# Instructions for Installing and Using the Downloadable Model Package in TRNSYS

**A Report of IEA SHC - Task 35**

**PV/Thermal Solar Systems**

**November 13, 2009**

Mike Collins
Veronique Delisle


Nov 2009

# Instructions for Installing and Using the Downloadable Model Package in TRNSYS

by

Mike Collins
Veronique Delisle

A report of Subtask B

Mike Collins
Dept. of Mechanical Engineering
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1
Phone: 1.519.888.4567x33655
E-mail: mcollins@uwaterloo.ca

Veronique Delisle
CANMET Energy Tech. Centre
Natural Resources Canada
Varennes, Quebec, Canada
J3X 1S6
Phone:1.450.652.7948
E-mail: Veronique.Delisle@NRCan.gc.ca

Nov 2009

## Contents



## Disclaimer

*The information contained in this document detail new TRNSYS models, that have been modified from existing TRNSYS models, to suit the needs of IEA SHC Task 35. Some general opinion on how the user may these models has also been expressed.*

*Neither the Author, nor any member of the IEA SHC Programme, make no warranties about the accuracy or completeness of any information contained herein. Further, the Author does not accept any liability for any damages or losses whatsoever, arising out of, or in any way related to, the use of this information.*

SOLAR HEATING & COOLING PROGRAMME
INTERNATIONAL ENERGY AGENCY

# IEA Solar Heating and Cooling Programme

The *International Energy Agency* (IEA) is an autonomous body within the framework of the Organization for Economic Co-operation and Development (OECD) based in Paris. Established in 1974 after the first "oil shock," the IEA is committed to carrying out a comprehensive program of energy cooperation among its members and the Commission of the European Communities.

The IEA provides a legal framework, through IEA Implementing Agreements such as the *Solar Heating and Cooling Agreement*, for international collaboration in energy technology research and development (R&D) and deployment. This IEA experience has proved that such collaboration contributes significantly to faster technological progress, while reducing costs; to eliminating technological risks and duplication of efforts; and to creating numerous other benefits, such as swifter expansion of the knowledge base and easier harmonization of standards.

The *Solar Heating and Cooling Programme* was one of the first IEA Implementing Agreements to be established. Since 1977, its members have been collaborating to advance active solar and passive solar and their application in buildings and other areas, such as agriculture and industry. Current members are:

| | | |
|---|---|---|
| Australia | Finland | Portugal |
| Austria | France | Spain |
| Belgium | Italy | Sweden |
| Canada | Mexico | Switzerland |
| Denmark | Netherlands | United States |
| European Commission | New Zealand | |
| Germany | Norway | |

A total of 44 Tasks have been initiated, 33 of which have been completed. Each Task is managed by an Operating Agent from one of the participating countries. Overall control of the program rests with an Executive Committee comprised of one representative from each contracting party to the Implementing Agreement. In addition to the Task work, a number of special activities—Memorandum of Understanding with solar thermal trade organizations, statistics collection and analysis, conferences and workshops—have been undertaken.

To find Solar Heating and Cooling Programme publications and learn more about the Programme visit
**www.iea-shc.org** or contact the SHC Executive Secretary, Pamela Murphy, e-mail: pmurphy@kmgrp.net.

Nov 2009

The Tasks of the IEA Solar Heating and Cooling Programme, both underway and completed are as follows:

**C**urrent Tasks & Working Group:

| | |
|---|---|
| Task 35 | *PV/Thermal Solar Systems* |
| Task 36 | *Solar Resource Knowledge Management* |
| Task 37 | *Advanced Housing Renovation with Solar & Conservation* |
| Task 38 | *Solar Thermal Cooling and Air Conditioning* |
| Task 39 | *Polymeric Materials for Solar Thermal Applications* |
| Task 40 | *Net Zero Energy Solar Buildings* |
| Task 42 | *Compact Solar Thermal Energy Storage* |
| Working Group | *Daylight Research Group* |

**C**ompleted Tasks:

| | |
|---|---|
| Task 1 | *Investigation of the Performance of Solar Heating and Cooling Systems* |
| Task 2 | *Coordination of Solar Heating and Cooling R&D* |
| Task 3 | *Performance Testing of Solar Collectors* |
| Task 4 | *Development of an Insolation Handbook and Instrument Package* |
| Task 5 | *Use of Existing Meteorological Information for Solar Energy Application* |
| Task 6 | *Performance of Solar Systems Using Evacuated Collectors* |
| Task 7 | *Central Solar Heating Plants with Seasonal Storage* |
| Task 8 | *Passive and Hybrid Solar Low Energy Buildings* |
| Task 9 | *Solar Radiation and Pyranometry Studies* |
| Task 10 | *Solar Materials R&D* |
| Task 11 | *Passive and Hybrid Solar Commercial Buildings* |
| Task 12 | *Building Energy Analysis and Design Tools for Solar Applications* |
| Task 13 | *Advance Solar Low Energy Buildings* |
| Task 14 | *Advance Active Solar Energy Systems* |
| Task 16 | *Photovoltaics in Buildings* |
| Task 17 | *Measuring and Modeling Spectral Radiation* |
| Task 18 | *Advanced Glazing and Associated Materials for Solar and Building Applications* |
| Task 19 | *Solar Air Systems* |
| Task 20 | *Solar Energy in Building Renovation* |
| Task 21 | *Daylight in Buildings* |
| Task 23 | *Optimization of Solar Energy Use in Large Buildings* |
| Task 22 | *Building Energy Analysis Tools* |
| Task 24 | *Solar Procurement* |
| Task 25 | *Solar Assisted Air Conditioning of Buildings* |
| Task 26 | *Solar Combisystems* |
| Task 28 | *Solar Sustainable Housing* |
| Task 27 | *Performance of Solar Facade Components* |
| Task 29 | *Solar Crop Drying* |
| Task 31 | *Daylighting Buildings in the 21$^{st}$ Century* |
| Task 32 | *Advanced Storage Concepts for Solar and Low Energy Buildings* |
| Task 33 | *Solar Heat for Industrial Processes* |
| Task 34 | *Testing and Validation of Building Energy Simulation Tools* |

**C**ompleted Working Groups:

*CSHPSS, ISOLDE, Materials in Solar Thermal Collectors, and the Evaluation of Task 13 Houses*

# IEA SHC Task 35 PV/Thermal Solar Systems

## Objective

The objectives of the Task are to catalyze the development and market introduction of high quality and commercial competitive PV/Thermal Solar Systems and to increase general understanding and contribute to internationally accepted standards on performance, testing, monitoring and commercial characteristics of PV/Thermal Solar Systems in the building sector.

## The Task is organized in 5 subtasks:

- Subtask A: Market and Commercialization of PV/T
- Subtask B: Energy Analysis and Modeling
- Subtask C: Product and System Development, Tests and Evaluation
- Subtask D: Demonstration Projects
- Subtask E: Dissemination

## Organisation

IEA SHC Task 35 "PV/Thermal Solar Systems" is a three year Task initiated by the International Energy Agency (IEA) Solar Heating and Cooling (SHC) Programme in January 2005.
The Danish Energy Authority, acting through Mr. Henrik Sørensen, Esbensen Consulting Engineers A/S, Denmark, is designated as Operating Agent for the Task.

Task 35 is a so-called "minimum-level" collaboration task with IEA PVPS (Photovoltaic Power Systems Programme). At this level, experts selected by the PVPS Executive Committee participate in experts meetings of the Task managed by the SHC Executive Committee. The Task is fully defined and managed by the SHC Executive Committee with appropriate input from the PVPS Executive Committee. In this project Israel participated as a PVPS country member.

The official participants in the Task are listed in the table below:

| **Country** | **Organization** | **Person** |
|---|---|---|
| Canada | Dept. of Mechanical Engineering, University of Waterloo, Waterloo, Ontario, Canada | Mike Collins |
| Denmark | Esbensen Consulting Engineers A/S | Henrik Sørensen |
| | Solar Energy Center, Danish Technological Institute | Ivan Katic |
| Israel | Millennium Electric | Ami Elazari |
| Sweden | Lund Technical University | Björn Karlsson<br>Johan Nilsson<br>Bengt Perers |
| The Netherlands | ECN (Energy Research Centre of the Netherlands) | Wim van Helden<br>Herbert Zondag<br>Marco Bakker |

Apart from the above mentioned a number of manufacturers, universities, and research institutes from the countries Germany, Greece, Hong Kong, Italy, South Korea, Thailand, and Spain have been involved in the work.

Visit the Task 35 website: http://www.iea-shc.org/task35 for more details on activities and results.

## INTRODUCTION

This document has been prepared in support of IEA Task 35: Combined Photovoltaic and Solar Thermal (PV/Thermal) Systems – Subtask B: Energy Analysis and Modelling. In a previous report [1], a comprehensive review was performed of PV, Solar Thermal, and PV/Thermal models. That review demonstrated that TRNSYS was the predominant source of these models, but the required models either didn't exist or did not work well. This document is intended to describe the new models and the modifications made to existing models, and to provide instructions for installing the models contained in the downloadable package.

The TRNSYS package is required to use these models. If the reader does not possess this software, it is advised that s/he download the simplified stand-alone package, available via the IEA SHCP Task 35 web page.

The models included here are:

1) Type 201 - PV/Thermal Transpired Air Collector: This model is a modification of Summers [2] Unglazed Transpired Plate (UTC) model. A description of this model can be found in Delisle and Collins [3].

Modifications include

- Placement of PV in different positions on the panel
- Consideration of the panel geometry on the shading of the cells

Model inputs, outputs, and parameters are described on page 5.

2) Type 250 –PV/Thermal Flat Plate Collectors: This model is a modified version of the Type50d model included with the TRNSYS package [4]. The model allows for the analysis of glazed and unglazed flat plate PV/Thermal collectors that use any fluid (including air). A description of this model can be found in the TRNSYS help files provided with the TRNSYS software.

Modifications include:

- Simplification to Type50d only
- Correction of a division by zero error in the original model
- Correction of an error in the PV temperature coefficient
- Reorganization of model inputs, outputs, and parameters

Model inputs, outputs, and parameters are described on page 8.

3) Type 251 – PV/Thermal Concentrating Collectors: This model is a modified version of the Type50h model included with the TRNSYS package [4]. The model allows for the analysis of glazed and unglazed concentrating PV/Thermal collectors that use any fluid (including air). A description of this model can be found in the TRNSYS help files provided with the TRNSYS software.

Modifications include:

- Simplification to Type50h only
- Reorganization of model inputs, outputs, and parameters

Model inputs, outputs, and parameters are described on page 11.

Code listings for all models are included as Appendix A.

This document is not intended to be an instruction manual for creating and running user defined TRNSYS models, or for using the TRNSYS software. The reader, however, is referred to IEA SHC Task 35 Report DB-2 [5] for an example of their usage. In that report, these models were used to develop rating and characterization methods for PV/Thermal panels.

## EASY INSTALLATION (all models)

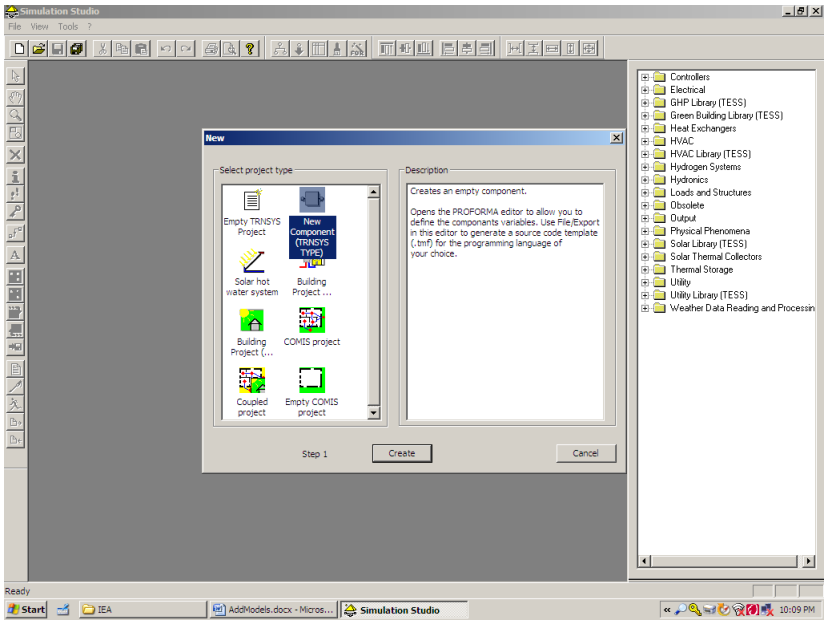
1. Place the included *.bmp and *.tmf files into the directory
   *Trnsys 16/Studio/Proformas/MyComponents*

   You may have to create the MyComponents folder yourself.

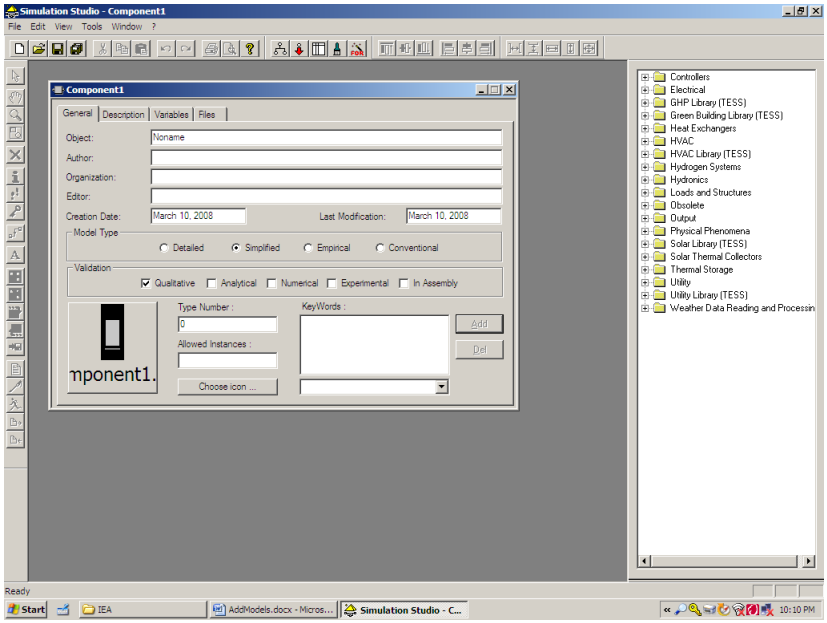
2. Drop the *.dll file in the folder *Trnsys16\UserLib\ReleaseDLLs.*

## COMPLETE INSTALLATION (all models)

1. Open the *Simulation Studio*
2. In the pull-down menu, select *File* and *New*. A '*new*' window will open.

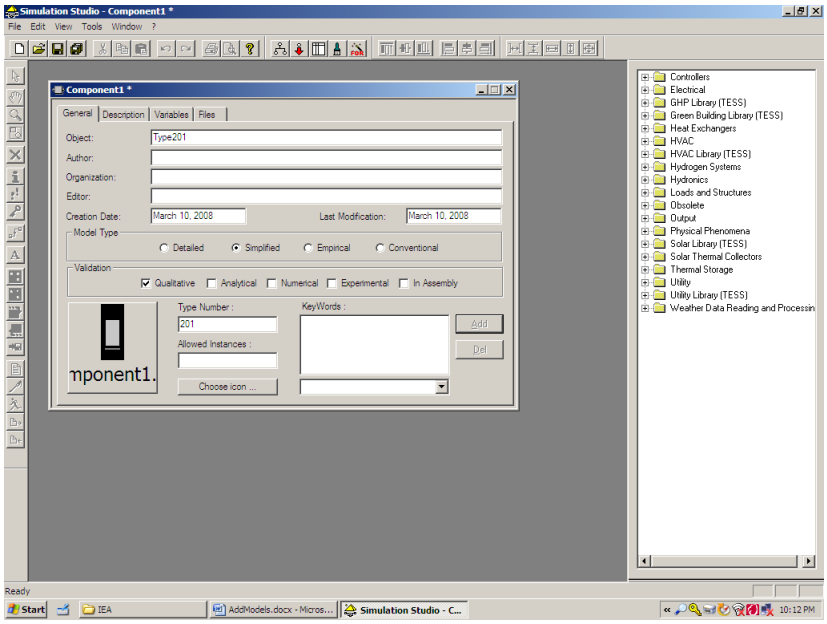



Select *New Component* and *Create*. A 'component' window will activate.

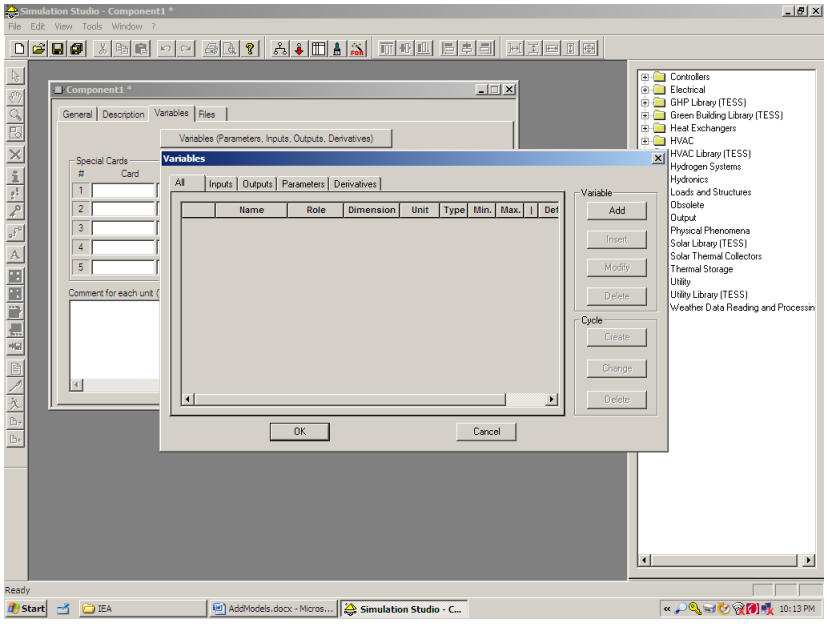


## TYPE201: PV/Thermal Transpired Air Collector

3. Under the *General* tab, enter "Type201" in the *Object* field, and "201" in the *Type Number* field.




4. Under the tab *Variables*, select *Variables (Parameters, Inputs, Outputs, Derivatives).* A 'variables' window will activate.



5. Complete the tabs *Parameters*, *Inputs*, and *Outputs* for the component being added according to Tables 201-1, 201-2, and 201-3.

**Table 201-1: Parameters List**

| | Name | Role | Dimension | Unit | Type | Min | Max | Default |
|---|---|---|---|---|---|---|---|---|
| 1 | Length of the base of the trapezoid | Par. | Length | m | Real | 0 | +inf | 0.15 |
| 2 | Length of the top of the trapezoid | Par. | Length | m | Real | 0 | +inf | 0.115 |
| 3 | Ribs height | Par. | Length | m | Real | 0 | +inf | 0.033 |
| 4 | Plate porosity | Par. | Any | Any | Real | 0 | 1 | 0.0025 |
| 5 | Hole pitch | Par. | Length | m | Real | 0 | +inf | 0.014 |
| 6 | Collector height | Par. | Length | m | Real | 0 | +inf | 2.49 |
| 7 | Distance between two corrugations | Par. | Length | m | Real | 0 | +inf | 0.2 |
| 8 | Plate width | Par. | Length | m | Real | 0 | +inf | 5.25 |
| 9 | Absorptance of the wall (or roof) on which the collector is mounted | Par. | Dimless | - | Real | 0 | 1 | 0.7 |
| 10 | Gap between the transpired plate and the wall (or roof) | Par. | Length | m | Real | 0 | +inf | 0.14 |
| 11 | Emissivity of the transpired plate back surface | Par. | Dimless | - | Real | 0 | 1 | 0.9 |
| 12 | Emissivity of the wall (or roof) on which the collector is mounted | Par. | Dimless | - | Real | 0 | 1 | 0.9 |
| 13 | Thickness of the transpired plate | Par. | Length | m | Real | 0 | +inf | 0.001 |
| 14 | Transpired plate absorptance | Par. | Dimless | - | Real | 0 | 1 | 0.94 |
| 15 | Transpired plate front surface emissivity | Par. | Dimless | - | Real | 0 | 1 | 0.9 |
| 16 | Overall loss coefficient of the building wall (or roof) | Par. | Heat Transfer Coeff | kJ/hrm2K | Real | 0 | +inf | 1.02 |
| 17 | PV Mode [1: No PV, 2: PV on the top of the corrugation, 3: PV everywhere] | Par. | Dimless | - | Int | 1 | 3 | 2 |
| 18 | PV temperature efficiency modifier (mu): EffPV=Effref+mu(TPV-Tref) | Par. | Inverse Temperature | 1/C | Real | -inf | +inf | -0.00056 |
| 19 | PV efficiency at reference conditions | Par. | Dimless | - | Real | 0 | 1 | 0.125 |
| 20 | PV reference temperature | Par. | Temperature | C | Real | -inf | +inf | 25 |
| 21 | PV absorptance-transmittance product | Par. | Dimless | - | Real | 0 | 1 | 0.8 |
| 22 | PV emissivity | Par. | Dimless | - | Real | 0 | 1 | 0.8 |
| 23 | Bypass collector in summer? [1 : Yes, 2 : No] | Par. | Dimless | - | Int | 0 | 1 | 0 |
| 24 | Bypass temperature | Par. | Temperature | C | Real | -inf | +inf | 18 |
| 25 | PV cells cover refraction index | Par. | Any | Any | Real | 0 | +inf | 1.526 |
| 26 | PV cells cover extinction coefficient | Par. | Inverse length | m^-1 | Real | -inf | +inf | 4 |
| 27 | PV cells cover thickness ( if set to zero, the effect of the incidence angle is not taken into account) | Par. | Length | m | Real | 0 | +inf | 0.002 |
| 28 | Proportion of PV cells on the surface on the top of the corrugations | Par. | Dimless | - | Real | 0 | 1 | 0.8 |
| 29 | Proportion of PV cells on the surface at the bottom of the corrugations | Par. | Dimless | - | Real | 0 | 1 | 0 |
| 30 | Proportion of PV cells on the surface on the sides of the corrugations | Par. | Dimless | - | Real | 0 | 1 | 0 |

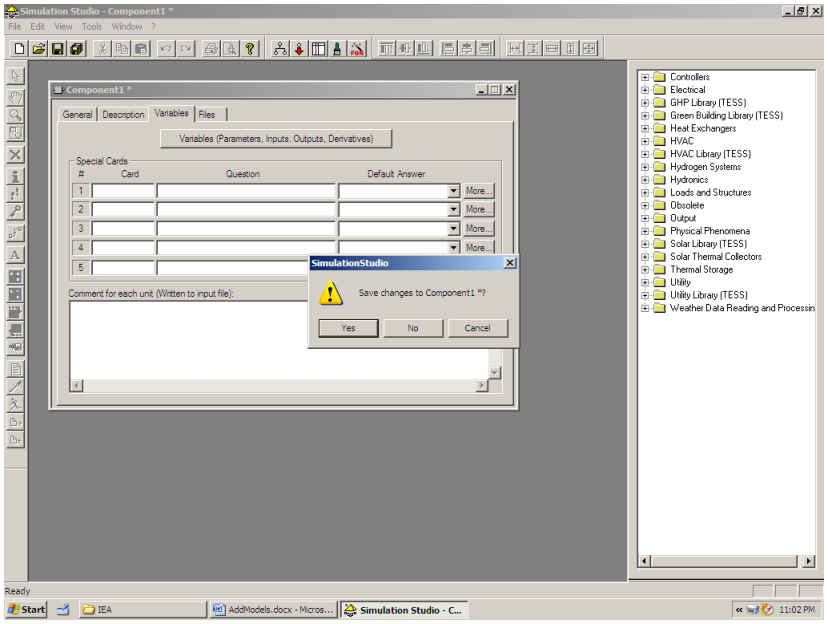
**Table 201-2: Inputs for Type201**

| | Name | Role | Dimension | Unit | Type | Min | Max | Default |
|---|---|---|---|---|---|---|---|---|
| 1 | Beam radiation on the collector surface | Input | Flux | kJ/hrm2 | Real | | | |
| 2 | Diffuse radiation on the collector surface | Input | Flux | kJ/hrm2 | Real | | | |
| 3 | Ground reflected radiation on the collector surface | Input | Flux | kJ/hrm2 | Real | | | |
| 4 | Horizontal beam radiation | Input | Flux | kJ/hrm2 | Real | | | |
| 5 | Zenith angle | Input | Dir.(Angle) | Degrees | Real | | | |
| 6 | Solar azimuth angle | Input | Dir.(Angle) | Degrees | Real | | | |
| 7 | Wind velocity | Input | Velocity | m/s | Real | | | 2.4 |
| 8 | Ambient temperature | Input | Temperature | C | Real | | | 5 |
| 9 | Atmospheric pressure | Input | Pressure | Pa | Real | | | 101325 |
| 10 | Sun incidence angle on the collector surface | Input | Dir.(Angle) | Degrees | Real | | | |
| 11 | Temperature inside the building | Input | Temperature | C | Real | | | 21 |
| 12 | Sky temperature | Input | Temperature | C | Real | | | |
| 13 | Minimum air flow rate through the collector | Input | Flow rate | Kg/hr | Real | | | |
| 14 | Maximum air flow rate through the collector | Input | Flow rate | Kg/hr | Real | | | |
| 15 | Collector slope | Input | Dir.(Angle) | degrees | Real | | | 90 |
| 16 | Collector azimuth angle | Input | Dir.(Angle) | degrees | Real | | | |
| 17 | Albedo | Input | Dimless | - | Real | | | 0.2 |
| 18 | Total horizontal radiation | Input | Flux | kJ/hrm2 | Real | | | |
| 19 | Horizontal diffuse radiation | Input | Flux | kJ/hrm2 | Real | | | |
| 20 | Temperature of the air that needs to be supplied to the building | Input | Temperature | C | Real | | | 28 |

**Table 201-3: Outputs for Type201**

| | Name | Role | Dimension | Unit | Type | Min | Max | Default |
|---|---|---|---|---|---|---|---|---|
| 1 | Transpired plate average surface temperature | Output | Temperature | C | Real | | | |
| 2 | Plenum average temperature | Output | Temperature | C | Real | | | |
| 3 | Mixed temperature (recirculated air and fresh air) | Output | Temperature | C | Real | | | |
| 4 | Collector outlet temperature | Output | Temperature | C | Real | | | |
| 5 | Mass fraction of outside air | Output | Dimless | - | Real | | | |
| 6 | Mass fraction of recirculated air | Output | Dimless | - | Real | | | |
| 7 | Mass flow rate through the collector | Output | Flow rate | Kg/hr | Real | | | |
| 8 | Collector heat exchange effectiveness | Output | Dimless | - | Real | | | |
| 9 | Collector thermal efficiency | Output | Dimless | - | Real | | | |
| 10 | Collector electrical efficiency | Output | Dimless | - | Real | | | |
| 11 | PV cells efficiency | Output | Dimless | - | Real | | | |
| 12 | Collector useful energy | Output | Power | W | Real | | | |
| 13 | Electrical power | Output | Power | W | Real | | | |
| 14 | Reduced wall heat losses | Output | Power | W | Real | | | |
| 15 | Bypass (0 if no bypass, 1 if bypass) | Output | Dimless | - | Int | | | |
| 16 | Energy absorbed by the collector | Output | Flux | W/m2 | Real | | | |

Select *OK* when all the variables are entered and close the *Component* window.



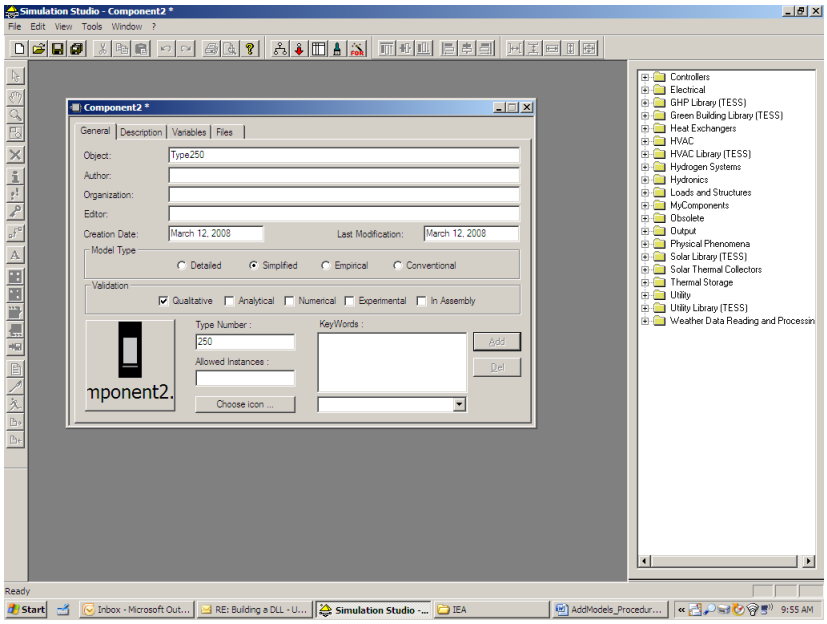

6. Select *Yes* to save changes to the new component. Name the file "Type201" and save the file in the folder *Trnsys16\Studio\Proformas\MyComponents.* If the folder *MyComponents* does not exist yet, create it.

7. The new component should now be available in the Simulation Studio environment with the other components of the standard library under the category *MyComponents*. If it does not appear, start a new project and go to *Direct Access/Refresh Tree.*

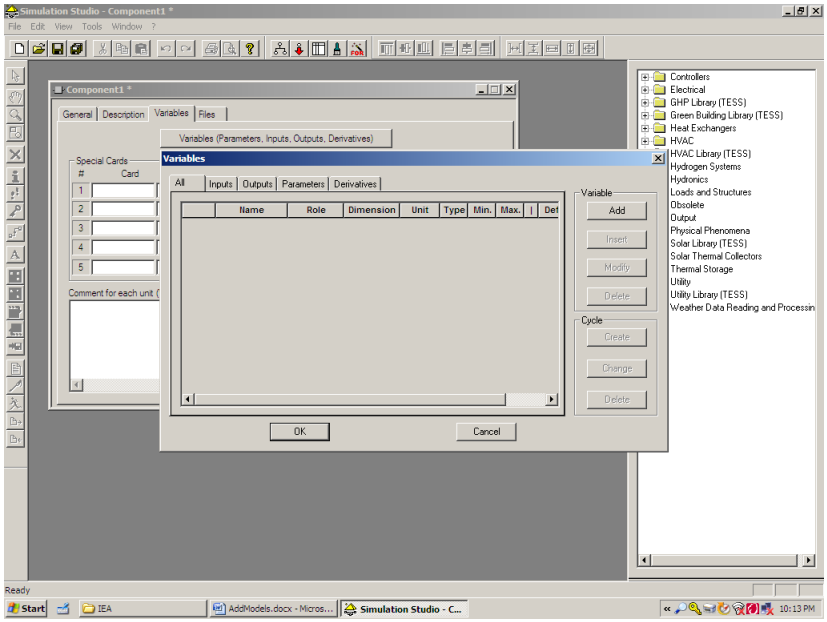
8. In order for the component to function, drop the file Type201.dll in the folder *Trnsys16\UserLib\ ReleaseDLLs.*

## TYPE250: PV/Thermal Flat Plate Collectors

3. Under the *General* tab, enter “Type250” in the *Object* field, and “250” in the *Type Number* field.




4. Under the tab *Variables*, select *Variables (Parameters, Inputs, Outputs, Derivatives).* A ‘variables’ window will activate.



5. Complete the tabs *Parameters*, *Inputs*, and *Outputs* for the component being added according to Tables 250-1, 250-2, and 250-3.

**Table 250-1: Parameters List**

| | Name | Role | Dimension | Unit | Type | Min | Max | Default |
|---|---|---|---|---|---|---|---|---|
| 1 | Collector Area | Param. | Area | $m^2$ | real | 0 | +Inf | 1 |
| 2 | Collector Efficiency Factor (FP) | Param. | dimensionless | - | real | 0 | 1 | 0.8 |
| 3 | Fluid Heat Capacity | Param. | Specific Heat | kJ/kg.K | real | 0 | +Inf | 4.19 |
| 4 | Number of Glass Covers | Param. | dimensionless | - | real | 0 | 10 | 1 |
| 5 | KL Product | Param. | dimensionless | - | real | 0 | +Inf | 0.06 |
| 6 | Back and Edge Loss Coefficient | Param. | Heat Transfer Coeff. | $kJ/hr.m^2K$ | real | 0 | +Inf | 10 |
| 7 | PV Absorptivity | Param. | dimensionless | - | real | 0 | 1 | 0.9 |
| 8 | PV Emissivity | Param. | dimensionless | - | real | 0 | 1 | 0.8 |
| 9 | PV Efficiency | input | dimensionless | - | real | | | 0.15 |
| 10 | PV Temperature Coeff | Param. | any | any | real | -Inf | +Inf | 0.004 |
| 11 | PV Ref Temperature | Param. | Temperature | C | real | -Inf | +Inf | 25 |
| 12 | PV Packing Factor | Param. | dimensionless | - | real | 0 | 1 | 0.9 |

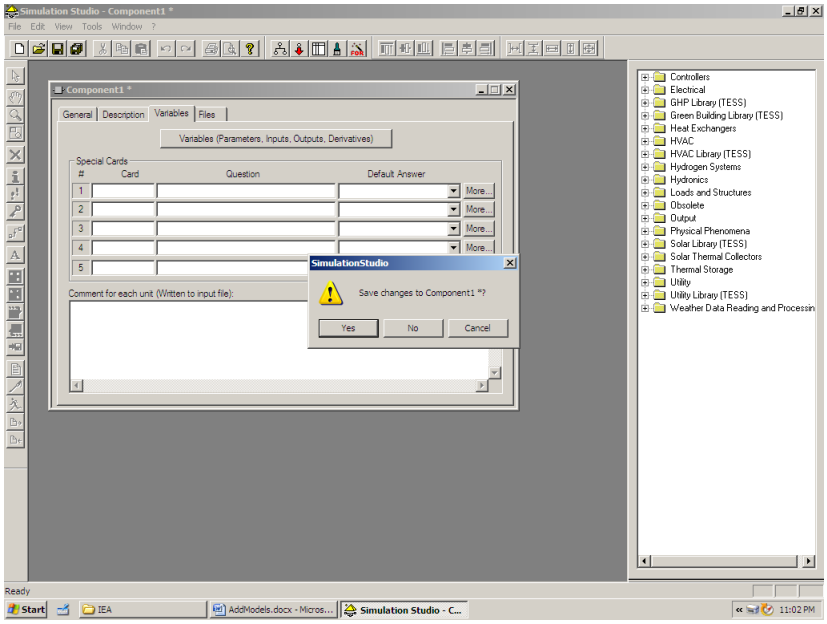
**Table 250-2: Inputs for Type250**

| | Name | Role | Dimension | Unit | Type | Min | Max | Default |
|---|---|---|---|---|---|---|---|---|
| 1 | Fluid Inlet Temperature | input | Temperature | C | real | | | 30 |
| 2 | Collector Specific Flowrate | input | Flow/area | $kJ/hr.m^2$ | real | | | 108 |
| 3 | Collector Slope | Param. | Direction (Angle) | degrees | real | 0 | +Inf | 45 |
| 4 | Ambient Temperature | input | Temperature | C | real | | | 30 |
| 5 | Incident Beam Radiation | input | Flux | $kJ/hr.m^2$ | real | | | 3600 |
| 6 | Incident Diffuse Radiation | input | Flux | $kJ/hr.m^2$ | real | | | 0 |
| 7 | Incident Angle | input | Direction (Angle) | degrees | real | | | 0 |
| 8 | Wind Speed | input | Velocity | m/s | real | | | 0.5 |

**Table 250-3: Outputs for Type250**

| | Name | Role | Dimension | Unit | Type | Min | Max | Default |
|---|---|---|---|---|---|---|---|---|
| 1 | Thermal Output | output | Power | kJ/hr | real | | | |
| 2 | Electrical Output | output | Power | kJ/hr | real | | | |
| 3 | Fluid Outlet Temperature | output | Temperature | C | real | | | |
| 4 | Flow Rate | output | Flow Rate | kg/hr | real | | | |
| 5 | tau-alpha | output | dimensionless | - | real | | | |
| 6 | Average Cell Temperature | output | Temperature | C | real | | | |
| 7 | Loss Coefficient | output | Heat Transfer Coeff. | $kJ/hr.m^2.K$ | real | | | |
| 8 | Apparent Loss | output | Heat Transfer Coeff. | $kJ/hr.m^2.K$ | real | | | |

Select *OK* when all the variables are entered and close the *Component* window.



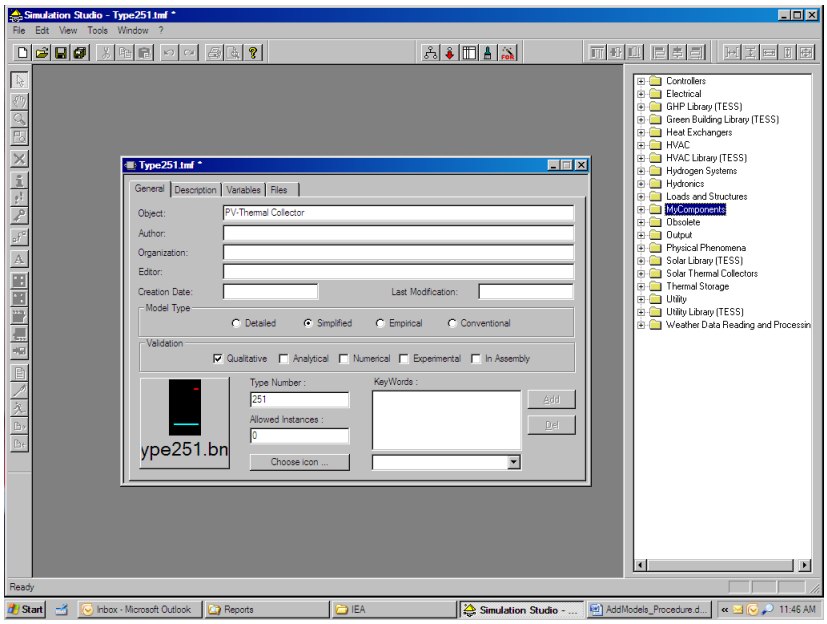

6. Select *Yes* to save changes to the new component. Name the file "Type250" and save the file in the folder *Trnsys16\Studio\Proformas\MyComponents.* If the folder *MyComponents* does not exist yet, create it.

7. The new component should now be available in the Simulation Studio environment with the other components of the standard library under the category *MyComponents*. If it does not appear, start a new project and go to *Direct Access/Refresh Tree.*

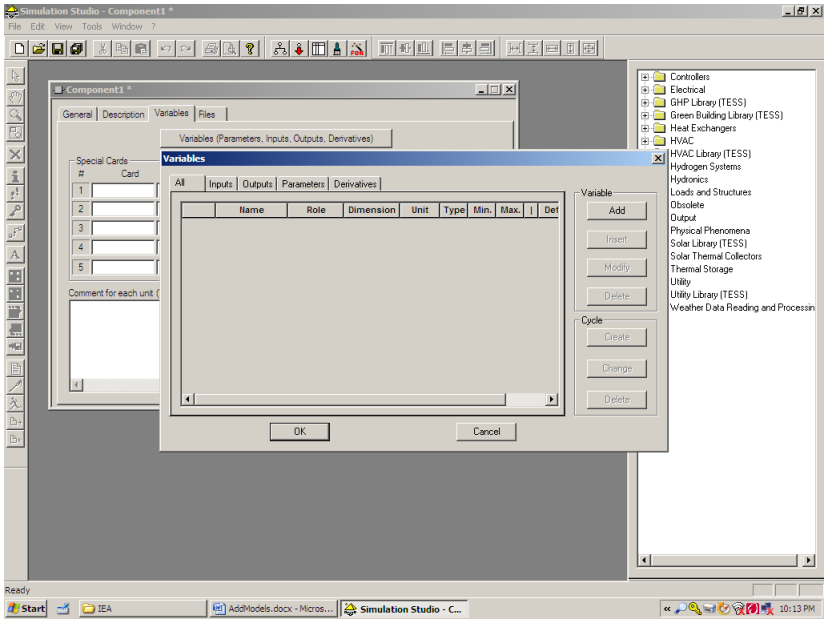
8. In order for the component to function, drop the file Type250.dll in the folder *Trnsys16\UserLib\ ReleaseDLLs.*

## TYPE251: PV/Thermal Concentrating Collectors

3. Under the *General* tab, enter "Type251" in the *Object* field, and "251" in the *Type Number* field.




4. Under the tab *Variables*, select *Variables (Parameters, Inputs, Outputs, Derivatives).* A 'variables' window will activate.



5. Complete the tabs *Parameters*, *Inputs*, and *Outputs* for the component being added according to Tables 251-1, 251-2, and 251-3.

**Table 250-1: Parameters List**

| | Name | Role | Dimension | Unit | Type | Min | Max | Default |
|---|---|---|---|---|---|---|---|---|
| 1 | Mode | param | dimensionless | - | integer | 8 | 8 | 8 |
| 2 | Collector Area | param | Area | $m^2$ | real | 0 | +Inf | 6.5 |
| 3 | Ratio of aperture to absorber area | param | dimensionless | - | real | 0 | 1 | 0.5 |
| 4 | Fluid thermal capacitance | param | Specific Heat | kJ/kg.K | real | 0 | +Inf | 4.19 |
| 5 | Plate absorptance | param | dimensionless | - | real | 0 | 1 | 0.9 |
| 6 | Fin efficiency area ratio | param | dimensionless | - | real | 0 | 1 | 0.9 |
| 7 | Back loss coefficient for no-flow condition | param | Heat Transfer Coeff. | kJ/hr.$m^2$.K | real | 0 | +Inf | 10 |
| 8 | Thermal conductance between cells and absorber | param | Overall Loss Coefficient | kJ/hr.K | real | 0 | +Inf | 720 |
| 9 | Heat transfer coefficient | param | Heat Transfer Coeff. | kJ/hr.$m^2$.K | real | 0 | +Inf | 5 |
| 10 | Cover plate transmittance | param | dimensionless | - | real | 0 | 1 | 0.9 |
| 11 | Number of glass covers | param | dimensionless | - | integer | 0 | 3 | 1 |
| 12 | Absorber plate emittance | param | dimensionless | - | real | 0 | 1 | 0.9 |
| 13 | Logical unit for SOLCEL data | param | dimensionless | - | integer | 10 | 30 | 15 |

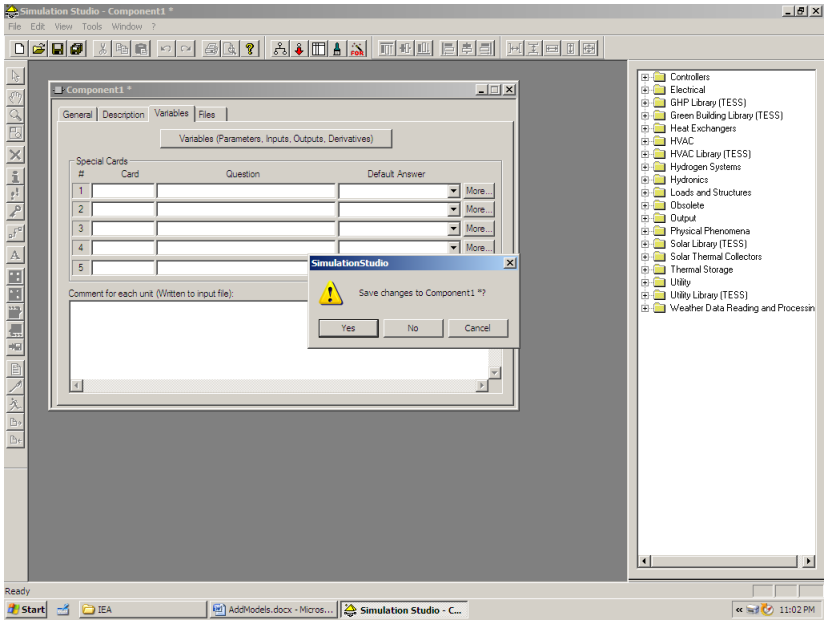
**Table 250-2: Inputs for Type251**

| | Name | Role | Dimension | Unit | Type | Min | Max | Default |
|---|---|---|---|---|---|---|---|---|
| 1 | Inlet fluid temperature | input | Temperature | C | real | -Inf | +Inf | 30 |
| 2 | Fluid mass flow rate | input | Flow Rate | kg/hr | real | 0 | +Inf | 108 |
| 3 | Collector slope | input | Direction (Angle) | degrees | real | 0 | +Inf | 45 |
| 4 | Voltage applied to array | input | Voltage | V | real | 0 | +Inf | 2 |
| 5 | Ambient temperature | input | Temperature | C | real | -Inf | +Inf | 20 |
| 6 | Incident radiation | input | Flux | kJ/hr.$m^2$ | real | 0 | +Inf | 3600 |
| 7 | Wind speed | input | Velocity | m/s | real | 0 | +Inf | 0.5 |

**Table 250-3: Outputs for Type251**

| | Name | Role | Dimension | Unit | Type | Min | Max | Default |
|---|---|---|---|---|---|---|---|---|
| 1 | Rate of useful energy gain | output | Power | kJ/hr | real | | | |
| 2 | Electrical power output | output | Power | kJ/hr | real | | | |
| 3 | Outlet fluid temperature | output | Temperature | C | real | | | |
| 4 | Fluid flowrate | output | Flow Rate | kg/hr | real | | | |
| 5 | Collector loss coefficient | output | Heat Transfer Coeff. | kJ/hr.$m^2$.K | real | | | |
| 6 | Apparent thermal loss coefficient | output | Heat Transfer Coeff. | kJ/hr.$m^2$.K | real | | | |
| 7 | Transmittance-absorptance product | output | dimensionless | - | real | | | |
| 8 | Average cell temperature | output | Temperature | C | real | | | |
| 9 | Cell temperature at collector inlet | output | Temperature | C | real | | | |
| 10 | Array voltage | output | Voltage | V | real | | | |
| 11 | Array current | output | Electric current | amperes | real | | | |

Select *OK* when all the variables are entered and close the *Component* window.




6. Select *Yes* to save changes to the new component. Name the file "Type251" and save the file in the folder *Trnsys16\Studio\Proformas\MyComponents.* If the folder *MyComponents* does not exist yet, create it.

7. The new component should now be available in the Simulation Studio environment with the other components of the standard library under the category *MyComponents*. If it does not appear, start a new project and go to *Direct Access/Refresh Tree.*

8. In order for the component to function, drop the file Type251.dll in the folder *Trnsys16\UserLib\ ReleaseDLLs.*

9. When using the Type251 model in TRNSYS, the model must be directed to the file 'SOLCEL.dat'. Double click the Type251 Icon in your performa, and then select the *External Files* Tab. Select *Brows*, and direct the model to the 'SOLCEL.dat' file.

   The SOLCEL.dat file contains information about the PV cells that are being modeled, and should be user created. The file should be formatted as follows:
   - 4 rows of data containing 2, 6, 6, and 6 numbers respectively.
   - Each number should contain 10 characters

| | | | | | |
|---|---|---|---|---|---|
| XICELL | CELLPF | | | | |
| TL | TH | CL | CH | VOCLL | ISCLL |
| ISCHL | ISCHL | ISCHH | ACELL | EG | AOMIN |
| DAODC | RSH | RSMIN | CONTR | DRSDC | NSER |

   Descriptions of these variables are contained in Section 8.4.4.14 of the TRNSYS Manual, and an example file has been provided in the downloadable package for the user.

## END NOTES

1) To add a component to TRNSYS only the .DLL file is required. The Fortran code has been provided in case one desires to examine or modify the code. Should the user desire to modify the code, a new .DLL file will have to be created using a FORTRAN compiler.

2) To create this new .DLL file, you will need a modern Fortran Compiler. After step 6 of the instructions, with the proforma still opened, select *File/Export* as Fortran from the main pull-down menu. In the Fortran Compiler environment, replace the content of the Fortran skeleton newly created by the Fortran code provided. When the desired modifications are completed, build and compile the DLL. Refer to the Getting Started volume of the TRNSYS Help Manual for further explanation on how to create a new component.

3) If you get TRNSYS ERROR # 1000 when you try to run a project, it might be because you have dll files in the folder *Trnsys16\UserLib*. If it is the case, they have to be moved to *Trnsys16\UserLib\ ReleaseDLLs* or *Trnsys16\UserLib\ DebugDLLs* depending on the mode on which they were compiled.

The TRNSYS help files, provided with the TRNSYS software, provide significant help for those attempting to develop their own models. The user is referred to those files should s/he need further assistance.

## REFERENCES

[1] Collins, M. (2008) “A Review of PV, Solar Thermal, and PV/Thermal Collector Models in TRNSYS”, IEA Report for IEA SHC - Task 35, Report DB-1.

[2] Summers, David N. (1995) "Thermal Simulation and Economic Assessment of Unglazed Transpired Collectors", M.S. Thesis in Mechanical Engineering, University of Wisconsin-Madison.

[3] Delisle, V., Collins, M.R., “PVT Unglazed Corrugated Transpired Solar Collector Model”, Canadian Solar Buildings Conference, Calgary, June 2007.

[4] Evans, D.L., Facinelli, W.A., and Otterbein, R.T. (1978) "Combined Photovoltaic / Thermal System Studies" DOE Technical Report Dep. NTIS, PC A09/MF A01, 183 pages.

[5] Collins, M. (2008) “Recommended Standard for Characterization and Monitoring of PV/Thermal Systems”, IEA Report for IEA SHC - Task 35, Report DB-2.

# APPENDIX A

```
                      SUBROUTINE TYPE201 (TIME,XIN,OUT,T,DTDT,PAR,INFO,ICNTRL,*)
************************************************************************
C Object: Type201
C Simulation Studio Model: Type201
C
C Author: David Summers, 1995 (UTC model)
C Editor: Veronique Delisle, 2007 (PV/Thermal UTC model)
C                       Changes were made to account for the wind effects,
C                       the corrugated shape of the plate (absorbed solar
C                       energy) and the fact that PV cells are mounted
C                       on the absorber plate
C Last modified: November 19, 2007
C
C       This models predicts the thermal and electrical performance of a
C       PV/Thermal Collector unglazed transpired collector with PV cells
C       mounted directly on the absorber plate. The absorber plate has a
C       trapezoidal corrugated shape. In Mode 1, the collector
C       is solved as a stand-alone UTC. In Mode 2, the PV cells can be
C       mounted only on the top surfaces of the corrugation. In Mode 3,
C       the PV cells can be mounted on every surface of the collector.
C ***
C *** Model Parameters
C ***
C       1.Length of the base of the trapezoid (a) [m]
C   2.Length of the top of the trapezoid (b) [m]
C       3.Ribs height (h_trap) [m]
C       4.Plate porosity (Por)
C       5.Hole pitch (pitch) [m]
C       6.Plate length/height (length) [m]
C       7.Distance between two corrugations (dist) [m]
C       8.Plate width (width) [m]
C       9.Abstorbtance of the wall/roof on which the collector is mounted (alphaW)
C       10.Gap between the corrugated plate and the wall (Gap) [m]
C       11.Back of the collector emissivity (e_back)
C       12.Wall emissivity (e_wall)
C       13.Corrugated plate thickness (th) [m]
C       14.Corrugated plate absorptivity at normal incidence angle (AlphaPl)
C       15.Collector surface emissivity (e_coll)
C       16.Overall loss coefficient of the building wall (Uwall) [kJ/m2Khr]
C       17.PV mode (1: No PV;2: PV on surface 3;3:PV on surfaces 1,2,3,4) (PVmode)
C       18.Efficiency modifier-Temperature (EffT) [1/degC]
C       19.PV efficiency at reference conditions [EffRef]
C       20.PV Cell reference temperature (Tref) [degC]
C       21.PV absorptance-transmittance product (TauAlfPV)
C       22.PV emissivity (e_PV)
C       23.Bypass collector in the summer (BypTemp) [1: Yes 0: No]
C       24.Bypass temperature (Tbypass) [degC]
C       25.PV cells glazing refraction index (ngl)
C       26.PV cells glazing extinction coefficient (kgl) [1/m]
C       27.PV cells glazing thickness (Lgl)             [m]
C       28.Proportion of PV cells on the surface at the top of the corrugation (PPV(3))
C         Defined as the ratio of the area of PV cells to that of on surface
c         at the top of the corrugations (APV on the top surfaces of the corr/b*Length)
C       29.Proportion of PV cells on the surface at the bottom of the corrugations (PPV(1))
C         APV on one surface at the bottom of the grooves/(d-b)*Length
C       30.Proportion of PV cells on the surface on the sides of the grooves (PPV(2)=PPV(4))
c         APV on one surface on the side of the groove (2 or 4)/(tTrap*Length)
C ***
C *** Model Inputs
C ***
C       1.Beam radiation on the collector surface (Gb[1,3]) [kJ/hr m2]
C       2.Sky diffuse radiation on the collector surface (Gd[1,3])[kJ/hr m2]
C       3.Ground reflected diffuse on the collector surface (Gg[1,3]) [kJ/hr m2]
C       4.Beam radiation on horizontal surface (GbH)[kJ/hr m2]
C       5.Incidence angle on horizontal surface (ThetaZ) [deg]
C       6.Solar azimuth angle (GammaS) [deg]
C       7.Wind velocity                 (Uwind) [m/s]
C       8.Ambient temperature (Tamb) [C]
C       9.Atmospheric pressure (Patm) [Pa]
C       10.Incidence angle on vertical surface (Theta(1)) [deg]
C       11.Building temperature (Tblg) [degC]
C       12.Sky Temperature (Tsky) [degC]
C       13.Minimum air flow rate through collector (MinFlow) [kg/hr]
C       14.Maximum air flow rate through collector (MFlow) [kg/hr]
C       15.Collector Slope (Beta(1)) [deg]
C       16.Collector azimuth angle [deg]
C       17.Ground reflectance (GrndRef)
C       18.Total radiation on horizontal surface (GH) [kJ/hrm2]
C       19.Diffuse radiation on the horizont                    (Gdh)           [kJ/hrm2]
C       20.Temperature of air that needs to be supplied to the building (Tsup) [degC]
C ***
C *** Model Outputs
C ***
C       1.Collector surface temperature (Tcol) [degC]
C       2.Plenum temperature (Tplen) [degC]
C       3.Mixed temperature (Tmix) [degC]
C       4.Collector outlet temperature (Tout) [degC]
C       5.Mass fraction of outside air (gam)
C       6.Mass fraction of recirculated air (1-gam)
C       7.Mass flow rate of air through the collector (MFlow*gam) [Kg/hr]
```

```
C               8.Collector heat exchange effectiveness (ehx)
C               9.Collector thermal efficiency (nth)
C               10.Collector electrical efficiency (nel)
C               11.PV cells efficiency (npv)
C               12.Collector useful energy (Qu) [W]
C               13.Maximum electrical power output (Qe) [W]
C               14.Reduced wall heat losses (Qred) [W]
C               15.Collector is bypassed? NO: bypass=0, YES: bypass=1
C               16.Absorbed solar energy (Qabs) [W/m2]

c                               |---b---|
C                               -------  -      --------        -------
C                               /                3   \             | h_trap/   3   \      /   3   \
C               -----/ 8     4 \-------/ 2      4\-----/ 2      6 \------
C                7                               1              1              5
c                               |---a------|
C       |---dist----------|

C*************************************************************************
C   TRNSYS acess functions (allow to acess TIME etc.)
    USE TrnsysConstants
    USE TrnsysFunctions
***************************************************************************
C   REQUIRED BY THE MULTI-DLL VERSION OF TRNSYS
    !DEC$ATTRIBUTES DLLEXPORT :: TYPE201        !SET THE CORRECT TYPE NUMBER HERE
***************************************************************************
C   TRNSYS DECLARATIONS
    IMPLICIT NONE
               DOUBLE PRECISION XIN,OUT,TIME,PAR,STORED,T,DTDT
               INTEGER*4 INFO(15)
               INTEGER*4 NP,NI,NOUT,ND
               INTEGER*4 NPAR,NIN,NDER,IUNIT,ITYPE,ICNTRL
               CHARACTER*3 OCHECK,YCHECK
***************************************************************************
C   USER DECLARATIONS - SET THE MAXIMUM NUMBER OF PARAMETERS (NP), INPUTS (NI),
C   OUTPUTS (NOUT), AND DERIVATIVES (ND) THAT MAY BE SUPPLIED FOR THIS TYPE
    PARAMETER (NP=30,NI=20,NOUT=16,ND=0)
***************************************************************************
C   REQUIRED TRNSYS DIMENSIONS
    DIMENSION XIN(NI),OUT(NOUT),PAR(NP),YCHECK(NI),OCHECK(NOUT),
              1   T(ND),DTDT(ND)
***************************************************************************
C               Common variables
               COMMON/PVT/Uwind,Tamb,Flow,Tsup,Gam,EffT,EffRef
               COMMON/PVT/Tref,t_trap,ThetaT,h_trap,a,b,dist
               COMMON/PVT/Por,rho_amb,visc_amb,AreaW,k_amb
               COMMON/PVT/pitch,length,width,hp,th,D,ecol
               COMMON/PVT/e_back,e_wall,Uwall,Tblg,Trad
               COMMON/PVT/L,cp_amb,Area_cs
               COMMON/PVT/QBsurf,QDsurf,Qsurf,PPV
               COMMON/PVT/Psh,Area,AlphaB,AlphaDcoll
               COMMON/PVT/PVOn
               COMMON/PVT/SHCASE,nb_corr,PVMode
               COMMON/PVT/IUNIT,ITYPE

C   ADD DECLARATIONS AND DEFINITIONS FOR THE USER-VARIABLES HERE
C   PARAMETERS
    DOUBLE PRECISION a,b,h_trap,d,pitch,length,dist,width,AlphaW
               DOUBLE PRECISION hp,e_back,e_wall,th,AlphaPl,e_coll,Uwall
               DOUBLE PRECISION EffT,EffRef,Tref,AlphaPV,PropPV
               DOUBLE PRECISION e_pv,Tbypass,Kgl,Lgl,ngl,TauAlfPV
               INTEGER*4 PVmode,BypTemp

C   INPUTS
               DOUBLE PRECISION GbW,GdW,GgW,GbH,GdH
               DOUBLE PRECISION ThetaZ,Gh,GammaS,Tsup,Gam,Tsky
               DOUBLE PRECISION Uwind,Tamb,Patm,Tblg,MFlow,betaW
               DOUBLE PRECISION MinFlow,GammaW,GrndRef

C              CONSTANTS
               DOUBLE PRECISION PI,Rair,SB,G

C              INTERNAL AND COMMON VARIABLES
               DOUBLE PRECISION Gb,Gd,Gg,GamMIN,Theta,Beta,lowg,hig,oldg,difg
               DOUBLE PRECISION t_trap,ThetaT,diag,Por,rho_amb
               DOUBLE PRECISION T_comp,T_crit,Trad,visc_amb,Flow,ThetaEnd,k_amb
               DOUBLE PRECISION Gamma,Thetar,Tau_a,TauC,Thetar_0,Tau_0
               DOUBLE PRECISION Thetard,Thetarg,Tau_ad,Tau_ag,Taud,Taug
               DOUBLE PRECISION prop_sh,Area,L,PS
               DOUBLE PRECISION AlphaDcoll,AlphaB,cp_amb,Wend,ecol,Area_cs
               DOUBLE PRECISION Tground,AreaW,Zeta,FB,FD
               DOUBLE PRECISION mx_dd,ma_augmdd,ma_augmdb,mx_db,ma_augmdg,mx_dg
               DOUBLE PRECISION Jdb,Gdb,Jdd,Gdd,Jdg,Gdg,Rb
               DOUBLE PRECISION QBsurf,QDsurf,Qsurf,Psh,PPV
               DOUBLE PRECISION RhoB,RhoGcoll,AlphaG,SsU1,SsU2,AlphaS
               DOUBLE PRECISION RhoDColl,ThetadW,ThetabW,ThetagW
               DOUBLE PRECISION EffPV,QE,EffEl,EffTh,QradW
               DOUBLE PRECISION hfilm,Tsa,QredW,Qpot,Tmix,FCS, FCG
               DOUBLE PRECISION Time0,TFINAL,DELT

               INTEGER*4 nb_surf,Bypass,PVSurf,PVOn
               INTEGER*4 i,j,nb_case,N,NbJ,SHCASE,FPASS,nb_corr,COUNT
```

```
      PARAMETER (nb_case=5,nbJ=11,nb_surf=8)
      DIMENSION Gamma(nbJ),Theta(nbJ),Beta(nbJ),Gd(nbJ),Gg(nbJ),Rb(nbJ)
      DIMENSION mx_dd(nbJ),Jdd(NbJ),Jdg(NbJ),mx_dg(nbJ),Gdg(nb_surf)
      DIMENSION Gb(nb_surf),Psh(nb_surf),PPV(nb_surf),Area(nb_surf)
      DIMENSION RhoB(nb_surf),RhoGcoll(nb_surf),PVSurf(nb_surf)
      DIMENSION RhoDColl(nb_surf*2),L(nb_surf*2),Jdb(nb_surf*2)
      DIMENSION mx_db(nb_surf*2),ma_augmdb((nb_surf*2),((nb_surf*2)+1))
      DIMENSION Gdb(nb_surf*2),FB((nb_surf*2),(nb_surf*2))
      DIMENSION prop_sh(nb_surf,(nb_case+1)),Gdd(nb_surf)
      DIMENSION PS(nb_surf),QBsurf(nb_surf),QDsurf(nb_surf)
      DIMENSION Qsurf(nb_surf),PVOn(nb_surf),AlphaG(nb_surf)
      DIMENSION ma_augmdd(nbJ,(nbJ+1)),ma_augmdg(nbJ,(nbJ+1))
      DIMENSION FD(NbJ,NbJ),AlphaB(nb_surf),AlphaDcoll(nb_surf)
      DIMENSION Thetar(nb_surf),Tau_a(nb_surf),TauC(nb_surf)

C     PVTtrcSOLVE Variables
      DOUBLE PRECISION Tcol,Tplen,Qrad_cs,Qrad_wc,Twall
      DOUBLE PRECISION Qconv_wa,Qwind,Qconv_ca,Qcond_wT,Qabs
      DOUBLE PRECISION Tout,Vs,Qu,effhx
      INTEGER*4 k

C     DATA STATEMENTS
      DATA PI/3.141592654/,Rair/0.2870/,g/9.8/
*******************************************************************************
c     TRNSYS FUNCTIONS
      TIME0=getSimulationStartTime()
      TFINAL=getSimulationStopTime()
      DELT=getSimulationTimeStep()

C    SET THE VERSION INFORMATION FOR TRNSYS
     IF(INFO(7).EQ.-2) THEN
        INFO(12)=16
        RETURN 1
      ENDIF
*******************************************************************************
C    DO ALL THE VERY LAST CALL OF THE SIMULATION MANIPULATIONS HERE
     IF (INFO(8).EQ.-1) THEN
        RETURN 1
      ENDIF
*******************************************************************************
C    DO ALL THE VERY FIRST CALL OF THE SIMULATION MANIPULATIONS HERE
     IF (INFO(7).EQ.-1) THEN
                IUNIT=INFO(1)
                ITYPE=INFO(2)

C      SET SOME INFO ARRAY VARIABLES TO TELL THE TRNSYS ENGINE HOW THIS TYPE IS TO WORK
       INFO(6)=NOUT
       INFO(9)=1
        INFO(10)=0 !STORAGE FOR VERSION 16 HAS BEEN CHANGED

C      CALL THE TYPE CHECK SUBROUTINE TO COMPARE WHAT THIS COMPONENT REQUIRES TO WHAT IS SUPPLIED IN
C      THE TRNSYS INPUT FILE
        CALL TYPECK(1,INFO,NI,NP,ND)

        DATA YCHECK/'IR1','IR1','IR1','IR1','DG1','DG1','VE1','TE1',
    1                   'PR3','DG1','TE1','TE1','MF1','MF1','DG1','DG1','DM1',
    1                   'IR1','IR1','TE1'/

        DATA OCHECK/'TE1','TE1','TE1','TE1','DM1','DM1','MF1','DM1',
    1                   'DM1','DM1','DM1','PW2','PW2','PW2','DM1','IR2'/
c       CALL RCHECK(INFO,YCHECK,OCHECK)
       RETURN 1
     ENDIF
*******************************************************************************
C    DO ALL OF THE INITIAL TIMESTEP MANIPULATIONS HERE - THERE ARE NO ITERATIONS AT THE INTIAL TIME
     IF (TIME .LT.(TIME0+DELT/2.D0)) THEN

C      SET THE UNIT NUMBER FOR FUTURE CALLS
       IUNIT=INFO(1)
       ITYPE=INFO(2)

c      Read the values of the parameters in sequential order
       a=PAR(1)
       b=PAR(2)
       h_trap=PAR(3)
       Por=PAR(4)
       pitch=PAR(5)
       length=PAR(6)
       dist=PAR(7)
       width=PAR(8)
       AlphaW=PAR(9)
       hp=PAR(10)
       e_back=PAR(11)
       e_wall=PAR(12)
       th=PAR(13)
       AlphaPl=PAR(14)
       e_coll=PAR(15)
       Uwall=PAR(16)
       PVMode=PAR(17)
       EffT=PAR(18)
       EffRef=PAR(19)
       Tref=PAR(20)
       TauAlfPV=PAR(21)
```

```
                e_PV=PAR(22)
                BypTemp=PAR(23)
                Tbypass=PAR(24)
                ngl=PAR(25)
                kgl=PAR(26)
                Lgl=PAR(27)
                PPV(3)=PAR(28)
                PPV(1)=PAR(29)
                PPV(2)=PAR(30)

C     CHECK THE PARAMETERS FOR PROBLEMS AND RETURN FROM THE SUBROUTINE IF AN ERROR IS FOUND
                IF(a.LE.0) CALL TYPECK(-4,INFO,0,1,0)
     IF(b.GE.a) CALL TYPECK(-4,INFO,0,2,0)
                IF(b.LE.0) CALL TYPECK(-4,INFO,0,2,0)
                IF(h_trap.LE.0) CALL TYPECK(-4,INFO,0,3,0)
                IF(Por.LE.0) CALL TYPECK(-4,INFO,0,4,0)
                IF(pitch.LE.0) CALL TYPECK(-4,INFO,0,5,0)
                IF(length.LE.0.OR.length.LE.a.OR.
     &          length.LE.b) CALL TYPECK(-4,INFO,0,6,0)
                IF(a.GE.dist.OR.dist.LE.0) CALL TYPECK(-4,INFO,0,7,0)
                IF(width.LE.dist.OR.width.LE.0) CALL TYPECK(-4,INFO,0,8,0)
                IF(AlphaW.LE.0.OR.AlphaW.GT.1)CALL TYPECK(-4,INFO,0,9,0)
                IF(hp.LE.0)CALL TYPECK(-4,INFO,0,10,0)
                IF(e_back.LT.0.OR.e_back.GT.1) CALL TYPECK(-4,INFO,0,11,0)
                IF(e_wall.LT.0.OR.e_wall.GT.1) CALL TYPECK(-4,INFO,0,12,0)
                IF(th.LE.0) CALL TYPECK(-4,INFO,0,13,0)
                IF(AlphaPl.LE.0.OR.AlphaPl.GE.1) CALL TYPECK(-4,INFO,0,14,0)
                IF(e_coll.LT.0.OR.e_coll.GT.1) CALL TYPECK(-4,INFO,0,15,0)
                IF(Uwall.LE.0) CALL TYPECK(-4,INFO,0,16,0)
                IF(PVMode.LT.1.OR.PVMode.GT.3) CALL TYPECK(-4,INFO,0,17,0)
                IF(EffRef.LE.0.OR.EffRef.GT.1) CALL TYPECK(-4,INFO,0,19,0)
                IF(TauAlfPV.LT.0.OR.TauAlfPV.GT.1) CALL TYPECK(-4,INFO,21,0)
                IF(e_PV.LE.0.OR.e_PV.GT.1) CALL TYPECK(-4,INFO,22,0)
                IF(BypTemp.LT.0.OR.BypTemp.GT.1)CALL TYPECK(-4,INFO,24,0)
                IF(PPV(3).LT.0.OR.PPV(3).GT.1)CALL TYPECK(-4,INFO,28,0)
                IF(PPV(1).LT.0.OR.PPV(1).GT.1)CALL TYPECK(-4,INFO,29,0)
                IF(PPV(2).LT.0.OR.PPV(2).GT.1)CALL TYPECK(-4,INFO,30,0)

     RETURN 1
     ENDIF
*******************************************************************************
C     RE-READ IN THE VALUES OF THE PARAMETERS IN SEQUENTIAL ORDER
                IF(INFO(1).NE.IUNIT) THEN

c               RESET THE UNIT NUMBER
                IUNIT=INFO(1)
                ITYPE=INFO(2)

                a=PAR(1)
                b=PAR(2)
                h_trap=PAR(3)
                Por=PAR(4)
                pitch=PAR(5)
                length=PAR(6)
                dist=PAR(7)
                width=PAR(8)
                AlphaW=PAR(9)
                hp=PAR(10)
                e_back=PAR(11)
                e_wall=PAR(12)
                th=PAR(13)
                AlphaPl=PAR(14)
                e_coll=PAR(15)
                Uwall=PAR(16)
                PVMode=PAR(17)
                EffT=PAR(18)
                EffRef=PAR(19)
                Tref=PAR(20)
                TauAlfPV=PAR(21)
                e_PV=PAR(22)
                BypTemp=PAR(23)
                Tbypass=PAR(24)
                ngl=PAR(25)
                kgl=PAR(26)
                Lgl=PAR(27)
                PPV(3)=PAR(28)
                PPV(1)=PAR(29)
                PPV(2)=PAR(30)
                ENDIF

C     RETRIEVE THE CURRENT VALUES OF THE INPUTS TO THIS MODEL FROM THE XIN ARRAY
                GbW=XIN(1)
                GdW=XIN(2)
                GgW=XIN(3)
                GbH=XIN(4)
                ThetaZ=XIN(5)
                GammaS=XIN(6)
                Uwind=XIN(7)
                Tamb=XIN(8)
                Patm=XIN(9)
                ThetabW=XIN(10)
                Tblg=XIN(11)
                Tsky=XIN(12)
```

```
            MinFlow=XIN(13)
            MFlow=XIN(14)
            BetaW=XIN(15)
            GammaW=XIN(16)
            GrndRef=XIN(17)
            Gh=XIN(18)
            GdH=XIN(19)
            Tsup=XIN(20)

C   CHECK THE INPUTS FOR PROBLEMS
    IF(Uwind.LT.0.) CALL TYPECK(-3,INFO,7,0,0)
    IF(Patm.LT.0.) CALL TYPECK(-3,INFO,9,0,0)
            IF(MinFlow.LT.0)CALL TYPECK(-3,INFO,13,0,0)
            IF(MFlow.LT.0)CALL TYPECK(-3,INFO,14,0,0)
            IF(MinFlow.GT.MFlow)CALL TYPECK(-3,INFO,14,0,0)
            IF(GrndRef.GT.1)CALL TYPECK(-3,INFO,17,0,0)

            IF(ErrorFound()) RETURN 1
*******************************************************************************
C           SET THE PROPORTION OF PV CELLS ON EACH SURFACE
            DO i=1,nb_surf
                        PVsurf(i)=0
            ENDDO

            IF (PVMode.EQ.1)THEN
                        DO i=1,nb_surf
                                    PPV(i)=0.0
                                    PVsurf(i)=0
                        ENDDO
            ELSE IF (PVMode.EQ.2)THEN
                        DO i=1,nb_surf
                                    IF (i.EQ.3) THEN
                                                PPV(i)=PPV(i)
                                                PVsurf(i)=1
                                    ELSE
                                                PPV(i)=0.0
                                                PVsurf(i)=0
                                    ENDIF
                        ENDDO
            ELSE
                        DO i=1,nb_surf
                                    IF (i.LE.4)THEN
                                                PPV(i)=PPV(i)
                                                PVsurf(i)=1
                                    ELSE
                                                PPV(i)=0.0
                                                PVsurf(i)=0
                                    ENDIF
                        ENDDO
            ENDIF

            IF (MFlow.LE.0.0)THEN
                        GamMIN=0.0
            ELSE
                        GamMIN=MinFlow/MFlow
            ENDIF
C           CALCULATE DIAMETER WITH THE RELATION OF Van Decker
            D=((4*POR*Pitch*Pitch)/PI)**0.5
*******************************************************************************
C           SET CONSTANTS VALUE
            SB=(5.67e-8)*3.6            !Stefan Boltzmann constant [kJ/hr*m2*K4]

C           MODIFY AND CONVERT PARAMETERS & INPUTS
            Uwind=Uwind*3600            !Convert m/s in m/h
            Tamb=Tamb+273.15
            Tsup=Tsup+273.15
            Tblg=Tblg+273.15
            Tsky=Tsky+273.15
            Tground=Tamb

C           CALCULATE TRAD
            FCS=(1+COSD(BetaW))/2.0
            FCG=(1-COSD(BetaW))/2.0
            Trad=(((Tground**4)*FCG)+((Tsky**4)*FCS))**0.25

C           CALCULATE TOTAL RADIATION ON THE COLLECTOR SURFACE
            QradW=GbW+GdW+GgW

C           IF NIGHTTIME, THE COLLECTOR IS AUTOMATICALLY BYPASSED
            Bypass=0
            IF(QradW.LT.1.0)THEN
                        Bypass=1
                        GO TO 110
            ENDIF

C           IF SUMMER, BYPASS COLLECTOR WHEN THE BYPASS OPTION IS SET TO 1
            IF (Tamb.GT.(Tbypass+273.15))THEN
                        IF(BypTemp.GE.1)THEN
                                    Bypass=1
                        ELSE
                                    Bypass=0
                        ENDIF
            ELSE
```

```
                    Bypass=0
            ENDIF

110         CONTINUE
*******************************************************************************
C           CALCULATE PV/THERMAL COLLECTOR GEOMETRIC CONSTANTS  AND
C           AREA FOR EACH SURFACE INCLUDING HOLES
*******************************************************************************
            t_trap=sqrt((h_trap**2)+(((a-b)/2)**2))       ![m]
            ThetaT=asin(h_trap/t_trap)    ![rad]
            ThetaEnd=ATAN(h_trap/(((a-b)/2)+Wend))
            diag=(((dist-((a+b)/2))**2)+(h_trap**2))**0.5
            nb_corr=AINT((width+dist-a)/dist)
            Wend=(width-a-(dist*nb_corr)+dist)/2

            i=1
            DO i=1,nb_surf
                    Area(i)=0.0
            ENDDO

            Area(1)=(dist-a)*length
            Area(2)=t_trap*length
            Area(3)=b*length
            Area(4)=Area(2)
            Area(5)=Wend*length
            Area(6)=Area(2)
            Area(7)=Area(5)
            Area(8)=Area(2)

            AreaW=width*length
            Area_cs=(width*hp)+(nb_corr*h_trap*((a+b)/2))
*******************************************************************************
C           CALCULATE AIR PROPERTIES AT AMBIENT TEMPERATURE WITH IDEAL GAS LAW
C           AND SUTHERLAND LAW
*******************************************************************************
            rho_amb=(Patm*0.001)/(Rair*Tamb)                                    ![kg/m3]
            visc_amb=(((1.71e-5)*((Tamb/273)**1.5)*((273+110.4)/
   &                (Tamb+110.4)))*3600                                                  ![kg/mh]
            cp_amb=(28.11+(0.001967*Tamb)+(0.4802e-5*Tamb*Tamb)-
   &                (1.966e-9*Tamb*Tamb*Tamb))*(1/28.97)   ![kJ/kgK]
            k_amb=(((2.414e-2)*((Tamb/273)**1.5)*((273+194.4)/
   &                (Tamb+194.4)))*3.6                                                                   ![kJ/mC]

            Flow=Mflow/rho_amb                                                                                  ![m3/hr]
*******************************************************************************
C           CALCULATE SOLAR ANGLES AND IRRADIATION FOR EACH SURFACE
*******************************************************************************
            DO i=1,NbJ
                    Gamma(i)=0.0
                    Beta(i)=0.0
                    Theta(i)=0.0
                    Rb(i)=0.0
                    Gg(i)=0.0
                    Gd(i)=0.0
            ENDDO

C           AZIMUTH ANGLE (Gamma)
            DO i=1,9,2
                    Gamma(i)=GammaW                         ![deg]
            ENDDO

            Gamma(2)=GammaW+ACOSD(SIND(BetaW)*COS(ThetaT))          ![deg]
            Gamma(4)=GammaW-ACOSD(SIND(BetaW)*COS(ThetaT))          ![deg]
            Gamma(6)=Gamma(4)
            Gamma(8)=Gamma(2)
            Gamma(10)=GammaW+ACOSD(SIND(BetaW)*COS(ThetaEnd))       ![deg]
            Gamma(11)=GammaW-ACOSD(SIND(BetaW)*COS(ThetaEnd))       ![deg]

            DO i=1,NbJ
                    IF(Gamma(i).GT.180)THEN
                            Gamma(i)=Gamma(i)-360.0
                    ELSE IF(Gamma(i).LT.(-1.0*180.0))THEN
                            Gamma(i)=Gamma(i)+360.0
                    ENDIF
            ENDDO

C           SLOPE (Beta)
            DO i=1,9,2
                    Beta(i)=BetaW
            ENDDO

            DO i=2,8,2
                    Beta(i)=ACOSD(COS(ThetaT)*COSD(BetaW))          ![deg]
            ENDDO
            Beta(10)=ACOSD(COS(ThetaEnd)*COSD(BetaW))       ![deg]
            Beta(11)=ACOSD(COS(ThetaEnd)*COSD(BetaW))

C           INCIDENCE ANGLE (Theta)
            DO i=1,NbJ
                    Theta(i)=ACOSD((COSD(ThetaZ)*COSD(Beta(i)))+
   &                (SIND(ThetaZ)*SIND(Beta(i))*COSD(GammaS-Gamma(i))))             !DEG
            ENDDO
```

```
C       SKIP THIS STEP WHEN NO RADIATION ON THE COLLECTOR
        IF (QradW.LE.(1.0)) GO TO 150

C       CALCULATE RATIO OF BEAM RADIATION (Rb)
        DO i=1,NbJ
                IF(COSD(ThetaZ).LE.0.001)THEN
                        Rb(i)=COSD(Theta(i))/0.001
                        ELSE
                        Rb(i)=(MAX(0.0,(COSD(Theta(i)))))/(COSD(ThetaZ))
                ENDIF
        ENDDO

C       BEAM RADIATION ON EACH SURFACE (GB)-ONLY FOR REAL SURFACES
        DO i=1,nb_surf
                Gb(i)=Rb(i)*GbH
        ENDDO

C       GROUND REFLECTED RADIATION ON EVERY FICTITOUS SURFACE (GG)
C       SKY DIFFUSE RADIATION ON EVERY FICTITIOUS SURFACE (GD)
C       ISOTROPIC MODEL
        Gg(9)=GgW
        Gd(9)=GdW
        DO i=10,11
                Gg(i)=0.5*(1-(COSD(Beta(i))))*GrndRef*Gh
                Gd(i)=GdH*(1+((COSD(Beta(i)))/2.0))
        ENDDO

        DO i=9,11
                IF (Gg(i).LE.0.001)THEN
                        Gg(i)=0.0
                        Gd(i)=0.0
                ENDIF
        ENDDO
        GO TO 160

150     CONTINUE
        DO i=1,Nb_surf
                Gb(i)=0.0
                Gd(i)=0.0
                Gg(i)=0.0
        ENDDO

160     CONTINUE
********************************************************************************
C       CALCULATE SOLAR OPTICAL PROPERTIES FOR EACH SURFACE
********************************************************************************
        DO i=1,nb_surf
                AlphaB(i)=0.0
                AlphaG(i)=0.0
                RhoB(i)=0.0
                RhoGcoll(i)=0.0

                Thetar(i)=0.0
                Tau_a(i)=0.0
                TauC(i)=0.0
        ENDDO

C       FIND EFFECTIVE SKY DIFFUSE AND GROUND REFLECTED ANGLES [DUFFIE AND BECKMAN, 1991]
        ThetadW=59.68-(0.1388*BetaW)+(0.001497*BetaW*BetaW)
        ThetaGW=90.0-(0.5788*BetaW)+(0.002693*BetaW*BetaW)

        IF (Lgl.LE.0.001)GO TO 170

        DO i=1,nb_surf
                Thetar(i)=ASIND(SIND(Theta(i))/ngl)
                Tau_a(i)=exp(-1*Kgl*Lgl/COSD(Thetar(i)))
                TauC(i)=Tau_a(i)*(1-(0.5*((((SIND(Thetar(i)-Theta(i)))**2)/
     &                      ((SIND(Thetar(i)+Theta(i)))**2))+
     &                      (((TAND(Thetar(i)-Theta(i)))**2)/
     &                      ((TAND(Thetar(i)+Theta(i)))**2)))))
        ENDDO

        DO i=1,nb_surf
                Thetar(i)=Thetar(i)
                Tau_a(i)=Tau_a(i)
                TauC(i)=TauC(i)
        ENDDO

        Tau_0=exp(-1*Kgl*Lgl)*(1-(((ngl-1)/(ngl+1))**2))/
     &              (1+(((ngl-1)/(ngl+1))**2))


        Thetard=ASIND(SIND(ThetadW)/ngl)
        Thetarg=ASIND(SIND(ThetagW)/ngl)

        Tau_ad=exp(-1*Kgl*Lgl/COSD(Thetard))
        Tau_ag=exp(-1*Kgl*Lgl/COSD(Thetarg))

        Taud=Tau_ad*(1-(0.5*((((SIND(Thetard-ThetadW))**2)/
     &                      ((SIND(Thetard+ThetadW))**2))+
     &                      (((TAND(Thetard-ThetadW))**2)/
     &                      ((TAND(Thetard+ThetadW))**2)))))
        Taug=Tau_ag*(1-(0.5*((((SIND(Thetarg-ThetagW))**2)/
```

```
     &                                    ((SIND(Thetarg+ThetagW))**2))+
     &                                    (((TAND(Thetarg-ThetagW))**2)/
     &                                    ((TAND(Thetarg+ThetagW))**2)))))

                DO i=1,nb_surf

                 Rhob(i)=PPV(i)*(1-TauC(i)-(1-Tau_a(i)))+((1-PPV(i))*(1-AlphaPl))
                 Rhogcoll(i)=PPV(i)*(1-Taug-(1-Tau_ag))+((1-PPV(i))*(1-AlphaPl))
                 RhoDcoll(i)=PPV(i)*(1-Taud-(1-Tau_ad))+((1-PPV(i))*(1-AlphaPl))

                 Alphab(i)=(PPV(i)*(TauC(i)/Tau_0)*TauAlfPV)+((1-PPV(i))*AlphaPl)
                 Alphadcoll(i)=(PPV(i)*(Taud/Tau_0)*TauAlfPV)+((1-PPV(i))*AlphaPl)

                ENDDO

170             CONTINUE
C               IF NO GLAZING ON THE PV CELLS, DO NOT TAKE INTO ACCOUNT THE
C               EFFECT OF THE INCIDENCE ANGLE
                IF (Lgl.LE.0.001)THEN
                 DO i=1,nb_surf
                                Rhob(i)=(PPV(i)*(1-TauAlfPV))+((1-PPV(i))*(1-AlphaPl))
                                Rhogcoll(i)=(PPV(i)*(1-TauAlfPV))+((1-PPV(i))*(1-AlphaPl))
                                RhoDcoll(i)=(PPV(i)*(1-TauAlfPV))+((1-PPV(i))*(1-AlphaPl))

                                Alphab(i)=(PPV(i)*TauAlfPV)+((1-PPV(i))*AlphaPl)
                                Alphadcoll(i)=(PPV(i)*TauAlfPV)+((1-PPV(i))*AlphaPl)
                 ENDDO
                ENDIF

C               SET RHOB, RHOD AND RHOG FOR EACH SURFACE
C               RHODCOLL(9 TO 16) ARE THE SHADED SURFACES TO CALCULATE THE BEAM DUE TO BEAM RADIATION
                RhoDcoll(10)=RhoDcoll(1)
                RhoDcoll(9)=RhoDcoll(1)

                DO i=11,16
                                RhoDcoll(i)=RhoDcoll(i-nb_surf)
                ENDDO

C               CALCULATE COLLECTOR EMISSIVITY AS A WEIGHTED AVERAGE OF THE
C               PANEL AND PV CELLS EMISSIVITIES
                IF (PVMode.EQ.1)THEN
                                PropPV=0.0
                                ecol=e_coll

                ELSEIF (PvMode.EQ.2)THEN
                                PropPV=PPV(3)*Area(3)*Nb_corr/(AreaW)
                                ecol=((1-PropPV)*e_coll)+(PropPV*e_pv)

                ELSE
                                PropPV=((PPV(1)*Area(1)*(Nb_corr-1))+
     &                  (PPV(3)*Area(3)*Nb_corr)+
     &                    (PPV(2)*Area(2)*(Nb_corr-1)*cos(ThetaT))+
     &                    (PPV(4)*Area(4)*(Nb_corr-1)*cos(ThetaT)))/AreaW
                                ecol=((1-PropPV)*e_coll)+(PropPV*e_pv)
                ENDIF
*********************************************************************************
C               FIND SHADING PORTION AND VIEW FACTORS
*********************************************************************************
                AlphaS=90.0-ThetaZ              ![deg]

C               CRITICAL ANGLE T_crit
                T_crit=atan(h_trap/(dist-a+((a-b)/2))) ![rad]

C               CALCULATE SCALAR PRODUCT S.U1 AND S.U2
                SsU1=(-1.0*COSD(GammaW)*COSD(AlphaS)*SIND(GammaS))+
     &           (COSD(AlphaS)*COSD(GammaS)*SIND(GammaW))
                SsU2=(COSD(AlphaS)*SIND(GammaS)*SIND(GammaW)*SIND(BetaW))
     &           +(COSD(AlphaS)*COSD(GammaS)*COSD(GammaW)*SIND(BetaW))+
     &           (SIND(AlphaS)*COSD(BetaW))

C               AVOID DIVIDING BY ZERO!
                IF (SsU1.EQ.0)THEN
                                T_comp=ACOS(ABS(SsU2))
                ELSE
                                T_comp=ATAN(ABS(SsU2)/ABS(SsU1))
                ENDIF

                IF(SsU1.LT.0.AND.SsU2.GE.0)THEN
                                T_comp=PI-T_comp
                ELSE IF(SsU1.LT.0.AND.SsU2.LT.0)THEN
                                T_comp=-1.0*(PI-T_comp)
                ELSE IF(SsU1.GE.0.AND.SsU2.LT.0)THEN
                                T_comp=-1.0*T_comp
                ELSE
                                T_comp=T_comp
                ENDIF

                DO i=1,nb_surf
                                PS(i)=0.0
                                DO j=1,(nb_case+1)
                                prop_sh(i,j)=2.0
                                ENDDO
                ENDDO
```

```
      DO i=1,(nb_surf*2)
            L(i)=0.0
                  DO j=1,(nb_surf*2)
                        FB(i,j)=0.0
                  ENDDO
      ENDDO

      SHCASE=0
      FPASS=1
200   CONTINUE
      IF (GbW.GT.0) THEN

C           "CASE 1"
            IF(T_comp.GT.(0.0).AND.T_comp.LE.T_crit)THEN

                  SHCASE=1
                  !CALCULATE SHADING

                  prop_sh(1,1)=1.0
                  prop_sh(2,1)=1.0
               prop_sh(4,1)=(diag*sin(T_crit-T_comp))/
     &                  (t_trap*sin(T_comp+ThetaT))
                  prop_sh(3,1)=0.0
                  prop_sh(5,1)=0.0
                  prop_sh(6,1)=0.0
                  prop_sh(8,1)=1.0
                  prop_sh(7,1)=1.0

                  DO i=1,nb_surf
                        PS(i)=prop_sh(i,1)
                  ENDDO

!                 CALCULATE length OF EACH SURFACE
                  CALL CALCULDIM(PS,DIST,A,T_TRAP,B,Wend,L)

!                 CALCULATE RELEVANT SHAPE FACTORS

               CALL CALCULSF2(L(9),L(4),t_trap,(PI-ThetaT),FB(9,4))
                  CALL CALCULSF6(L(12),L(11),(Dist-a),(PI-ThetaT),FB(12,11))
                  CALL CALCULSF4(L(4),L(11),(Dist-a),(PI-ThetaT),FB(4,11))
                  CALL CALCULSF1(L(15),L(16),(PI-ThetaT),FB(15,16))
                  CALL CALCULSF1(L(9),L(12),(PI-ThetaT),FB(9,12))
                  CALL CALCULSF1(L(9),L(11),(PI-ThetaT),FB(9,11))
                  CALL CALCULSF1(L(5),L(6),(PI-ThetaT),FB(5,6))

                  FB(4,9)=(FB(9,4)*L(9))/(L(4))
                  FB(11,4)=(FB(4,11)*L(4))/(L(11))
                  FB(12,9)=(FB(9,12)*L(9))/(L(12))
                  FB(11,12)=(FB(12,11)*L(12))/(L(11))
                  FB(11,9)=(FB(9,11)*L(9))/(L(11))
                  FB(16,15)=(FB(15,16)*L(15))/(L(16))
                  FB(6,5)=(FB(5,6)*L(5))/(L(6))

C           "CASE 2"
                  ELSE IF(T_comp.GT.T_crit.AND.T_comp.LE.ThetaT)THEN

                  SHCASE=2

                  prop_sh(1,2)=(h_trap-(tan(T_comp)*((a-b)/2)))/
     &                              ((dist-a)*tan(T_comp))
                  prop_sh(2,2)=1.0
                  prop_sh(3,2)=0.0
                  prop_sh(4,2)=0.0
                  prop_sh(5,2)=0.0
                  prop_sh(6,2)=0.0
                  prop_sh(7,2)=(h_trap-(tan(T_comp)*((a-b)/2)))/
     &                              (Wend*tan(T_comp))

                  IF((prop_sh(7,2)).GT.(1.0))THEN
                        prop_sh(7,2)=1.0
                  ENDIF
                  prop_sh(8,2)=1.0

                  DO i=1,nb_surf
                        PS(i)=prop_sh(i,2)
                  ENDDO

!                 CALCULATE length OF EACH SURFACE
                  CALL CALCULDIM(PS,DIST,A,T_TRAP,B,Wend,L)

!                 CALCULATE RELEVANT SHAPE FACTORS
                  CALL CALCULSF1(L(1),L(4),(PI-ThetaT),FB(1,4))
                  CALL CALCULSF1(L(9),L(11),(PI-ThetaT),FB(9,11))
                  CALL CALCULSF5(L(9),L(4),(Dist-a),(PI-ThetaT),FB(9,4))
                  CALL CALCULSF3(L(4),(Dist-a),(PI-ThetaT),FB(4,11))
                  CALL CALCULSF5(L(1),L(11),(Dist-a),(PI-ThetaT),FB(1,11))
                  CALL CALCULSF1(L(15),L(16),(PI-ThetaT),FB(15,16))
                  FB(16,15)=(FB(15,16)*L(15))/(L(16))

                  IF((PS(7)).LT.(1.0)) THEN
                        CALL CALCULSF5(L(7),L(16),Wend,(PI-ThetaT),FB(7,16))
                        FB(16,7)=(FB(7,16)*L(7))/(L(16))
```

```
                ENDIF

                CALL CALCULSF1(L(5),L(6),(PI-ThetaT),FB(5,6))

                FB(4,1)=(L(1)*FB(1,4))/(L(4))
                FB(4,9)=(FB(9,4)*L(9))/(L(4))
                FB(11,4)=FB(4,11)
                FB(11,9)=(FB(9,11)*L(9))/(L(11))
                FB(11,1)=(FB(1,11)*L(1))/(L(11))
                FB(6,5)=(FB(5,6)*L(5))/(L(6))

C               "CASE 3"
                ELSE IF(T_comp.GT.ThetaT.AND.T_comp.LE.(PI-ThetaT))THEN
                SHCASE=3

                prop_sh(1,3)=0.0
                prop_sh(2,3)=0.0
                prop_sh(3,3)=0.0
                prop_sh(4,3)=0.0
                prop_sh(5,3)=0.0
                prop_sh(6,3)=0.0
                prop_sh(7,3)=0.0
                prop_sh(8,3)=0.0

                DO i=1,nb_surf
                        PS(i)=prop_sh(i,3)
                ENDDO

!               CALCULATE length OF EACH SURFACE
                CALL CALCULDIM(PS,DIST,A,T_TRAP,B,Wend,L)

!               CALCULATE SHAPE FACTOR FOR THE EXISTING SURFACES

                CALL CALCULSF1 (L(1),L(4),(PI-ThetaT),FB(1,4))
                CALL CALCULSF1 (L(1),L(2),(PI-ThetaT),FB(1,2))
                CALL CALCULSF1 (L(7),L(8),(PI-ThetaT),FB(7,8))
                CALL CALCULSF1 (L(5),L(6),(PI-ThetaT),FB(5,6))
                CALL CALCULSF3(L(2),(Dist-a),(PI-ThetaT),FB(2,4))

                FB(2,1)=(FB(1,2)*L(1))/(L(2))
                FB(4,1)=(FB(1,4)*L(1))/(L(4))
                FB(4,2)=FB(2,4)
                FB(8,7)=(FB(7,8)*L(7))/(L(8))
                FB(6,5)=(FB(5,6)*L(5))/(L(6))


C               "CASE 4"
                ELSE IF(T_comp.GT.(PI-ThetaT).AND.
     &                  T_comp.LE.(PI-T_crit))THEN
                SHCASE=4

                prop_sh(1,4)=(h_trap-(tan(PI-T_comp)*((a-b)/2)))/
     &                                  ((dist-a)*tan(PI-T_comp))

                prop_sh(2,4)=0.0
                prop_sh(3,4)=0.0
                prop_sh(4,4)=1.0
                prop_sh(5,4)=(h_trap-(tan(PI-T_comp)*((a-b)/2)))/
     &                                  (Wend*tan(PI-T_comp))
                IF((prop_sh(5,4)).GT.(1.0))THEN
                        prop_sh(5,4)=1.0
                ENDIF

                prop_sh(6,4)=1.0
                prop_sh(7,4)=0.0
                prop_sh(8,4)=0.0

                DO i=1,nb_surf
                        PS(i)=prop_sh(i,4)
                ENDDO

!               CALCULATE length OF EACH SURFACE
                CALL CALCULDIM(PS,DIST,A,T_TRAP,B,Wend,L)

!               CALCULATE RELEVANT SHAPE FACTORS
                CALL CALCULSF1 (L(10),L(12),(PI-ThetaT),FB(10,12))
                CALL CALCULSF1 (L(1),L(2),(PI-ThetaT),FB(1,2))
                CALL CALCULSF1 (L(7),L(8),(PI-ThetaT),FB(7,8))
                CALL CALCULSF1 (L(13),L(14),(PI-ThetaT),FB(13,14))
                CALL CALCULSF5 (L(1),L(12),(Dist-a),(PI-ThetaT),FB(1,12))
                CALL CALCULSF3 (L(12),(Dist-a),(PI-ThetaT),FB(12,2))
                CALL CALCULSF5 (L(10),L(2),(Dist-a),(PI-ThetaT),FB(10,2))

                IF((PS(5)).LT.(1.0)) THEN
                 CALL CALCULSF5 (L(5),L(14),Wend,(PI-ThetaT),FB(5,14))
                        FB(14,5)=(FB(5,14)*L(5))/(L(14))
                ENDIF

                FB(12,10)=(FB(10,12)*L(10))/(L(12))
                FB(12,1)=(FB(1,12)*L(1))/(L(12))
                FB(2,12)=FB(12,2)
                FB(2,10)=(FB(10,2)*L(10))/(L(2))
                FB(2,1)=(FB(1,2)*L(1))/(L(2))
```

```
                                        FB(8,7)=(FB(7,8)*L(7))/(L(8))
                                        FB(14,13)=(FB(13,14)*L(13))/(L(14))

C                               "CASE 5"
                                ELSE IF(T_comp.GT.(PI-T_crit).AND.T_comp.LE.PI)THEN
                                        SHCASE=5

                                        prop_sh(1,5)=1.0
                                  prop_sh(2,5)=(diag*sin(T_comp-PI+T_crit))/
     &                                          (t_trap*sin(PI-T_comp+ThetaT))
                                        prop_sh(3,5)=0.0
                                        prop_sh(4,5)=1.0
                                        prop_sh(6,5)=1.0
                                        prop_sh(7,5)=0.0
                                        prop_sh(8,5)=0.0
                                        prop_sh(5,5)=1.0


                                        DO i=1,nb_surf
                                                PS(i)=prop_sh(i,5)
                                        ENDDO

!                                       CALCULATE length OF EACH SURFACE
                                        CALL CALCULDIM(PS,DIST,A,T_TRAP,B,Wend,L)

!                                       CALCULATE RELEVANT SHAPE FACTORS
                                        CALL CALCULSF1 (L(9),L(12),(PI-ThetaT),FB(9,12))
                                        CALL CALCULSF1 (L(9),L(11),(PI-ThetaT),FB(9,11))
                                        CALL CALCULSF1 (L(7),L(8),(PI-ThetaT),FB(7,8))
                                        CALL CALCULSF1 (L(13),L(14),(PI-ThetaT),FB(13,14))
                                        CALL CALCULSF6 (L(11),L(12),(Dist-a),(PI-ThetaT),FB(11,12))
                                        CALL CALCULSF4 (L(2),L(12),(Dist-a),(PI-ThetaT),FB(2,12))
                                        CALL CALCULSF2 (L(9),L(2),t_trap,(PI-ThetaT),FB(9,2))

                                        IF((PS(5)).LT.(1.0)) THEN
                                         CALL CALCULSF5 (L(5),L(14),Wend,(PI-ThetaT),FB(5,14))
                                                FB(14,5)=(FB(5,14)*L(5))/(L(14))
                                        ENDIF

                                        FB(12,9)=(FB(9,12)*L(9))/L(12)
                                        FB(12,11)=(FB(11,12))*L(11)/L(12)
                                        FB(12,2)=(FB(2,12)*L(2))/(L(12))
                                        FB(11,9)=(FB(9,11)*L(9))/(L(11))
                                        FB(2,9)=(FB(9,2)*L(9))/(L(2))
                                        FB(8,7)=(FB(7,8)*L(7))/(L(8))
                                        FB(14,13)=(FB(13,14)*L(13))/(L(14))

                                ENDIF

                                DO i=1,nb_surf
                                        prop_sh(i,(nb_case+1))=MIN(prop_sh(i,1),prop_sh(i,2),
     &                                                          prop_sh(i,3),prop_sh(i,4),prop_sh(i,5))

                                ENDDO

                        ENDIF

                        DO i=1,nb_surf
                                IF(prop_sh(i,(nb_case+1)).GE.2.0)THEN
                                        prop_sh(i,(nb_case+1))=1.0
                                ELSE
                                        prop_sh(i,(nb_case+1))=prop_sh(i,(nb_case+1))
                                ENDIF
                        ENDDO

                        IF (FPASS.EQ.2) THEN
                                go to 210
                        ENDIF
*******************************************************************************
C               CALCULATE ABSORBED SOLAR RADIATION DUE TO BEAM
*******************************************************************************
C               SET MATRIX MA_AUGMDB

                        DO i=1,(nb_surf*2)
                                mx_db(i)=0.0
                                DO j=1,((nb_surf*2)+1)
                                        ma_augmdb(i,j)=0.0
                                ENDDO
                        ENDDO

                        DO i=1,(nb_surf*2)
                                DO j=1,(nb_surf*2)
                                        IF(i.eq.j)THEN
                                                ma_augmdb(i,j)=1.0
                                        ELSE
                                                ma_augmdb(i,j)=-1.0*RhoDcoll(i)*FB(i,j)

                                        ENDIF
                                ENDDO
                        ENDDO

                        DO i=1,nb_surf
                                ma_augmdb(i,17)=RhoB(i)*Gb(i)
```

```
            ENDDO


            CALL SOLVEMATRIX((nb_surf*2),ma_augmdb,mx_db)

            DO i=1,(nb_surf*2)
                  Jdb(i)=mx_db(i)
            ENDDO

C           RE CALCULATE CONFIGURATION FACTORS
            FPASS=2

            DO i=1,(nb_surf*2)
                  DO j=1,(nb_surf*2)
                        FB(i,j)=0.0
                  ENDDO
            ENDDO

C           RE CALCULATE SHAPE FACTORS
            GO TO 200
210         CONTINUE

            DO i=1,(nb_surf*2)
                  Gdb(i)=0.0
                  DO j=1,(nb_surf*2)
                        Gdb(i)=Gdb(i)+(FB(i,j)*Jdb(j))
                  ENDDO
            ENDDO
*******************************************************************************
C           CALCULATE ABSORBED SOLAR RADIATION DUE TO SKY DIFFUSE
C           AND GROUND REFLECTED RADIATION
*******************************************************************************
C           CALCULATE VIEW FACTORS
            FPASS=1
220         CONTINUE

            DO i=1,(NbJ)
                  DO j=1,(NbJ)
                        FD(i,j)=0.0
                  ENDDO
            ENDDO

            CALL CALCULSF1 ((dist-a),t_trap,(PI-ThetaT),FD(1,2))
            FD(2,1)=(FD(1,2)*(dist-a))/(t_trap)
            CALL CALCULSF3(t_trap,(Dist-a),(PI-ThetaT),FD(2,4))
            FD(4,2)=FD(2,4)
            FD(1,4)=FD(1,2)
            FD(4,1)=FD(2,1)

            CALL CALCULSF1((dist-b),t_trap,ThetaT,FD(9,2))
            FD(2,9)=(FD(9,2)*(dist-b))/t_trap
            FD(4,9)=FD(2,9)
            FD(9,2)=0.0

            CALL CALCULSF1 (Wend,t_trap,(PI-ThetaT),FD(5,6))
            FD(6,5)=(FD(5,6)*Wend)/t_trap
            FD(8,7)=FD(6,5)
            FD(7,8)=FD(5,6)
            CALL CALCULSF7 ((Dist-a),(Dist-b),h_trap,FD(1,9))

            FD(7,10)=1-FD(7,8)
            FD(8,10)=1.0-FD(8,7)
            FD(5,11)=FD(7,10)
            FD(6,11)=FD(8,10)

            IF(FPASS.EQ.3)GO TO 240

C           Set matrix ma_augmdd and ma_augmdg
            DO i=1,nbJ
                  mx_dd(i)=0.0
                  mx_dg(i)=0.0
                  DO j=1,(nbJ+1)
                        ma_augmdd(i,j)=0.0
                        ma_augmdg(i,j)=0.0
                  ENDDO
            ENDDO

            DO i=1,nbJ
                  DO j=1,nbJ
                        IF(i.eq.j)THEN
                              ma_augmdd(i,j)=1.0
                              ma_augmdg(i,j)=1.0
                        ELSE
                              IF(i.LE.nb_surf)THEN
                                    ma_augmdd(i,j)=(-1.0*RhoDcoll(i)*FD(i,j))
                                    ma_augmdg(i,j)=(-1.0*RhoDcoll(i)*FD(i,j))
                              ELSE
                                    ma_augmdd(i,j)=0.0
                                    ma_augmdg(i,j)=0.0
                              ENDIF
                        ENDIF
                  ENDDO
            ENDDO
```

```
            ma_augmdd(3,12)=RhoDcoll(3)*GdW
            ma_augmdd(9,12)=Gd(9)
            ma_augmdd(10,12)=Gd(10)
            ma_augmdd(11,12)=Gd(11)

            ma_augmdg(3,12)=RhoGcoll(3)*GgW
            ma_augmdg(9,12)=Gg(9)
            ma_augmdg(10,12)=Gg(10)
            ma_augmdg(11,12)=Gg(11)

            IF (FPASS.eq.2)THEN
                  go to 230
            ENDIF

            CALL SOLVEMATRIX(NbJ,ma_augmdd,mx_dd)

            DO i=1,NbJ
                  Jdd(i)=mx_dd(i)
            ENDDO

c           RE CALCULATE CONFIGURATION FACTORS
            FPASS=2
            GO TO 220
230         CONTINUE

            CALL SOLVEMATRIX(NbJ,ma_augmdG,mx_dG)
            DO i=1,NbJ
                  Jdg(i)=mx_dg(i)
            ENDDO

c           RE CALCULATE CONFIGURATION FACTORS
            FPASS=3
            GO TO 220
240         CONTINUE

            DO i=1,nb_surf
                  Gdd(i)=0.0
                  Gdg(i)=0.0
            ENDDO

            DO i=1,nb_surf
                  IF(i.EQ.3)THEN
                        Gdd(i)=GdW
                        Gdg(i)=GgW
                  ELSE
                        DO j=1,NbJ
                              Gdd(i)=Gdd(i)+(FD(i,j)*Jdd(j))
                              Gdg(i)=Gdg(i)+(FD(i,j)*Jdg(j))
                        ENDDO
                  ENDIF
            ENDDO
*******************************************************************************
C           DETERMINE QABS ON EACH TYPE OF SURFACE i
*******************************************************************************
C           DETERMINE % SHADING
            DO i=1,Nb_surf
                  Psh(i)=0.0
                  Psh(i)=prop_sh(i,(nb_case+1))
                  QBsurf(i)=0.0
                  QDsurf(i)=0.0
                  Qsurf(i)=0.0
            ENDDO

            IF(QradW.GT.(1.0))THEN

C           FIND DIFFUSE AND BEAM PORTION OF THE TOTAL ABSORBED
C           SOLAR RADIATION
            DO i=1,Nb_surf
                  QBsurf(i)=(1-Por)*Area(i)*AlphaB(i)*Gb(i)*(1-Psh(i))

                  IF(i.EQ.1)THEN
                        IF(SHCASE.EQ.4)THEN
                              QDsurf(i)=(1-Por)*AlphaDcoll(i)*Area(i)*(Gdd(i)
     &                                    +Gdg(i)+(Gdb(i)*(1-Psh(i)))+
     &                                    (Gdb(i+9)*Psh(i)))
                        ELSE
                              QDsurf(i)=(1-Por)*AlphaDcoll(i)*Area(i)*(Gdd(i)
     &                                    +Gdg(i)+(Gdb(i)*(1-Psh(i)))+
     &                                    (Gdb(i+nb_surf)*Psh(i)))
                        ENDIF
                  ELSE IF(i.EQ.2)THEN
                              QDsurf(i)=(1-Por)*AlphaDcoll(i)*Area(i)*(Gdd(i)
     &                                    +Gdg(i)+(Gdb(i)*(1-Psh(i)))+
     &                                    (Gdb(i+9)*Psh(i)))

                  ELSE IF(i.EQ.3)THEN
                              QDsurf(i)=(1-Por)*AlphaDcoll(i)*Area(i)*
     &                                    (Gdd(i)+Gdg(i))

                  ELSE
                        QDsurf(i)=(1-Por)*AlphaDcoll(i)*Area(i)*(Gdd(i)+
     &                                    Gdg(i)+(Gdb(i)*(1-Psh(i)))+
```

```
   &                                                                (Gdb(i+nb_surf)*Psh(i)))

                        ENDIF
                        Qsurf(i)=QBsurf(i)+QDsurf(i)
                ENDDO

                ELSE
                        DO N=1,Nb_surf
                                QBsurf(i)=0.0
                                QDsurf(i)=0.0
                                Qsurf(i)=QBsurf(i)+QDsurf(i)
                        ENDDO
                ENDIF
*******************************************************************************
C               OPTIMIZE FLOW RATE AND SOLVE SUBROUTINE FOR UNKNOWN HEAT FLOWS AND TEMPERATURES
C               In this model, contrary to Summers' (1995), the collector is automatically bypassed
C    at night and both the thermal and electrical output are set to zero.
C               In the summer, however, the user has the choice to enable or disable an option
C               that will automatically open the collector bypass damper if the ambient
C               temperature becomes greater than a selected bypass temperature, Tbypass.
C               In such a case, the collector will be solved under bypass conditions, i.e.
C               with a air mass flowrate of zero. If this option is disabled, the mass flowrate of
C               air going through the collector (m) will be set to (mmin).In winter time,
C               during the day, if at the minimum flowrate, the mixed temperature is found
C               to be lower than Tsup, then the mass flowrate is set to mmin. If for the lowest
C               and highest value of Gamma, the mixed temperature is found to be higher
C               than Tsup, then the flowrate in the collector is also set to mmin, unless
C               the summer bypass option was enabled by the user. In any other cases,
C               the mass flowrate, m, that minimizes the auxiliary heat required, i.e.
C               when Tmix=Tsup, is determined using the bissection method.
*******************************************************************************
C               SET THE FIRST GAMMA TO THE MINIMUM GAMMA
                Gam=GamMIN
                IF(bypass.GE.1)THEN
                        Gam=0.0
                ENDIF

C               CALL SUBOURINTE
                CALL PVTtrcSOLVE(Tcol,Tplen,Qrad_cs,Qrad_wc,Twall,
   &                                        Qconv_wa,Qwind,Qconv_ca,Qcond_wT,Qabs,
   &                                        Qu,Tout,effhx,Vs)
C               CALCULATE TMIX
                Tmix=((Gam*Tout)+((1-Gam)*(Tblg)))

C               EXIT IF COLLECTOR IS BYPASSED
                IF (Bypass.GE.1) THEN
                        Tmix=(GamMIN*Tamb)+((1-GamMIN)*Tblg)
                GO TO 260
                ENDIF

c               EXIT IF SUMMER AND NO BYPASS
                IF(Tamb.GE.(Tbypass+273.15))GO TO 260

C               WINTER DAYTIME
C               OPTIMIZE FLOW RATE THROUGH COLLECTOR TO MINIMIZE REQUIRED AUXILIARY ENERGY
                IF (Tmix.LT.Tsup)THEN
C               MORE FLOW RATE WILL GIVE MORE ENERGY BUT WILL REQUIRE MORE QAUX, SET GAM TO GAMMIN
                                GAM=GamMIN
                                GO TO 260
                ELSE
C               Tmix>Tsup=>TRY GAM=1
                        Gam=1.0
                        CALL PVTtrcSOLVE(Tcol,Tplen,Qrad_cs,Qrad_wc,Twall,
   &                                        Qconv_wa,Qwind,Qconv_ca,Qcond_wT,Qabs,
   &                                        Qu,Tout,effhx,Vs)

                        Tmix=((Gam*Tout)+((1-Gam)*(Tblg)))

                        IF(Tmix.LT.Tsup)THEN
C                       FIND GAM FOR Tmix=Tsup USING BISSECTION METHOD
                        COUNT=0
                        lowg=GamMIN
                        hig=1.0
                        GAM=(lowg+hig)/2.0
250                     CONTINUE
                        COUNT=COUNT+1
                        CALL PVTtrcSOLVE(Tcol,Tplen,Qrad_cs,Qrad_wc,Twall,
   &                                        Qconv_wa,Qwind,Qconv_ca,Qcond_wT,Qabs,
   &                                        Qu,Tout,effhx,Vs)

                        Tmix=((Gam*Tout)+((1-Gam)*(Tblg)))
                                IF(Tmix.LT.Tsup)THEN
                                        hig=GAM
                                ELSE
                                        lowg=GAM
                                ENDIF
                                oldg=Gam
                                Gam=(lowg+hig)/2.0
                                difg=abs(GAM-oldg)
255                                     IF((difg.gt.0.01).AND.(COUNT.LT.150))GO TO 250
C                                       ERROR MESSAGE IN CASE IT DOESN'T CONVERGE

                        ELSE
```

```
C                       Tsup IS TOO HOT
                                IF (BypTemp.GE.1)THEN
C                       SUMMER BYPASS SHOULD HAVE BEEN OPENED BECAUSE Tsup IS TOO HOT
                                        Gam=0.0
                                        Bypass=1.0
                                        CALL PVTtrcSOLVE(Tcol,Tplen,Qrad_cs,
     &                          Qrad_wc,Twall,
     &                          Qconv_wa,Qwind,Qconv_ca,Qcond_wT,Qabs,
     &                          Qu,Tout,effhx,Vs)
                                        Tmix=(GamMIN*Tamb)+((1-GamMIN)*Tblg)
                                        GO TO 260
                                ELSE
C                       NO SUMMER BYPASS AND GAM IS SET TO GAMMIN
                                        Gam=GamMIN
                                        CALL PVTtrcSOLVE(Tcol,Tplen,Qrad_cs,
     &                          Qrad_wc,Twall,
     &                          Qconv_wa,Qwind,Qconv_ca,Qcond_wT,Qabs,
     &                          Qu,Tout,effhx,Vs)

                                ENDIF
                        ENDIF
                ENDIF

260             CONTINUE
*******************************************************************************
C               CALCULATE THE REDUCED WALL HEAT LOSSES, THE ELECTRICITY PRODUCED
C               AND THE THERMAL AND ELECTRICAL EFFICIENCIES
*******************************************************************************
C               SET FILM COEFFICIENT TO 15 W/m2 C [ENERMODAL, 1994]
                hfilm=15*3.6  !Film heat transfer coefficient [kJ/hm2C]

C               FIND SOL-AIR TEMPERATURE [ASHRAE, 1993]
                Tsa=Tamb+((AlphaW*QradW)/hFilm)         ![K}
                Qpot=AreaW*Uwall*(Tblg-Tsa)                                             ![kJ/hr]
                QredW=Qpot-(AreaW*Uwall*(Tblg-Tplen))![kJ/hr]
                QE=0.0
                EffPV=0.0

                IF (PVMode.EQ.1) GO TO 300
                IF (QradW.LT.1.0)GO TO 300

                EffPV=Effref+(EffT*((Tcol-273.15)-Tref))

C               TURN PV OFF ON SURFACES WHERE PV CELLS ARE SHADED
                DO i=1,Nb_surf
                        IF((PVsurf(i).GE.0.5).AND.(PSh(i).GT.0.0))THEN
                                PVOn(i)=0
                        ENDIF
                ENDDO

                IF (PVMode.EQ.2)THEN
                        QE=Qsurf(3)*PPV(3)*EffPV*nb_corr

                ELSE IF (PVMode.EQ.3)THEN
                        IF((PVON(1).LE.0).OR.(PVON(2).LE.0).OR.(PVON(4).LE.0))THEN
C                       SHADING, TAKE THE MINIMUM OF THE DIFFUSE RADIATION
                                QE=EffPV*(MIN((QDsurf(3)/(Area(3)*AlphaDcoll(3))),
     &                          (QDsurf(1)/(Area(1)*AlphaDcoll(1))),
     &                          (QDsurf(2)/(Area(2)*AlphaDcoll(2))),
     &                          (QDsurf(4)/(Area(4)*AlphaDcoll(4))))*
     &                          ((Nb_corr*Area(3)*PPV(3))+
     &                          ((Nb_corr-1)*((Area(1)*PPV(1))+(Area(2)*PPV(2))+
     &                          (Area(4)*PPV(4)))))
                        ELSE
C                       NO SHADING, TAKE THE MINIMUM OF THE TOTAL RADIATION
                                QE=EffPV*(MIN((QBsurf(3)/(AlphaB(3)*Area(3)))+
     &                          (QDsurf(3)/(Area(3)*AlphaDcoll(3))),
     &                          ((QBsurf(1)/(Area(1)*AlphaB(1)))+
     &                          (QDsurf(1)/(Area(1)*AlphaDcoll(1)))),
     &                          ((QBsurf(2)/(Area(2)*AlphaB(2)))+
     &                          (QDsurf(2)/(Area(2)*AlphaDcoll(2)))),
     &                          ((QBsurf(4)/(Area(4)*AlphaB(4)))+
     &                          (QDsurf(4)/(Area(4)*AlphaDcoll(4))))))*
     &                          ((Nb_corr*Area(3)*PPV(3))+
     &                          ((Nb_corr-1)*(Area(1)*PPV(1)+Area(2)*PPV(2)+
     &                          Area(4)*PPV(4))))
                        ENDIF
                ENDIF
300             CONTINUE

                IF (QradW.GT.1.0)THEN
                        IF(Qu.GT.1.0)THEN
                                Effth=(Qu/(AreaW*QradW))
                        ELSE
                                Qu=0.0
                                Effth=0.0
                        ENDIF
                        IF(Qe.GT.1.0)THEN
                                Effel=QE/(AreaW*QradW)
                        ELSE
                                Qe=0.0
                                Effel=0.0
                        ENDIF
```

```
            ELSE
                  Effth=0.0
                  Effel=0.0
                  effhx=0.0
            ENDIF

C           CONVERT TEMPERATURES FROM CELCIUS TO KELVIN
            Tcol=Tcol-273.15
            Tplen=Tplen-273.15
            Tout=Tout-273.15
            Tmix=Tmix-273.15
            QU=QU*1000/3600
            QE=QE*1000/3600
            QredW=QredW*1000/3600
            Qabs=Qabs*1000/3600/AreaW

C   SET THE OUTPUTS FROM THIS MODEL IN SEQUENTIAL ORDER AND GET OUT
            OUT(1)=Tcol             !degC
            OUT(2)=Tplen            !degC
            OUT(3)=Tmix             !degC
            OUT(4)=Tout             !degC
            OUT(5)=GAM                      !0 TO 1
            OUT(6)=1-GAM            !0 TO 1
            OUT(7)=GAM*MFLOW        !kg/hr
            OUT(8)=effhx
            OUT(9)=Effth
            OUT(10)=Effel
            OUT(11)=EffPV
            OUT(12)=QU              !W
            OUT(13)=QE              !W
            OUT(14)=QredW
            OUT(15)=Bypass
            OUT(16)=Qabs            !W/m2

            RETURN 1
    END SUBROUTINE Type201
*************************************************************************
            SUBROUTINE PVTtrcSOLVE (Tcol,Tplen,Qrad_cs,Qrad_wc,Twall,
     &                                                  Qconv_wa,Qwind,Qconv_ca,Qcond_wT,Qabs,
     &                                                  Qu,Tout,effhx,Vs)
     IMPLICIT NONE
C           PVTtrcSOLVE common variables
            COMMON/PVT/Uwind,Tamb,Flow,Tsup,Gam,EffT,EffRef
            COMMON/PVT/Tref,t_trap,ThetaT,h_trap,a,b,dist
            COMMON/PVT/Por,rho_amb,visc_amb,AreaW,k_amb
            COMMON/PVT/pitch,length,width,hp,th,D,ecol
            COMMON/PVT/e_back,e_wall,Uwall,Tblg,Trad
            COMMON/PVT/L,cp_amb,Area_cs
            COMMON/PVT/QBsurf,QDsurf,Qsurf,PPV
            COMMON/PVT/Psh,Area,AlphaB,AlphaDcoll
            COMMON/PVT/PVOn
            COMMON/PVT/SHCASE,nb_corr,PVMode
            COMMON/PVT/IUNIT,ITYPE

C           SETTING PVTtrcSOLVE Variables
            DOUBLE PRECISION Tcol,Tplen,Qrad_cs,Qrad_wc,Twall
            DOUBLE PRECISION Qconv_wa,Qwind,Qconv_ca,Qcond_wT,Qabs
            DOUBLE PRECISION Tout,Vs,Qu,effhx
            INTEGER*4 k

c           COMMON VARIABLES DECLARATION
            DOUBLE PRECISION Uwind,Tamb,Flow,Tsup,Gam,EffT,EffRef
            DOUBLE PRECISION Tref,t_trap,ThetaT,h_trap,a,b,dist
            DOUBLE PRECISION Por,rho_amb,visc_amb,AreaW,k_amb
            DOUBLE PRECISION pitch,length,width,hp,th,D,ecol
            DOUBLE PRECISION e_back,e_wall,Uwall,Tblg,Trad
            DOUBLE PRECISION L,cp_amb,Area_cs
            DOUBLE PRECISION QBsurf,QDsurf,Qsurf,PPV
            DOUBLE PRECISION Psh,Area,AlphaB,AlphaDcoll
            INTEGER*4 PVon
            INTEGER*4 SHCASE,Nb_corr,PVMode
            INTEGER*4 IUNIT,ITYPE

            DIMENSION QBsurf(8),QDsurf(8),Qsurf(8),PPV(8)
            DIMENSION Psh(8),Area(8),AlphaB(8),AlphaDcoll(8)
            DIMENSION PVon(8)

c           INTERNAL VARIABLES
c           DOUBLE PRECISION Qcond_w,TAU,C1,C2,C3
            DOUBLE PRECISION SB,PR,Rew,Res,Reb,Reh,ef,eb,eh,dif
            DOUBLE PRECISION Ma,Mb,Mx,ResC
c           DOUBLE PRECISION SB,PR,Rew,Res,Reb,Reh,ef,eb,eh,dif
            DOUBLE PRECISION hconv_wa,Tsur,Twall_in,diflim,kmax,hwind
            DOUBLE PRECISION ReL,NuL,Rexc,ReVOP,ReUA,NuFlat,Nu,hFP
            double precision Qcond_wpotato,TAUpotato,C1potato,C2potato,C3p
c           INTEGER*4 PVon,SHCASE,Nb_corr,PVMode
            INTEGER*4 nb_case,nb_surf
c           INTEGER*4 IUNIT,ITYPE
            INTEGER SPASS
            INTEGER*4 neq,i,j,R,N

            pARAMETER (nb_surf=8,nb_case=5,Neq=8)
            DIMENSION L(nb_surf*2),Ma(10,11), Mb(10),Mx(10),ResC(10)
```

```
C               SETTING CONSTANTS
                SB=(5.67e-8)*3.6                !Stefan-Boltzmann constant [kJ/hrm2K4]
                Pr=0.71
                diflim=0.01
                kmax=100

                SPASS=1
100             CONTINUE
                IF (SPASS.GT.1)GO TO 160

C               SET Vs
                Vs=(Flow*Gam)/AreaW                     !m/hr
*******************************************************************************
C               CALCULATE hconv,wall-air
*******************************************************************************
C               IF NO FLOW, HCONV_WA=0.1 W/M2C Maurer [2004]
                IF (Vs.LE.0)THEN
                        hconv_wa=0.36                   ![kJ/m2 h C]
                        GO TO 110
                ENDIF

                ReL=(rho_amb*((AreaW*Vs)/Area_cs)*0.5*length)/(visc_amb)
                Rexc=500000

c               Verify if the flow is laminar or if there is transition
                IF (ReL.LE.Rexc)THEN
                        !Laminar
                        NuL=0.664*(ReL**0.5)*(Pr**0.33333333)
                ELSE IF (ReL.GT.Rexc)THEN
                        !Transition
                        NuL=((0.037*(ReL**0.8))-871)*(Pr**0.3333333)
                ENDIF

                hconv_wa=(k_amb*NuL)/length             ![kJ/m2 h C]
110             CONTINUE
*******************************************************************************
C               CALCULATE HEAT EXCHANGER EFFECTIVENESS (Van Decker and Hollands, 2001)
C               Assume: Asymptotic region (Boundary layer thickness is invariant)
C                                       Convection at the back comes only from the back of the hole
*******************************************************************************
                IF (Vs.GT.0)THEN
                        Rew=Uwind*pitch*rho_amb/visc_amb
                        Res=Vs*pitch*rho_amb/visc_amb
                        Reb=Vs*pitch*rho_amb/(visc_amb*Por)
                        Reh=Vs*D*rho_amb/(visc_amb*Por)

                        ef=1-(1/(1+((1/Res)*max(17.7,(0.708*(Rew**0.5))))))
                        eb=1-(1.0/(1+(3.4*(Reb**(-0.3333333333)))))
                        eh=1-exp((-0.0204*(pitch/D))-(20.62*th/(Reh*D)))

                        effhx=1-((1-ef)*(1-eb)*(1-eh))
                ELSE
                        effhx=0.0
                ENDIF
*******************************************************************************
C               CALCULATE  HWIND
*******************************************************************************
                IF(Vs.GT.(0.01))THEN
C               SWIFT SOFTWARE CORRELATION (TO USE)
                hwind=MIN((0.02*(Uwind/Vs)),(2.8+(3.0*
    &           (Uwind/3600))))                                             ![W/m2K]
C               STRL MODEL (FOR SMALL PANEL)
c               hwind=6+(4*Uwind/3600)-(76*Vs/3600)

                ELSE
                hwind=(2.8+(3.0*((Uwind)/3600)))        ![W/m2K]
c               hwind=6+(4*Uwind/3600)
                ENDIF
*******************************************************************************
C               SOLVE SIMULTANEOUS EQUATIONS FOR UNKNOWN TEMPERATURES AND HEAT FLOWS
C               [A][X]=[B]
*******************************************************************************
                Tsur=Trad
                Twall_in=Tblg

C               SET MATRIX Ma
C               SET A VALUE OF Tcol AND Twall TO START
                Tcol=Tamb+20
                Twall=Tamb+10

                DO i=1,10
                        Mx(i)=0.0
                        DO j=1,11
                                Ma(i,j)=0.0
                        ENDDO
                ENDDO

C               SET EQUATION 1 :effhx
                Ma(1,1)=-1.0*effhx
                Ma(1,2)=1.0
```

```
            Ma(1,11)=Tamb*(1.0-effhx)

C           SET EQUATION 2:Qrad,col-sur
            Ma(2,3)=-1.0

            Ma(2,1)=SB*(Tcol+Tsur)*((Tcol**2)+
     &                  (Tsur**2))*(1-Por)*(ecol*AreaW)

            Ma(2,11)=SB*(Tcol+Tsur)*((Tcol**2)+(Tsur**2))*
     &                  (1-Por)*Tsur*(ecol*AreaW)

C           SET EQUATION 3: Qrad,wall-col
            Ma(3,1)=(SB*AreaW*((Twall**2)+(Tcol**2))*
     &                          (Twall+Tcol))/
     &                          (((1-e_wall)/e_wall)+
     &                          ((1-e_back)/e_back)+1.0)
            Ma(3,4)=1.0
            Ma(3,9)=(-1.0*SB*AreaW*((Twall**2)+(Tcol**2))*
     &                          (Twall+Tcol))/
     &                          (((1-e_wall)/e_wall)+
     &                          ((1-e_back)/e_back)+1.0)

C           SET EQUATION 4:Qconv,wall-air
            Ma(4,5)=1.0
            Ma(4,9)=-1.0*hconv_wa*AreaW
            Ma(4,2)=hconv_wa*AreaW

C           SET EQUATION 5:         Qconv,col-amb
            Ma(5,6)=1.0
            Ma(5,1)=-1.0*hwind*3.6*AreaW
            Ma(5,11)=-1.0*hwind*3.6*AreaW*Tamb

C           SET EQUATION 6:         Qconv,col-air
            Ma(6,7)=1.0
            Ma(6,2)=-1.0*rho_amb*Vs*cp_amb*AreaW
            Ma(6,11)=-1.0*rho_amb*Vs*cp_amb*AreaW*Tamb

C           SET EQUATION 7 :Qabs+Qrad_wc-Qconv_ca-Qwind-Qrad_cs=0
            Ma(7,8)=1.0
            Ma(7,4)=1.0
            Ma(7,7)=-1.0
            Ma(7,6)=-1.0
            Ma(7,3)=-1.0

C           SET EQUATION 8: Qabs=(1-EffPV)*Qsurf
            MA(8,8)=1.0
            IF (PvMode.EQ.1)THEN
                        Ma(8,11)=(Qsurf(5)+Qsurf(6)+Qsurf(7)+Qsurf(8)+
     &                          (Qsurf(3)*Nb_corr)+
     &                          ((Qsurf(1)+Qsurf(2)+Qsurf(4))*(Nb_corr-1)))

            ELSE IF (PvMode.EQ.2)THEN
                        Ma(8,1)=Qsurf(3)*EffT*Nb_corr*PPV(3)
                        Ma(8,11)=Qsurf(5)+Qsurf(6)+Qsurf(7)+Qsurf(8)+
     &                          ((Qsurf(1)+Qsurf(2)+Qsurf(4))*(Nb_corr-1))+
     &                          (Qsurf(3)*Nb_corr*(1-(PPV(3)*EffRef)+
     &                          (PPV(3)*EffT*(Tref+273.15))))

            ElSE IF (PvMode.EQ.3)THEN
                        Ma(8,1)=EffT*((Nb_corr*Qsurf(3)*PPV(3))+
     &                          ((Nb_corr-1)*(Qsurf(1)*PPV(1)+Qsurf(2)*
     &                          PPV(2)+Qsurf(4)*PPV(4))))
                        Ma(8,11)=Qsurf(5)+Qsurf(6)+Qsurf(7)+Qsurf(8)+
     &                          (Qsurf(1)*(Nb_corr-1)*(1-(PPV(1)*EffRef)+
     &                          (PPV(1)*(Tref+273.15)*EffT)))+
     &                          (Qsurf(2)*(Nb_corr-1)*(1-(PPV(2)*EffRef)+
     &                          (PPV(2)*(Tref+273.15)*EffT)))+
     &                          (Qsurf(4)*(Nb_corr-1)*(1-(PPV(4)*EffRef)+
     &                          (PPV(4)*(Tref+273.15)*EffT)))+
     &                          (Qsurf(3)*Nb_corr*(1-(PPV(3)*EffRef)+
     &                          (PPV(3)*(Tref+273.15)*EffT)))
            ENDIF

C           SET EQUATION 9:         Qconv,wall-air+Qrad,wall-col-Qcond_wall=0
            Ma(9,5)=1.0
            Ma(9,4)=1.0
            Ma(9,10)=-1.0

C           SET EQUATION 10:Qcond_wallT
            Ma(10,10)=1.0
            Ma(10,9)=Uwall*AreaW
            Ma(10,11)=Uwall*AreaW*Twall_in

*******************************************************************************
C           START ITERATION
*******************************************************************************
            k=0
130         CONTINUE
            k=k+1

            CALL SOLVEMATRIX(10,Ma,Mx)
            Tcol=Mx(1)
            Twall=Mx(9)
```

```
150     CONTINUE
        SPASS=SPASS+1
C       RECALCULATE CONFIGURATION FACTORS
        GO TO 100
160     CONTINUE

        DO i=1,10
                DO j=1,11
                        Ma(i,j)=0.0
                ENDDO
        ENDDO

C       SET EQUATION 1 :effhx
        Ma(1,1)=-1.0*effhx
        Ma(1,2)=1.0
        Ma(1,11)=Tamb*(1.0-effhx)

C       SET EQUATION 2:Qrad,col-sur
        Ma(2,3)=-1.0

        Ma(2,1)=SB*(Tcol+Tsur)*((Tcol**2)+
     &          (Tsur**2))*(1-Por)*(ecol*AreaW)

        Ma(2,11)=SB*(Tcol+Tsur)*((Tcol**2)+(Tsur**2))*
     &          (1-Por)*Tsur*(ecol*AreaW)

C       SET EQUATION 3: Qrad,wall-col
        Ma(3,1)=(SB*AreaW*((Twall**2)+(Tcol**2))*
     &                  (Twall+Tcol))/
     &                  (((1-e_wall)/e_wall)+
     &                  ((1-e_back)/e_back)+1.0)
        Ma(3,4)=1.0
        Ma(3,9)=(-1.0*SB*AreaW*((Twall**2)+(Tcol**2))*
     &                  (Twall+Tcol))/
     &                  (((1-e_wall)/e_wall)+
     &                  ((1-e_back)/e_back)+1.0)

C       SET EQUATION 4:Qconv,wall-air
        Ma(4,5)=1.0
        Ma(4,9)=-1.0*hconv_wa*AreaW
        Ma(4,2)=hconv_wa*AreaW

C       SET EQUATION 5:         Qconv,col-amb
        Ma(5,6)=1.0
        Ma(5,1)=-1.0*hwind*3.6*AreaW
        Ma(5,11)=-1.0*hwind*3.6*AreaW*Tamb

C       SET EQUATION 6:         Qconv,col-air
        Ma(6,7)=1.0
        Ma(6,2)=-1.0*rho_amb*Vs*cp_amb*AreaW
        Ma(6,11)=-1.0*rho_amb*Vs*cp_amb*AreaW*Tamb

C       SET EQUATION 7 :Qabs+Qrad_wc-Qconv_ca-Qwind-Qrad_cs=0
        Ma(7,8)=1.0
        Ma(7,4)=1.0
        Ma(7,7)=-1.0
        Ma(7,6)=-1.0
        Ma(7,3)=-1.0

C       SET EQUATION 8: Qabs=(1-EffPV)*Qsurf
        MA(8,8)=1.0
        IF (PvMode.EQ.1)THEN
                Ma(8,11)=(Qsurf(5)+Qsurf(6)+Qsurf(7)+Qsurf(8)+
     &                  (Qsurf(3)*Nb_corr)+
     &                  ((Qsurf(1)+Qsurf(2)+Qsurf(4))*(Nb_corr-1)))

        ELSE IF (PvMode.EQ.2)THEN
                Ma(8,1)=Qsurf(3)*EffT*Nb_corr*PPV(3)
                Ma(8,11)=Qsurf(5)+Qsurf(6)+Qsurf(7)+Qsurf(8)+
     &                  ((Qsurf(1)+Qsurf(2)+Qsurf(4))*(Nb_corr-1))+
     &                  (Qsurf(3)*Nb_corr*(1-(PPV(3)*EffRef)+
     &                  (PPV(3)*EffT*(Tref+273.15))))

        ElSE IF (PvMode.EQ.3)THEN
                Ma(8,1)=EffT*((Nb_corr*Qsurf(3)*PPV(3))+
     &                  ((Nb_corr-1)*(Qsurf(1)*PPV(1)+Qsurf(2)*
     &                  PPV(2)+Qsurf(4)*PPV(4))))
                Ma(8,11)=Qsurf(5)+Qsurf(6)+Qsurf(7)+Qsurf(8)+
     &                  (Qsurf(1)*(Nb_corr-1)*(1-(PPV(1)*EffRef)+
     &                  (PPV(1)*(Tref+273.15)*EffT)))+
     &                  (Qsurf(2)*(Nb_corr-1)*(1-(PPV(2)*EffRef)+
     &                  (PPV(2)*(Tref+273.15)*EffT)))+
     &                  (Qsurf(4)*(Nb_corr-1)*(1-(PPV(4)*EffRef)+
     &                  (PPV(4)*(Tref+273.15)*EffT)))+
     &                  (Qsurf(3)*Nb_corr*(1-(PPV(3)*EffRef)+
     &                  (PPV(3)*(Tref+273.15)*EffT)))

        ENDIF

C       SET EQUATION 9:         Qconv,wall-air+Qrad,wall-col-Qcond_wall=0
        Ma(9,5)=1.0
        Ma(9,4)=1.0
```

```
            Ma(9,10)=-1.0

C           SET EQUATION 10:Qcond_wallT
            Ma(10,10)=1.0
            Ma(10,9)=Uwall*AreaW
            Ma(10,11)=Uwall*AreaW*Twall_in
*******************************************************************************
C           CALCULATE RESIDUAL
*******************************************************************************
            DO i=1,10
                        Mb(i)=0.0
                        DO j=1,10
                                    Mb(i)=Mb(i)+((Ma(i,j))*(Mx(j)))
                        ENDDO
            ENDDO

            dif=0.0
            DO i=1,10
                        ResC(i)=0.0
                        ResC(i)=Mb(i)-Ma(i,11)
                        dif=dif+ABS(ResC(i))
            ENDDO

C           VERIFY IF CONVERGENCE IS OBTAINED
C           IF NOT, REPEAT NEW Tcol AND Twall
            IF (dif.gt.diflim.and.k.lt.kmax) go to 130

C           CONVERGENCE IS OBTAINED
            Tcol=Mx(1)
            Tplen=Mx(2)
            Qrad_cs=Mx(3)
            Qrad_wc=Mx(4)
            Qconv_wa=Mx(5)
            Qwind=Mx(6)
            Qconv_ca=Mx(7)
            Qabs=Mx(8)
            Twall=Mx(9)
            Qcond_wT=Mx(10)

C           CALCULATE Tout
            IF(Vs.GT.0.0)THEN
                        Tout=Tamb+((Qconv_wa+Qconv_ca)/(rho_amb*Vs*AreaW*cp_amb))
            ELSE
                        Tout=Tamb
            ENDIF
            Qu=rho_amb*Vs*cp_amb*AreaW*(Tout-Tamb)

400         CONTINUE

            Tcol=Tcol
            Tplen=Tplen
            Qrad_cs=Qrad_cs
            Qrad_wc=Qrad_wc
            Twall=Twall
            Qconv_wa=Qconv_wa
            Qwind=Qwind
            Qconv_ca=Qconv_ca
            Qcond_wT=Qcond_wT
            Qabs=Qabs
            Qu=Qu
            Tout=Tout
            effhx=effhx
            Vs=Vs

            RETURN
            END
*******************************************************************************
!           SUBROUTINE CALCULDIM
*******************************************************************************
            SUBROUTINE              CALCULDIM(PS,DIST,A,T_TRAP,B,L_END,L)
            IMPLICIT NONE

            DOUBLE PRECISION PS,L
            DOUBLE PRECISION DIST,A,T_TRAP,B,L_END
            PARAMETER NBSURF=8
            DIMENSION PS(NBSURF)
            DIMENSION L(NBSURF*2)

            L(1)=(dist-a)*(1.0-PS(1))
            L(2)=t_trap*(1.0-PS(2))
            L(3)=b
            L(4)=t_trap*(1.0-PS(4))
            L(5)=L_end*(1.0-PS(5))
            L(6)=t_trap*(1.0-PS(6))
            L(7)=L_end*(1.0-PS(7))
            L(8)=t_trap*(1.0-PS(8))
            L(9)=(dist-a)*PS(1)
            L(10)=(dist-a)*PS(1)
            L(11)=t_trap*PS(2)
            L(12)=t_trap*(PS(4))
            L(13)=L_end*PS(5)
            L(14)=t_trap*PS(6)
            L(15)=L_end*PS(7)
```

```
      L(16)=t_trap*PS(8)

      END
*******************************************************************************
!     SUBROUTINE CALCULSF1
*******************************************************************************
      SUBROUTINE CALCULSF1 (LENGTH1,LENGTH2,ALPHA,F12)
      IMPLICIT NONE

      DOUBLE PRECISION AF,PI,LENGTH1,LENGTH2,F12
      DOUBLE PRECISION ALPHA

      PI=3.141592654
      AF=LENGTH2/LENGTH1
      F12=(AF+1-(((AF*AF)+1-(2*AF*(COS(ALPHA))))**0.5))/2

      END
*******************************************************************************
!     SUBROUTINE CALCULSF2
*******************************************************************************
      SUBROUTINE CALCULSF2 (LENGTH1,LENGTH2,LENGTH3,ALPHA,F12)
      IMPLICIT NONE

      DOUBLE PRECISION LENGTH1,LENGTH2,F12,X,Y,PI
      DOUBLE PRECISION LENGTH3,ALPHA
      PI=3.141592654

      X=((LENGTH1**2)+((LENGTH3-LENGTH2)**2)-
     &  (2*LENGTH1*(LENGTH3-LENGTH2)*COS(ALPHA)))**0.5
      Y=((LENGTH1**2)+(LENGTH3**2)-
     &  (2*LENGTH1*LENGTH3*COS(ALPHA)))**0.5

      F12=(X+LENGTH2-Y)/(2*LENGTH1)

      END
*******************************************************************************
!     SUBROUTINE CALCULSF3
*******************************************************************************
      SUBROUTINE CALCULSF3(LENGTH1,LENGTH2,ALPHA,F12)
      IMPLICIT NONE

      DOUBLE PRECISION LENGTH1,LENGTH2,F12,X,Y,PI
      DOUBLE PRECISION ALPHA,ZETA
      PI=3.141592654

      X=((LENGTH1**2)+(LENGTH2**2)-
     &  (2*LENGTH1*LENGTH2*COS(ALPHA)))**0.5
      ZETA=ASIN((SIN(ALPHA)*LENGTH1)/X)
      Y=((LENGTH1**2)+(X**2)-
     &  (2*LENGTH1*X*COS(ALPHA-ZETA)))**0.5


      F12=((2.0*X)-Y-LENGTH2)/(2*LENGTH1)
      END
*******************************************************************************
!     SUBROUTINE CALCULSF4
*******************************************************************************
      SUBROUTINE CALCULSF4(LENGTH1,LENGTH2,LENGTH3,ALPHA,F12)
      IMPLICIT NONE

      DOUBLE PRECISION LENGTH1,LENGTH2,LENGTH3,F12,X,Y,Z,W,PI
      DOUBLE PRECISION ALPHA,ZETA
      PI=3.141592654

      X=((LENGTH2**2)+(LENGTH3**2)-
     &  (2*LENGTH2*LENGTH3*COS(ALPHA)))**0.5
      Z=(((LENGTH2-LENGTH1)**2)+(LENGTH3**2)-
     &  (2*(LENGTH2-LENGTH1)*(LENGTH3)*COS(ALPHA)))**0.5
      ZETA=ASIN((LENGTH2*SIN(ALPHA))/X)
      W=((LENGTH2**2)+(X**2)-
     &  (2*LENGTH2*X*COS(ALPHA-ZETA)))**0.5
      Y=((W**2)+(LENGTH1**2)-(2*W*LENGTH1*COS(PI-ALPHA)))**0.5


      F12=(X+Y-W-Z)/(2*LENGTH1)
      END
*******************************************************************************
!     SUBROUTINE CALCULSF5
*******************************************************************************
      SUBROUTINE CALCULSF5(LENGTH1,LENGTH2,LENGTH3,ALPHA,F12)
      IMPLICIT NONE

      DOUBLE PRECISION LENGTH1,LENGTH2,LENGTH3,F12,X,Y,PI
      DOUBLE PRECISION ALPHA
      PI=3.141592654

      X=((LENGTH2**2)+((LENGTH3-LENGTH1)**2)-
     &  (2*LENGTH2*(LENGTH3-LENGTH1)*COS(ALPHA)))**0.5
      Y=((LENGTH2**2)+(LENGTH3**2)-
     &  (2*LENGTH3*LENGTH2*COS(ALPHA)))**0.5

      F12=(X+LENGTH1-Y)/(2*LENGTH1)
      END
```

```
*******************************************************************************
!               SUBROUTINE CALCULSF6
*******************************************************************************
                SUBROUTINE CALCULSF6(LENGTH1,LENGTH2,LENGTH3,ALPHA,F12)
                IMPLICIT NONE

                DOUBLE PRECISION LENGTH1,LENGTH2,LENGTH3,F12,PI
                DOUBLE PRECISION ALPHA,X,Y,Z,ZETA
                PI=3.141592654

                X=((LENGTH3**2)+(LENGTH1**2)-(2*LENGTH3*LENGTH1*
     &          COS(ALPHA)))**0.5
                Y=((LENGTH3**2)+(LENGTH2**2)-(2*LENGTH3*LENGTH2*
     &          COS(ALPHA)))**0.5
                ZETA=ASIN((LENGTH1*sin(ALPHA))/X)
                Z=((X**2)+(LENGTH2**2)-(2*X*LENGTH2*
     &          COS(ALPHA-ZETA)))**0.5
                F12=(X+Y-Z-LENGTH3)/(2*LENGTH1)
                END
*******************************************************************************
!               SUBROUTINE CALCULSF7
*******************************************************************************
                SUBROUTINE CALCULSF7(LENGTH1,LENGTH2,LENGTH3,F12)
                IMPLICIT NONE

                DOUBLE PRECISION LENGTH1,LENGTH2,LENGTH3,F12
                DOUBLE PRECISION ALPHA,X,Y


                ALPHA=ATAN(LENGTH3/((LENGTH2-LENGTH1)/2.0))
                X=((LENGTH3**2)+(((LENGTH2-LENGTH1)/2.0)**2.0))**0.5


                Y=((LENGTH2**2)+(X**2)-(2*LENGTH2*X*
     &          COS(ALPHA)))**0.5

                F12=((2.0*Y)-(2.0*X))/(2*LENGTH1)
                END
*******************************************************************************
!               SUBROUTINE USED TO SOLVE MATRIX BY THE ELIMINATION METHOD                                       *
!               SUPPLEMENTED BY A SEARCH FOR THE LARGEST PIVOTAL ELEMENT AT EACH STAGE   *
!               DEVELOPED BY M.COLLINS
                                                        *
*******************************************************************************

                SUBROUTINE SOLVEMATRIX(N,A,XSOL)
                IMPLICIT NONE

                INTEGER N
                DOUBLE PRECISION A(N,(N+2))
                DOUBLE PRECISION XSOL(N)
                DOUBLE PRECISION CMAX,TEMP,C,Y,D
                DOUBLE PRECISION ABS
                INTEGER NM1,NP1,NP2,I,J,L,LP,NOS,NI,NJ

                NM1=N-1
                NP1=N+1
                NP2=N+2

                DO I=1,N
                        A(I,NP2)=0.0
      ! DO 1 J=1,NP1     ! TODO ?
                END DO

                DO I=1,N
                        DO J=1,NP1
                                A(I,NP2)=A(I,NP2)+A(I,J)
                        END DO
                END DO

                DO L=1,N-1
                        CMAX=A(L,L)
                        LP=L+1
                        NOS=L

                        DO I=LP,N
                                IF(ABS(CMAX).LT.ABS(A(I,L)))THEN
                                        CMAX=A(I,L)
                                        NOS=I
                                ENDIF
                        END DO

!       SWAP ROWS
                        IF (NOS.NE.L) THEN
                                DO J=1,NP2
                                        TEMP=A(L,J)
                                        A(L,J)=A(NOS,J)
                                        A(NOS,J)=TEMP
                                END DO
                        END IF

                        DO I=LP,N
                                C=0.0
```

```
                                        Y=-A(I,L)/A(L,L)
                                        DO J=L,NP2
                                                A(I,J)=A(I,J)+Y*A(L,J)
                                        END DO
                                        DO J=L,NP1
                                                C=C+A(I,J)
                                        END DO
                        END DO
                END DO

!  NOW BACKSUBSTITUTE
                XSOL(N)=A(N,NP1)/A(N,N)
                DO I=1,NM1
                        NI=N-I
                        D=0.0
                        DO J=1,I
                                NJ=N+1-J
                                D=D+A(NI,NJ)*XSOL(NJ)
                        END DO
                        XSOL(NI)=(A(NI,NP1)-D)/A(NI,NI)
                END DO

                END
```

```
      SUBROUTINE TYPE250(TIME,XIN,OUT,T,DTDT,PAR,INFO,ICNTRL,*)

C************************************************************************
C.    THIS COMPONENT SIMULATES THE THERMAL PERFORMANCE OF A
C.    FLAT-PLATE SOLAR COLLECTOR USING THE MODEL DEVELOPED BY
C.    HOTTEL, WHILLIER, AND BLISS.
C     LAST MODIFIED 3/93 - JWT
C     3/04 - TPM - FOR TRNSYS 16
C     12/05 - DAA - Added units checking with RCHECK
C     02/07 - MRC - Simplified to Type d only
C               - 'flowrate' input changed to 'specific flowrate'
C               - parameter/input defaults corrected and reorganized
C               - variable declarations corrected
C               - Temperature Coefficient is now a -ve value
C************************************************************************
C.
C.    PARAMETERS
C.    A         COLLECTOR AREA
C.    ALF       ABSORPTANCE OF THE COLLECTOR PLATE SURFACE
C.    BR        TEMPERATURE COEFFICIENT OF THE CELLS
C.    CELLPF    RATIO OF CELL AREA TO ABSORBER AREA
C.    CPF       THERMAL CAPACITANCE OF THE COLLECTOR FLUID
C.    EP        THERMAL EMITTANCE OF THE COLLECTOR PLATE SURFACE
C.    FP        COLLECTOR GEOMETRY EFFICIENCY FACTOR
C.    NG/XNG    NUMBER OF GLASS COVERS
C.    PVEFF     CELL EFFICIENCY
C.    TR        TEMPERATURE OF CELL REFERENCE EFFICIENCY
C.    UBE       CONTRIBUTION TO UL DUE TO BOTTOM AND EDGE
C.    XKL       PRODUCT OF THE EXTINCTION COEFFICIENT AND THE
C.              THICKNESS OF EACH GLASS COVER
C.
C.    INPUT
C.    ANGLE     TILT OF THE COLLECTOR WITH RESPECT TO HORIZONTAL
C.    FLWRT     COLLECTOR FLUID SPECIFIC FLOWRATE
C.    HBT       INSTANTANEOUS BEAM RADIATION ON THE COLLECTOR SURFACE
C.    HDT       INSTANTANEOUS DIFFUSE RADIATION ON THE COLLECTOR SURFACE
C.    TA        AMBIENT TEMPERATURE
C.    TIN       INLET FLUID TEMPERATURE
C.    THETA1    ANGLE OF INCIDENCE OF RADIATION ON THE COLLECTOR
C.    WIND      WINDSPEED
C.
C.    OUTPUT
C.    TAUALF    PRODUCT OF THE TRANSMITTANCE OF THE GLASS
C.              AND THE ABSORPTANCE OF THE COLLECTOR PLATE SURFACE
C.    TCELL     AVERAGE CELL TEMPERATURE
C.    TOUT      OUTLET FLUID TEMPERATURE
C.    QE        ELECTRICAL POWER OUTPUT
C.    QU        USEFUL ENERGY COLLECTION RATE PER UNIT AREA
C.    UL        OVERALL ENERGY LOSS COEFFICIENT. SEE KLEIN
C.    ULO       APPARENT THERMAL LOSS COEFFICIENT
C.
C.    OTHER
C.    TAU60     TRANSMITTANCE OF THE GLASS COVER SYSTEM AT AN
C.              INCIDENCE ANGLE OF 60 DEGREES, AS ASSUMED FOR DIFFUSE
C.              RADIATION SEE HOTTEL AND WOERTZ
C.    HR        TOTAL RADIATION INCIDENT ON THE TILTED COLLECTOR SURFACE
C.    REFIND    REFRACTIVE INDEX OF THE GLASS
C.    TM        MEAN FLUID TEMPERATURE
C-------------------------------------------------------------------------------------------------------------
! Copyright © 2004 Solar Energy Laboratory, University of Wisconsin-Madison. All rights reserved.
C-------------------------------------------------------------------------------------------------------------
C     USE STATEMENTS
      USE TrnsysFunctions
C-------------------------------------------------------------------------------------------------------------
C   REQUIRED BY THE MULTI-DLL VERSION OF TRNSYS
    !DEC$ATTRIBUTES DLLEXPORT :: TYPE250                                    !SET THE CORRECT TYPE NUMBER HERE

C     TRNSYS DECLARATIONS
      IMPLICIT NONE
      DOUBLE PRECISION XIN,OUT,TIME,PAR,STORED,T,DTDT
      INTEGER*4 INFO(15),NP,NI,NOUT,ND
      INTEGER*4 NPAR,NIN,NDER,IUNIT,ITYPE
      INTEGER*4 ICNTRL, NSTORED
      CHARACTER*3 OCHECK,YCHECK
C-------------------------------------------------------------------------------------------------------------
C     USER DECLARATIONS
      PARAMETER (NP=12,NI=8,NOUT=8,ND=0,NSTORED=0)
C-------------------------------------------------------------------------------------------------------------
C     REQUIRED TRNSYS DIMENSIONS
      DIMENSION XIN(NI),OUT(NOUT),PAR(NP),YCHECK(NI),OCHECK(NOUT),
     &          STORED(NSTORED),T(ND),DTDT(ND)

C-------------------------------------------------------------------------------------------------------------
C     DECLARATIONS AND DEFINITIONS FOR THE USER-VARIABLES
      DOUBLE PRECISION IC,K1,K2,EG,PI,SB,ATR,BTR,CTR,TAU040,REFIND,AR,
     & UBE,A,FP,CPF,ALF,UL,TAU,BR,TR,CELLPF,TAUALF,XNG,EP,ANGLE,XKL,
     & TAU60,FE,UB,CB,UF,UT,TIN,FLWRT,TA,TC,BA,HR,ETAR,ETAA,WIND,HBT,
     & HDT,THETA1,ICT,TILT,V,COSTH1,THETA2,COSTH2,HWIND,TM,TMC,TAC,F,C,
     & STF1,STF2,S,SINC,DENOM,TP,TCELLR,TCELL,TEMP,PR,BETA,PRS,ULAB,ULO,
     & FR,HRT,QU,QE,TOUT,TPR,PRSAVE,BETAS,TIME0,TFINAL,DELT,PVEFF
      INTEGER MODE,NPP,ITER,LUIN,NG
      DIMENSION ATR(3),BTR(3),CTR(3),TAU040(3)
      DATA EG/0.88/,PI/3.1415927/,SB/5.678E-08/
```

```
      DATA ATR/-2.9868,-1.4214,-0.74816/
      DATA BTR/-3.7360,-5.7356,-6.5262/
      DATA CTR/4.3541,5.7723,6.3769/
      DATA TAU040/0.92,0.845,0.785/
      DATA IUNIT/0/,REFIND/1.526/,NPP/0/,AR/1./,UBE/0./
C-----------------------------------------------------------------------------------------------------------
C     TRNSYS FUNCTIONS
      TIME0=getSimulationStartTime()
      TFINAL=getSimulationStopTime()
      DELT=getSimulationTimeStep()
C-----------------------------------------------------------------------------------------------------------
C     SET THE VERSION INFORMATION FOR TRNSYS
      IF(INFO(7).EQ.-2) THEN
            INFO(12)=16
            RETURN 1
      ENDIF
C-----------------------------------------------------------------------------------------------------------
C     DO THE VERY LAST CALL OF THE SIMULATION MANIPULATIONS HERE
      IF (INFO(8).EQ.-1) THEN
            RETURN 1
      ENDIF
C-----------------------------------------------------------------------------------------------------------
C     PERFORM ANY "AFTER-ITERATION" MANIPULATIONS THAT ARE REQUIRED
      IF(INFO(13).GT.0) THEN
            RETURN 1
      ENDIF
C-----------------------------------------------------------------------------------------------------------
C     DO THE VERY FIRST CALL OF THE SIMULATION MANIPULATIONS HERE
      IF (INFO(7).EQ.-1) THEN
C     RETRIEVE THE UNIT AND TYPE NUMBER FOR THIS COMPONENT FROM THE INFO ARRAY
            IUNIT=INFO(1)
            ITYPE=INFO(2)
C     SET SOME INFO ARRAY VARIABLES TO TELL THE TRNSYS ENGINE HOW THIS TYPE IS TO WORK
            INFO(6)=8
            INFO(9)=1
            INFO(10)=0
C     SET THE # OF INPUTS, PARAM AND DERIVATIVES
            MODE=PAR(1)
            MODE=4
            NPAR=INFO(4)
            CALL TYPECK (1,INFO,8,NPAR,0)
            DATA YCHECK/'TE1','MF1','DG1','TE1','IR1','IR1','DG1','VE1'/
            DATA OCHECK/'PW1','PW1','TE1','MF1','DM1','TE1','HT1','HT1'/
            CALL RCHECK(INFO,YCHECK,OCHECK)
            RETURN 1
      ENDIF
C-----------------------------------------------------------------------------------------------------------


C     DO THE INITIAL TIMESTEP MANIPULATIONS HERE
      IF (TIME.LT.(TIME0+DELT/2.D0)) THEN
            IUNIT=INFO(1)
            ITYPE=INFO(2)
            ITER=0
            MODE=4
            A=PAR(1)
            FP=PAR(2)
            CPF=PAR(3)
            XNG=PAR(4)
            XKL=PAR(5)
            UBE=PAR(6)
            ALF=PAR(7)
            EP=PAR(8)
            PVEFF=PAR(9)
            BR=PAR(10)
            TR=PAR(11)
            CELLPF=PAR(12)
            UL=0.
            NG=XNG
            TAU60=DEXP(-1.21453*XNG*XKL)*(1.-DEXP((ATR(NG)+(BTR(NG)+
     &      CTR(NG)*0.5)*0.5)*0.5))
            CALL SOLARCELL (TCELLR,TEMP,SINC,PR,BETA,IC,V,ITER,NPP,-1,A,
     &      AR,MODE,LUIN)
            RETURN 1
      ENDIF
C-----------------------------------------------------------------------------------------------------------
C     RE-READ THE PARAMETERS IF ANOTHER UNIT OF THIS TYPE HAS BEEN CALLED
      IF(INFO(1).NE.IUNIT) THEN
            IUNIT=INFO(1)
            ITYPE=INFO(2)
            ITER=0
            MODE=4
            A=PAR(1)
            FP=PAR(2)
            CPF=PAR(3)
            XNG=PAR(4)
            XKL=PAR(5)
            UBE=PAR(6)
            ALF=PAR(7)
            EP=PAR(8)
            PVEFF=PAR(9)
            BR=PAR(10)
            TR=PAR(11)
```

```
                        CELLPF=PAR(12)
                        UL=0.
                        NG=XNG
                        TAU60=DEXP(-1.21453*XNG*XKL)*(1.-DEXP((ATR(NG)+(BTR(NG)+
     &          CTR(NG)*0.5)*0.5)*0.5))
                ENDIF

C-------------------------------------------------------------------------------------------------------------
C               RETRIEVE THE CURRENT VALUES OF THE INPUTS
                TIN=XIN(1)
                FLWRT=XIN(2)
                ANGLE=XIN(3)
                TA=XIN(4)
                IF(BR.EQ.0.0)THEN
                        BR=0.0000001
                ENDIF
                TC=TR-1./BR
                IF(TC.EQ.TA) TC=TA+0.1
                BA=1./(TC-TA)
                HBT=XIN(5)
                HDT=XIN(6)
                THETA1=XIN(7)
                WIND=XIN(8)
                ETAR=PVEFF*CELLPF
                ETAA=ETAR*(1.+BR*(TA-TR))
                HR=HBT+HDT
                TAU=0.0001
                IF (THETA1.GT.85.) GO TO 23
                IF (HR.LE.1.0E-10) GO TO 23
                THETA1=THETA1*2.*PI/360.0
                COSTH1=DCOS(THETA1)
                THETA2=DASIN(DSIN(THETA1)/REFIND)
                COSTH2=DCOS(THETA2)
                TAU=TAU040(NG)
                IF (COSTH1.LT.0.766) THEN
                        TAU=1.0-DEXP((ATR(NG)+(BTR(NG)+CTR(NG)*COSTH1)*COSTH1)*COSTH1)
                ENDIF
                TAU=HBT/HR*TAU*DEXP(-XNG*XKL/COSTH2)
                TAU=TAU+HDT/HR*TAU60
                IF(XNG.EQ.0)THEN
                        TAU=1.0
                ENDIF
23              TAUALF=ALF*TAU
                UL=UL-TAU*HR*ETAA*BA
                ICT=0
                HWIND=5.7+3.8*WIND
                TM=TIN
25              IF (ITER.EQ.0) ICT=ICT+1
                IF (ICT.GT.2) GO TO 45
                TMC=TM+273.15
                TAC=TA+273.15
                IF (TMC.LE.TAC) TMC=TAC+1.0
                F=(1.0-0.04*HWIND+5.0E-04*HWIND*HWIND)*(1.0+0.091*XNG)
                IF (XNG.LT..5) F=1.
                C=365.9*(1.0-0.00883*ANGLE+0.0001298*ANGLE*ANGLE)
                STF1=C/TMC*((TMC-TAC)/(XNG+F))**0.33
                STF1=XNG/STF1+1.0/HWIND
                STF1=1.0/STF1
                STF2=1.0/(EP+0.05*XNG*(1.0-EP))+(2.*XNG+F-1.)/EG-XNG
                STF2=SB*(TMC*TMC+TAC*TAC)*(TMC+TAC)/STF2
                UL=(STF1+STF2)*3.6+UBE
                UL=UL-TAU*HR*ETAA*BA
                IF (FLWRT*A-1.E-5) 34,34,30
30              IF ((FP*UL/(FLWRT*CPF)).LT.0.01) GO TO 31
                FR=FLWRT*CPF*(1.0-DEXP(-FP*UL/(FLWRT*CPF)))/UL
                GO TO 32
31              FR=FP
32              HRT=HR*(1.-ETAA/ALF)
                QU=FR*(HRT*TAUALF-UL*(TIN-TA))
                QE=TAU*HR*ETAA*(1.-BA*(FR*(TIN-TA)+(TAUALF*HRT/UL)*(1.-FR)))
                TOUT=QU/FLWRT/CPF+TIN
                GO TO 36
34              QU=0.0
                ULO=UL+TAU*HR*ETAA*BA
                QE=TAU*HR*ETAA*(ULO-TAUALF*HR*BA)/UL
                FLWRT=0.0
                TOUT=(TAUALF*HR-QE)/ULO+TA
36              TM=(TIN+TOUT)/2.0
                TCELL=TM
                IF(HR.LE.1.E-5)THEN
                        TCELL=TA
                        GO TO 25
                        IF(ETAR.LT.1.E-5)THEN
                                TCELL=TC
                                GO TO 25
                        ENDIF
                        IF(TAU.LE.1.E-5)THEN
                                TCELL=TA
                                GO TO 25
                        ENDIF
                ENDIF
                TCELL=TC-(QE/(TAU*HR*ETAR))*(TC-TR)
                GO TO 25
```

```
45          CONTINUE
            OUT(1)=QU*A
            OUT(2)=QE*A
            OUT(3)=TOUT
            OUT(4)=FLWRT*A
            OUT(5)=TAUALF
            OUT(6)=TCELL
            OUT(7)=UL
            OUT(8)=ULO
46          CONTINUE
            RETURN 1
            END

C-------------------------------------------------------------------------------------------------------------
```

```
      SUBROUTINE TYPE251(TIME,XIN,OUT,T,DTDT,PAR,INFO,ICNTRL,*)

C*************************************************************************
C.              THIS COMPONENT SIMULATES THE THERMAL PERFORMANCE OF A
C.              FLAT-PLATE SOLAR COLLECTOR USING THE MODEL DEVELOPED BY
C.              HOTTEL, WHILLIER, AND BLISS.
C               LAST MODIFIED 3/93 - JWT
C               3/04 - TPM - FOR TRNSYS 16
C               12/05 - DAA - Added units checking with RCHECK
C               02/07 - MRC - Simplified to Type h only
C*************************************************************************
C.
C.              PARAMETERS
C.              A               COLLECTOR AREA
C.              ALF             ABSORPTANCE OF THE COLLECTOR PLATE SURFACE
C.              AR              APERTURE AREA TO ABSORBER AREA
C.              CB              THERMAL CONDUCTANCE BETWEEN CELLS AND ABSORBER
C.              CPF             THERMAL CAPACITANCE OF THE COLLECTOR FLUID
C.              EP              THERMAL EMITTANCE OF THE COLLECTOR PLATE SURFACE
C.              FE              APPROPRIATE FIN EFFICIENCY
C.              NG/XNG          NUMBER OF GLASS COVERS
C               TAU             COVER TRANSMITTANCE
C.              UB              BACK LOSS COEFFICIENT
C.              UF              FILM COEFFICIENT BETWEEN FLUID AND ABSORBER
C
C.              INPUT
C.              FLWRT           COLLECTOR FLUID FLOWRATE
C.              HR              TOTAL RADIATION INCIDENT ON THE TILTED COLLECTOR SURFACE
C.              TA              AMBIENT TEMPERATURE
C.              TILT            COLLECTOR SLOPE
C.              TIN             INLET FLUID TEMPERATURE
C.              V               VOLTAGE APPLIED TO ARRAY
C.              WIND            WINDSPEED
C.
C.              OUTPUT
C.              CURREN          ARRAY OUTPUT CURRENT
C.              QE              ELECTRICAL POWER OUTPUT
C.              QU              USEFUL ENERGY COLLECTION RATE PER UNIT AREA
C.              TAUALF          PRODUCT OF THE TRANSMITTANCE OF THE GLASS
C.                              AND THE ABSORPTANCE OF THE COLLECTOR PLATE SURFACE
C.              TOUT            OUTLET FLUID TEMPERATURE
C.              TCELL           AVERAGE CELL TEMPERATURE
C.              TCELLR          CELL TEMPERATURE AT INLET
C.              UL              OVERALL ENERGY LOSS COEFFICIENT. SEE KLEIN
C.              ULO             APPARENT THERMAL LOSS COEFFICIENT
C.
C-------------------------------------------------------------------------------------------------------------
! Copyright © 2004 Solar Energy Laboratory, University of Wisconsin-Madison. All rights reserved.
C-------------------------------------------------------------------------------------------------------------
C               USE STATEMENTS
                USE TrnsysFunctions
C-------------------------------------------------------------------------------------------------------------
C               REQUIRED BY THE MULTI-DLL VERSION OF TRNSYS
    !DEC$ATTRIBUTES DLLEXPORT :: TYPE251                                    !SET THE CORRECT TYPE NUMBER HERE

C               TRNSYS DECLARATIONS
                IMPLICIT NONE
                DOUBLE PRECISION XIN,OUT,TIME,PAR,STORED,T,DTDT
                INTEGER*4 INFO(15),NP,NI,NOUT,ND
                INTEGER*4 NPAR,NIN,NDER,IUNIT,ITYPE
                INTEGER*4 ICNTRL,NSTORED
                CHARACTER*3 OCHECK,YCHECK
C-------------------------------------------------------------------------------------------------------------
C               USER DECLARATIONS
                PARAMETER (NP=13,NI=7,NOUT=11,ND=0,NSTORED=0)
C-------------------------------------------------------------------------------------------------------------
C               REQUIRED TRNSYS DIMENSIONS
                DIMENSION XIN(NI),OUT(NOUT),PAR(NP),YCHECK(NI),OCHECK(NOUT),
     &          STORED(NSTORED),T(ND),DTDT(ND)
C-------------------------------------------------------------------------------------------------------------
C               DECLARATIONS AND DEFINITIONS FOR THE USER-VARIABLES
                DOUBLE PRECISION IC,K1,K2,EG,PI,SB,ATR,BTR,CTR,TAU040,REFIND,AR,
     & UBE,A,FP,CPF,ALF,UL,TAU,BR,TR,CELLPF,TAUALF,XNG,EP,XKL,
     & TAU60,FE,UB,CB,UF,UT,TIN,FLWRT,TA,TC,BA,HR,ETAR,ETAA,WIND,HBT,
     & HDT,THETA1,ICT,TILT,V,COSTH1,THETA2,COSTH2,HWIND,TM,TMC,TAC,F,C,
     & STF1,STF2,S,SINC,DENOM,TP,TCELLR,TCELL,TEMP,PR,BETA,PRS,ULAB,ULO,
     & FR,HRT,QU,QE,TOUT,TPR,PRSAVE,BETAS,TIME0,TFINAL,DELT
                INTEGER MODE,NPP,ITER,LUIN,NG
                DIMENSION ATR(3),BTR(3),CTR(3),TAU040(3)
                DATA EG/0.88/,PI/3.1415927/,SB/5.678E-08/
                DATA ATR/-2.9868,-1.4214,-0.74816/
                DATA BTR/-3.7360,-5.7356,-6.5262/
                DATA CTR/4.3541,5.7723,6.3769/
                DATA TAU040/0.92,0.845,0.785/
                DATA IUNIT/0/,REFIND/1.526/,NPP/0/,AR/1./,UBE/0./
C-------------------------------------------------------------------------------------------------------------
C               TRNSYS FUNCTIONS
                TIME0=getSimulationStartTime()
                TFINAL=getSimulationStopTime()
                DELT=getSimulationTimeStep()
C-------------------------------------------------------------------------------------------------------------
C               SET THE VERSION INFORMATION FOR TRNSYS
                IF(INFO(7).EQ.-2) THEN
```

```
                        INFO(12)=16
                        RETURN 1
                ENDIF
C---------------------------------------------------------------------------------------------------------------
C               DO ALL THE VERY LAST CALL OF THE SIMULATION MANIPULATIONS HERE
                IF (INFO(8).EQ.-1) THEN
                        RETURN 1
                ENDIF
C---------------------------------------------------------------------------------------------------------------
C               PERFORM ANY "AFTER-ITERATION" MANIPULATIONS THAT ARE REQUIRED
                IF(INFO(13).GT.0) THEN
                        RETURN 1
                ENDIF
C---------------------------------------------------------------------------------------------------------------
C               DO ALL THE VERY FIRST CALL OF THE SIMULATION MANIPULATIONS HERE
                IF (INFO(7).EQ.-1) THEN
C     RETRIEVE THE UNIT NUMBER AND TYPE NUMBER FOR THIS COMPONENT FROM THE INFO ARRAY
                        IUNIT=INFO(1)
                        ITYPE=INFO(2)
C               SET SOME INFO ARRAY VARIABLES TO TELL THE TRNSYS ENGINE HOW THIS TYPE IS TO WORK
                        INFO(6)=11
                        INFO(9)=1
                        INFO(10)=0
C     SET THE REQUIRED # OF INPUTS, PARAMETERS AND DERIVATIVES
                        MODE=PAR(1)
                        NPAR=INFO(4)
                        CALL TYPECK (1,INFO,7,NPAR,0)
                        CALL RCHECK(INFO,YCHECK,OCHECK)
                RETURN 1
                ENDIF
C---------------------------------------------------------------------------------------------------------------
C   DO THE INITIAL TIMESTEP MANIPULATIONS HERE
                IF (TIME.LT.(TIME0+DELT/2.D0)) THEN
                        IUNIT=INFO(1)
                        ITYPE=INFO(2)
                        ITER=0
                        MODE=PAR(1)
                        A=PAR(2)
                        AR=PAR(3)
                        CPF=PAR(4)
                        ALF=PAR(5)
                        NPP=1
                        FE=PAR(6)
                        UB=PAR(7)
                        CB=PAR(8)
                        UF=PAR(9)
                        UF=UF*(1.+FE)
                        IF (CB.LT.1.E-5) CALL TYPECK (4,INFO,0,0,0)
                        TAU=PAR(10)
                        TAUALF=TAU*ALF
                        XNG=PAR(11)
                        EP=PAR(12)
                        LUIN=-1
                        IF(INFO(4).EQ.13) LUIN=PAR(13)
                        CALL SOLARCELL (TCELLR,TEMP,SINC,PR,BETA,IC,V,ITER,NPP,-1,A,
   & AR,MODE,LUIN)
                        RETURN 1
                ENDIF
C---------------------------------------------------------------------------------------------------------------
C   RE-READ THE PARAMETERS IF ANOTHER UNIT OF THIS TYPE HAS BEEN CALLED
                IF(INFO(1).NE.IUNIT) THEN
                        IUNIT=INFO(1)
                        ITYPE=INFO(2)
                        ITER=0
                        MODE=PAR(1)
                        A=PAR(2)
                        AR=PAR(3)
                        CPF=PAR(4)
                        ALF=PAR(5)
                        NPP=1
                        FE=PAR(6)
                        UB=PAR(7)
                        CB=PAR(8)
                        UF=PAR(9)
                        UF=UF*(1.+FE)
                        IF (CB.LT.1.E-5) CALL TYPECK (4,INFO,0,0,0)
                        TAU=PAR(10)
                        TAUALF=TAU*ALF
                        XNG=PAR(11)
                        EP=PAR(12)
                        LUIN=-1
                        IF(INFO(4).EQ.13) LUIN=PAR(13)
                ENDIF
C---------------------------------------------------------------------------------------------------------------
C   RETRIEVE THE CURRENT VALUES OF THE INPUTS
                TIN=XIN(1)
                TM=TIN
                FLWRT=XIN(2)
                TILT=XIN(3)
                V=XIN(4)
                TA=XIN(5)
                HR=XIN(6)
                WIND=XIN(7)
```

```
                HWIND=5.7+3.8*WIND
                ICT=0
25              CONTINUE
                IF (ITER.EQ.0) ICT=ICT+1
                IF (ICT.GT.2) GO TO 45
                TMC=TM+273.15
                TAC=TA+273.15
                IF (TMC.LE.TAC) TMC=TAC+1.0
                F=(1.0-0.04*HWIND+5.0E-04*HWIND*HWIND)*(1.0+0.091*XNG)
                IF (XNG.LT..5) F=1.
                STF1=365.9/TMC*((TMC-TAC)/(XNG+F))**0.33
                STF1=XNG/STF1+1.0/HWIND
                STF1=1.0/STF1
                STF2=1.0/(EP+0.05*XNG*(1.0-EP))+(2.*XNG+F-1.)/EG-XNG
                STF2=SB*(TMC*TMC+TAC*TAC)*(TMC+TAC)/STF2
                UL=(STF1+STF2)*3.6+UBE
                S=HR*TAUALF*AR
                SINC=S/ALF
                UT=UL
                DENOM=UT*(CB+UF)+CB*UF
                K1=CB/DENOM
                IF (ICT.NE.1) GO TO 27
                TP=K1*(S-UT*(TIN-TA))+TIN
                TCELLR=UF*(TP-TIN)/CB+TP
                IF (FLWRT.LT.1.E-5) THEN
                        TCELLR=(S+TA*(UT+CB*UB/(CB+UB)))/(UT+CB-CB**2/(CB+UB))
                ENDIF
                TCELL=TCELLR
27              TEMP=TCELL
                CALL SOLARCELL (TCELLR,TEMP,SINC,PR,BETA,IC,V,ITER,NPP,INFO(7),A,
     & AR,MODE,LUIN,*28)
                CALL LINKCK('TYPE50','SOLARCEL',1,99)
28              K2=1.+K1*PR*BETA*(CB+UF)/CB
                IF (S.LT.1.E-5) PRS=0.
                IF (S.GE.1.E-5) PRS=PR/S
                FP=UF*K1*(1.-PRS*(1.+BETA*(TA-TCELLR)))/K2
                ULAB=UF*K1*(UT+PR*BETA)/(K2*FP)
                UL=ULAB/AR
                ULO=UT/AR
                IF (FLWRT-1.E-5) 34,34,30
30              IF ((FP*UL*A/(FLWRT*CPF)).LT.0.01) GO TO 31
                FR=FLWRT*CPF*(1.0-DEXP(-FP*UL*A/(FLWRT*CPF)))/(A*UL)
                GO TO 32
31              FR=FP
32              HRT=HR
                QU=FR*(HRT*TAUALF-UL*(TIN-TA))
                QE=PR*(1.+BETA*(TCELL-TCELLR))/AR
                TOUT=QU/FLWRT*A/CPF+TIN
                GO TO 36
34              QU=0.0
                QE=PR/AR
                TCELLR=(S-PR+TA*(UT+CB*UB/(CB+UB)))/(UT+CB-CB**2/(CB+UB))
                TCELL=TCELLR
                TEMP=TCELL
                TOUT=TCELL*(1.-UB/(UB+CB))+TA*UB/(UB+CB)
                FLWRT=0.
                GO TO 39
36              TM=(TIN+TOUT)/2.0
                TP=QU*AR/UF+TIN
                TCELLR=UF*(TP-TIN)/CB+TP
                TPR=QU*AR/UF+TM
                TCELL=UF*(TPR-TM)/CB+TPR
                TM=TCELL
                TEMP=TCELL
39              PRSAVE=PR
                BETAS=BETA
                ITER=1
                CALL SOLARCELL (TCELLR,TEMP,SINC,PR,BETA,IC,V,ITER,NPP,INFO(7),A,
     & AR,MODE,LUIN,*391)
                CALL LINKCK('TYPE50','SOLARCEL',1,99)
391             IF (PR.LT.1.E-5.AND.PRSAVE.LT.1.E-5) GO TO 40
                IF (PR.LT.1.E-5) GO TO 25
                IF (DABS((PR-PRSAVE)/PR).GT.0.05) GO TO 25
40              IF (BETA.LT.1.E-5.AND.BETAS.LT.1.E-5) GO TO 41
                IF (BETA.LT.1.E-5) GO TO 25
                IF (DABS((BETA-BETAS)/BETA).GT.0.05) GO TO 25
41              ITER=0
                GO TO 25
45              OUT(1)=QU*A
                OUT(2)=QE*A
                OUT(3)=TOUT
                OUT(4)=FLWRT
                OUT(5)=ULO
                OUT(6)=UL
                OUT(7)=TAUALF
                OUT(8)=TCELL
                OUT(9)=TCELLR
                OUT(10)=V
                OUT(11)=QE*A/(V*3.6)
                RETURN 1
                END
C-------------------------------------------------------------------------------------------------------------------------
```