500
HEADER NODE DATA, BAR
     1, 293.15, 1.0  $ Heated end of bar
     2, 293.15, 1.0
     3, 293.15, 1.0
     4, 293.15, 1.0
    -5, 300.0, -1.0 $ Boundary End of bar
HEADER CONDUCTOR DATA, BAR
     1, 1, 2, 2.0 $ Heated end, half node
     2, 2, 3, 1.0
     3, 3, 4, 1.0
     4, 4, 5, 2.0 $ Boundary end, half node
HEADER SOURCE DATA, BAR
     1, 10.0 $ 10 W load
HEADER OPERATIONS DATA,
BUILD ALL
M     CALL STDSTL
HEADER OUTPUT CALLS, BAR
M     CALL TPRINT('ALL')
END OF DATA
```



Where k = L = A_x = 1:

A=1

B=0.5

C=0.5

D=1

E= 0.16667

F= 0.16667

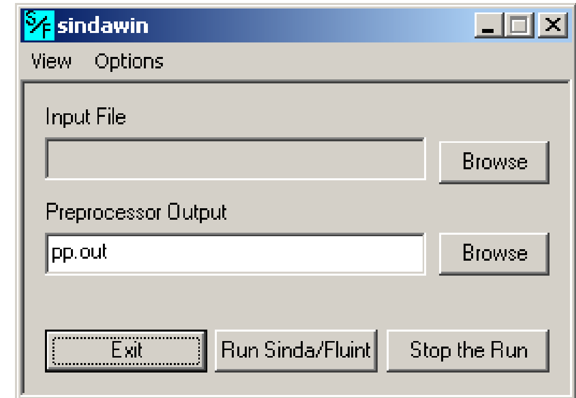
G=0.33333

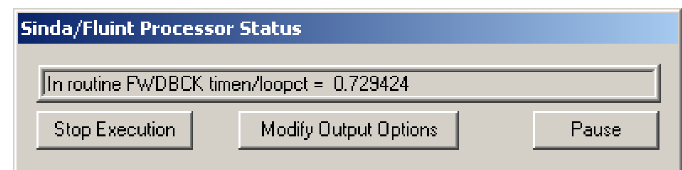
H=0.33333

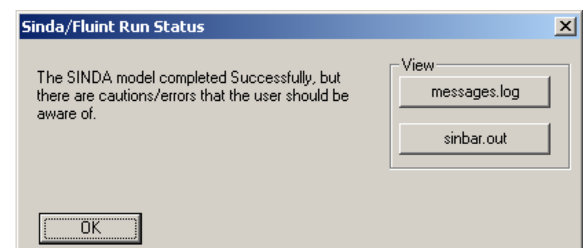
I = 2

Running SINDA

- Create the SINDA deck in your favorite text editor (TextPad, UltraEdit, WordPad, Notepad, etc)
- Find Run SINDA/FLUINT in your Start Menu
  - Typically: Start – Programs – SindaFluint – Run SINDA FLUINT
- Select Browse beside the Input file and navigate to your file location and select it
- Then click "Run Sinda/Fluint"
  - You should see it got through Preprocess, Compile/Link, the Processor
  - A window, similar to the one to the right, should pop up to indicate progress
  - If it does not, then look at the pp.out (or whatever you specified as the filename) for any preprocessor errors.
  - If no Pre-Processor errors, check messages.txt or messages.log for any compile or link errors







SINDA Model Example - Output

```
SYSTEMS IMPROVED NUMERICAL DIFFERENCING ANALYZER WITH FLUID INTEGRATOR           PAGE     2

MODEL = CENTROID                                                                     CENTROID CASE   STDSTL

 SUBMODEL NAME = BAR

                                          CALCULATED                                  ALLOWED
        MAX DIFF DELTA T PER ITER         DRLXCC(                      0)=   1.00100      VS. DRLXCA= 1.000000E-03
        MAX ARITH DELTA T PER ITER        ARLXCC(                      0)=   1.00100      VS. ARLXCA= 1.000000E-03
        FRACTIONAL SYSTEM LEVEL ENERGY IMBALANCE =   0.00000       VS. EBALSA= 1.000000E-02
        ENERGY INTO AND OUT OF SUB        ESUMIS                     =   0.00000          ESUMOS=   0.00000
        MAX NODAL ENERGY BALANCE          EBALNC(                      0)=   0.00000      VS. EBALNA=   0.00000
        NUMBER OF ITERATIONS              LOOPCT                     =            0       VS. NLOOPS=            500
        PROBLEM TIME                      TIMEN                      =   0.00000      VS. TIMEND=   0.00000

                                        DIFFUSION NODES IN INPUT NODE NUMBER ORDER
  T        1=   293.15      T        2=  293.15      T        3=   293.15      T        4=   293.15
                                        ARITHMETIC NODES IN INPUT NODE NUMBER ORDER
                                                           ++NONE++
                                        BOUNDARY NODES IN INPUT NODE NUMBER ORDER
  T        5=   300.00
```

NOTE: Initial Condition

SINDA Model Example - Output

```
 SYSTEMS IMPROVED NUMERICAL DIFFERENCING ANALYZER WITH FLUID INTEGRATOR          PAGE     3

 MODEL = CENTROID                                                                    CENTROID CASE
STDSTL


  SUBMODEL NAME = BAR

         CONVERGENCE STATUS: SUBMODEL CONVERGED

                                     CALCULATED                                           ALLOWED
         MAX DIFF DELTA T PER ITER      DRLXCC(BAR                       4)=   0.00000      VS. DRLXCA= 1.000000E-03
         MAX ARITH DELTA T PER ITER     ARLXCC(                          0)=   0.00000      VS. ARLXCA= 1.000000E-03
         FRACTIONAL SYSTEM LEVEL ENERGY IMBALANCE =   0.00000     VS. EBALSA= 1.000000E-02
         ENERGY INTO AND OUT OF SUB     ESUMIS                          =  10.0000          ESUMOS=  10.0000
         MAX NODAL ENERGY BALANCE       EBALNC(BAR                       4)=   0.00000      VS. EBALNA=  0.00000
         NUMBER OF ITERATIONS           LOOPCT                          =            2      VS. NLOOPS=          500
         PROBLEM TIME                   TIMEN                           =  0.00000      VS. TIMEND=  0.00000

                                      DIFFUSION NODES IN INPUT NODE NUMBER ORDER
   T       1=  330.00      T       2=  325.00      T       3=  315.00      T       4=  305.00
                                      ARITHMETIC NODES IN INPUT NODE NUMBER ORDER
                                                           ++NONE++
                                      BOUNDARY NODES IN INPUT NODE NUMBER ORDER
   T       5=  300.00
```

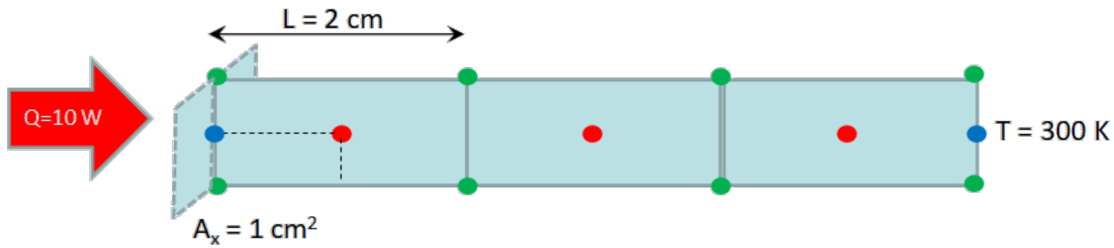
NOTE: Final Solution

Results as expected !

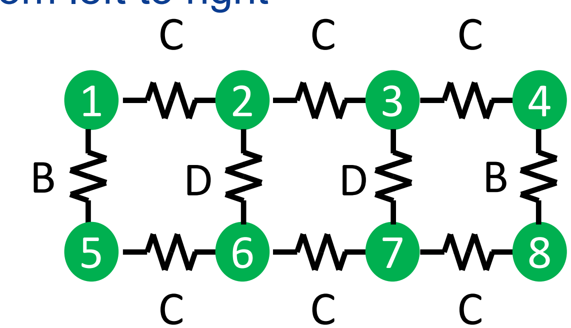
SINDA Model Example – Input Deck (Edge Node)

```
...
HEADER NODE DATA, BAR
     1, 293.15, 1.0  $ Heated end of bar
     2, 293.15, 1.0
     3, 293.15, 1.0
    -4, 300.0, 1.0 $ Boundary End of bar
     5, 293.15, 1.0  $ Heated end of bar
     6, 293.15, 1.0
     7, 293.15, 1.0
    -8, 300.0, -1.0 $ Boundary End of bar
HEADER CONDUCTOR DATA, BAR
     1, 1, 2, 0.25
     2, 2, 3, 0.25
     3, 3, 4, 0.25
     4, 5, 6, 0.25
     5, 6, 7, 0.25
     6, 7, 8, 0.25
     7, 1, 5, 1.0
     8, 2, 6, 2.0
     9, 3, 7, 2.0
     10, 4, 8, 1.0
HEADER SOURCE DATA, BAR
     1, 5.0 $ 10 W load (half)
     5, 5.0 $ 10 W load (half)
...
```

- Now let's investigate a bar twice as long (6 cm) but with an Edge Node Formulation



- The closed form solution still shows a linear distribution from 360 K to 300 K
  - Green nodes would be 360 K, 340 K, 320 K, and 300 K from left to right



Where k = A_x = 1 and L = 2:
B=1, C=0.25, D=2

SINDA Model Example – Output (Edge Node)

```
MODEL = EDGENODE                                                                                   EDGENODE CASE
  STDSTL


 SUBMODEL NAME = BAR

        CONVERGENCE STATUS: SUBMODEL CONVERGED

                                  CALCULATED                                     ALLOWED
        MAX DIFF DELTA T PER ITER     DRLXCC(BAR                  7)=  0.00000      VS. DRLXCA= 1.000000E-03
        MAX ARITH DELTA T PER ITER    ARLXCC(                     0)=  0.00000      VS. ARLXCA= 1.000000E-03
        FRACTIONAL SYSTEM LEVEL ENERGY IMBALANCE =  0.00000      VS. EBALSA= 1.000000E-02
        ENERGY INTO AND OUT OF SUB    ESUMIS                    =  10.0000          ESUMOS=  10.0000
        MAX NODAL ENERGY BALANCE      EBALNC(BAR                  7)=  0.00000      VS. EBALNA=  0.00000
        NUMBER OF ITERATIONS          LOOPCT                    =           2       VS. NLOOPS=        500
        PROBLEM TIME                  TIMEN                     =  0.00000      VS. TIMEND=  0.00000
                                        DIFFUSION NODES IN INPUT NODE NUMBER ORDER
  T        1=  360.00      T        2=  340.00      T        3=  320.00      T        5=  360.00      T        6=  340.00      T        7=  320.00
                                        ARITHMETIC NODES IN INPUT NODE NUMBER ORDER
                                                          ++NONE++
                                        BOUNDARY NODES IN INPUT NODE NUMBER ORDER
  T        4=  300.00      T        8=  300.00
```

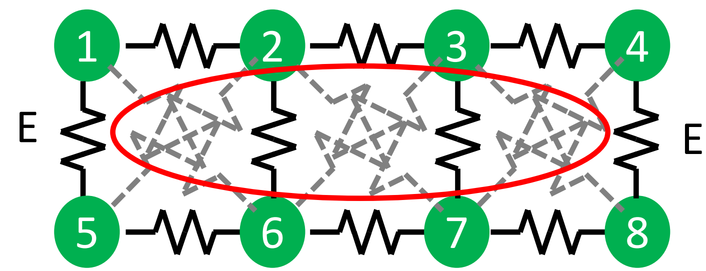
Results as expected !

But what about the Finite Element formulation, with the negative conductance?

SINDA Model Example – Finite Element

```
...
HEADER CONDUCTOR DATA, BAR
     1, 1, 2, -0.166667
     2, 2, 3, -0.166667
     3, 3, 4, -0.166667
     4, 5, 6, -0.166667
     5, 6, 7, -0.166667
     6, 7, 8, -0.166667
     7, 1, 5, 0.583333
     8, 2, 6, 1.166667
     9, 3, 7, 1.166667
     10, 4, 8, 0.583333
     11, 1, 6, 0.4166667
     12, 2, 7, 0.4166667
     13, 3, 8, 0.4166667
     14, 2, 5, 0.4166667
     15, 3, 6, 0.4166667
     16, 4, 7, 0.4166667
...
```

Where k = A_x = 1 and L = 2:
E=0.58333
F= *-0.16667*
G=0.41667
H=1.16667



Note the 6 conductors with a negative value…
While this may seem illegal, the FE formulation can result in negative conductors. It just means that the heat flow cannot be represented as a simple Q=GΔT

```
DIFFUSION NODES IN INPUT NODE NUMBER ORDER
  T        1=  360.00      T        2=  340.00      T        3=  320.00      T        5=  360.00      T        6=  340.00      T        7=  320.00
                                        ARITHMETIC NODES IN INPUT NODE NUMBER ORDER
                                                          ++NONE++
                                        BOUNDARY NODES IN INPUT NODE NUMBER ORDER
  T        4=  300.00      T        8=  300.00
```

Results still as expected !

Consider an energy flow into node 2 (5 W expected):

(-0.1667)*(360-340) + (0.41667)*(360-340) + 1.16667*(340-340) =(0.25)*20 **= <u>5 !</u>**

<u>So, energy balances can still be checked, just not at an individual conductor level</u>

Intermediate SINDA Outline

- More than a basic Node or Conductor
- Registers, User Data Global, and User Data Local
- Defining Array Data
- Logic Blocks
- Basic FORTRAN for simple tasks

SINDA File Structure Review

- The following HEADERs are available:

– **HEADER OPTIONS DATA**	Titles, I/O file names, options
– **HEADER NODE DATA,smn**	Node descriptions
– **HEADER CONDUCTOR DATA,smn**	Conductor descriptions
– **HEADER SOURCE DATA,smn**	Nodal heat source descriptions
– **HEADER CONTROL DATA,global or smn**	Execution control "constants"-all smn's
– **HEADER REGISTER DATA**	User variables, all smns
– **HEADER USER DATA,global**	User variables, all smns
– **HEADER USER DATA,smn**	User variables (numbers) one smn
– **HEADER ARRAY DATA,smn**	User arrays
– **HEADER CARRAY DATA,smn**	User character arrays (strings)
– *HEADER OPERATIONS DATA*	Analysis sequence (main driver)
– *HEADER OUTPUT CALLS,smn*	Output operations and logic
– *HEADER VARIABLES 0,smn*	User logic: time-dependence
– *HEADER VARIABLES 1,smn*	User logic: temperature-dependence
– *HEADER VARIABLES 2,smn*	User logic: wrap-up
– *HEADER SUBROUTINE DATA*	Additional user written subroutines

- **Bold** indicates Data block, *Italics* are Logic Blocks, Red covered in Intermediate SINDA, all others covered in Basic SINDA, Headers with ",smn" are related to a specific submodel

More than a Basic Node or Conductor

- Certain 3 character codes can be used to make specialized nodes or conductors
- GEN allows multiple nodes or conductors to be created
  - `GEN 1,3,2,70.0,13.0` would generate 3 Nodes (1,3,and 5) all with initial temperatures of 70.0 and Capacitances of 13.0
  - `GEN 1,5,2,1,3,2,4,10.0` would generate Conductors 1,3,5,7,and 9 connecting [1,2],[4,6],[7,10],[10,14],and [13,18] all with Conductances of 10.0
- SIV defines nodes or conductors as temperature varying
  - `SIV 1,70.0,A5,10.0` would generate node 1 with an initial temperature of 70.0 and a capacitance that is determined by the (T,C) entries in Array 5 multiplied by 10.0
  - `SIV 3,1,3,A6,5.0` would generate conductor 3 connecting [1,3] with a conductance determined by Avg(T1,T3) and the (T,G) values in Array 6 multiplied by 5.0
- Options also exist for SIM (multiple, temperature dependent conductors) DIV (temperature dependent based on two materials),DIM (multiple temperature dependent based on two materials)
- In addition to SIV, SIM, DIV, and DIM there are also SPV, SPM, DPV, and DPM which treats the array entries as a polynomial instead of a lookup

HEADER REGISTER DATA

- REGISTERs are the preferred method for parameterizing a model
  - Limited to 32 characters and by default are REALs, may include underscore
  - May be declared as INTEGERs or DOUBLEs if needed
  - Often compared to spreadsheet values, in that their inter-dependence propagates automatically within model
    - *BoxThickness = 0.1*
    - *BoxDensity = 100.0*
    - *BoxMass = BoxDensity * BoxThickness*
    - *Changing BoxThickness or BoxDensity will automatically change BoxMass*
  - May be used in Data blocks (such as NODE or CONDUCTOR DATA)
    - *NODE DATA Line: GEN 1,3,1,70.0,1770.0 * BoxThickness*
    - *CONDUCTOR DATA Line: 15,BOX.1,BASEPLATE.1,BoxThickness * 267.0 * 0.01 / 0.1*
  - However, personally, I have not found this "spreadsheet like" feature to be all that useful when making adjustments in VARIABLEs blocks. In essence, it works well for data blocks, but not as well for logic blocks (in my opinion)

```
HEADER REGISTER DATA
   BoxThk = 1.0
   BoxDens = 100.0
   INT:NoBoxes = 5    $ Integer Register
   DP:BoxSurfArea = 25.0    $ Double Precision Register
   BoxMass = BoxThk*BoxSurfArea * BoxDens    $ Calculated Register
```

HEADER USER DATA, GLOBAL

LEGACY INFO

- USER DATA,GLOBAL is an alternate way to parameterize a model with named variables
  - Largely replaced by REGISTERs now, legacy feature
  - **MUST** come before any submodel specific USER DATA blocks
  - Follows FORTRAN implicit rules for naming (I-N are integers, all others REALs)
  - Variables only exist in FORTRAN based logic blocks
    - *OPERATIONS,VARIABLES 0,1,2,OUTPUT CALLS,SUBROUTINE*
  - Cannot be used in Data Blocks
    - *However, the user may set node capacitances or conductor values in logic blocks and use named variables there*

```
HEADER USER DATA,GLOBAL
     BoxThk = 1.0       $ Real Value
     BoxDens = 100.0    $ Real Value
     NoBoxes = 5        $ Integer value (based on N)
     BoxSurfArea = 0.0  $ Real value, calculated later
     BoxMass = 0.0      $ Real value, calculated later
HEADER OPERATIONS DATA
M       BoxMass = BoxThk * BoxSurfArea * BoxDens $ Calculated Value
M       MYSUBMODEL.C101 = BoxMAss * 980.0        $ Assign mCp to MYSUBMODEL.101
```

HEADER USER DATA, submodel

- USER DATA, submodel is the last method of creating globally accessible variables
- Basic Format
  - ***N#, Value***
  - Number can be referenced as either REAL or INTEGER in logic blocks
  - XK# in Logic is REAL value
  - K# is INTEGER value
  - Be careful mixing the two as the actual storage locations in memory are equivalenced (more on this in the advanced SINDA)
  - Often used as Heater On/Off flags

```
HEADER USER DATA,MYSUBMODEL
        1,10.0    $ Real Value
      101,1
      102,0
HEADER OPERATIONS DATA
M      MYSUBMODEL.Q101 = 5.0 * MYSUBMODEL.K102
M      MYSUBMODEL.C101 = XK1 * 950.0
M      MYSUBMODEL.T101 = XK101 $ This is very dangerous!!! It was assigned as
  an integer value and now is being references as a REAL.  No type conversion
  checks are made
```

HEADER ARRAY DATA, submodel

- ARRAY DATA is where independent/dependent data may be entered, such as Conductivity as a function of Temperature or Flux as a function of Time
- Basic Format
  - ***A#,Value1, Value2, Value3…., ValueN, END <u>OR</u>***
  - ***A# = Value1, Value2, Value3…., ValueN***
  - But choose one and only one format, they cannot be mixed!!
- Three types of arrays are commonly entered
  - Singlet: one variable (e.g. all Times or all fluxes)
  - Doublet: pairs of (Independent, Dependent) variables (e.g. Temperature, Density)
  - Bivariate: groups of (Independent1,Independent2,and Dependent) variables (e.g. Temperature, InputHeat, Power)
- Entries must be all of the same type (REAL or INTEGER) unless MIXARRAY was specified in OPTIONS DATA
- Array entries are often strictly enforced to be monotonically increasing
- Arrays may span multiple lines, but if so, ***<u>do not</u>*** include a comma at the end of the line

```
HEADER ARRAY DATA,MYSUBMODEL
   5 = 1.0,2.0,3.0,4.0
       5.0,6.0,7.0,8.0
```

Examples: HEADER ARRAY DATA,submodel

LEGACY INFO

- Single Array Example

```
    5 = 1.0,2.0,3.0,4.0
        5.0,60.,7.0,8.0
    6 = 1,2,3,4
        5,6,7,8
```

- Double Array Example

```
   15 = 1.0,100.0
        2.0,110.0
        3.0,120.0
        4.0,130.0
```

- BiVariate Array Example – 3 in this case indicated how many entries there are for the first independent variable and should be an integer!!

```
25 = 3,0.0,20.0,50.0
     1.0,0.01,0.05,0.10
     2.0,0.03,0.10,0.20
     3.0,0.05,0.20,0.40
```

- CARRAY Example – allows strings to be entered for use in Logic Blocks

```
HEADER CARRAY DATA,MYSUBMODEL
    5 = This line describes something and is padded to fill up 1024 characters
    6 = This line describes something else
    7 = Any of these lines may be referenced by MYSUBMODEL.UCA5-7 in Logic
    8 = Often used to include descriptions of nodes that can be output in Logic
```

Logic Block Variables

- Logic blocks provide access to nearly everything in the SINDA environment
  - *Implicitly defined SINDA Variables*
    - Temperatures: **SUB1.T101** is temperature of node 101 in the SUB1 submodel
    - Heat Loads: **SUB2.Q101** is applied heat of node 101 in the SUB2 submodel
    - Capacitance: **SUB3.C101** is m*Cp of node 101 in the SUB3 submodel
    - Conductance: **SUB4.G101** is conductance of conductor 101 in the SUB4 submodel
    - User Data Submodel: **SUB5.XK202** is the REAL variable #202 in the SUB5 submodel
    - User Data Submodel: **SUB6.K1** is the INTEGER variable #1 in the SUB6 submodel
    - Array Data: **SUB7.A5** is Array number 5 in the SUB7 submodel
    - Array Data: **SUB7.NA6** is an INTEGER Array number 6 in the SUB7 submodel
    - Character Array Data: **SUB8.UCA5** is Character Array number 5 in SUB8 submodel
    - *If any of these variables do not have their submodel prefix, then it is assumed to be the submodel associated with the current HEADER or the submodel defined by DEFMOD*
  - *Explicitly defined SINDA Control Variables (e.g. DRLXCA, DTIMEI, etc)*
    - SINDA creates ITEST-NTEST as global INTEGER variables and ATEST-HTEST and OTEST-ZTEST as global REAL variables
  - *Explicitly defined SINDA Global Variables (REGISTER and USER DATA,GLOBAL)*
  - *Local FORTRAN variables defined by user*
  - *SINDA Function Library*
    - Hundreds of Functions and Subroutines available

HEADER VARIABLES 0|1|2, submodel

- Logic that is time dependent should be in VARIABLES 0
  - *Interpolation of environmental heat loads*
  - *Assignment of constant power dissipations*
  - *Assignment of time varying conductances (e.g. articulating arrays, deployments)*
- Logic that is temperature dependent should be in VARIABLES 1
  - *Heater simulation*
  - *Assignment of power dissipations that are temperature dependent*
  - *Assignment of conductance values that are temperature dependent*
  - *Assignment of capacitances that are temperature dependent*
- Logic for processing data after convergence should be in VARIABLES 2
  - *Heater Duty cycle/On time calculation*
  - *Check of current state*
  - *Predictor/Corrector*
- Logic for results output should be in OUTPUT CALLS
  - *Send results to OUTPUT or SAVE files*
  - *User specified output*

General Functions

- Interpolation/Extrapolation
  - *2 singlet arrays,1 double array*
  - *Piecewise linear, quadratic*
- Input
  - *Load initial temperatures*
- Output
  - *Results, Heat flows, Sink Temperatures*
- Actual to Relative Conversion
  - *Nodes, Conductors, Arrays, Constants, etc*
- State change
  - *Temporary hold as HEATER or BOUNDARY nodes*
- Specialized Applications
  - *Heater Simulation*
  - *Heatpipes*
  - *TECs*
  - *Phase Change*
  - *PID*
  - *Ablation*

Some Commonly Used Routines

LEGACY INFO

•Common Interpolation/Extrapolation Routines

```
CALL D1DEG1(TIMEN,A5,Q101)          $ A5 is a doublet array (Power vs. Time)
CALL D1D1DA(TIMEN,A6,A7,Q101)       $ A6 and A7 are singlet arrays (Time and Power respectively)
CALL D11MDA(TIMEN,A6,A7,2.0,Q101) $ Same as above but multiplied by 2

CALL D1DEG2(TIMEN,A5,Q101)          $ Like D1DEG1 but parabolic interpolation
CALL D1D2DA(TIMEN,A6,A7,Q101)       $ Like D1D1DA but parabolic interpolation
CALL DA11CY(TIMEN,PERIOD,A1,A2,Q101)      $ Cyclical interpolation (2 singlets)
CALL DA11MC(TIMEN,PERIOD,A1,A2,2.0,Q101) $ Cyclical interpolation (2 singlets) with multiplier
```

•Common Interpolation/Extrapolation Routines

•Common Input Routines

```
CALL RESTAR(NREC)                         $ NREC is the record number in the SAVE file from
which to retrieve
```

•Common Output Routines

```
CALL TPRINT('ALL') $ Print Temperatures for all Submodels to Output File
CALL QPRINT('ALL') $ Print Heat Loads for all Submodels to Output File
CALL CPRINT('ALL') $ Print Capacitances for all Submodels to Output File
CALL GPRINT('ALL') $ Print Conductances for all Submodels to Output File
CALL HNQCAL('ALL') $ Calculate Heater Node Power for all Submodels - Stored in Q variables
CALL NODMAP('MYSUBMODEL',101,0) $ Print Heat Map for node MYSUBMODEL.101 to Output file
CALL QMAP('MYSUBMODEL'),'DA',0) $ Print Heat Map for diffusion and arithmetic nodes in
                                          MYSUBMODEL to Output file
CALL SUBMAP                 $ Print Submodel to Submodel Heat Flows to Output file
CALL SAVE('TQGCR',0)        $ Write Temps, Heat Loads, Conds, Caps, and Registers to Save file
```

Some Commonly Used Routines

LEGACY INFO

- Common Actual to Relative Conversion Routines

```
CALL NODTRN('MYSUBMODEL',101,ITEST) $ Find index in FORTRAN T array for node MYSUBMODEL.101
                                       and return to ITEST
ITEST = INTNOD('MYSUBMODEL',101)    $ Same as above, but a function instead of a subroutine
CALL CONTRN('MYSUBMODEL',101,ITEST) $ Find index in FORTRAN G array for cond MYSUBMODEL.101
                                      and return to ITEST
ITEST = INTCON('MYSUBMODEL',101)    $ Same as above, but a function instead of a subroutine
CALL MODTRN('MYSUBMODEL',ITEST)     $ Find index in FORTRAN submodel array for submodel
                                      MYSUBMODEL and return to ITEST
ITEST = MDLTRN('MYSUBMODEL')        $ Same as above, but a function instead of a subroutine
CALL ARYTRN('MYSUBMODEL',101,ITEST) $ Find index in FORTRAN A array for array MYSUBMODEL.101
                                      and return to ITEST
```

More on Actual and Relative Numbering in Advanced SINDA

- Common State Change Routines

```
CALL HTRNOD('MYSUBMODEL',101)    $ Makes node SUBMODEL.101 a heater node (must have been
                                   defined as Diffusion or arithmetic originally)
CALL RELNOD('MYSUBMODEL',101)    $ Releases node SUBMODEL.101 back to its original type
CALL HTRMOD('MYSUBMODEL','AD')   $ Makes all Diffusion and Arithmetic nodes in submodel
                                   MYSUBMODEL into heater nodes
CALL RELMOD('MYSUBMODEL','A')    $ Releases all Arithmetic nodes in submodel MYSUBMODEL  back to
                                   their original type
```

Useful for temporarily suspending nodes or submodels from the solution while including them in a BUILD (for example unconnected nodes)

Some Commonly Used Routines

LEGACY INFO

- Common Application Routines
  - Heater Simulation

```
C Return 1.0 if T101 is above 5.0 (off),0.0 if below -5.0 (on) to XK101. Value is unchanged
C if between -5.0 and 5.0
M      CALL THRMST(T101,-5.0,5.0,XK101) $ Sets XK101 to 0.0 or 1.0 based on thermostat state
C
C Adds 10.0 W to Q101 if T101 would result in heater on. ZTEST and YTEST hold how long the
C heater has been on and how many times it switched. XK101 is 0.0 if on and 1.0 if off
M      CALL HEATER(T101,Q101,10.0,-5.0,5.0,ZTEST,YTEST,XK101)
C
C Adds heat proportional to temperature within band (10.0 W if T101 < -5.0,4.0 W if T =
C 1.0,  C 0.0W if T > 5.0). YTEST is how long the heater has been on. XK101 is the heat
C added
M      CALL PHEATER(T101,Q101,0.0,10.0,-5.0,5.0,YTEST,XK101)
```

  - Heatpipes: `HEATPIPE,HEATPIPE2,HPUNITS,HPGLOC`
  - Thermo Electric Coolers: `BISTEL,TECUNITS,TECINFO,TEC1,TEC2`
  - Phase Change (Solid to Liquid): `FUSION`
  - PID Control: `PIDSET,PIDSETWPIDINIT,PID,PIDS,PIDTAB,PIDGET`
  - Ablation: `ABLATESET,ABLATE,ABLATERATE`

See SINDA/FLUINT Manual for specifics on these functions

Basic FORTRAN

- FORTRAN 77 statements must start in column 7 or greater
  - Should not go beyond column 72
  - Character in column 6 indicates continuation of the previous line
  - Numbers in columns 1-5 reserved for optional line numbers
- Data types should be declared
  - Data types are implicitly defined by first character: I-N is INTEGER, all others REAL
  - INTEGER: 32 bit whole numbers, INTEBGER*8 for 64 bit
  - REAL: 32 bit floating point numbers, REAL*8 for 64 bit
  - CHARACTER: string values
- DIMENSION used to specify arrays of specified data type
  - Cannot dynamically allocate space though
- GOTO command allows jump to specified line number (bad practice)

Conditional Statements:

```
IF (Condition1) THEN
…
ELSEIF (Condition2) THEN
…
ELSE
…
ENDIF
```

Loop Statements:

```
DO ITEST = 1,10
 …
END DO

WHILE (ITEST.EQ.1)
 …
END WHILE
```

Conditions:

Not Equal:	.NE.
Equal:	.EQ.
Greater Than:	.GT.
Greater Than or Equal To:	.GE.
Less Than:	.LT.
Less Than or Equal To:	.LE.
Logical AND:	.AND.
Logical OR:	.OR.

Basic FORTRAN (I/O)

- FORTRAN allows the user to read and write data to and from files
- First a file must be opened for access
  - `OPEN(UNIT=Spec, FILE='Filename', STATUS='UNKNOWN')`
  - e.g. `OPEN(UNIT=123, FILE='../Heater.dc', STATUS='UNKNOWN')`
- The data can be read or written
  - `READ(Spec, FormatLine) Variable1,Variable2,…`
  - `WRITE(Spec, FormatLine) Variable1,Variable2,…`
  - e.g. `WRITE(123,*) 'T101',MYSUBMODEL.T101`
- FORMAT statements tell FORTRAN what to expect or how to write
  - `FORMAT(FmtSpec1,FmtSpec2,FmtSpec3,…)`
  - **F10.6**: decimal notation,10 characters wide,6 after decimal place (321.000000)
  - **E10.4**: scientific notation,10 characters wide,4 after decimal place (3.2100E+00)
  - **I3**: integer notation (321); **3X**: 3 spaces; **A**: string
- Lastly, the file should be closed when no longer needed
  - `CLOSE(UNIT=Spec)`

```
C234567
F      OPEN(UNIT=123, FILE='..\MyFile.txt', STATUS='UNKNOWN')
M      WRITE(123,10) 'Node 101 T:',T101
F      WRITE(123,10) 'Node 101 Q:',
M     1  Q101 $ Note that F and M type statements may be split across lines
F10    FORMAT(A,1X,F10.6)
F      CLOSE(UNIT=123)
```

Advanced SINDA

- How does SINDA solve for temperatures?
- The SOLVER – optimization within SINDA
- Relative vs. Actual Numbering and Equivalencing
- Extending SINDA with FORTRAN

How Does SINDA Solve for Temperatures?

- Depending on the MATMET control constant, SINDA will solve for temperatures either iteratively or through direct matrix inversion
- The iterative approach *may* be faster for certain classes of problems
  - Having a node coupled to two other nodes, one with a very large conductor and the other much smaller, may cause oscillations
  - Avoid “hard coupling” with very large conductor
    1. Iterative solves for Node i, holding all other nodes as boundary temperatures.
    2. Then moves on to Node i+1,continuously using the newer temperature where applicable.
    3. Once all nodes have been solved, this is one iteration.
    4. It then checks if all node temperature changes since the last iteration are within tolerance.
    5. If so, solution found. If not, continue process until convergence or max loops reached
- The matrix methods are certainly faster for FE based or larger models
  - Two algorithms available (MATMET = [1 or 2] or [11 or 12])
- Both require linearization of radiative terms
  - MATMET=1/2 uses more memory and is older approach (YSMP – Yale Sparse Matrix Package) – Gaussian Elimination
  - MATMET=11/12 uses less memory and is newer approach (AMGCG – Advanced Multi Grid Conjugate Gradient)
  - Set up [G]*[T] = [Q] matrix, invert [G] and multiply by [Q] to get [T]
  - [G] is often very sparse and may need to be regenerated due to linearization by [T]
- Which is faster depends on your model; don’t be afraid to play around

The SOLVER: Optimization within SINDA

- The SOLVER was introduced as an optimization scheme within the constructs of a SINDA model
- User must define the following:
  - Goal: singular value to be minimized or maximized (OBJECT)
  - Design Variables: what is allowed to be varied and by how much
  - Constraints: what other factors may be considered for a valid solution
- New HEADERs introduced for SOLVER
  - HEADER SOLVER DATA: control constants for SOLVER (e.g. GOAL)
  - HEADER PROCEDURE: define OBJECT, call Thermal Solution
    - OBJECT best to define as Root Sum Square, but other methods available
  - HEADER DESIGN DATA: LowLimit <= RegisterName <= HighLimit
  - HEADER CONSTRAINT DATA: LowLimit <= ConstraintName <= HighLimit
    - Constraint Name limited to 8 characters and may not be a Global User Data or Register. These represent alternative factors to consider a solution "good"
  - HEADER SOLOGIC0: Like Variables 0 within SOLVER
  - HEADER SOLOGIC1: Like Variables 1 within SOLVER
  - HEADER SOLOUTPUT: Like OUTPUT CALLS within SOLVER

The SOLVER: Sample Problem

- 0.5 W Detector heat strapped to Aluminum radiator
- Allowable Radiator Envelope: 0.35 m x 0.1 m
- Minimum Radiator Thickness: 0.0005 m
- Sink Temperature of 40 K, $\varepsilon = 0.85$
- Major GOAL : Minimize Radiator Gradient (70% importance)
- Sub-GOAL: Achieve -120°C on Detector (20% importance)
- Sub-GOAL: Keep mass low (10% importance)

The SOLVER: Sample – Basic SINDA Setup

```
HEADER OPTIONS DATA
TITLE Radiator Optimization
  OUTPUT = RadOpt.out
  MODEL = TEST
HEADER REGISTER DATA
  RadX = 0.1
  RadY = 0.1
  RadThk = 0.005
  kAl = 167.9
  densAl = 2770.0
  Mass = RadX*RadY*RadThk*densAl  $ As a Register, automatically updated with other changes
  TMax = 0.0
  TMin = 0.0
HEADER NODE DATA,MAIN
  GEN 1,100,1,25.0,-1.0       $ 10 x 10 radiator
    998,25.,-1.0       $ Detector
   -999,-233.,-1.0     $ Sink Temperature of 40 K  HEADER CONDUCTOR DATA,MAIN
HEADER CONDUCTOR DATA,MAIN
C X Conduction - Assume Ax = RadY/10 and L = RadX/9 - for centroids
   GEN  1,9,1, 1,1, 2,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
   GEN 11,9,1,11,1,12,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
   GEN 21,9,1,21,1,22,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
   GEN 31,9,1,31,1,32,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
   GEN 41,9,1,41,1,42,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
   GEN 51,9,1,51,1,52,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
   GEN 61,9,1,61,1,62,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
   GEN 71,9,1,71,1,72,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
   GEN 81,9,1,81,1,82,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
   GEN 91,9,1,91,1,92,1,kAl * RadThk * (RadY / 10.) / (RadX / 9.0)
C Y Conduction - Assume Ax = RadX/10 and L = RadY/9 - for centroids
   GEN  101,10,1, 1,1,11,1,kAl * RadThk * (RadX / 10.) / (RadY / 9.0)
   GEN  111,10,1,11,1,21,1,kAl * RadThk * (RadX / 10.) / (RadY / 9.0)
   GEN  121,10,1,21,1,31,1,kAl * RadThk * (RadX / 10.) / (RadY / 9.0)
   GEN  131,10,1,31,1,41,1,kAl * RadThk * (RadX / 10.) / (RadY / 9.0)
   GEN  141,10,1,41,1,51,1,kAl * RadThk * (RadX / 10.) / (RadY / 9.0)
   GEN  151,10,1,51,1,61,1,kAl * RadThk * (RadX / 10.) / (RadY / 9.0)
   GEN  161,10,1,61,1,71,1,kAl * RadThk * (RadX / 10.) / (RadY / 9.0)
   GEN  171,10,1,71,1,81,1,kAl * RadThk * (RadX / 10.) / (RadY / 9.0)
   GEN  181,10,1,81,1,91,1,kAl * RadThk * (RadX / 10.) / (RadY / 9.0)
C Radiation
   GEN -1001,100,1,1,1,999,0,0.85 * (RadX/10.0) * (RadY/10.0)
C Strap to Radiator, note that this comes near the lower left corner
   999,998,78,kAl * 0.025 * 0.003 / 0.1 $ 10 cm long,2.5 cm wide,0.3 cm thick,Al
```

OPTIONS DATA

REGISTERs to parameterize the model

10 x 10 nodes for Radiator, 1 for Detector, 1 for Sink

Conduction in X Direction

Conduction in Y Direction

Radiation to sink
Heat Strap to Center

The SOLVER: Sample – SOLVER Specific Setup

```
HEADER CONTROL DATA,GLOBAL
    SIGMA=5.67E-8
    ABSZRO = -273.15
    TIMEND = 1000.0
    NLOOPS = 100000
    DRLXCA = 1.0E-5
HEADER OPERATIONS
BUILD ALL
      CALL SOLVER
M     CALL TPRINT('ALL')
HEADER DESIGN DATA
    0.0005 <= RadThk <= 0.01
    0.05 <= RadX <= 0.1
    0.05 <= RadY <= 0.35
HEADER PROCEDURE
F     TMax = -200.0
F     TMin = 200.0
F     DO ITEST = 1,100
F       IF (T(INTNOD('MAIN',ITEST)).LT.TMIN) TMIN = T(INTNOD('MAIN',ITEST))
F       IF (T(INTNOD('MAIN',ITEST)).GT.TMAX) TMAX = T(INTNOD('MAIN',ITEST))
F     END DO
M     OBJECT = (0.1*Mass) + (0.2*ABS(MAIN.T998-(-120.)) + (0.7*(Tmax-Tmin))) $ Hold -120C while minimizing Gradient
M     GOAL = 0.0  $ Try to get Object to zero
      CALL STEADY
HEADER OUTPUT CALLS, MAIN
M     TMax = -200.0
M     TMin = 200.0
F     DO ITEST = 1,100
F       IF (T(INTNOD('MAIN',ITEST)).LT.TMIN) TMIN = T(INTNOD('MAIN',ITEST))
F       IF (T(INTNOD('MAIN',ITEST)).GT.TMAX) TMAX = T(INTNOD('MAIN',ITEST))
F     END DO
M     OBJECT = (0.1*Mass) + (0.2*ABS(MAIN.T998-(-120.)) + (0.7*(Tmax-Tmin))) $ Hold -120C while minimizing Gradient
      WRITE(NOUT,*) OBJECT
      WRITE(NOUT,*) 'RadX ',RadX
      WRITE(NOUT,*) 'RadY ',RadY
      WRITE(NOUT,*) 'RadThk ',RadThk
      WRITE(NOUT,*) 'Mass ',Mass
      WRITE(NOUT,*) 'Tmax ',Tmax
      WRITE(NOUT,*) 'Tmin ',Tmin
      WRITE(NOUT,*) 'Temp ',MAIN.T998
HEADER VARIABLES1,MAIN
M      Q998 = 0.5 $ 0.5 W of detector dissipation
END OF DATA
```

CONTROL DATA

OPERATIONS
Call to SOLVER
DESIGN DATA

PROCEDURE

Find Tmin and Tmax for Gradient

Define Object to optimize and Call Thermal Solution (STEADY)

Find Tmin and Tmax for Gradient

Redefine value of OBJECT based on updated temperatures

Output relevant information

Add 0.5 W of dissipation to detector

The SOLVER: Example – Output

```
  6.51941670031157
RadX    0.100000000000000
RadY    0.100000000000000
RadThk    5.000000000000000E-003
Mass    0.138500000000000
Tmax    -93.1916848597173
Tmin    -93.6913708397929
Temp    -89.2210674287068
  6.51136858371543
RadX    0.100000000000000
RadY    0.100000000000000
RadThk    5.100000000000000E-003
Mass    0.141270000000000
Tmax    -93.1990505119184
Tmin    -93.6889479403427
Temp    -89.2284330809078
  6.70060747965006
RadX    9.800000000000000E-002
RadY    0.100000000000000
RadThk    5.000000000000000E-003
Mass    0.135730000000000
Tmax    -92.2846087730940
Tmin    -92.7843698417608
Temp    -88.3139913420835
  6.34282065935889
RadX    0.100000000000000
RadY    0.102000000000000
RadThk    5.000000000000000E-003
Mass    0.141270000000000
Tmax    -94.0762859879434
Tmin    -94.5760393747227
Temp    -90.1056685569329
  5.88865222883556
RadX    0.100000000000000
RadY    0.107368120157434
RadThk    5.016789586505306E-003
Mass    0.149204184994584
Tmax    -96.3481208564904
Tmin    -96.8470244213934
Temp    -92.3775034254799
...
```

```
...
   0.435140508791916
RadX    0.100000000000000
RadY    0.212863762437403
RadThk    5.257559652919626E-003
Mass    0.310002868321797
Tmax    -123.977300279171
Tmin    -124.552734068211
Temp    -120.006682848161
  0.434524556462765
RadX    0.100000000000000
RadY    0.212801959035887
RadThk    5.257798101008654E-003
Mass    0.309926916902417
Tmax    -123.966605028417
Tmin    -124.541932720208
Temp    -119.995987597406
  0.434292286303274
RadX    0.100000000000000
RadY    0.212840155638721
RadThk    5.257650731984784E-003
Mass    0.309973858424841
Tmax    -123.973215520157
Tmin    -124.548608781059
Temp    -120.002598089147
  0.434353676501999
RadX    0.100000000000000
RadY    0.212841864083419
RadThk    5.257644140513062E-003
Mass    0.309975957936477
Tmax    -123.973511156074
Tmin    -124.548907349925
Temp    -120.002893725063
  0.433767999879875
RadX    0.100000000000000
RadY    0.212825565837466
RadThk    5.257707021932317E-003
Mass    0.309955928730251
Tmax    -123.970690714441
Tmin    -124.546058929185
Temp    -120.000073283431
```

FROM HEADER PROCEDURE...

```
OBJECT = (0.1*Mass) + (0.2*ABS(MAIN.T998-(-120.)) +
(0.7*(Tmax-Tmin)) $ Hold -120C while minimizing
Gradient
WRITE(NOUT,*) OBJECT
WRITE(NOUT,*) 'RadX ',RadX
WRITE(NOUT,*) 'RadY ',RadY
WRITE(NOUT,*) 'RadThk ',RadThk
WRITE(NOUT,*) 'Mass ',Mass
WRITE(NOUT,*) 'Tmax ',Tmax
WRITE(NOUT,*) 'Tmin ',Tmin
WRITE(NOUT,*) 'Temp ',MAIN.T998
```

FINAL SOLUTION

RadX: 0.05 <= 0.1 <= 0.1

RadY: 0.05 <= 0.21282 <= 0.35

RadThk: 0.0005 <= 0.000525 <= 0.01 T998 = -120.0000 (Goal of -120°C)

Mass: 0.30996 kg

Gradient: 0.5754°C

Note: if the RadX dimension is allowed to go to 0.2, the optimum solution is a square, which intuitively makes sense

```
  0.373113566535000
RadX    0.145831259373365
RadY    0.145831259373365
RadThk    5.105345046710294E-003
Mass    0.300750335885368
Tmax    -123.967934258405
Tmin    -124.457222684727
Temp    -119.997316827394
```

SINDA Data Structures

- Numbers for Nodes, Arrays, Conductors, etc entered by the user are referred to as Actual Number
- These values are all compressed into singular arrays for Temperatures, Heat Loads, Capacitances, Conductances, Arrays etc. The corresponding index of a given entry is the Relative Number
  - Nodes are first grouped by Input Order (Submodel),then by type (D,A,B,H)
  - Conductors are grouped by Input Order (Submodel),then type (L,R)
  - Arrays are grouped by Input Order (Submodel)
- INTNOD or NODTRN for node conversion
- INTCON or CONTRN for conductor conversion
- ARYTRN for array conversion

```
HEADER NODE DATA,MYSUBMODEL
    101, 70.0, 2.0 $ Diffusion Node
    102, 70.0, 2.0 $ Arithmetic Node
   -103, 70.0, 2.0 $ Boundary Node
   -104, 70.0, 2.0 $ Heater Node
HEADER NODE DATA,MYSUBMODEL
   -103, 70.0, 2.0 $ Boundary Node
    102, 70.0, 2.0 $ Arithmetic Node
    101, 70.0, 2.0 $ Diffusion Node
```

Actual	Relative
MYSUBMODEL.T(101)	T(1)
MYSUBMODEL.T(102)	T(2)
MYSUBMODEL.T(103)	T(3)
MYSUBMODEL.T(104)	T(4)
MYSUBMODEL2.T(101)	T(5)
MYSUBMODEL2.T(102)	T(6)
MYSUBMODEL2.T(103)	T(7)

SINDA Array Data Structures

- Arrays are handled a bit differently
  - All HEADER ARRAY data from user stored in one contiguous A array
  - There is a FORTRAN EQUIVALENCEd NA array with the A array
  - What this means is that both NA and A reference the same location in memory
  - Retrieving something from an index location in A will treat the bits as if they were stored as a REAL number (i.e. floating point)
  - Retrieving something from the same index location in NA will treat the bits as if they were stored as an INTEGER number
  - This allows both INTEGER and REAL data to be stored in a contiguous block of memory
  - Each index contains only one valid data entry (either INTEGER or REAL), but not both
- SINDA also stores one more index than the number of entries
  - This first index indicates how many entries exist in the array

```
HEADER ARRAY DATA,MYSUBMODEL

  1 = 1.0,2.0,3.0,4.0
 10 = 2.3,3.2,3.3
 11 = 1,2,3
 21 = 2,1.0    $ Only Valid if MIXARRAY specified in OPTIONS DATA!!!
     3,2.0
```

Index	A	NA	Start
1	?	4	← A1
2	1.0	?	
3	2.0	?	
4	3.0	?	
5	4.0	?	
6	?	3	← A10
7	2.3	?	
8	3.2	?	
9	3.3	?	
10	?	3	← A11
11	?	1	
12	?	2	
13	?	3	
14	?	4	← A21
15	?	2	
16	1.0	?	
17	?	3	
18	2.0	?	

SINDA User Variable Data Structures

- USER DATA, Submodel uses a similar EQUIVALENCE trick
  - All USER DATA is stored in one contiguous XK array
  - There is a FORTRAN EQUIVALENCEd K array with the XK array
  - What this means is that both K and XK reference the same location in memory
  - Retrieving something from an index location in XK will treat the bits as if they were stored as a REAL number (i.e. floating point)
  - Retrieving something from the same index location in K will treat the bits as if they were stored as an INTEGER number
  - This allows both INTEGER and REAL data to be stored in a contiguous block of memory
  - Each index contains only one valid data entry (either INTEGER or REAL), but not both

```
HEADER USER DATA,MYSUBMODEL
  1, 1.0
  2, 0.0
HEADER USER DATA,MYSUBMODEL2
  1, 1
  2, 0
  3, 0.0
HEADER OPERATIONS
C This one will luck out and return zero
M     WRITE(NOUT,*) MYSUBMODEL2.K3
C This one will be unpredictable!!
M     WRITE(NOUT,*) MYSUBMODEL2.XK1
```

Index	XK	K	Variable
1	1.0	?	MYSUBMODEL.1
2	0.0	?	MYSUBMODEL.2
3	?	1	MYSUBMODEL2.1
4	?	0	MYSUBMODEL2.2
5	0.0	?	MYSUBMODEL2.3

Extending SINDA with FORTRAN

- Most of SINDA's data as accessible to the user through FORTRAN
  - The only real exception is what is happening internal to any of their library routines
- Including a user SUBROUTINE with an M-type CALL COMMON line, will include all common blocks for SINDA, granting the user access to the T, Q, G, C, etc arrays as well as all the control constants and other SINDA data structures
- Variety of Variables are at your disposal
  - NNOD is the number of nodes
  - MMODS is the number of submodels
  - NDNAM(Size=NNOD) array contains the Submodel Name for each node
  - NDINT(Size=NNOD) array contains the node number for each node
  - NMOD(Size=MMODS) array contains list of Submodel Names
  - NSTRT(Size=MMODS) array contains starting node index for specified Submodel Index
  - NMDIF(Size=MMODS) contains # of Diffusion nodes for specified Submodel Index
  - NMARI(Size=MMODS) contains # of Arithmetic nodes for specified Submodel Index
  - NMBD(Size=MMODS) contains # of Boundary nodes for specified Submodel Index
  - NMHT(Size=MMODS) contains # of Heater nodes for specified Submodel Index
  - ARLXCA, ARLXCC, DRLXCA, DRLXCC, CSGMIN, etc (Size=NMODS) all contain the corresponding SINDA variable for the specified submodel index

Sample: TMGPRTDP
TMG-Like output with User Specified format

```
C---------------------------------------------------------------------------------------------------------
C  Routine to write out SINDA/FLUINT data in TMG-like format with formatting as supplied by the user
C  !!! Double Precision Only !!!
C---------------------------------------------------------------------------------------------------------
F      SUBROUTINE TMGQPRT(UNITSP, STRFMT)
M         CALL COMMON
F        INTEGER UNITSP
F        INTEGER TMGK, TMGJ, TMGI, N
F        CHARACTER*42 NODENAME
F        CHARACTER*8 NODENO
F        CHARACTER*32 STRFMT
F        CHARACTER*42 TMPFMT
C
C
F        IF (STRFMT.EQ.'                                ') THEN
F          WRITE(UNITSP,'(A)') 'ERROR: No format specifier given!!!'
F        ELSE
F          TMPFMT='(A42,1X,' // STRFMT // ')'
M          IF(NSOL.LE.1) THEN
F            WRITE (UNITSP,'(I6,1X,A)') -99999, 'SS'
F          ELSE
F            WRITE (UNITSP,'(I6,1X,E18.11)') -99999, TIMEN
F          ENDIF
F          DO TMGI=1,NNOD
F            WRITE(UNIT=NODENO,FMT='(I8)') NDINT(TMGI)
F            NODENAME = NDNAM(TMGI)
F            TMGJ = 1
F            DO WHILE (NODENO(TMGJ:TMGJ).EQ.' ')
F              TMGJ = TMGJ + 1
F            END DO
F            DO N = 1,LEN(NODENAME)
F              IF (NODENAME(N:N).EQ.' ') THEN
F                NODENAME = NODENAME(1:N-1) // '.' // NODENO(TMGJ:)
F                GOTO 15
F              ENDIF
F            END DO
F15          CONTINUE
F              WRITE(UNITSP,TMPFMT) NODENAME, Q(TMGI)
F          END DO
F        END IF
F      END
```

Call to Common to Access SINDA

Variable Declarations

Check that input was provided

Assemble Format String

Output Appropriate Header (SS vs. TR)

Loop through all nodes
Store NodeNo as Char*8,Get Submodel

Find Trailing whitespace for NodeNo

Find Trailing whitespace for Submodel
And assemble as Submodel.NodeNo

Get Double Precision Temperature and write to File in user specified format

Sample: TMGPRTDP Output

```
HEADER OPERATIONS DATA
F      CHARACTER*32 STRFMT
...
C Open a temperature and heat file for TMG-like output
F      OPEN(UNIT=680,FILE='..\Output.tmg',STATUS = 'UNKNOWN')
C
M      CALL STDSTL
C
...
M      CLOSE(680)

HEADER OUTPUT CALLS,MAIN
F      CHARACTER*32 STRFMT
C
M      IF((TIMEN.GE.(27812.0)).OR.(LEOP.EQ.1)) THEN
F        STRFMT='F18.9‘
F        CALL TMGPRTDP(680,STRFMT)
M      ENDIF

END OF DATA
```

```
-99999 SS
CDH.1002                         13.714309132
CDH.1003                          8.691085165
CDH.1004                          7.724342888
CDH.1005                         17.094802865
CDH.1006                          9.321929573
CDH.1007                          6.824265046
CDH.1008                         10.917145871
CDH.1101                          8.855667787
CDH.1102                          4.851964618
...
CDH.2206                         -1.132241624
CDH.2207                         -1.214881071
CDH.2208                         -1.255354058
CDH.2209                         -1.362248950
DFU.1                            -0.209298608
DFU.2                            -0.210654863
DFU.3                            -0.241078186
DFU.4                            -0.259009639
DFU.5                            -0.110150270
DFU.6                            -0.211591894
...
UBS_MLI.105002                   86.173589647
UBS_MLI.105003                    7.831876488
UBS_MLI.105004                  -52.959941162
UBS_MLI.105005                  -53.982811265
UBS_MLI.105006                   82.727261291
UBS_MLI.105007                   32.061672761
UBS_MLI.105008                  -38.585060777
UBS_MLI.105009                  -39.619218036
-99999  0.28200000000E+05
CDH.1002                         13.230735342
CDH.1003                          8.195073628
CDH.1004                          7.225618560
CDH.1005                         16.596811058
CDH.1006                          8.818744199
...
```

Sample: Conv_Trace

```
C-----------------------------------------------------------
C !!! Convergence Trace !!!
C-----------------------------------------------------------
F      SUBROUTINE CONV_TRACE(FileSpec)
M      CALL COMMON
F      INTEGER FileSpec,I,CONVERGED
F      DO I=1,MMODS
M        IF(NSOL.EQ.1) THEN
F          CONVERGED = 1
F          IF(ABS(DRLXCC(I)).GT.DRLXCA(I)) CONVERGED = 0
F          IF(ABS(ARLXCC(I)).GT.ARLXCA(I)) CONVERGED = 0
F          IF (CONVERGED.EQ.0) THEN
F            WRITE(FileSpec,10) TIMEN,','
F            WRITE(FileSpec,11) LOOPCT,','
F            WRITE(FileSpec,9) NMOD(I),','
F            WRITE(FileSpec,10) DRLXCA(I),','
F            WRITE(FileSpec,10) DRLXCC(I),','
F            WRITE(FileSpec,11) NDRLXN(I),','
F            WRITE(FileSpec,10) ARLXCA(I),','
F            WRITE(FileSpec,10) ARLXCC(I),','
F            WRITE(FileSpec,11) NARLXN(I),','
F            WRITE(FileSpec,10) EBALSA(I),','
F            WRITE(FileSpec,13) ESUMIS(I),','
F            WRITE(FileSpec,13) ESUMOS(I),','
F            WRITE(FileSpec,13) EBALSC,','
F            WRITE(FileSpec,13) TSUMIS,','
F            WRITE(FileSpec,14) TSUMOS,','
F          ENDIF
M        ELSE
...
```

```
...
F          CONVERGED = 1
F          IF(ABS(DRLXCC(I)).GT.DRLXCA(I)) CONVERGED = 0
F          IF(ABS(ARLXCC(I)).GT.ARLXCA(I)) CONVERGED = 0
F          IF(ABS(DTMPCC(I)).GT.DTMPCA(I)) CONVERGED = 0
F          IF(ABS(ATMPCC(I)).GT.ATMPCA(I)) CONVERGED = 0
F          IF (CONVERGED.EQ.0) THEN
F            WRITE(FileSpec,10) TIMEN,','
F            WRITE(FileSpec,11) LOOPCT,','
F            WRITE(FileSpec,9) NMOD(I),','
F            WRITE(FileSpec,10) DRLXCA(I),','
F            WRITE(FileSpec,10) DRLXCC(I),','
F            WRITE(FileSpec,11) NDRLXN(I),','
F            WRITE(FileSpec,10) ARLXCA(I),','
F            WRITE(FileSpec,10) ARLXCC(I),','
F            WRITE(FileSpec,11) NARLXN(I),','
F            WRITE(FileSpec,10) DTMPCA(I),','
F            WRITE(FileSpec,10) DTMPCC(I),','
F            WRITE(FileSpec,11) NDTMPN(I),','
F            WRITE(FileSpec,16) ATMPCA(I),','
F            WRITE(FileSpec,10) ATMPCC(I),','
F            WRITE(FileSpec,12) NATMPN(I),','
F          ENDIF
M        ENDIF
F9       FORMAT(A8,A2\)
F10      FORMAT(F10.2,A,\)
F11      FORMAT(I10,A,\)
F12      FORMAT(I10,A)
F13      FORMAT(F11.4,A\)
F14      FORMAT(F11.4,A)
F15      FORMAT(D11.4,A\)
F16      FORMAT(E11.2,A\)
F      END DO
F      END
```

Sample: Conv_Trace Output

Lists:
TimeStep,LoopCt,Submodel,DRLXCA,DRLXCC,NDRLXN,ARLXCA,ARLXCC,NARLXN,EBALSA,ESUMIS,ESUMOS,EBLASC,TSUMIS,TSUMOS for any submodels which did not meet convergence criteria when this subroutine was called

```
      0.00,  5000,PY_DIODE,  0.00,  0.00, 10190, 0.00, 0.00,  10111,  0.01, 107.7526, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,MY_DIODE,  0.00,  0.00, 10450, 0.00, 0.08,  10521,  0.01, 76.3841, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,KAHTR   ,  0.00,  0.00,     0, 0.00,-0.01,   4071,  0.01, 0.0000, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,KAPR    ,  0.00,  0.00, 16524, 0.00, 0.05,  44302,  0.01,1371.5599, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,KUHTR   ,  0.00,  0.00,     0, 0.00, 0.16,  11432,  0.01, 0.0000, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,KUPR    ,  0.00,  0.09, 51432, 0.00, 0.02,  15114,  0.01,3418.2185, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,GMI     ,  0.00, -0.10,  1227, 0.00,-0.10,   7991,  0.01,1168.1204, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,RF      ,  0.00,  0.00, 11014, 0.00, 0.00, 102460,  0.01, 65.8247, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,PY_ACT  ,  0.00,  0.00,   132, 0.00, 0.00,   1132,  0.01, 10.4051, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,ACS5    ,  0.00,  0.00,    22, 0.00, 0.00,  10020,  0.01, 5.4127, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,ACS6    ,  0.00,  0.00,    22, 0.00, 0.02,  10032,  0.01, 5.2209, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,ACS     ,  0.00, -0.01,     3, 0.00, 0.03, 200003,  0.01, 91.6071, 0.0000, -0.0102, 0.0000, 0.0000,
...
      0.00,  5000,MY_SA   ,  0.00, -0.26, 24127, 0.00, 0.13, 124126,  0.01,9846.1025, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,NEA1    ,  0.00,  0.00,  2001, 0.00, 0.00,  11043,  0.01, 4.5772, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,NEA2    ,  0.00,  0.00,  1009, 0.00,-0.02,  11021,  0.01, 4.7244, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,NEA5    ,  0.00,  0.00,  5020, 0.00, 0.00,  11012,  0.01, 5.3195, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,NEA8    ,  0.00,  0.00,  6003, 0.00,-0.02,  16037,  0.01, 5.0826, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,PROP    ,  0.00, -0.01,     1, 0.00, 0.01,   200140,  0.01, 94.5073, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,PROPRNG ,  0.00,  0.00,    62, 0.00, 0.01,   100063,  0.01, 155.8436, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,PRPDECK ,  0.00,  0.00,    55, 0.00, 0.02, 200047,  0.01, 170.0131, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,PY_BOOM ,  0.00,  0.00,  3051, 0.00, 0.00,  10118,  0.01, 77.9972, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,PY_HARN ,  0.00,  0.00,     2, 0.00, 0.00,    223,  0.01, 11.6835, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,PY_SA   ,  0.00,  0.06, 21214,  0.00, -0.03, 121216,  0.01,14190.6582, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,PY_SADA ,  0.00,  0.00,   123, 0.00,-0.01, 200104,  0.01, 10.6203, 0.0000, -0.0102, 0.0000,   0.0000,
      0.00,  5000,RAD_ACE ,  0.00,  0.00,  1145, 0.00, 0.00, 101105,  0.01, 12.8975, 0.0000, -0.0102, 0.0000,   0.0000,
      0.00,  5000,RAD_AM  ,  0.00,  0.01, 11148, 0.00, 0.00, 101300,  0.01, 45.9346, 0.0000, -0.0102, 0.0000,   0.0000,
      0.00,  5000,SA_HNG  ,  0.00, -0.01,   200, 0.00, 0.00, 100200,  0.01, 41.2432, 0.0000, -0.0102, 0.0000,   0.0000,
      0.00,  5000,TAM     ,  0.00, -0.01,    15, 0.00, 0.00, 100002,  0.01, 1.7128, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,UBSTRUSS,  0.00,  0.00, 14034,  0.00,  0.00, 144032,  0.01, 48.8929, 0.0000, -0.0102, 0.0000, 0.0000,
      0.00,  5000,UBS_DECK,  0.00,  0.89, 209266,   0.00, -0.06, 10147,  0.01, 88.8816, 0.0000, -0.0102, 0.0000, 0.0000,
```

Sample: Get_Orbit

```
C---------------------------------------------------------------------------------------
C !!! GETORBIT !!! -Used to determine if in eclipse or not for power purposes
C---------------------------------------------------------------------------------------
F     SUBROUTINE GETORBIT(TIME_ARRAY, ECLIPSE_ENTRY, ECLIPSE_EXIT, PERIOD)
M     CALL COMMON
F     INTEGER NoEntries, I
F     REAL TIME_ARRAY, TIME_ARRAY1, ECLIPSE_ENTRY, ECLIPSE_EXIT
F     REAL CUR_TIME, PREV_TIME, MAX_DELTAF DIMENSION TIME_ARRAY(1)
F     EQUIVALENCE (TIME_ARRAY1, NoEntries)
F     TIME_ARRAY1 = TIME_ARRAY(1)
C
F     ECLIPSE_ENTRY = 0.
F     ECLIPSE_EXIT = 0.
F     PERIOD = TIME_ARRAY(1+NoEntries)
F     MAX_DELTA = PERIOD/1000.
F     DO I=2,NoEntries
F       PREV_TIME = TIME_ARRAY(I)
F       CUR_TIME = TIME_ARRAY(I+1)
F       IF (((CUR_TIME-PREV_TIME).LT.MAX_DELTA).AND.(ECLIPSE_ENTRY.EQ.0.)) THEN
F         ECLIPSE_ENTRY = CUR_TIME
F       ELSEIF (((CUR_TIME-PREV_TIME).LT.MAX_DELTA).AND.(ECLIPSE_ENTRY.NE.0.)) THEN
F         ECLIPSE_EXIT = CUR_TIME
F       ENDIF
F     END DO
C Check if no eclipse is present
F     IF ((ECLIPSE_ENTRY.EQ.0.0).AND.(ECLIPSE_EXIT.EQ.0.0)) THEN
C Now check to make sure that ECL_EXIT-ECL_ENTRY < half of PERIOD
F     ELSEIF ((ECLIPSE_EXIT-ECLIPSE_ENTRY).LT.(PERIOD/2.)) THEN
C       ECL_EXIT-ECL_ENTRY > half of PERIOD, swap em
F     ELSE
F       CUR_TIME = ECLIPSE_EXIT
F       ECLIPSE_EXIT = ECLIPSE_ENTRY
F       ECLIPSE_ENTRY = CUR_TIME
F     ENDIF
F     END
```

Call to Common to Access SINDA Variable Declarations

Equivalence trick to access first entry of array as integer

Get Period from last entry

Loop through all entries looking for when Dtime is small enough to suggest eclipse

Check for full sun orbit

Check that eclipse duration is longer than sunlit duration

Sample: SUB_MAX_RANGE

```
C-------------------------------------------------------------------------------------
C SUB_MAX_RANGE – Used to find Maximum Temperature for nodes in specified Submodel and
C within specified node range
C-------------------------------------------------------------------------------------
F       REAL FUNCTION SUB_MAX_RANGE(MYSUBNAM, STARTNO, ENDNO)
M        CALL COMMON
F        INTEGER I_VAR, J_VAR, STARTNO, ENDNO, LEN_MYSUBNAM
F        CHARACTER*32 MYSUBNAM
F        CHARACTER*32 MYSUBNAM2
F        CHARACTER*32 MYSUBCHK
F        DOUBLE PRECISION TEMP
F        DOUBLE PRECISION TMIN, TMAX
F        TMIN = 100000.0
F        TMAX = -100000.0
F        LEN_MYSUBNAM = 0
F        DO I_VAR=1,LEN(MYSUBNAM)
F          IF (ICHAR(MYSUBNAM(I_VAR:I_VAR)).EQ.0) THEN
F            IF (LEN_MYSUBNAM.EQ.0) THEN
F              LEN_MYSUBNAM = I_VAR-1
F            ENDIF
F          ENDIF
F        END DO
F        DO I_VAR=1,MMODS
F          MYSUBCHK = NMOD(I_VAR)
F          IF ( (LEN(TRIM(MYSUBCHK)).EQ.LEN_MYSUBNAM).AND.
F     1        (MYSUBNAM(1:LEN_MYSUBNAM) .EQ. MYSUBCHK(1:LEN_MYSUBNAM)) ) THEN
F            DO J_VAR=NSTRT(I_VAR), NSTRT(I_VAR)+NMDIF(I_VAR) + NMARI(I_VAR) + NMHT(I_VAR) + NMBD(I_VAR)-1
F              IF ((NDINT(J_VAR).GE.STARTNO).AND.(NDINT(J_VAR).LE.ENDNO)) THEN
F                CALL DPTEMP(MYSUBCHK,NDINT(J_VAR),'GET',TEMP)
F                IF(TEMP.GT.TMAX) TMAX = TEMP
F                IF(TEMP.LT.TMIN) TMIN = TEMP
F              ENDIF
F            END DO
F          END IF
F        END DO
F        SUB_MAX_RANGE = REAL(TMAX)
F        RETURN
F      END
```

Call to Common to Access SINDA Variable Declarations

Get Period from last entry

Loop through all entries to get length of MYSUBNAM with no spaces

Loop through all submodels

Check that Sub length is correct and that they match MYSUBNAM

If so, then start looking through nodes in this submodel and if they are in range, get temps and store Min and Max

Return Maximum Temp found

Thermal Calculations

What's Happening When SINDA runs?

RECALL: In the end, SINDA models are really just compiled FORTRAN programs

- The preprocessor begins by expanding all the INSERT and INCLUDE directives in the inp file. This expanded SINDA file is then processed to identify the:
  - DATA portions (Nodes, Conductors, Arrays, etc) which are stored in binary files
  - LOGIC portions (Operations, Variables 1, Output Calls, etc) which are written as FORTRAN instructions, making the necessary conversions from ACTUAL numbers to RELATIVE numbers in the code

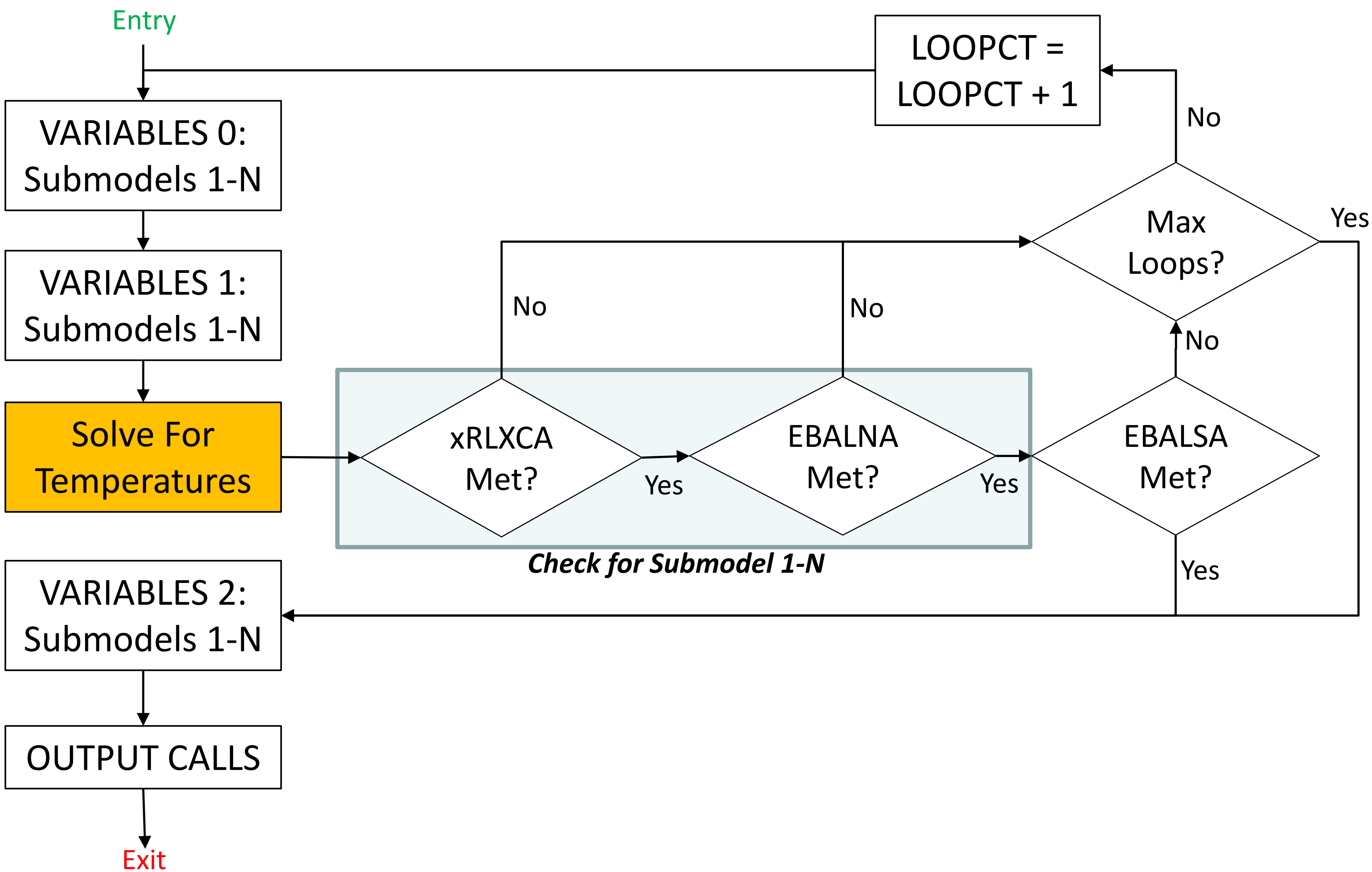
- Once the model is preprocessed, the complete FORTRAN file along with the SINDA library files are compiled to generate the executable, which is subsequently executed to perform the thermal simulation

- The entry point for the program is the OPERATIONS block, which must include a BUILD directive to specify which submodels to include in the solution. Most often, the OPERATIONS block also includes calls to STEADY or TRANSIENT to execute the desired simulation as requested by the user

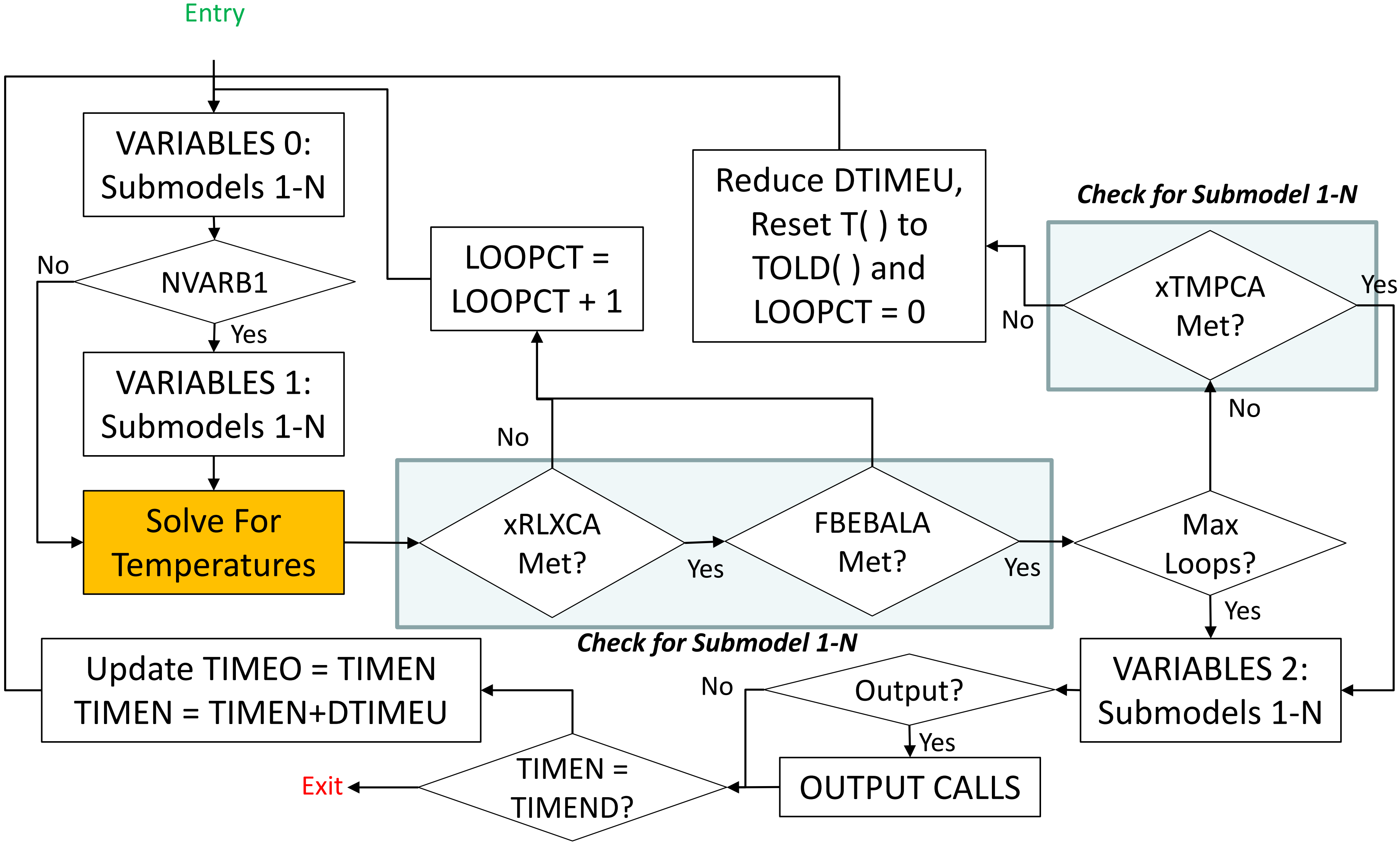
What's Happening When SINDA runs?

- When a STEADY or TRANSIENT solution routine is called, a sequence of subroutines are then called, one for each submodel included in the BUILD statement
  - **VARIABLES 0**: instructions to be performed at the start of each solution or timestep *(best location for time dependent logic)*
  - **VARIABLES 1**: instructions to be performed at the start of each iteration during steady state solutions or timestep during transient *(best location for temperature dependent logic)*
    - *Note that using the NVARB1 control constant can alter the default behavior to execute this logic after each iteration during TRANSIENT runs as well, but to enable this, it must be specified by the user*
  - **VARIABLES 2:** instructions to be performed upon convergence or when maximum number of loops is reached during solution iteration *(best location for state logic, e.g. heater on)*
- After the last submodel's VARIABLES 1 instructions have been executed, SINDA performs the actual temperature solution for the current iteration
  - For STEADY solutions, the solution continues until convergence is met, defined by:
    - *A/DRLXCA – maximum allowable temperature change between iterations for Arith. and Diff. nodes*
    - *EBALNA – maximum allowable nodal energy imbalance (W)*
    - *EBALSA – total allowable SYSTEM energy imbalance (%)*
    - *NOTE that A/DRLXCA and EBALNA may vary for each submodel, allowing tighter local convergence*
  - For TRANSIENT
    - *A/DRLXCA – maximum allowable temperature change between iterations for Arith. and Diff. nodes*
    - *A/DTMPCA – maximum allowable temperature change between timesteps for Arith. and Diff. nodes*
    - *FBEBALA – maximum allowable nodal energy imbalance in transient (%)*

What's Happening When STEADY runs?



What's Happening When TRANSIENT runs?



Solving for Temperatures

- SINDA has two types of methods for solving for temperatures:
  - **<u>Matrix Inversion (MATMET <>0)</u>: all conductors are processed to represent the nodal energy balance system of equations in a matrix format [G][T] = [Q] where:**
    - [G] is a matrix where each (i,j) entry represents all connections between node i and j and where radiative terms are linearized by $(T_i + T_j)(T_i^2+T_j^2)$
    - [T] is a vector of temperatures to be solved
    - [Q] represents any impressed loads as well as stored/released energy based on mCp dT/dt as the temperature solution is evolved through iterations due to the linearization and mCP effects
    - Solving for [T] requires inversion of [G] such that $[T] = [Q][G]^{-1}$; this is the computationally expensive part
    - Two solution approaches in SINDA (YSMP: Yale Sparse Matrix Package [1,2] and AMG-CG: Advanced Multigrid Conjugate Gradient [11,12]) Can be applied for the entire model or solving at a submodel level.
    - Default is generally MATMET=12: AMG-CG for entire model
  - **<u>Iterative (MATMET=0)</u>: each node is solved one at a time for its energy balance based on capacitance, adjoining conductors and their associated nodes**
    - Was the original SINDA solution approach. Matrix solutions are considerably faster for solving most classes of problems, so this method is generally preserved for backwards compatibility, but is not usually recommended for general use (think of perturbations as a pebble in a pond…how long for a wave generated by the perturbation to reach the shore? (i.e. how long before T_i change is felt at T_j)
    - Many more iterations necessary to propagate changes throughout the model, but faster to solve one full iteration than matrix solutions, which advance the solution simultaneously for all nodes
    - As nodes are solved from i = 1:n, Temperatures of nodes already solved during current iteration are used for subsequent node solutions within current iteration. As such, the order of the solution is varied [Start, End,Step] of [1,n,1], then [n,1,-1], then [1,n,2] and [n,1,-2] to fill in remaining nodes)…see NSOLOR

Extrapolating to Reach Best Solution

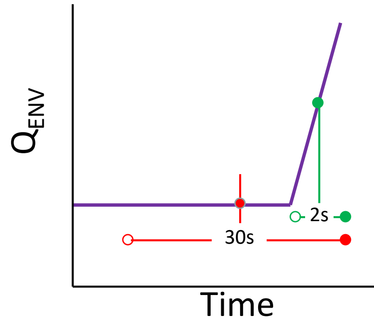
- SINDA has multiple methods to help accelerate convergence to the best temperature solution based on iterative kinds of solutions (which due to the non-linear behavior of radiation applies to both the Iterative and Matrix Inversion discussed previously)
- Extrapolation is handled by $T_{new} = T_{old} + \omega(T_{new} - T_{old})$ where ω is the extrapolation scale factor. ITERXT and EXTLIM control constants define ω.
- ITERXT: ITERations before eXTrapolations (Default=3)
  - 0: ω = EXTLIM
  - 1: $\omega = 2 / (1 + (1-\rho)^{1/2})$ where ρ = Maximum ($|T_{new} - T_{old}| / |T_{new} - T_{old}|$ from previous iteration)
  - 2: $\omega = 2 / (1 + (1-\rho^2)^{1/2})$ where ρ = Maximum ($|T_{new} - T_{old}| / |T_{new} - T_{old}|$ from previous iteration)
  - 3: Aitken's Del Squared Method is used: $T^i = (T^i T^{i-2} - (T^{i-1})^2) / (T^i + T^{i-2} - 2T^{i-1})$ which requires at least 3 iterations to have been solved and i represents the current iteration. NOTE, this approach is only applied if the 3 iterations are monotonically increasing or decreasing
- EXTLIM: EXTrapolation LIMit (Default=0.5)
  - Ignored if ITERXT =1 or 2
  - If <1: Damping factor -- helps force convergence if oscillating (under relaxation)
  - If >1: Acceleration factor -- helps accelerate solution, but may result in oscillations (over relaxation)
- SINDA will often detect oscillations during steady state solutions and apply heavy damping at a submodel level (ITERXT = 0, EXTLIM=0.5) to try to dampen out oscillations and reach convergence
  - Less often needed in TRANSIENT, where mCp terms act as damping factor to sudden changes

Selecting a Timestep

- SINDA defaults to automatic timestep selection (DTIMEI=0). A term called CSGMIN (Capacitance Divided by Sum of Conductors) is useful to see where conditions might exist that may cause the timesteps to be smaller than a user might like. Note that the units for CSG are units of time (W/K / (J/K) = s):
  - Small Capacitance → small CSG. **If capacitance is very small, make node arithmetic.**
  - Large Conductance → small CSG. **If conductance is too large, consider merging into a single node and eliminating large conductance.**
  - **<u>NEVER use a large conductance (e.g. 500 W/K) to make two nodes go to the same temperature. This can lead to numerical instability in the solution.</u>**
  - Also, be careful of very small thicknesses in Thermal Desktop that may lead to a high conductance (e.g. modeling the actual thickness of the adhesive for a heater mat contactor to the mounting substrate)
  - To troubleshoot small timestep with DTIMEI=0, search for "THIS SUBMODEL CONTAINS TIME STEP LIMITING NODE:" in the .out file. This will highlight the particular node and submodel that is driving the timestep. May not be the same throughout the solution…
- DTIMEL/DTIMEH can be used to prevent timesteps from going too small or large
  - If calculated timestep > DTIMEH, then it is set to DTIMEH
  - If calculated timestep < DTIMEL, **<u>then the run terminates with an error message</u>**
  - More often than not, the calculated timestep is smaller that what a user would like…

Selecting a Timestep

- A user may also set DTIMEI to an explicit value to force a particular timestep
- **AUTHOR's OPINION**:: Although the software vendor cautions against doing this, my experience has been that with larger models assembled from a variety of organizations and model developers, a reasonable timestep to achieve results in a timely manner is acceptable rather than waiting for the optimum timestep to be determined by the code
    - Unless great care is taken throughout the entire model to ensure capacitances are not too small and conductors are not too large, the solution may suffer from small timesteps due to one localized area. Finding and fixing this may take more work than is desired…
- A sample Low Earth Orbit model explored the difference between a user selected timestep of 30 s and automatic timestep calculation
    - 49.7 s to run with DTIMEI=30; 554.3 s with DTIMEI=0 *(Factor of 10x slower…)*
    - All Orbit Average nodal temperatures were within 1°C over all nodes
    - Some MLI nodes did show considerable differences in temperatures *at particular timesteps*. This is most likely due to the TIMEM variable (TIMEN + TIMEO)/2, which is used for interpolation of the environmental fluxes.
    - Taking an example as shown to the right, the purple line represents the env flux. With the same TIMEN (solid dot), but different timesteps, the TIMEO (open dot) is different as is the TIMEM used for interpolation. So, based on the timestep, a different Qenv would be used for the same TIMEN…



Troubleshooting a Slow Running Model

- To troubleshoot a slow running model, there are basically three areas to address:
  - Model is taking too long for each iteration
  - Model is taking too many iterations for steady-state or at each timestep
  - Model is taking too small a timestep to finish in a timely manner
- Too Long per Iteration…
  - **Usual Cause: too many conductors.** Can the number of Radks be reduced without significantly impacting the accuracy? Solution time tends to be linear with number of conductors…
  - Use of SPARSEG might help. This tries to ignore the smaller terms. For a SPARSEG of 0.001, any conductors that are less than 0.001*Sum Conductors for a given node are temporarily set to zero for the solution to "sparsify" the matrix, making it less computationally demanding to invert
- Too Many Iterations…
  - **Usual Cause: convergence criteria may be too tight, perturbations in steady state (e.g. heaters)**
  - Determine which submodels are needing more loops to converge. A/DRLXCC (calculated value for comparison to A/DRLXCA).
  - NDRLXC and NDRLXN list node and submodel ID for node used for DRLXCC
  - NARLXC and NARLXN list node and submodel ID for node used for ARLXCC
  - Hint: CONV_TRACE code in Advanced SINDA Section allows this data to be output at the request of a user. Often useful to output every N iterations (maybe 25) to show what has not converged and where
- Too Small a Timestep…
  - **Usual Cause: Automatic timestep calculation (DTIMEI=0) used and presence of very small capacitance or very large conductors**
  - **Secondary Cause: too small a value for A/DTMPCA and instantaneous large flux change**

OpenTD Application Programming Interface

OpenTD API

- *With the release of Thermal Desktop 6.0, an application programming interface was also included*
- *Since then, each subsequent release has allowed greater access to model data*
- *The API allows users to develop computer programs that can access and manipulate data and objects in a Thermal Desktop model*
- *As a general rule, there are three types of applications that could be developed:*
  - *Manipulate, create, or remove model objects*
  - *Extract data from a model*
  - *Alter the model execution process*

OpenTD API Application Examples

- *Add user logic between generation of inp/cc files and SINDA execution*
- *Display objects or node numbers based on a user input list of nodes*
- *List object counts to file*
- *List all submodel to submodel connections (Conductors/Contactors)*
- *List Symbol values (or over ride CaseSet values)*
- *Extract and Display optical and material properties*
- *User manipulation of Radk output (user defined output criteria)*
- *Merge selected Tri elements into Quad*
- *Model conversion to other formats*
- *Post processing utilities*
- *Saved Views and export of images for reports/documentation*
- *Domain Tag Set Importer*

Getting Started

- *To take advantage of the API, the Dynamic Linked Libraries need to be added as a reference to a .NET project*
  - *C:\Windows\Microsoft.NET\assembly\GAC_64\...\OpenTDvXX.Results.dll*
  - *C:\Windows\Microsoft.NET\assembly\GAC_MSIL\...\OpenTDvXX.dll*
- *Create a TD object (VB language):*

```
Dim TD As New OpenTDv63.Thermal Desktop
TD.Connect("C:\MyDrawing.dwg")
```

- *The above lines connect to an open instance of AutoCAD with MyDrawing.dwg or opens a new instance with the file. From there, a user is free to query or modify the drawing based on their needs*
- *There are methods of the Thermal Desktop class to get single objects (based on their handle or name) or all objects of a specified type as a List(Of …) object*

```
Dim AllCaseSets as List(Of OpenTDv63.CaseSet)
AllCaseSets = TD.GetCaseSets
```

- *Limited documentation for the API at this time. Vendor does provide Getting Started Guides for 6.1, 6.2, and 6.3. Also, some ICES papers exist on the topic and Vendor includes forums on the API*

A bit about the internal hierarchy…

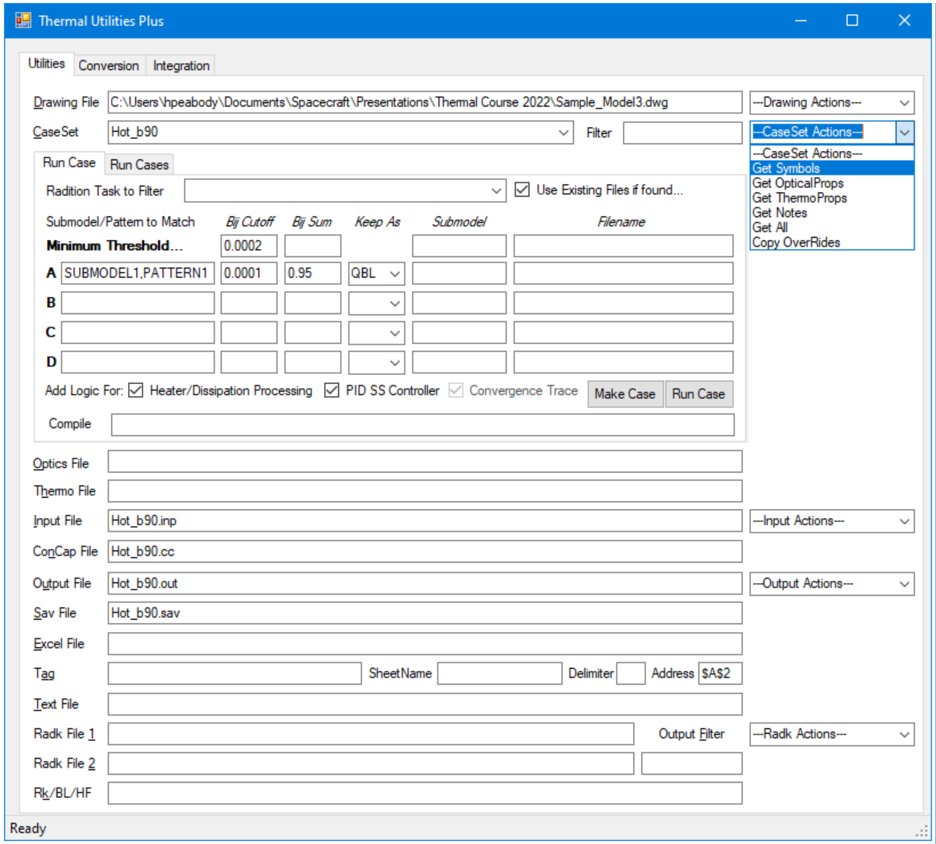
- *AutoCAD Handles (the 4 or 6 character hex codes unique to each object) are a key to working with the relationships between TD objects*
- *There are various lists of handles to work with. Which of these apply depends on the object type (In []) and lists the handles of associated objects:*
  - *AttachedConicHandles [Nodes] -> Surfaces, FDSolids, Measures*
  - *AttachedObjectHandles [Nodes] -> Conductors, FE, HeatLoad, Contactor, Heater*
  - *AttachedNodeHandles [Surfaces, FDSolids, FE] -> Nodes*
  - *AttachedHeatLoads [Surfaces, FDSolids, FE] -> HeatLoads*
  - *AttachedConductor [Surfaces, FDSolids, FE] -> Conductors*
  - *From and To [Contactors,Conductors] -> Nodes, Surfaces, FDSolids, FE*
  - *ApplyConnections [Heaters, Heatloads] -> Nodes, Surfaces, FDSolids, FE*
- *The GetEntityTypes method allows a list of Handles to be processed to determine the object type for each handle*
- *Note: INS nodes are not objects that can be directly retrieved. To determine these, all surfaces need to be polled to see if they create INS and what the properties are*
- *Some objects may also allow Domain Tag Sets. If these are encountered, they need to be dereferenced into their constituent objects, which can be retrieved through the DomainManager*

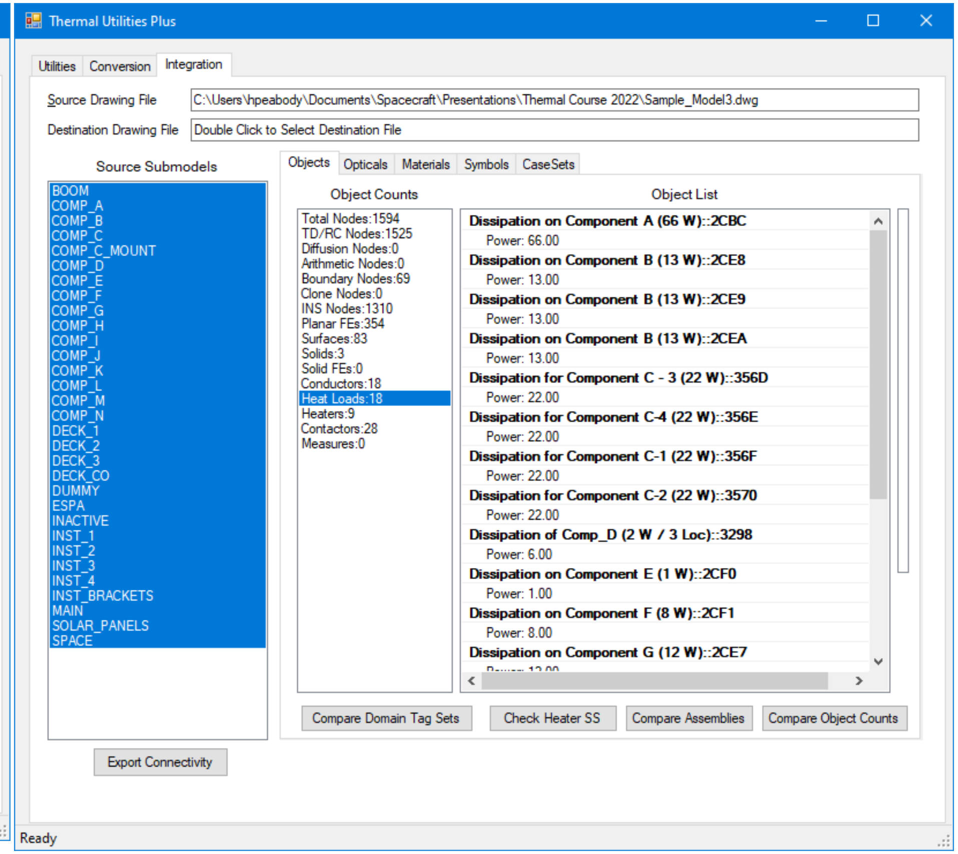
Some Functions developed at GSFC using the API

Framework Function	Purpose
GenerateHeaterDissipationLogic	Process SINDA .inp file and add logic to .htr file to output Heater and Dissipation output logic for every timestep
ProcessHeaterDissipationResults	Process SINDA .out file and retrieve output generated from GenerateHeaterDissipationLogic and import into Excel template workbook
GenerateDampedPropHeatersInSSForPIDs	Process SINDA .inp file and add logic to .pid file to apply Predictor/Corrector approach for all PID controllers
GenerateHtrDisSummaryCompare	Generate Compare_Summary sheet highlighting the differences between two HtrDis postprocessing files
Write2DArrayToCSVFile	Convert 2D array to comma-separated value output file (can include Header row and Output Mask)
ImportFileIntoExcel	Import specified text file into specified Excel location, parsing on delimiter
ExtractTaggedLinesAndImportIntoExcel	Extract lines beginning with TagID from OutFile and import into specified Excel location, parsing on specified delimiter
GetTDGlobalVisibilityStates	Return data structure with deterministic global visibility state for each TD object type (e.g. TD/RC nodes, Surfaces, HeatLoads, etc)
TurnNodeIDsOnForSelectedNodes	Evaluate user provided node list and turn node number visibility on in GUI
GetTDObjectCounts	Retrieve counts for each TD object type for each submodel
GetTDReferencedProperties	Retrieve list of Material and Optical properties used by each submodel
GetTDObjectCountsAndReferencedProperties	Retrieve counts for each TD object type and lists of Material and Optical properties for each submodel
GetAllReferencedRCOFiles	Make list of all Optical files that were referenced throughout all CaseSets
GetAllReferencedTDPFiles	Make list of all Material files that were referenced throughout all CaseSets
RemoveRadk1FromRadk2	Outputs file with all the Radks that appear only in Radk File 2
ReplaceRadk2MinusRadk1WithBackloads	Outputs file with inputs and logic to represent all the Radks in Radk2 but not in Radk1 as Backloads along with radiation to sink
ReplaceRadk2MinusRadk1WithHeatFlowsIJ	Outputs file with inputs and logic to represent all the Radks in Radk2 but not in Radk1 as a Heat Flow
EvaluateSymbolsInDWGFile	Generates temporary case set, outputs CC file and processes this for evaluated symbol values
EvaluateSymbolsForSpecifiedCaseSet	Generates temporary case set spawned from user specified case, outputs CC file and processes this for evaluated symbol values
ExtractSymbolEvaluatedValuesFromCCFile	Process CC file header and retrieves symbol names and evaluated values
GetTDOptProps	Read TD object and extract optical properties and store in GMM_OpticalProperties collection
GetTDThermoProps	Read TD object and extract thermophysical properties and store in GMM_ThermophiysicalProperties collection
WritePropsToXL	Extract Material/Optical property data and generate Tables in Excel workbook along with temperature dependent material property plots
GetTDNotes	Read TD object and extract Notes data and Splice together all Tabs into single output text file
GetTDRunDirectory	Return path to either DWG file if No CaseSet specified with UserDirectory, or to UserDirectory
RunSpecifiedCaseSet	Execute case set in its own directory with options to add heater dissipation and convergence trace logic

Italics indicates a requirement to use the OpenTD API to connect to Thermal Desktop

Some Applications developed at GSFC using the API





Good Modeling Practices and Other Miscellaneous Tips

Best Practices: DO's

- DO: Use appropriate nodalization: most for thin, low k; fewer for thick, high k paths
- DO: Use MLI offset of 100000/200000 (if on both sides) for easy identification of MLI by node number
- DO: Use logical ranges of node numbers to identify components within a submodel (e.g. 1000-1999 for motor, 2000-2999 for shaft, 3000-3999 for mirror, etc)
- DO: Verify node numbers are unique (or duplicates are intended)
- DO: Fill in comment fields
  - Heatload: Location, Dissipation
  - Heater: Location, On/Off setpoints, Control Type, Available Power; Also add meaningful Register Append String
  - Contactor: From, To, Value
  - Conductor: From, To, Value
- DO: Ensure nodes are merged. Also, check for free edges
- DO: Use a starting number like 1001 to indicate that the node numbers are not defaults (i.e. starting from 1)
- DO: Ensure contactors make contact as intended
- DO: Verify Active sides for Radiation for all groups
- DO: Verify MLI is included where intended
- DO: Verify Optical and Material properties are as intended and document source of the data
- DO: Check that thicknesses are reasonable – showing wireframe outline preferred method
- DO: Check for overlapping surfaces and correct if needed
- DO: Ensure symbol names are logical, organized, and easily interpreted
- DO: Use "Output Expression" wherever possible rather than resultant value
- DO: Provide meaningful names for Radk, HR, and SINDA files in CaseSets. Avoid accepting defaults
- DO: Assign a NoRad property (a=e=1) for all inactive sides. Don't accept DEFAULT property
- DO: Purge unused Symbols, Properties, Radiation Groups, Submodels, Layers, Domain Tag Sets, etc
- DO: Use Notes tabs to document model status and configuration
- DO: Ensure the radks to space are included or excluded as applicable in Radk output (e.g. internal Radks)
- DO: Perform conductivity check with 1 linear boundary condition and no radiation to ensure connectivity

Best Practices: DON'Ts

- DON'T: Use duplicate descriptions for different objects (each object should be uniquely described)
- DON'T: Couple nodes together with a very high conductance (or very small thickness)
- DON'T: Fire more rays than necessary for the accuracy desired. Bear in mind which small radks will not be output based on Bij Cutoff and Sum values
- DON'T: Have more radks than necessary for accuracy desired
- DON'T: Set Oct Cell objects too low or it may spend time going down too many levels without making progress
- DON'T: Have large timesteps if thermal responses are fast (e.g. 30 s timestep for a heat pulse of 5 s)
- DON'T: Use two tri-elements where a quad-element would be better
- DON'T: Use automeshing codes, or be sure to exercise control over mesh density to minimize node count
- DON'T: Have very small capacitance nodes where an arithmetic node is a better choice (e.g. MLI)
- DON'T: Use midpoint or fixed duty cycle for heaters in steady state, unless you are sure this will apply over all analysis cases (Damped Proportional preferred), or use the program feature to change behavior for various CaseSets
- DON'T: Use (X == 1) ? 0 : 1 syntax in output to FORTRAN blocks if outputting as expression
- DON'T: Use the Enable/Disable feature where the same effect can be accomplished using a 1 or 0 flag with HeatLoads, Heaters, and Conductors. This preserves the state allowing it be changed during a run, where disabling does not output the logic to SINDA at all. (Keeps same number and order of all objects)
- DON'T: Allow zero value heatloads, heaters, etc to be skipped on output to SINDA. If the object is not output as an expression *and* the value is zero, it will not go through to SINDA. Better to see a heatload or heater with a zero value than not. Outputting as an expression avoids this.
- DON'T: Run any analysis for the first-time before you have performed hand calculations or estimates to know what to expect. If you don't know what to expect, how can you know if the model is right?
- DON'T: Keep aliases in a model once the material or optical is known
- DON'T: Output more timesteps of output than are needed (e.g. last 2 orbits enough, don't need first 8)
- DON'T: Change object Defaults unless you are sure the fields will be updated, multi-edit feature better method
- DON'T: Rerun Radiation Calculations unless necessary (i.e. be careful running CaseSets in different folders)
- DON'T: Change Headers in a user logic block unless you also reset the Header back to the assigned one at the end
- DON'T: Go overboard with submodels. Too many submodels makes Model Browser navigation unwieldy

Miscellaneous Tips

- For construction of the model, it can sometimes be easier to add construction lines based on the CAD snap points, then move them a fixed distance away to build the model. This prevents AutoCAD from spending time searching for snap points over the CAD geometry during model construction. Once complete, move the model back to overlay the CAD by the same fixed distance
- Specification of points does not need to use the object snaps (e.g. end point). To move something 10 units in X, the from point van be entered as 0,0,0 and the to point as 10,0,0. Alternately, the @ sign preceding a point makes that point reference the first point. Polar notation can also be used (dist<angle e.g. @1<45)
- Planar elements can be extruded to make solid elements. If using solid elements, it is often necessary to shell coat them with zero thickness planar elements on the free faces. This allows for radiation (note that radiation properties cannot be applied to solid elements, only planar), contactors, heaters and heatloads to be applied to faces. The thickness should remain zero, since the solid elements should provide the mCp contributions and not the shell coat.
- For tight heater control, the initial inclination may be to reduce the deadband and control with a proportional heater. However, in effect this increases the Pgain term in a controller, which may lead to oscillations instead of better control. May be better to widen the range instead of narrowing it.
- For HR tasks, options exist to add Pre and Post logic. This can be an easy way to disable heatrates by placing and IF statement around the call to assign the heat rates. This has been used to ensure that chained heat rates calls (in time) do not apply a load for the end of the first set and start of the second set.
- If moving objects that are attached to an articulator, care must be exercised. If a node is moved by 1 unit in X and the articulator is also moved by 1 unit in X in a separate command, then the net effect will be that the node will be moved by 2 units in X. Selecting both the nodes and articulator in a single move does not produce this behavior. Alternatively, making the articulator inactive allows it to be moved without impacting the attached objects
- A cleanup process has been developed to prepare models for integration (both source model cleanup and destination model cleanup) as well as the steps to integrate the cleaned models. They are described on the next two slides…

Preparing Model for Integration

1. Turn on all layers and turn visibility on for all objects.
2. Turn off global visibility for all TD objects and delete all unneeded CAD or construction lines
3. Turn on Global visibility for all objects again
4. Delete all TD Geometry not needed for integration at higher level
5. Remove all user nodes not needed at higher level
6. Remove all empty Assemblies/Trackers
7. Remove all empty Material Orienters
8. Remove any Heat Loads, Contactors, Conductors that reference empty tag sets (area = 0)
9. Purge all empty Domain Tag Sets
10. Make backup copy of tdp and rco files (BOL and EOL)
11. Remove any unused Aliases (Materials and Opticals)
12. Delete unused material properties
13. Delete unused optical properties from both BOL and EOL rco files
14. Remove any unneeded LogicObjects
15. Export All CaseSets to temporary file and delete from CaseSet Manager
16. Purge all unused symbols (search in text) multiple time until none remain (excepting hrZZZ)
17. Import CaseSets from Step 15 and check for any Symbols referenced but not defined in Model browser. If found, remove from CaseSet reference or resolve the symbol reference
18. Run TDPurgeBlocks
19. Check for any remaining layers that are not needed (orphaned blocks, empty tdText objects, etc)
20. Might need to delete object by selecting using 'Filter with specified layer, or select from ModelBrowser when viewing by Layer

Integrating Source Model into Destination

1. Use CLASSICINSERT AutoCAD command to import Source file into destination file (this preserved Domain Tag Sets)
2. Import Optical Properties (BOL and EOL files)
3. Import Material properties
4. Import Logic Objects
5. Import Symbols
6. Import Orbits (if needed)
7. Check ModelBrowser for any undefined Optical Properties
8. Check ModelBrowser for any undefined Material Properties
9. Check ModelBrowser for any Referenced, but undefined symbols
10. Check ModelBrowser for any empty Domain Tag sets
11. Manually reconcile CaseSet symbol overrides from Source CaseSets with Destination CaseSets
12. Verify correct Radiation activity (merge Radiation groups if needed)
13. Do a side by side comparison at a submodel level in the Model Browser to see that the same number of Nodes, User nodes, Insulation Nodes, Surfaces, FDSolids, Solid Fes, Contactors, Conductors, Heat Loads, Heaters, and Measures are associated with each submodel from the Source Model and the Integrated model. If differences exist, reconcile the discrepancy.
14. Make thermal connections between source and destination objects (e.g. contactors, conductors, etc)
15. Run sample case to ensure functionality

Miscellaneous Tips

- If a SINDA model crashes with no indication in any out or log files, a method for troubleshooting is to add Status Points into the model.
  - A status point opens a file, writes a unique string to indicate progress, and then immediately closes the file.
  - This technique ensures that the buffer gets flushed to the file.
  - Add as many Status Points as needed to isolate where in the code the issue is occurring

```
HEADER OPERATIONS
F       OPEN(UNIT=667,FILE='..\DEBUG.TXT',STATUS='UNKNOWN')
F       WRITE(667,*) 'GOT TO START OF STDSTL'
F       CLOSE(UNIT=667)
M       CALL STDSTL
F       OPEN(UNIT=667,FILE='..\DEBUG.TXT',STATUS='UNKNOWN')
F       WRITE(667,*) 'GOT TO END OF STDSTL'
F       CLOSE(UNIT=667)
...
HEADER VARIABLES 0, SUB1
F       OPEN(UNIT=667,FILE='..\DEBUG.TXT',STATUS='UNKNOWN')
F       WRITE(667,*) 'GOT TO START OF SUB1.VAR0'
F       CLOSE(UNIT=667)
...SUB1 logic...
F       OPEN(UNIT=667,FILE='..\DEBUG.TXT',STATUS='UNKNOWN')
F       WRITE(667,*) 'GOT TO END OF SUB1.VAR0'
F       CLOSE(UNIT=667)
HEADER VARIABLES 1, SUB1
```

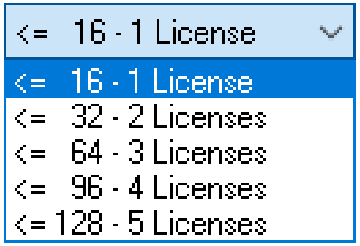
- If Thermal Desktop is not the end software for delivery (e.g. a model conversion is needed) avoid disabling of nodes to make holes, MLI overrides and property overrides. Better to make explicit sub-surfaces to make conversion easier
- If mapping thermal results to a structural model (FEM) for thermal distortion analysis, it is best to associate multiple groups and break both models into smaller subsections to prevent bleedover at interfaces. Use of Domain Tag Sets for mapping in TD model highly recommended. Also, be cautious if mapping tolerance needs to be too large just to map all FEM grid points. Might be better to only map within specified tolerance and allow NASTRAN to “fill in” unmapped grid points.

Breakout models

- Breakout models are a good way to explore design options whose effect is localized to a portion of the design without the need to run a full detailed model
- A key to breakout models is to determine the boundary conditions to be used for evaluation
  - *Conductive Interface Temperatures can often be extracted from the detailed model*
  - *The radiative environment can often be replicated by backloads or sinks if needed*
- One common approach to determine the conductance through a complex part with two interfaces is to utilize a mesher to quickly generate a model, apply a boundary temperature condition at one interface, and a heat load at the other.
  - *The effective conductivity can then be estimated from G = Q / DT, where Q is the heat load applied and DT represents the difference between the average temperature at each interface*
- This can be extended to multiple interfaces using the SuperNetwork feature in Thermal Desktop
  - *A similar approach can be utilized by quickly generating a detailed mesh. Then assign all the nodes <u>not</u> representing an interface as part of the SubNetwork.*
  - *Generation of the CC file will include only the nodes in the SuperNetwork and the effective conductances between them. These can then be extracted from the CC file and utilized in the detailed model*
  - *Internally, it is setting each SuperNetwork node temperature to 1 while all others are set to zero and then using super-position to determine the temperature field, heat flows, and conductances*

Optimizing CPU Resources

- Most modern PCs now feature processor chips with multiple cores, which allows parallel processing to be employed by software and algorithms written to take advantage of this
- Thermal Desktop radiation calculations are inherently parallelizable, with the ray trace for Surface i begin completely independent from Surface i+1
  - *That said, Thermal Desktop will utilize more RadCad licenses for machines with many cores*
  - *1 License (<=16 cores), 2 Licenses (<=32 Cores), 3 Licenses (<=64 cores), etc.*



- On the other hand, SINDA/FLUINT algorithms are not easily broken into multiple threads to take advantage of parallel resources
  - *During Steady State solutions, 2 cores may be used, but Transient solutions currently use a single core*
- To take full advantage of CPU resources when running many cases, it is advised to run all Radiation calculations first (which utilizes all available cores) and only *generate* the SINDA input files
- Once all the radiation files have been generated, the SINDA/FLUINT jobs can then be run in parallel
  - ***Use of the paths.txt file is strongly suggested***
  - *Specifying a file in the same folder as the .inp file named paths.txt and specifying search folders for INSERT files in this file allows a unique Run Folder to be defined that contains only the .inp and the paths.txt file*
  - *Executing the run in this unique folder for each job allows compartmentalization of all the output files per job and avoids any potential file conflicts associated with running jobs in parallel*
  - *The number of jobs that can be submitted in parallel is limited only by the available licenses and memory*
  - *Use of SINDAWIN outside of Thermal Desktop to execute multiple jobs allows for better resource utilization and furthermore does not utilize a license for Thermal Desktop while SINDA is running*

Analyzing the Model Predictions

Modeling vs. Analysis

- There is an important difference between Modeling and Analysis

- Modeling is the generation of predicted behavior
  - This can be done by the computer

- Analysis is the interpretation of those predictions to infer or discover design insufficiencies, strengths, and weaknesses.
  - This should be done by the Engineer, although tools exist to help with the interpretation

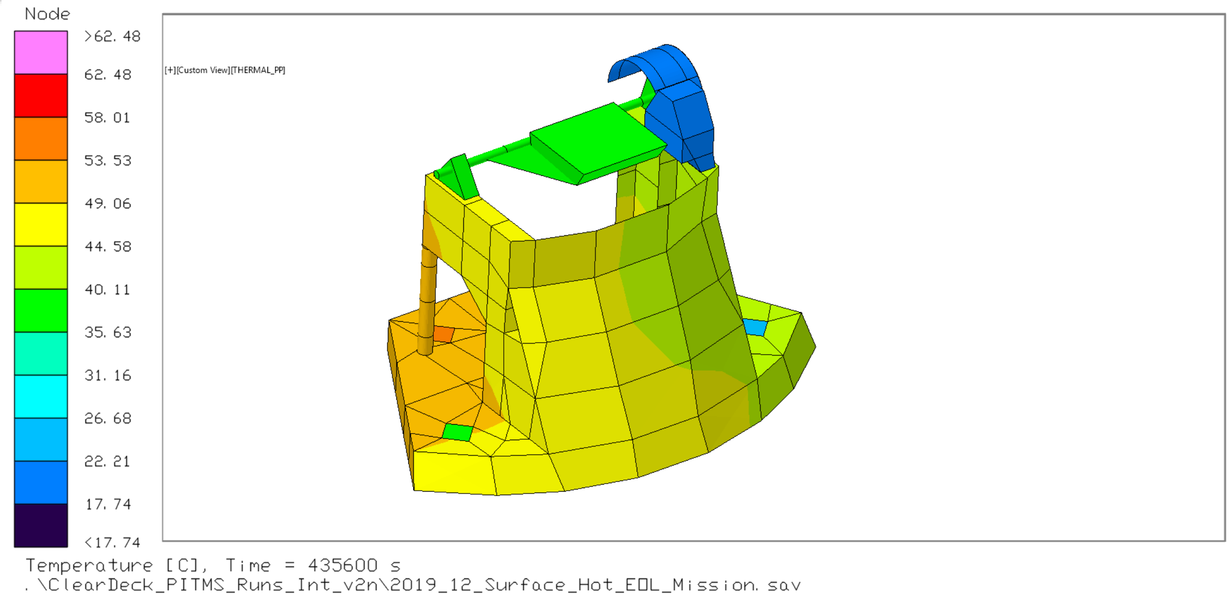
Good Thermal Analysts do more than generate predictions. They study the data to understand the design…

Verify Results

- Verify Results refers specifically to the process of confirming that the model and predictions are consistent with the physics of the design.
- Model correlation is a measure of how accurately the model predicts actual measured values under the same conditions as tested…this happens well after a model is built, much later in a program
- Configuring your model with specific user logic and output can help produce critical results without the need to spend as much effort on post processing, but does require more effort up front
- The first thing to check are temperatures. A contour plot is useful for this, as it will highlight any gradients. Sharp discontinuities in contours are often an indication of a modeling issue…
- If the contours look reasonable (what should be cold is cold, what should be hot is hot), the next thing to compare is the temperatures against design limits. XY plots are good to see the local max and min values over an orbit, but it is hard to do this for every node of importance. Comparison of the hottest temperature and coldest temperature of a component (or group of nodes) over the orbit to the design limit is best accomplished using a spreadsheet and is often represented in a tabular form.
- XY plots are also useful to display the nodal stability and compare against any stability requirements
- For heaters, it is important to confirm that the heater power is sufficient. No more than 70% of the available heater power should be used per GSFC GOLD rules. The required amount of power to maintain a component within limits does not depend on heater size, but duty cycle does…
- A variety of external tools exist for post processing. Thermal Desktop offers XY plotting, Contour Plots, Animated Contour Plots and methods for: querying Max/Mins, writing output text files, and calculating sinks and heat flows post-solution
- ***Lastly, to really understand a design, there is no substitute for investigating heat flows. Heat flows identify areas of a design where better coupling is needed, better isolation is needed, or unexpected heat inputs or leaks are present. This takes a fair bit of time to set up, but the understanding of a model/design is far better after having gone through the effort.***

Typical Thermal Analysis Data Visualizations

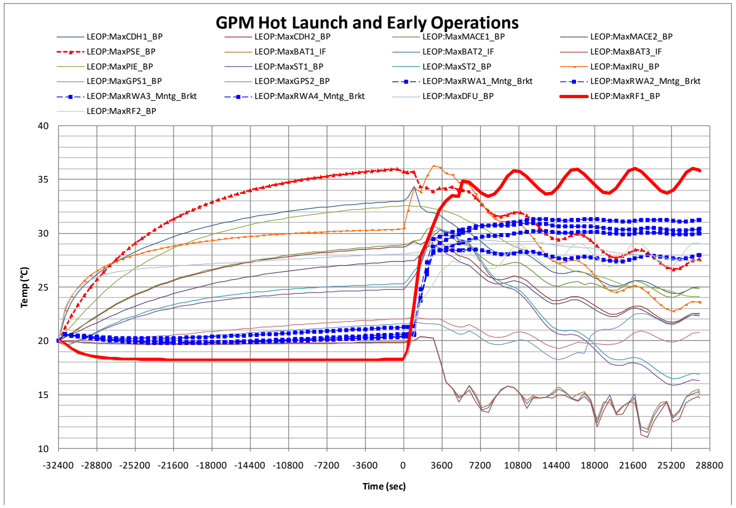
Contour Plot



Heater Power Table

Heater Power Estimates (W)	CB90 Normal Ops	CB00 Safehold	CB00 Normal Ops
Battery	53.49	9.16	8.71
Propulsion	133.88	119.56	81.01
SADDS	59.40	61.31	45.56
HGAS	25.52	30.78	26.74
RF	1.15	0.00	0.00
Avionics	94.61	23.73	0.00
Inst IF	10.16	0.00	8.34
RWA	83.91	54.20	25.39
TOTALS	462.14	298.75	195.75

Data vs. Time Plot

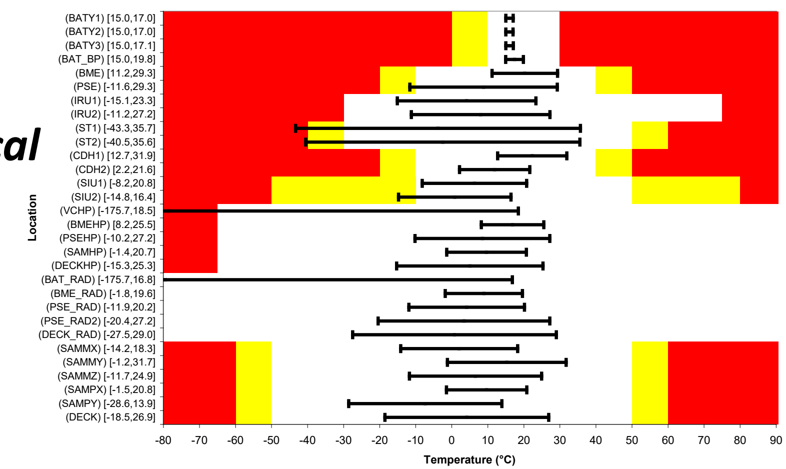


Typical Thermal Analysis Data Visualizations

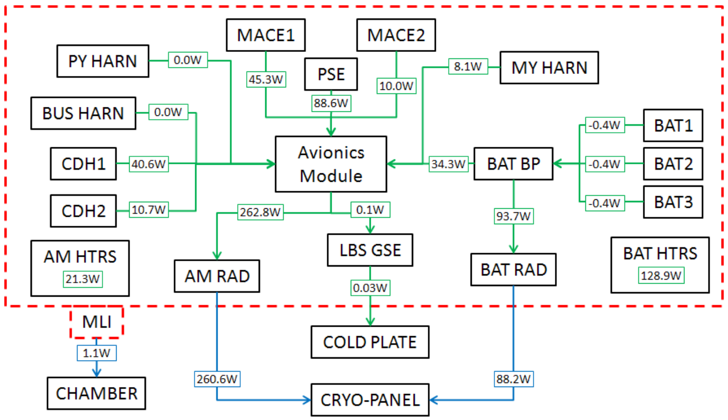
Tabular Data (Critical Components)

Sheet:	NodeInfo						CB00 Safehold			CB90 Normal Ops			HB00 Normal Ops			HB90 Safehold			CB90 DSC			HB00 ARC			HB20 Normal Ops		
Parameter:	Surv_Cold	Op_Cold	GOLD_Cold	GOLD_Hot	Op_Hot	Surv_Hot	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
Node/Group							[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]	[°C]
KaPR																											
Ka_TR_Unit	-30	-20	-15	50	55	60	-8	-7	-6	-8	-8	-8	31	32	33	-5	-4	-2	-11	-9	-8	31	32	33	34	35	36
Ka_TDA	-30	-20	-15	50	55	60	-1	-1	0	-3	-3	-3	29	31	33	-3	-2	-1	-6	-4	-3	29	31	33	32	34	36
Ka_TX_BPF	-30	-20	-15	50	55	60	-2	-2	-1	-6	-6	-5	26	29	32	-2	-1	-1	-8	-6	-5	26	29	32	29	32	35
Ka_RDA	-30	-20	-15	50	55	60	-1	-1	0	-3	-3	-3	29	31	32	-3	-2	-1	-6	-4	-3	29	31	32	31	33	35
Ka_RX_BPF	-30	-20	-15	50	55	60	-1	-1	0	-5	-4	-4	28	29	31	-1	-1	0	-8	-5	-4	28	29	31	31	32	33
Ka_Div/Comb1	-30	-20	-15	50	55	60	-7	-5	-4	-8	-7	-7	26	29	30	-3	1	5	-12	-8	-6	26	29	30	29	31	33
Ka_Div/Comb2	-30	-20	-15	50	55	60	-2	-1	0	-3	-3	-3	28	29	30	-3	-2	-1	-6	-4	-3	28	29	30	30	32	33
Ka_Hyb	-30	-20	-15	50	55	60	-1	0	1	-3	-3	-2	29	30	31	-3	-2	-1	-6	-4	-2	29	30	31	32	33	34
Ka_CPS	-30	-20	-15	50	55	60	-1	0	1	-2	-2	-2	28	30	33	-3	-2	-1	-5	-3	-2	28	30	33	31	33	36
Ka_FCIF	-30	-20	-15	50	55	60	-4	-3	-2	-6	-5	-4	24	29	34	-5	-4	-3	-8	-6	-4	24	29	34	27	32	38
Ka_SCDP_B	-30	-20	-15	50	55	60	-1	0	0	-2	-2	-2	29	30	31	-3	-2	-2	-5	-3	-2	29	30	31	32	33	34
Ka_PS_SW	-30	-20	-15	50	55	60	-6	-5	-4	-6	-6	-6	24	26	27	-8	-7	-4	-9	-6	-4	24	26	27	27	30	32
Ka_IF_Box	-30	-20	-15	50	55	60	-8	-5	-3	-7	-7	-6	21	24	28	-9	-7	-4	-11	-7	-1	21	24	28	23	27	33
Ka_Terminator	-30	-30	-25	55	60	60	-10	-7	-3	-11	-10	-9	27	33	35	-3	5	15	-18	-12	-9	27	33	35	29	35	38
KaPR_Flexures_Instr-Side	-30	-25	-20	40	45	50	-3	1	3	-5	-1	2	22	23	26	7	9	12	-7	-1	2	22	23	26	23	24	29
KaPR_Flexures_SC-Side	-30	-25	-20	40	45	50	-5	1	6	-14	-4	3	6	16	22	15	21	28	-16	-4	3	6	16	22	10	19	25
KuPR																											
Ku_TR_Unit	-30	-20	-15	50	55	60	-7	-2	1	-8	-2	3	30	32	34	33	35	37	-8	-2	5	30	32	34	30	32	34
Ku_TDA	-30	-20	-15	50	55	60	-7	-2	2	-10	-3	3	28	30	32	36	37	38	-10	-4	3	28	30	32	28	30	33
Ku_RX_BPF	-30	-20	-15	50	55	60	-7	-3	2	-10	-4	2	28	30	32	36	37	39	-10	-5	2	28	30	32	28	30	32
Ku_Div/Comb1	-30	-20	-15	50	55	60	-11	-3	13	-12	-2	14	22	28	31	28	36	43	-12	-2	14	22	28	31	22	28	31
Ku_Div/Comb2	-30	-20	-15	50	55	60	-9	-2	11	-11	-1	13	25	28	31	34	37	41	-11	-2	13	25	28	31	25	28	31
Ku_Hyb	-30	-20	-15	50	55	60	-8	-3	2	-11	-6	0	26	29	30	37	40	42	-12	-7	0	26	29	31	26	29	31
Ku_CPS	-30	-20	-15	50	55	60	-8	-3	2	-10	-2	5	28	30	33	34	36	37	-10	-3	5	28	30	33	28	30	34
Ku_FCIF	-30	-20	-15	50	55	60	-8	-3	3	-10	-2	6	27	29	31	33	34	35	-10	-2	7	27	29	31	27	29	31
Ku_PS_SW	-30	-20	-15	50	55	60	-7	-3	1	-10	-5	0	29	31	32	37	39	40	-10	-6	0	29	31	32	29	31	33
Ku_IF_Box	-30	-20	-15	50	55	60	-10	-7	-6	-12	-9	-5	30	31	32	29	30	31	-12	-8	-4	30	31	33	31	32	34
Ku_Terminator	-30	-30	-25	55	60	60	-12	-5	-1	-12	-7	-3	24	30	33	27	32	36	-12	-7	-2	24	30	34	24	31	34
KuPR_Flexures_Instr-Side	-30	-25	-20	40	45	50	-10	-1	5	-18	-5	1	9	18	22	16	23	31	-19	-6	1	9	18	22	12	20	26
KuPR_Flexures_SC-Side	-30	-25	-20	40	45	50	0	2	4	-2	1	3	21	22	24	11	13	14	-2	1	3	21	22	24	21	22	23
GMI																											
ICA	-40	-15	-15	48	53	75	10	20	32	10	19	30	25	37	47	17	28	40	9	19	31	25	37	**48**	21	31	41
SMA	-35	0	0	45	50	55	4	8	18	5	10	23	7	14	22	6	13	24	5	10	23	7	14	22	17	21	28
EPC	-35	-10	-10	55	60	85	-13	-13	-13	26	26	26	38	39	40	-10	-10	-9	25	25	26	38	39	41	43	44	45
EDC	-35	-10	-10	45	50	85	-13	-13	-13	7	7	7	17	17	18	-11	-11	-10	6	6	7	17	17	19	22	22	23
RF_Boxes	-35	-10	-10	40	45	85	-16	-9	-2	-1	7	14	9	18	26	-12	-3	0	-1	7	14	9	18	27	14	24	31
HF Mixer	-40	-10	-10	50	55	90	-4	-2	-1	-3	-1	3	13	16	18	-3	-3	-2	-3	-1	3	13	16	20	22	25	28
Hot Load	-60	-33	-28	52	57	120	-9	-8	-8	-2	-2	-2	9	10	11	-6	-5	-3	-3	-2	0	9	10	12	23	24	26
Reflector	-100	-100	-95	95	100	150	-40	32	83	-8	0	5	-42	6	60	98	115	129	-28	-4	5	-42	6	60	-40	13	60
Cold Sky Reflector	-100	-100	-95	127	132	132	-34	-29	-23	-41	-41	-41	-21	-10	1	-31	-28	-26	-41	-37	-30	-21	-10	14	-8	4	22
RDA	-140	-140	-135	110	115	115	-36	15	53	25	28	32	-40	3	31	22	59	84	25	28	37	-40	2	35	-35	12	45
RDA LR	-80	-80	-75	145	150	150	-34	-9	35	-63	-22	5	-29	-10	11	-17	24	113	-70	-24	5	-29	-10	11	-31	-5	15
RDA LR Struts	-70	-70	-65	65	70	80	-18	-14	-5	-41	-21	-2	-12	-9	-6	-2	2	6	-42	-22	-2	-12	-9	-6	-10	-4	-1
IBS LR	-60	-60	-55	115	120	120	-19	-8	18	-24	-8	16	-11	2	34	-16	-1	26	-24	-8	17	-11	2	35	-6	6	29
Cal Arm LR	-100	-80	-75	145	150	180	-31	-12	-5	-7	13	43	-17	11	51	-14	-4	1	-7	14	51	-17	11	51	-2	29	71

Graphical Table

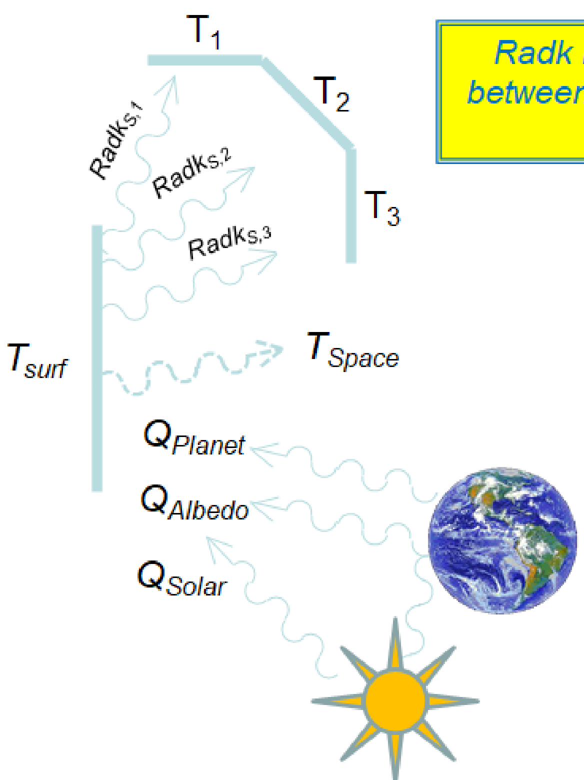


Heat Flow Diagrams



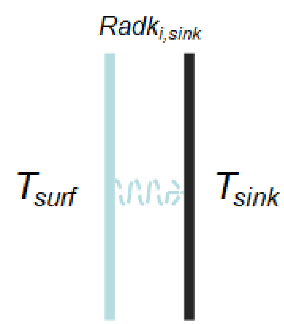
Equivalent Sink Temperature

- ***Equivalent Sink Temperatures are a useful byproduct derived from thermal analysis***
  - For any surface on a spacecraft, there is a *single* temperature that represents the external scene
  - This technique is generally applied to major thermal surfaces, such as radiators
    - *MLI, apertures, etc., which are not major design heat rejection paths, would not need this*
  - The sink temperature is a very useful value for sizing calculations in spreadsheet based or systems tools
  - This is essentially how Thermal determines cryo-panel or heater panel temperatures for testing



Radk is a Radiation Coupling between two surfaces defined as $Radk_{ij} = A_i \varepsilon_i B_{ij}$

=



$$T_{i,Sink}^4 = \frac{\sigma \sum_{j}^{1,N} Radk_{ij} T_j^4 + Q_{env}}{\sigma \sum_{j}^{1,N} Radk_{ij}}$$

$$Radk_{i,Sink} = \sum_{j}^{1,N} Radk_{ij}$$

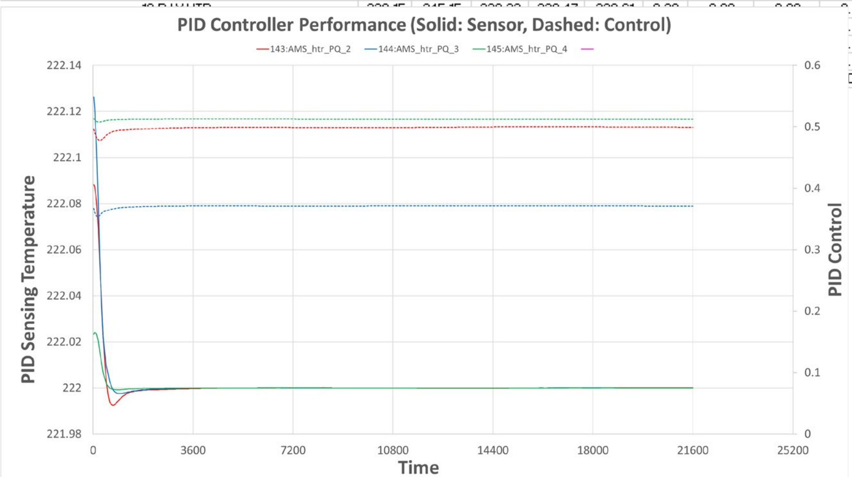
Post-Processing Steps

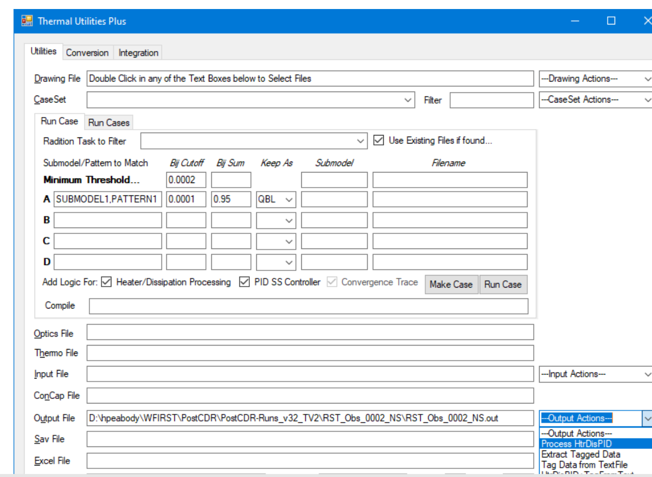
- Determine the Data Products to produce
  - Temperature vs Limit Tables
  - Heater Power Tables
  - Temperature/Heat vs. Time XY Plots
  - Contour Plots
  - Heat Flow Diagrams
  - Backloads or Equivalent Sink Temperatures
- Determine what data needed is from input and output files:
  - Temperatures (almost always)
  - Nodal Heat (often)
  - Conductors (sometimes)
  - Capacitances (sometimes)
  - Registers (For Heater Processing, although Nodal Heat may be sufficient)
  - Environmental Loads
  - Radiation Couplings (particularly to Space)
    - CYGWIN or Text Editor with robust search capability can make this easier
- Determine which locations (nodes) are needed
  - This is where node ranges and specific numbering can really help
  - Can MLI nodes be ignored?
  - Consider if groups of nodes may be averaged to get bulk effects
- Determine what hand calcs can help verify data validity
  - Radiator Heat Flow to Space
  - Radiator View Factor Computations/Radk Sums
  - Heat flows across critical interfaces (major paths, isolating paths)
  - Overall system energy balance

GSFC Developed Post-Processing Application

- Recent developments at GSFC use the OpenTD API to process the CondCap file and identify all the HeatLoads, Heaters, and PID Controllers in the model. The associated logic is processed to create an include file prior to execution that outputs relevant information to the output file for each of these objects
- After model execution, the data is extracted from the output file and imported into a custom Microsoft Excel® template file for further evaluation
- This Workbook contains a Summary page with Temperatures, Heat, Duty Cycles, etc for all the HeatLoads, Heaters and PID controllers as well as basic plotting capabilities

Starting Timestep	18000	RowIndex	303								
Heater	OnTemp	OffTemp	MinTemp	AvgTemp	MaxTemp	DeltaT	MinPwr	AvgPwr	MaxPwr	AvailPwr	DutyCycle	Element
SYSTEM							3134.34	4964.14	5605.14			
2:JittDamp/Boom Heater 1	270.15	271.15	269.59	270.71	271.76	2.17	0.00	21.75	45.00	45.00	48.3%	BUS:Op_Htr
3:Damp 7-1	270.15	271.15	270.71	270.82	270.92	0.21	27.00	27.00	27.00	27.00	100.0%	BUS:Op_Htr
4:Ant Brckt Htr for AzActuator	283.15	290.15	193.08	193.15	193.22	0.14	0.00	0.00	0.00	0.00	0.0%	BUS:Op_Htr
5:Azimuth Act Ext Heater	238.15	245.15	229.43	229.56	229.69	0.26	0.00	0.00	0.00	0.00	0.0%	BUS:Op_Htr
6:Elevation Act Ext Heater	238.15	245.15	246.79	247.07	247.34	0.55	0.00	0.00	0.00	0.00	0.0%	BUS:Op_Htr
7:El Actuator (Y-axis) Heater	238.15	245.15	246.79	247.07	247.34	0.55	0.00	0.00	0.00	9.00	0.0%	BUS:Op_Htr
8:Az Actuator (X-axis) Heater	238.15	245.15	229.43	229.56	229.69	0.26	9.00	9.00	9.00	9.00	100.0%	BUS:Op_Htr
9:RJ Hinge Htr	238.15	245.15	263.26	263.27	263.27	0.00	0.00	0.00	0.00	12.75	0.0%	BUS:Op_Htr
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	.00	9.00	100.0%	BUS:Op_Htr
									.00	9.00	0.0%	BUS:Op_Htr
									.00	0.00	0.0%	BUS:Op_Htr
									.00	62.90	0.0%	BUS:Surv_Htr
									0.00	100.00	8.3%	BUS:Op_Htr





	A	B	C	D	E	F	G	H
1	Observatory	4986.18	6940.51	7579.86				
2	**Component**	MinPwr	AvgPwr	MaxPwr	MinTemp	AvgTemp	MaxTemp	Element
3	SYSTEM	1851.85	1976.38	2003.25				
4	Battery	0	0	0	287.19	287.69	288.15	BUS:Dis
5	CDH A	63.0135	63.0135	63.0135	281.4	281.41	281.42	BUS:Dis
6	CDH B	12.1505	12.1505	12.1505	271.29	271.3	271.31	BUS:Dis
7	Diplexer	1.14	1.14	1.14	264.14	264.29	264.43	BUS:Dis
8	Diplexer - RF Switch A	0.95	0.95	0.95	264.64	264.81	264.97	BUS:Dis

Post-Processing the Sample Problem

- ***The predictions from the sample problem need to be compared to their given limits to see if the design meets requirements***
- ***This is often easiest to do in a spreadsheet, like Microsoft Excel***
- ***First, each of the user .tbl files needs to be imported onto its own sheet***
  - The easiest way to do this is to open each all the .tbl files in a text editor that supports multiple files
  - Create a worksheet for each case, and copy the text from the file onto the worksheet
  - Use the Text to Columns feature under Data in Excel to parse based on Space and Treat Consecutive Delimiters as One
- ***Next, select all the sheets and perform the following actions:***
  - Delete Column A (which should be blank)
  - Cell A1 text: "Min", Cell A2 text: "Max"
  - In Cell B1 formula: "=MIN(B6:B100)". Fill Right through Column AB
  - In Cell B2 formula: "=MAX(B6:B100)". Fill Right through Column AB
- ***Now create a new Worksheet with the component listing and limits***
  - Row 1 after limits: H00, H15, H30…H90, C00, C15…C90,S00,S15,…@S0
  - Row 2 after limits: Max of any H case, Min for any C/S cases
  - Row 3 after limits: =INDIRECT("'" & F$1 & "'!R" & IF(F$2="Min",1,2) &"C" & MATCH($A3,INDIRECT("'" & F$1 & "'!R4C1:R4C100",FALSE),0),FALSE)
- ***Add two more column to the right of the columns for each case***
  - Row 2, 1st Col: "Max", Row 2, 2nd Col: "Min"
  - Row 3, 1st Col: "=max(E3:L3)", Fill Down, Row 4, 1st Col: "=min(m3:Z3)", Fill Down

Component	Surv Low (°C)	Op Low (°C)	Op High (°C)	Surv High (°C)
A	-20	-10	40	50
B1	-20	-10	40	50
B2	-20	-10	40	50
B3	-20	-10	40	50
Cmin	-20	-10	40	50
Cmax	-20	-10	40	50
D1	-44	-34	71	81
D2	-44	-34	71	81
D3	-44	-34	71	81
E	-20	-10	40	50
F	-20	-10	40	50
G	-20	-10	40	50
Hmin	0	10	30	40
Hmax	0	10	30	40
I	-20	-10	40	50
J	-20	-10	40	50
K	-20	-10	40	50
L	-20	-10	40	50
M	-20	-10	40	50
N1min	2	5	25	30
N1Max	2	5	25	30
N2Min	2	5	25	30
N2Max	2	5	25	30
N3Min	2	5	25	30
N3Max	2	5	25	30
N4Min	2	5	25	30
N4Max	2	5	25	30

Evaluating the Predictions

- ***What do those Excel functions do?***
  - INDIRECT Excel function: return Excel object for evaluated string (e.g. "A1" would return the value in cell A1, "A1:B30" would return the range A1:B30
  - MATCH returns index in Search range where Search Text is found
- ***Now, with model data, the "Thermal Analysis" portion can begin. Up to now, the effort could more correctly be described as "Thermal Modeling". But the "Analysis" portion is where the engineer is needed to interpret the data and make decision about design acceptability or if possible design modifications are needed.***
- ***Component_N looks badly outside of limits, with max and min temperatures well beyond the limits. Recall that this was a thin walled, low conductivity component with heaters. It is likely that the heater design needed to be revisited.***
- ***Numerous components are near or slightly below their operational limit. As most of these are mounted to Deck_1, it suggests an operational heater may be needed***
- ***Component_C is also predicting higher than the limit***

Component	SurvLow (°C)	OpLow (°C)	OpHigh (°C)	SurvHigh (°C)	Min	Max
A	-20	-10	40	50	-6.0	21.0
B1	-20	-10	40	50	-13.4	11.7
B2	-20	-10	40	50	-12.9	12.2
B3	-20	-10	40	50	-10.2	15.5
Cmin	-20	-10	40	50	-3.6	25.0
Cmax	-20	-10	40	50	17.7	48.9
D1	-44	-34	71	81	-11.6	30.2
D2	-44	-34	71	81	-8.0	27.0
D3	-44	-34	71	81	-14.2	13.6
E	-20	-10	40	50	-14.9	11.2
F	-20	-10	40	50	-12.4	16.7
G	-20	-10	40	50	-9.4	16.7
Hmin	0	10	30	40	8.4	15.8
Hmax	0	10	30	40	16.7	25.7
I	-20	-10	40	50	-15.2	10.0
J	-20	-10	40	50	-12.1	15.2
K	-20	-10	40	50	-19.1	20.9
L	-20	-10	40	50	-16.1	20.7
M	-20	-10	40	50	-16.1	10.0
N1min	2	5	25	30	-8.8	23.3
N1Max	2	5	25	30	9.0	48.0
N2Min	2	5	25	30	-8.8	23.3
N2Max	2	5	25	30	9.1	42.9
N3Min	2	5	25	30	-8.8	23.3
N3Max	2	5	25	30	8.9	42.0
N4Min	2	5	25	30	-8.8	23.3
N4Max	2	5	25	30	9.1	42.9

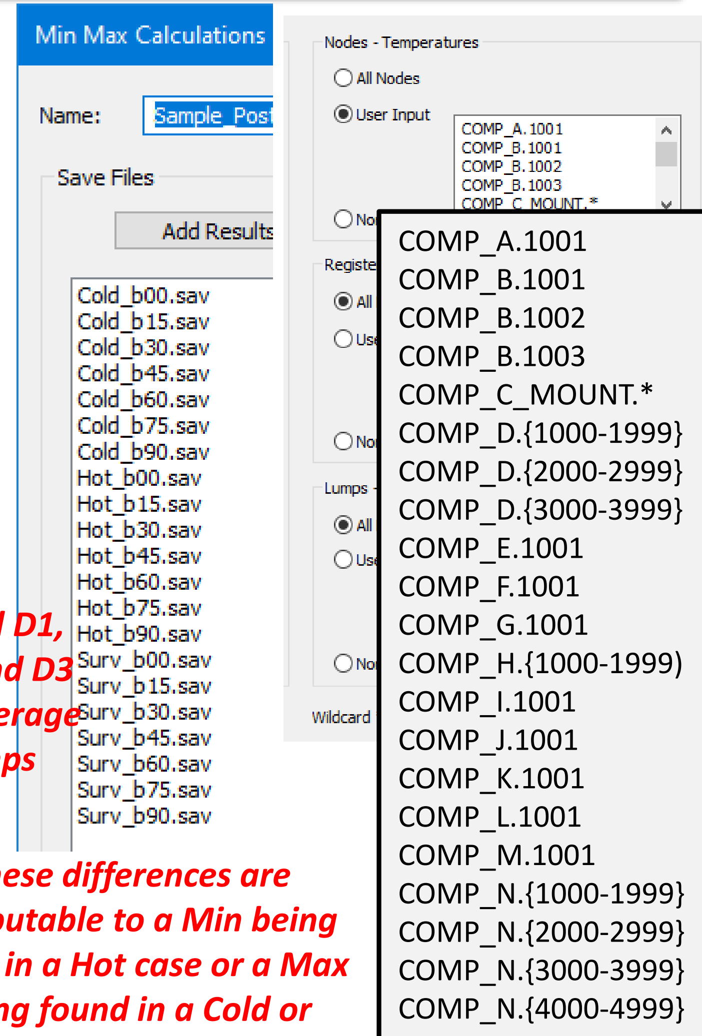
Post-Processing the Sample Problem (Alt)

- ***Alternately, the sav files from Thermal Desktop could be polled to gather similar information. Adding a Post Processing-Find Results Max Min task could be used to gather the Min and Max data for a table (Note Temperatures/Registers Only).***
- ***This creates 7 files in a Min_Max folder***
  - RegisterMinMax and RegisterMinMaxPerSave
  - SubmodelIdMinMax and SubmodelIdMinMaxPerSave (Nodes)
  - SubmodelIntegratedMinMaxPerSave (Average of Submodel)
  - SubmodelMinMax and SubmodelMinMaxPerSave (Submodel)
- ***In this case, the data in the SubmodelIDMinMax file is what should be imported. Going through a similar exercise as before to populate the table yields some slight discrepancies***

Component	Min	TD Min	Max	TD Max
A	-6.0	-6.0	21.0	21.0
B1	-13.4	-13.4	11.7	11.7
B2	-12.9	-12.9	12.2	12.2
B3	-10.2	-10.2	15.5	15.5
Cmin	-3.6	-3.6	25.0	25.0
Cmax	17.7	17.7	48.9	48.9
D1	-11.6	-11.7	30.2	30.2
D2	-8.0	-4.4	27.0	23.9
D3	-14.2	-4.4	13.6	-3.3
E	-14.9	-14.9	11.2	11.2
F	-12.4	-12.4	16.7	16.7
G	-9.4	-9.4	16.7	16.7
Hmin	8.4	7.7	15.8	15.8
Hmax	16.7	16.2	25.7	25.7

Component	Min	TD Min	Max	TD Max
I	-15.2	-15.2	10.0	10.0
J	-12.1	-12.1	15.2	15.2
K	-19.1	-19.1	20.9	20.9
L	-16.1	-16.1	20.7	20.7
M	-16.1	-16.1	10.0	10.0
N1min	-8.8	-8.8	23.3	23.9
N1Max	9.0	8.2	48.0	51.1
N2Min	-8.8	-8.8	23.3	23.6
N2Max	9.1	8.7	42.9	46.8
N3Min	-8.8	-8.7	23.3	22.9
N3Max	8.9	8.5	42.0	46.1
N4Min	-8.8	-8.6	23.3	22.7
N4Max	9.1	8.5	42.9	46.1

Recall D1, D2, and D3 are average temps



These differences are attributable to a Min being found in a Hot case or a Max being found in a Cold or Survival Case using Thermal Desktop's Post Processing

Checking the Predicts

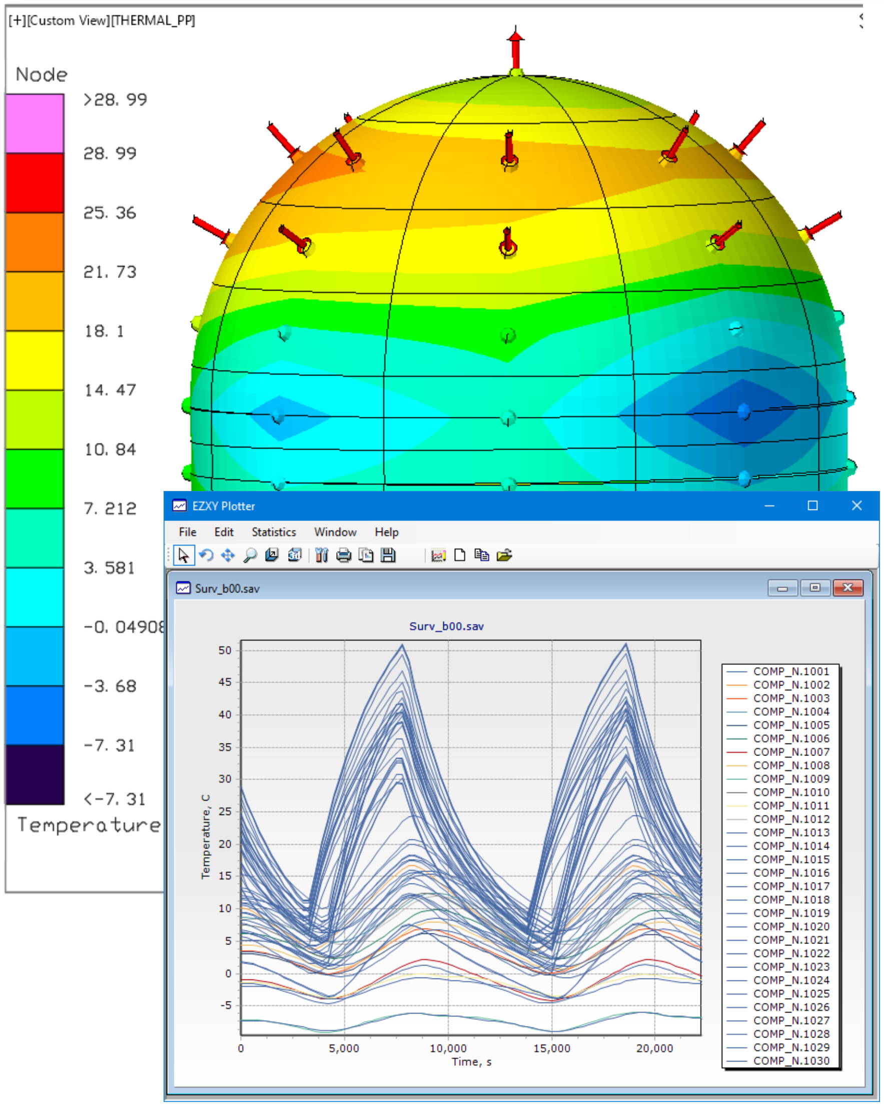
- *As a sanity check of the predicts, an investigation of the heat flows is advisable. From the predicts, it appears that a Hot b75 produces many of the hottest predicts. Rerun this case if needed and ensure that the output is set to All for the save file. This allows the heat flows to be computed by Thermal Desktop*
- *Heat flows can be done through Post Processing-QFLOW From Results, but sometimes it is easier to use the Model Browser-Options-Heat Flow Between Submodels option. In this case, specify Into Submodel as DECK_1 and From Submodel as (GLOBAL)*
- *Recall that Comp A, B (x3), E, F, G, I, J, K, L, and M were all mounted to DECK_1. Summing up their orbit average power and accounting for the power scale factor yields 243 W. Including 114 W from Comp C produces 357 W total*
- *Considering heat gained or lost through the MLI as negligible, the 357 W agrees fairly well with the 378 W rejected to space suggesting a reasonable solution (within 6%)*
- *Another sanity check is to sum all the radks between DECK_1 non mli nodes and space. This effort produced about 1560 in^2. Comparing to $0.89*\{\pi(28.5)^2 - (21.7x11.8)\}$ yields an effective view factor of about 76% for the radiator.*
- *Taking an average temperature of about 11 C for DECK_1, 3.66E-11 * 1560 * { (11+273)^4 – (4)^4} produces 371 W radiated to space. (Good Agreement)*
- ***Hand calcs and first principles hold !!***

Heat Flow Into DECK_1 from:

Submodel	Total Sum/Count	Linear/Count	Radiative/Count
COMP_A	85.8000 / 13	85.8000 / 13	0. / 0
COMP_B	50.7000 / 21	50.7000 / 21	0. / 0
COMP_D	1.5703 / 1123	0. / 0	1.5703 / 1123
COMP_E	1.3000 / 11	1.3000 / 11	0. / 0
COMP_F	10.4000 / 5	10.4000 / 5	0. / 0
COMP_G	15.6000 / 9	15.6000 / 9	0. / 0
COMP_H	-8.1988 / 1427	0. / 0	-8.1988 / 1427
COMP_I	44.2000 / 21	44.2000 / 21	0. / 0
COMP_J	11.7000 / 2	11.7000 / 2	0. / 0
COMP_K	7.5833 / 5	7.5833 / 5	0. / 0
COMP_L	16.4667 / 6	16.4667 / 6	0. / 0
COMP_M	2.6000 / 10	2.6000 / 10	0. / 0
DECK_1	1.568e-14/ 2974	4.843e-15/768	-1.604e-16/ 2206
DECK_2	33.3669 /18704	0. / 0	33.3669 /18704
ESPA	29.2116 / 5904	22.6403 / 40	6.5712 / 5864
INST_3	-0.0006149/ 6	0. / 0	-0.0006149/ 6
INST_BRACKETS	0.2046 / 264	0. / 0	0.2046 / 264
SPACE	-378.1775 / 207	0. / 0	-378.1775 / 207

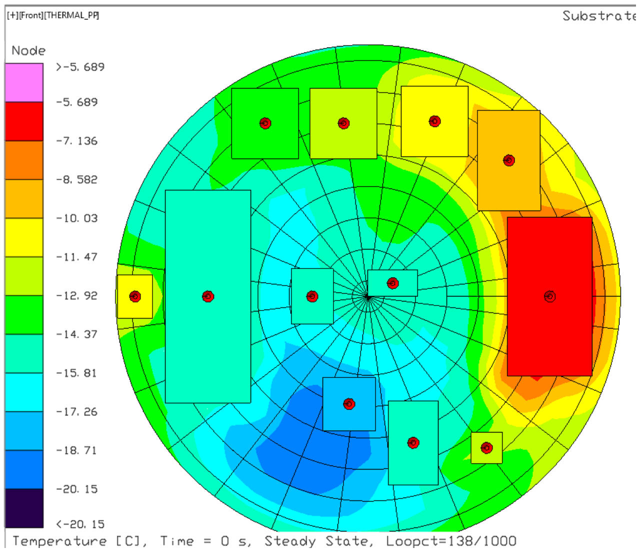
Heat Flow numbers are positive flowing into DECK_1

Analyzing the Sample Design – Comp N

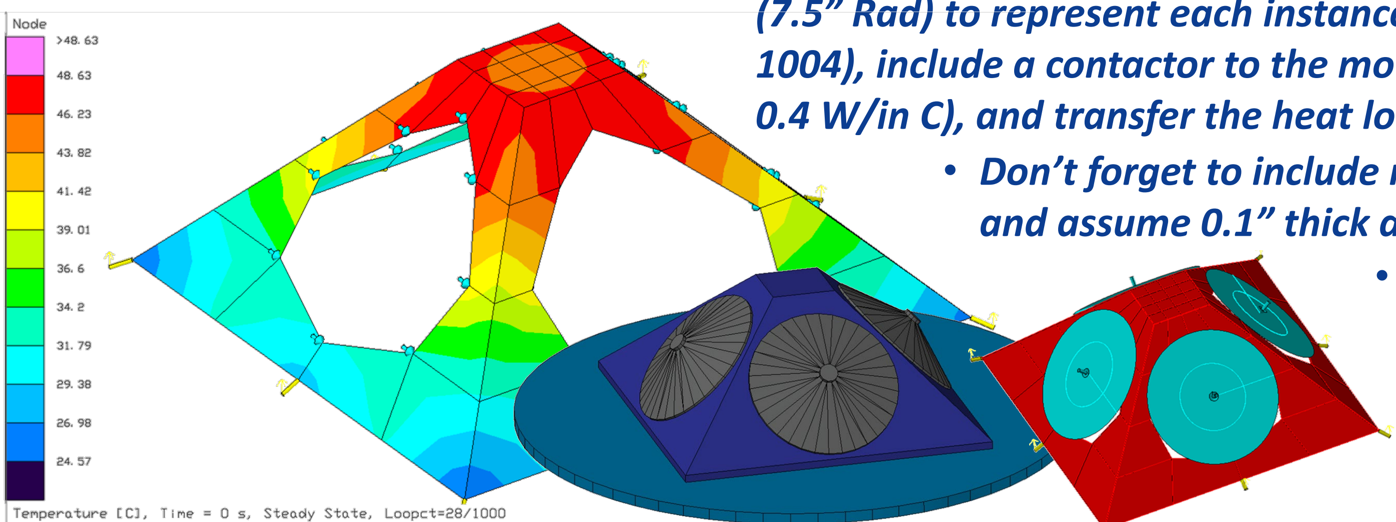


- ***Displaying a contour plot of Component N1 shows large gradients throughout, with high temperatures near the heaters and low temperatures at the deck interface***
- ***An XY plot shows the high temperatures over time as the heater is on, while the cold spots remain nearly the same.***
- ***This strongly suggests that the Top and Bottom heater approach will not work and a Top, Middle, Bottom approach is needed***
- ***A contour plot of Deck 1 also shows the coldest spot, which would be the ideal location for an operational heater sensing point***
- ***Make the necessary updates to the Component N heaters***

Analyzing the Sample Design – Comp C

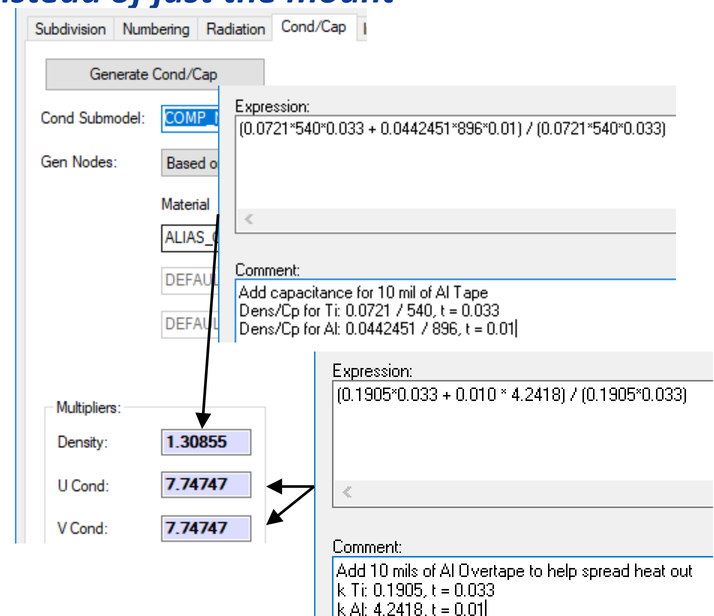


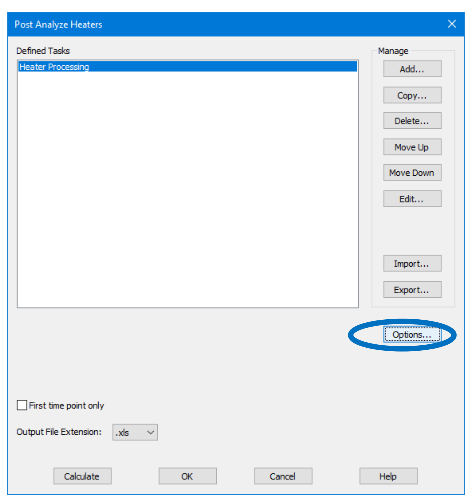
- *Displaying a contour plot of Deck_1 for the Surv_b90 case shows the coldest spot on the deck and the influence on the mounted components*
- *Adding a heater near the cold spot (75 W, -2/5 C) should help*
- *Investigating Component_C for the Hot_B75 case shows the hot spot near the top of the pyramid. With heat driven towards the base and a uniform distribution of heat around the circular openings, a hot spot near the top is not surprising*
- *But it neglects the baseplate conduction effects of the actual components and that impact on the mount. The choice to model only the mount was perhaps not optimum. Add a disc (7.5" Rad) to represent each instance of Component C (1001-1004), include a contactor to the mount (Edge type, Per length, 0.4 W/in C), and transfer the heat load to the new surfaces.*
  - *Don't forget to include radiation (Black Anodize) and assume 0.1" thick aluminum*
    - *Also update the custom output logic and PostProcessing task to output COMP_C instead of COMP_C_MOUNT*

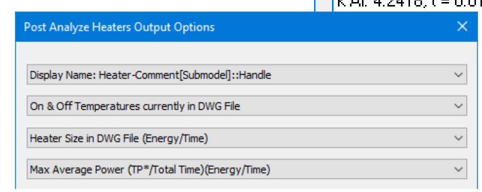


Post-Updates (Checking Against Requirements)

- *A 75 W heater (-2 to 5 C) was added to the Deck 1 for the avionics that were too cold*
- *10 W heaters were added to the mid band of Component N and the Top and Bottom were reduced to 3 W. 10 mils (0.01") of aluminum tape was also added to component N, since the thin walled titanium resulted in numerous hot spots. The four discrete mounting points for Component N to Deck 3 also created gradients, so the U Scaling was removed to represent better contact around the entire circumference*
- *Component C was modeled with actual component representations instead of just the mount*
- *All 21 cases were rerun and the Excel post processing was expanded to consider the possibility of minimum temperatures occurring in Hot Cases and maximum temperatures occurring in Cold/Survival*
- *A check of Heater Power usage was also performed using the Post Processing-Analyze Heaters From Results option. Using the Options button for Heater Processing, the Setpoints, Heater Size, and Max Average Power were output*
- *As a result of the high duty cycles (>70%) for Comp N, the heater size was increased from 10 W to 15 W*
- *Furthermore, with the added aluminum tape effect, the top and bottom heater on Comp N never came on and so they were removed from the model.*
- *With the update to the Component N heaters, the model was run again for all 21 cases.*



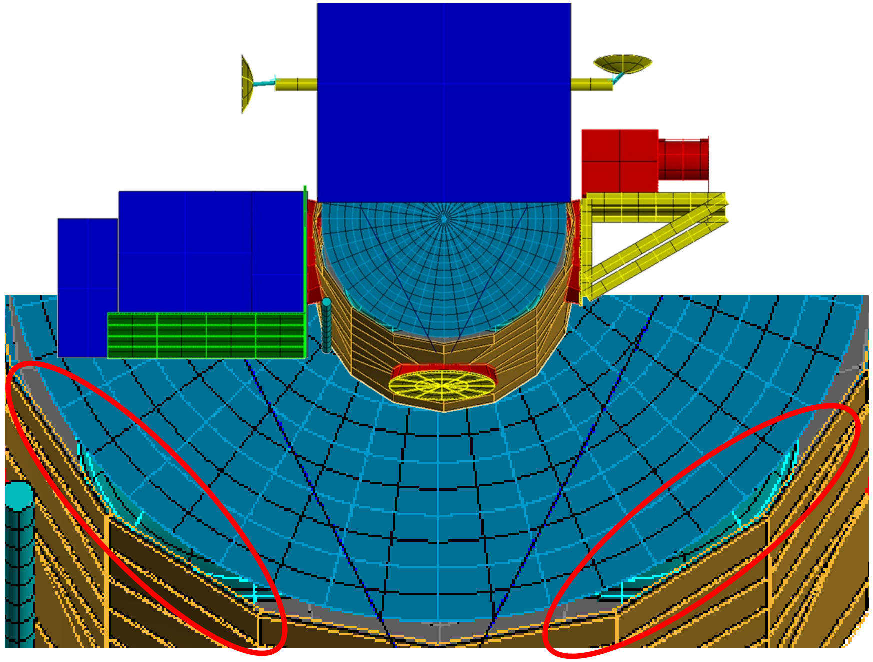




Troubleshooting Hot Exceedances

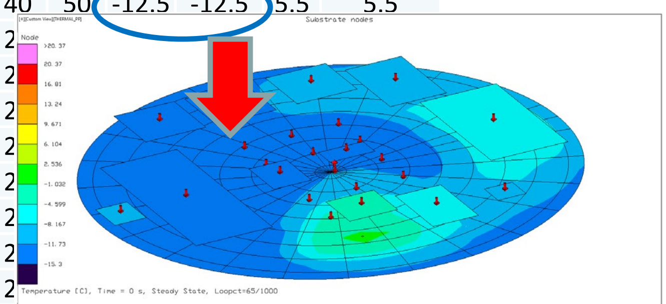
Component	Surv Low (°C)	Op Low (°C)	Op High (°C)	Surv High (°C)	Min	TD Min	Max	TD Max
A	-20	-10	40	50	-2.1	-2.0	21.0	21.0
B1	-20	-10	40	50	-9.4	-9.4	11.8	11.8
B2	-20	-10	40	50	-9.1	-9.1	12.3	12.3
B3	-20	-10	40	50	-6.6	-6.6	15.5	15.5
Cmin	-20	-10	40	50	8.6	8.6	34.3	34.3
Cmax	-20	-10	40	50	8.8	8.8	35.0	35.0
D1	-44	-34	71	81	-8.6	-8.7	30.2	30.2
D2	-44	-34	71	81	-4.0	0.5	27.1	24.1
D3	-44	-34	71	81	-9.4	-1.6	13.6	-0.3
E	-20	-10	40	50	-8.8	-8.8	11.3	11.3
F	-20	-10	40	50	-5.5	-5.5	16.8	16.8
G	-20	-10	40	50	-5.7	-5.7	16.7	16.7
Hmin	0	10	30	40	8.5	8.5	15.8	15.8
Hmax	0	10	30	40	16.7	16.5	25.7	25.7
I	-20	-10	40	50	-9.7	-9.7	10.1	10.1
J	-20	-10	40	50	-7.3	-7.3	15.3	15.3
K	-20	-10	40	50	-2.1	-2.1	21.0	21.0
L	-20	-10	40	50	-5.0	-5.0	20.9	20.9
M	-20	-10	40	50	-9.3	-9.3	10.0	10.0
N1min	2	5	25	30	6.7	6.7	24.7	24.7
N1Max	2	5	25	30	8.5	8.5	27.4	27.4
N2Min	2	5	25	30	6.7	6.6	24.7	24.6
N2Max	2	5	25	30	8.4	8.4	27.3	27.3
N3Min	2	5	25	30	6.7	6.3	24.7	23.8
N3Max	2	5	25	30	8.8	8.7	26.4	26.4
N4Min	2	5	25	30	6.7	6.4	24.7	23.6
N4Max	2	5	25	30	8.7	8.5	26.3	26.3

- ***Post processing this current set of predicts shows that Comp N exceeds limits. Further investigation shows this only occurs during the Hot b75 case.***
- ***This strongly suggests that this may be driven by heating from the orbital environment. Viewing the model from the sun, shows some direct solar loading is on Comp N at the first timestep.***
- ***The closeout is not quite doing its job. Increase the radius from 28.5 to 30 and rerun…***



Troubleshooting Cold Exceedances

Component	Surv Low (°C)	Op Low (°C)	Op High (°C)	Surv High (°C)	Min	TD Min	Max	TD Max
A	-20	-10	40	50	-5.1	-5.1	15.5	15.5
B1	-20	-10	40	50	-13.1	-13.1	5.8	5.8
B2	-20	-10	40	50	-12.6	-12.6	6.4	6.4
B3	-20	-10	40	50	-9.7	-9.7	9.6	9.6
Cmin	-20	-10	40	50	5.3	5.3	26.3	26.4
Cmax	-20	-10	40	50	6.0	6.0	26.5	26.5
D1	-44	-34	71	81	-6.3	-6.3	18.4	18.4
D2	-44	-34	71	81	-2.0	-3.5	16.0	16.1
D3	-44	-34	71	81	-13.8	-4.5	10.0	-3.5
E	-20	-10	40	50	-12.0	-12.0	6.4	6.4
F	-20	-10	40	50	-8.8	-8.8	12.1	12.1
G	-20	-10	40	50	-8.7	-8.7	10.8	10.8
Hmin	0	10	30	40	7.8	7.8	15.8	15.8
Hmax	0	10	30	40	16.7	16.4	25.7	25.7
I	-20	-10	40	50	-13.1	-13.1	5.2	5.2
J	-20	-10	40	50	-11.4	-11.4	9.2	9.2
K	-20	-10	40	50	-3.6	-3.6	17.2	17.2
L	-20	-10	40	50	-7.6	-7.6	16.8	16.8
M	-20	-10	40	50	-12.5	-12.5	5.5	5.5
N1min	2	5	2					
N1Max	2	5	2					
N2Min	2	5	2					
N2Max	2	5	2					
N3Min	2	5	2					
N3Max	2	5	2					
N4Min	2	5	2					
N4Max	2	5	2					



- ***Closing out the gap had the desired effect of getting all the hot temperatures within limits over the mission, but now some of the components are not meeting their minimum operating requirements***
- ***This is a likely effect of a modeling error, where the gap allows some solar or albedo energy in which provided some warming in colder cases***
- ***This loss of erroneous heat needs to be compensated by the addition of a second heater on Deck_1***
- ***Add a heater where indicated (75 W, -2/5) and increase each of the Comp_N heaters to 20 W***

Name	On (C)	Off (C)	Size (W)	Avg (W)	DC (%)
Heater-Component H Surv Heater (17/22 75 W)[COMP_H][COMP_H_Srv]	17	22	75	52.8	70.5
Heater-Comp N1 - Mid (7/14 15 W)[COMP_N][N_1_Mid]	7	14	15	11.2	74.5
Heater-Comp N2 – Mid (7/14 15 W)[COMP_N][N_2_Mid]	7	14	15	12.2	81.1
Heater-Comp N3 - Mid (7/14 15 W)[COMP_N][N_3_Mid]	7	14	15	12.2	81.2
Heater-Comp N4 - Mid (7/14 15 W)[COMP_N][N_4_Mid]	7	14	15	13.1	87.3
Heater-Deck 1 Operational Heater (-2/5 75 W)[DECK_1][DECK_1]	-2	5	75	58.1	77.4

Finishing Up (Final Products)

- ***Examining the predicted temperatures shows everything within limits, albeit without as much margin as desired, with Component C right near its hot operational limit.***
- ***Looking at the heater powers shows relatively healthy margins with around 30% heater power still available across all cases.***
- ***Your systems engineer colleague thanks you for your efforts and ensures your recommendations will be rolled up into their final proposal.***

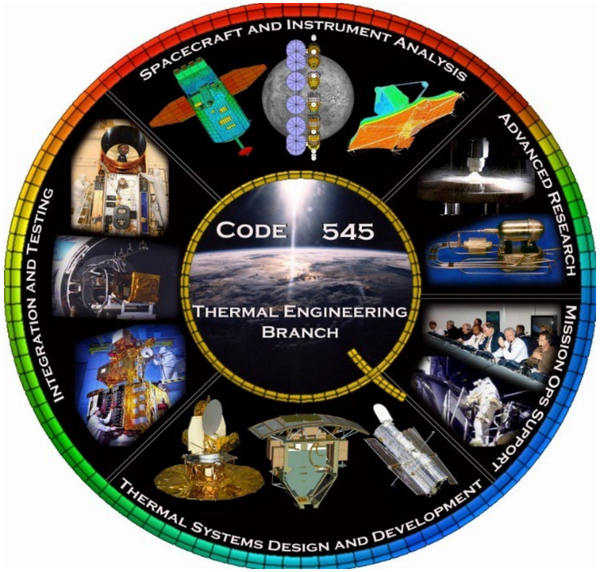
Name	On (C)	Off (C)	Size (W)	Avg (W)	DC (%)
Heater-Component H Surv Heater (17/22 75 W)[COMP_H][COMP_H_Srv]	17	22	75	52.5	70.0
Heater-Comp N1 - Mid (7/14 20 W)[COMP_N][N_1_Mid]	7	14	20	8.7	43.6
Heater-Comp N2 – Mid (7/14 20 W)[COMP_N][N_2_Mid]	7	14	20	9.6	48.1
Heater-Comp N3 - Mid (7/14 20 W)[COMP_N][N_3_Mid]	7	14	20	9.4	47.2
Heater-Comp N4 - Mid (7/14 20 W)[COMP_N][N_4_Mid]	7	14	20	10.4	52.2
Heater-Deck 1 Surv Heater A (-2/5 75 W)[DECK_1][DECK_1A]	-2	5	75	47.9	63.9
Heater-Deck 1 Surv Heater B (-2/5 75 W)[DECK_1][DECK_1A]	-2	5	75	47.5	63.3

Component	SurvLow (°C)	OpLow (°C)	OpHigh (°C)	SurvHigh (°C)	Min	TD Min	Max	TD Max
A	-20	-10	40	50	-2.5	-2.5	15.8	15.8
B1	-20	-10	40	50	-5.8	-5.8	6.5	6.5
B2	-20	-10	40	50	-6.4	-6.4	6.9	6.9
B3	-20	-10	40	50	-6.4	-6.4	10.0	10.0
Cmin	-20	-10	40	50	8.1	8.1	26.6	26.7
Cmax	-20	-10	40	50	8.7	8.7	26.9	26.9
D1	-44	-34	71	81	-4.9	-5.0	18.8	18.8
D2	-44	-34	71	81	-0.4	-0.3	16.6	16.5
D3	-44	-34	71	81	-10.1	-2.2	10.5	-0.9
E	-20	-10	40	50	-6.9	-6.9	6.9	6.8
F	-20	-10	40	50	-6.8	-6.8	12.4	12.4
G	-20	-10	40	50	-5.8	-5.8	11.1	11.1
Hmin	0	10	30	40	7.8	7.8	15.8	15.8
Hmax	0	10	30	40	16.7	16.5	25.6	25.6
I	-20	-10	40	50	-5.6	-5.6	5.8	5.8
J	-20	-10	40	50	-6.2	-6.2	9.7	9.7
K	-20	-10	40	50	-1.7	-1.7	17.6	17.6
L	-20	-10	40	50	-5.9	-5.9	17.1	17.1
M	-20	-10	40	50	-5.4	-5.4	6.0	6.0
N1min	2	5	25	30	6.7	6.7	12.7	12.7
N1Max	2	5	25	30	9.7	9.7	19.9	19.9
N2Min	2	5	25	30	6.7	6.5	12.7	12.8
N2Max	2	5	25	30	9.6	9.6	20.8	20.8
N3Min	2	5	25	30	6.7	6.3	12.7	12.5
N3Max	2	5	25	30	9.3	9.3	19.9	19.9
N4Min	2	5	25	30	6.7	6.4	12.7	12.5
N4Max	2	5	25	30	9.6	9.6	21.0	21.0

Closing Thoughts on Thermal Analysis

- *Every thermal analysis should begin with expected performance; without this, how can the validity of the predictions be judged?*
- *Analysis of the model predictions is the responsibility of the Thermal Engineer*
- *Trust the results after you have done some basic verification with hand calculations and first principles, but also trust your own judgement*
- *Anticipate the data that is needed from the model. Very often, only a small fraction of all model nodes are needed to evaluate the design*
- *Automating the post-processing may take some up front effort, but will pay dividends when the model needs to be run many times through many cases*
- *Very often, a fix implemented for one component can have unintended consequences for other components. The results need to be investigated as a system. Just because a fix brought one component within limits does not mean that something that was previously meeting requirements was unaffected*
- *If something unexpected occurred due to a model update, seek to understand the physics that is driving the behavior or make sure it is not a modeling error*
- *Learn to separate requirements that are not being met due to the design from requirements not being met due to modeling practices*

Wrapping it all up



Key Thermal Analysis Take-aways

- Thermal Analysis is a cost efficient method of using computer simulations to explore the design space and verify a design meets requirements
- Most spacecraft thermal analyses use a Geometry model to compute Radiative exchange and environments. These results feed into a Thermal network model to predict temperatures and heater powers
- A process exists for building models and verifying the inputs prior to execution
- While an understanding of what the code is doing behind the scenes is not required, it is extremely helpful to understand the algorithms when troubleshooting is needed
- It is the engineer's responsibility to ensure the predictions are reasonable based on the physics of the simulation and to synthesize the predicts produced by the model into useful products for stakeholders
- Physics is physics…if you model is telling you something that does not make sense, check the physics. If the design does not (or cannot) meet requirements, the laws of physics must still apply…

Acronyms

ASCII	American Standard Code for Information Interchange
API	Application Programming Interface
BOL	Beginning of Life
CAD	Computed Aided Design
CCHP	Constant Conductance Heat Pipe
DB	Database
ESPA	EELV (Evolved Expandable Launch Vehicle) Secondary Payload Adapter
EOL	End of Life
FD	Fiinte Difference
FE	Finite Element
FEM	Finite Element Model
FEMAP	Finite Element Modeling And Post-processing
FORTRAN	Formula Translation
GMM	Geometric Math Model
GSFC	Goddard Space Flight Center
HR	Heat Rate
Bij	Interchange Factor
MCRT	Monte Carlo Ray Trace
MLI	Multi Layer Insulation

NASA	National Aeronautics and Space Administration
NCG	Non-Condensable Gas
PCM	Phase Change Material
PID	Proportional-Integral-Derivative (controller)
GR/Radk	Radaiotn Coupling
RAAN	Right Ascension od Ascending Node
S/F	Sinda/Fluint
STEP	Standard for the Exchange of Product Data
SS	Steady State
STEP-TAS	STEP-Thermal Analsysi for Space
SINDA	Systems Integrated Numerical Difference Analyzer
TMM	Thermal Math Model
TRASYS	Thermal Radiation Analysis SYStem
TEC	Thermo Electric Cooler
TR	Transient
UCS	User Coordinate System
VCHP	Variable Conductance Heat Pipe
VF	View Factor