

Module Interface Specification for Solar Water Heating Systems Incorporating Phase Change Material

Brooks MacLachlan

August 5, 2016

Contents

1	Introduction	4
2	Notation	4
3	Module Decomposition	4
4	MIS of Control Module	5
4.1	Module	5
4.2	Uses	5
4.3	Syntax	5
4.3.1	Exported Access Programs	5
4.4	Semantics	5
4.4.1	State Variables	5
4.4.2	Environment Variables	6
4.4.3	Access Routine Semantics	6
5	MIS of Input Parameters Module	6
5.1	Module	6
5.2	Uses	6
5.3	Syntax	6
5.3.1	Exported Data Types	6
5.3.2	Exported Access Programs	6
5.4	Semantics	6
5.4.1	State Variables	6
5.4.2	Access Routine Semantics	7

6	MIS of Input Format Module	7
6.1	Module	7
6.2	Uses	7
6.3	Syntax	8
6.4	Exported Access Programs	8
6.5	Semantics	8
6.5.1	State Variables	8
6.5.2	Assumptions	8
6.5.3	Access Routine Semantics	8
7	MIS of Input Verification Module	9
7.1	Module	9
7.2	Uses	9
7.3	Syntax	9
7.3.1	Exported Access Programs	9
7.4	Semantics	9
7.4.1	Environment Variables	9
7.4.2	Assumptions	9
7.4.3	Access Routine Semantics	10
8	MIS of Temperature ODEs Module	10
8.1	Module	10
8.2	Uses	10
8.3	Syntax	11
8.3.1	Exported Access Programs	11
8.4	Semantics	11
8.4.1	State Variables	11
8.4.2	Assumptions	11
8.4.3	Access Routine Semantics	12
9	MIS of ODE Solver Module	12
9.1	Module	12
9.2	Uses	12
9.3	Syntax	13
9.3.1	Exported Access Programs	13
9.4	Semantics	13
9.4.1	State Variables	13
9.4.2	Local Constants	13
9.4.3	Access Routine Semantics	13
10	MIS of Energy Module	13
10.1	Module	13
10.2	Uses	13

10.3	Syntax	14
10.3.1	External Access Programs	14
10.4	Semantics	14
10.4.1	State Variables	14
10.4.2	Assumptions	14
10.4.3	Access Routine Semantics	15
10.4.4	Local Functions	15
11	MIS of Output Verification Module	16
11.1	Module	16
11.2	Uses	16
11.3	Syntax	16
11.3.1	Exported Access Programs	16
11.4	Semantics	16
11.4.1	State Variables	16
11.4.2	Environment Variables	16
11.4.3	Local Variables	16
11.4.4	Assumptions	16
11.4.5	Access Routine Semantics	17
11.4.6	Local Functions	17
12	MIS of Plotting Module	18
12.1	Module	18
12.2	Uses	18
12.3	Syntax	18
12.3.1	Exported Access Programs	18
12.4	Semantics	18
12.4.1	State Variables	18
12.4.2	Environment Variables	18
12.4.3	Assumptions	18
12.4.4	Access Routine Semantics	18
13	MIS of Output Module	19
13.1	Module	19
13.2	Uses	19
13.3	Syntax	19
13.3.1	Exported Constants	19
13.3.2	Exported Access Program	19
13.4	Semantics	19
13.4.1	State Variables	19
13.4.2	Environment Variables	19
13.4.3	Access Routine Semantics	19

1 Introduction

The following document details the Module Interface Specifications for the implemented modules in a program simulating a Solar Water Heating System with Phase Change Material. It is intended to ease navigation through the program for design and maintenance purposes.

Complementary documents include the System Requirement Specifications and Module Guide.

2 Notation

The following table summarizes the primitive data types used by SWHS.

Data Type	Notation	Description
character	char	a single symbol or digit
integer	\mathbb{Z}	a number without a fractional component in $(-\infty, \infty)$
real	\mathbb{R}	any number in $(-\infty, \infty)$

SWHS also uses some derived data types: arrays, strings, and structures. Arrays are lists filled with elements of the same data type. Strings are arrays of characters. Structures contain pairs of keys and values, where keys are unique variable names used to identify their corresponding value, and values can be of any data type. SWHS also uses functions, which are defined by the data types of their inputs and outputs. Functions are described by showing their input data types separated by multiplication symbols on the left side of an arrow, and their output data type on the right side.

3 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	Level 2
Hardware-Hiding Module	
Behaviour-Hiding Module	Input Format Module Input Parameters Module Input Verification Module Output Format Module Output Verification Module Temperature ODEs Module Energy Equations Module Control Module
Software Decision Module	Sequence Data Structure Module ODE Solver Module Plotting Module

Table 1: Module Hierarchy

4 MIS of Control Module

4.1 Module

main

4.2 Uses

parameters (5), load_params (6), verify_params (7), temperature (8), ODE Solvers Module (9), energy (10), verify_output (11), plot (12), output (13)

4.3 Syntax

4.3.1 Exported Access Programs

Name	In	Out	Exceptions
main	string	-	-

4.4 Semantics

4.4.1 State Variables

time: array of reals

tempW: array of reals

tempP: array of reals

latHeat: array of reals

eW: array of reals
eP: array of reals
eTot: array of reals

4.4.2 Environment Variables

win: 2D array of pixels displayed on the screen

4.4.3 Access Routine Semantics

main(*s*): transition: *time, tempW, tempP, latHeat eW, eP, eTot, win* := *results*[0],
results[1], *results*[2], *results*[3], *eW1||eW2||eW3, eP1||eP2||eP3*,
 $(\forall i \in [0..|post(eW)| - 1]) (post(eW[i]) + post(eP[i]))$, Prints infor-
 mation about the melting of PCM.
 exception: none

5 MIS of Input Parameters Module

5.1 Module

parameters

5.2 Uses

N/A

5.3 Syntax

5.3.1 Exported Data Types

parameters := structure

5.3.2 Exported Access Programs

N/A

5.4 Semantics

5.4.1 State Variables

params.L: real
params.diam: real
params.Vp: real
params.Ap: real
params.rho_p: real

params.Tmelt: real
params.C_ps: real
params.C_pl: real
params.Hf: real
params.Ac: real
params.Tc: real
params.rho_w: real
params.C_w: real
params.hc: real
params.hp: real
params.Tinit: real
params.tstep: real
params.tfinal: real
params.AbsTol: real
params.RelTol: real
params.ConsTol: real
params.Vt: real
params.Mw: real
params.tau_w: real
params.eta: real
params.Mp: real
params.tau_ps: real
params.tau_pl: real
params.Epmelt_init: real
params.Ep_melt3: real
params.Mw_noPCM: real
params.tau_w_no_PCM: real

5.4.2 Access Routine Semantics

N/A

6 MIS of Input Format Module

6.1 Module

load_params

6.2 Uses

parameters (5)

6.3 Syntax

6.4 Exported Access Programs

Name	In	Out	Exceptions
load_params	string	parameters	-

6.5 Semantics

6.5.1 State Variables

params: parameters

param: array of reals

6.5.2 Assumptions

The input string corresponds to an existing filename in the current directory. The input file is formatted correctly.

6.5.3 Access Routine Semantics

load_params(*s*): transition: *params.L*, *params.diam*, *params.Vp*, *params.Ap*,
params.rho_p, *params.Tmelt*, *params.C_ps*, *params.C_pl*,
params.Hf, *params.Ac*, *params.Tc*, *params.rho_w*,
params.C_w, *params.hc*, *params.hp*, *params.Tinit*,
params.tstep, *params.tfinal*, *params.AbsTol*,
params.RelTol, *params.ConsTol*, *params.Vt*, *params.Mw*,
params.tau_w, *params.eta*, *params.Mp*, *params.tau_ps*,
params.tau_pl, *params.Epmelt_init*, *params.Ep_melt3*,
params.Mw_noPCM, *params.tau_w_noPCM* := *param*[0],
param[1], *param*[2], *param*[3], *param*[4], *param*[5],
param[6], *param*[7], *param*[8], *param*[9], *param*[10],
param[11], *param*[12], *param*[13], *param*[14], *param*[15],
param[16], *param*[17], *param*[18], *param*[19], *param*[20],
param[21], *param*[22], *param*[23], *param*[24], *param*[25],
param[26], *param*[27], *param*[28], *param*[29], *param*[30],
param[31], where *param* is the array of parameters obtained
from the input file *s*

exception: none

7 MIS of Input Verification Module

7.1 Module

verify_params

7.2 Uses

parameters (5)

7.3 Syntax

7.3.1 Exported Access Programs

Name	In	Out	Exceptions
verify_valid	parameters	-	badLength, badDiam, badPCMVolume, badPCMAndTankVol, badPCMArea, badPCMDensity, badMeltTemp, badCoilAndInitTemp, badCoilTemp, badPCMHeatCapSolid, badPCMHeatCapLiquid, badHeatFusion, badCoilArea, badWaterDensity, badWaterHeatCap, badCoilCoeff, badPCMCoeff, badInitTemp, badFinalTime, badInitAndMeltTemp
verify_recommended	parameters	-	-

7.4 Semantics

7.4.1 Environment Variables

win: 2D array of pixels displayed on the screen.

7.4.2 Assumptions

All of the fields of the input parameters structure have been assigned a value.

7.4.3 Access Routine Semantics

verify_valid(<i>params</i>):	transition:	<i>win</i> : (error is thrown \Rightarrow Prints error message)
	exceptions:	$exc := (params.L \leq 0 \Rightarrow \text{badLength} \mid$ $params.diam \leq 0 \Rightarrow \text{badDiam} \mid$ $params.Vp \leq 0 \Rightarrow \text{badPCMVVolume} \mid$ $params.Vp \geq params.Vt \Rightarrow \text{badPCMAndTankVol} \mid$ $params.Ap \leq 0 \Rightarrow \text{badPCMArea} \mid$ $params.rho_p \leq 0 \Rightarrow \text{badPCMDensity} \mid$ $params.Tmelt \leq 0 \vee params.Tmelt \geq$ $params.Tc \Rightarrow \text{badMeltTemp} \mid params.Tc \leq$ $params.Tinit \Rightarrow \text{badCoilAndInitTemp} \mid$ $params.Tc \geq 100 \vee params.Tc \leq 0 \Rightarrow$ $\text{badCoilTemp} \mid params.C_{ps} \leq 0 \Rightarrow \text{badPCMHeatCapSolid} \mid$ $params.C_{pl} \leq 0 \Rightarrow \text{badPCMHeatCapLiquid} \mid$ $params.Hf \leq 0 \Rightarrow \text{badHeatFusion} \mid params.Ac \leq 0 \Rightarrow$ $\text{badCoilArea} \mid params.rho_w \leq 0 \Rightarrow \text{badWaterDensity} \mid$ $params.C_w \leq 0 \Rightarrow \text{badWaterHeatCap} \mid params.hc \leq 0 \Rightarrow$ $\text{badCoilCoeff} \mid params.hp \leq 0 \Rightarrow \text{badPCMCo-eff} \mid$ $params.Tinit \leq 0 \vee params.Tinit \geq 100 \Rightarrow \text{badInitTemp} \mid$ $params.tfinal \leq 0 \Rightarrow \text{badFinalTime} \mid params.Tinit \geq$ $params.Tmelt \Rightarrow \text{badInitAndMeltTemp})$ See Appendix (14) for the complete list of exceptions and associated error messages.
verify_recommended(<i>params</i>):	transition:	<i>win</i> : (Warning is thrown \Rightarrow Prints warning message)
	exception:	none

8 MIS of Temperature ODEs Module

8.1 Module

temperature

8.2 Uses

parameters (5)

8.3 Syntax

8.3.1 Exported Access Programs

Name	In	Out	Exceptions
temperature1	array of reals, array of reals, array of reals, parameters	array of functions	-
temperature2	array of reals, array of reals, array of reals, array of reals, parameters	array of functions	-
temperature3	array of reals, array of reals, array of reals, parameters	array of functions	-
event1	array of reals, array of reals, array of reals, parameters	function	-
event2	array of reals, array of reals, array of reals, array of reals, parameters	function	-

8.4 Semantics

8.4.1 State Variables

t : array of reals

$Tw1$: array of reals

$Tw2$: array of reals

$Tw3$: array of reals

$Tp1$: array of reals

$Tp2$: array of reals

$Tp3$: array of reals

$Qp2$: array of reals

8.4.2 Assumptions

All of the fields of the input parameters structure have been assigned a value. The values have been properly constrained.

8.4.3 Access Routine Semantics

temperature1($t, Tw1, Tp1, params$):	output:	$out := \{dT_w : real \times real \times real \rightarrow real, dT_p : real \times real \times real \rightarrow real\}$
	exception:	none
temperature2($t, Tw2, Tp2, Qp2, params$):	output:	$out := \{dT_w : real \times real \times real \times real \rightarrow real, dT_p : real \times real \times real \times real \rightarrow real, dQ_p : real \times real \times real \times real \rightarrow real\}$
	exception:	none
temperature3($t, Tw3, Tp3, params$):	output:	$out := \{dT_w : real \times real \times real \rightarrow real, dT_p : real \times real \times real \rightarrow real\}$
	exception:	none
event1($t, Tw1, Tp1, params$):	output:	$out := Ev : real \times real \times real \rightarrow real$
	exception:	none
event2($t, Tw2, Tp2, Qp2, params$):	output:	$out := Ev : real \times real \times real \times real \rightarrow real$
	exception:	none

9 MIS of ODE Solver Module

9.1 Module

ODE Solver Module

9.2 Uses

N/A

9.3 Syntax

9.3.1 Exported Access Programs

Name	In	Out	Exceptions
solve	function, array of reals, array of reals, function, real, real	array of reals (<i>NUM_EQS</i> of them)	-

9.4 Semantics

9.4.1 State Variables

results: array of reals (*NUM_EQS* of them)

9.4.2 Local Constants

NUM_EQS: integer

9.4.3 Access Routine Semantics

solve(*f*, *domain*, *ics*, *events*, *abstol*, *reltol*) output: *out* := *results*, where
results holds the solution to the ODE system generated by the solver.
exceptions: none

10 MIS of Energy Module

10.1 Module

energy

10.2 Uses

parameters (5)

10.3 Syntax

10.3.1 External Access Programs

Name	In	Out	Exceptions
energy1Wat	array of reals, parameters	array of reals	-
energy1PCM	array of reals, parameters	array of reals	-
energy2Wat	array of reals, parameters	array of reals	-
energy2PCM	array of reals, parameters	array of reals	-
energy3Wat	array of reals, parameters	array of reals	-
energy3PCM	array of reals, parameters	array of reals	-

10.4 Semantics

10.4.1 State Variables

eW1: array of reals

eP1: array of reals

eW2: array of reals

eP2: array of reals

eW3: array of reals

eP3: array of reals

10.4.2 Assumptions

All of the fields of the input parameters structure have been assigned a value. The values have been properly constrained.

10.4.3 Access Routine Semantics

energy1Wat($Tw1, params$):	transition:	$(\forall i \in [0.. Tw1 - 1]) (eW1[i] := \text{watEnergy}(Tw1[i], params))$
	output:	$out := eW1$
	exception:	none
energy1PCM($Tp1, params$):	transition:	$(\forall i \in [0.. Tp1 - 1]) (eP1[i] := \text{pcmEnergy1}(Tp1[i], params))$
	output:	$out := eP1$
	exception:	none
energy2Wat($Tw2, params$):	transition:	$(\forall i \in [0.. Tw2 - 1]) (eW2[i] := \text{watEnergy}(Tw2[i], params))$
	output:	$out := eW2$
	exception:	none
energy2PCM($Qp2, params$):	transition:	$(\forall i \in [0.. Qp2 - 1]) (eP2[i] := \text{pcmEnergy2}(Qp2[i], params))$
	output:	$out := eP2$
	exception:	none
energy3Wat($Tw3, params$):	transition:	$(\forall i \in [0.. Tw3 - 1]) (eW3[i] := \text{watEnergy}(Tw3[i], params))$
	output:	$out := eW3$
	exception:	none
energy3PCM($Tp3, params$):	transition:	$(\forall i \in [0.. Tp3 - 1]) (eP3[i] := \text{pcmEnergy3}(Tp3[i], params))$
	output:	$out := eP3$
	exception:	none

10.4.4 Local Functions

watEnergy: $\text{real} \times \text{parameters} \rightarrow \text{real}$

$\text{watEnergy}(Tw, params) \equiv params.C_w \times params.Mw \times (Tw - params.Tinit)$

pcmEnergy1: $\text{real} \times \text{parameters} \rightarrow \text{real}$

$\text{pcmEnergy1}(Tp, params) \equiv params.C_ps \times params.Mp \times (Tp - params.Tinit)$

pcmEnergy2: $\text{real} \times \text{parameters} \rightarrow \text{real}$

$\text{pcmEnergy2}(Qp, params) \equiv params.Epmelt_init + Qp$

pcmEnergy3: $\text{real} \times \text{parameters} \rightarrow \text{real}$

$$\text{pcmEnergy3}(Tp, \text{params}) \equiv \text{params.Epmelt_init} + \text{params.Ep_melt3} + \text{params.C_pl} \times \text{params.Mp} \times (Tp - \text{params.Tmelt})$$

11 MIS of Output Verification Module

11.1 Module

verify_output

11.2 Uses

parameters (5)

11.3 Syntax

11.3.1 Exported Access Programs

Name	In	Out	Exceptions
verify_output	array of reals, array of reals, array of reals, array of reals, array of reals, parameters	-	-

11.4 Semantics

11.4.1 State Variables

expEPCM: array of reals

expEWat: array of reals

errorWater: real

errorPCM: real

11.4.2 Environment Variables

win: 2D array of pixels displayed on the screen

11.4.3 Local Variables

11.4.4 Assumptions

All of the fields of the input parameters structure have been assigned a value. The values have been properly constrained. The input arrays are not empty.

11.4.5 Access Routine Semantics

$\text{verify_output}(t, Tw, Tp, Ew, Ep, \text{params})$: transition: $\text{expEPCM}, \text{expEWat}, \text{errorWater}, \text{errorPCM}$,
 $\text{win} := (\forall i \in [1..|t| - 1])$
 $(\text{expectedEp}(\text{traprule}(\text{delta}(t[i - 1], t[i]), Tw[i], Tp[i], Tw[i - 1], Tp[i - 1]), \text{params})),$
 $(\forall i \in [1..|t| - 1]) (\text{expectedEw}(\text{expectedEc}(\text{traprule}(\text{delta}(t[i - 1], t[i]), \text{params.Tc}, Tw[i], \text{params.Tc}, Tw[i - 1]), \text{params}), \text{post}(\text{expEPCM}))),$
 $\text{error}(\text{sum}(\text{post}(\text{expEWat})), \text{Ew}[|Ew| - 1]),$
 $\text{error}(\text{sum}(\text{post}(\text{expEPCM})), \text{Ep}[|Ep| - 1]), (\text{errorWater} > \text{ConsTol} \vee \text{errorPCM} > \text{ConsTol} \Rightarrow \text{Prints warning message(s)})$
 exception: none

11.4.6 Local Functions

$\text{delta}: \text{real} \times \text{real} \rightarrow \text{real}$

$\text{delta}(t1, t2) \equiv t2 - t1$

$\text{traprule}: \text{real} \times \text{real} \times \text{real} \times \text{real} \times \text{real} \rightarrow \text{real}$

$\text{traprule}(t, A1, B1, A2, B2) \equiv t \times (A1 - B1 + A2 - B2)/2$

$\text{expectedEc}: \text{real} \times \text{parameters} \rightarrow \text{real}$

$\text{expectedEc}(c, \text{params}) \equiv \text{params.hc} \times \text{params.Ac} \times c$

$\text{expectedEp}: \text{real} \times \text{parameters} \rightarrow \text{real}$

$\text{expectedEp}(p, \text{params}) \equiv \text{params.hp} \times \text{params.Ac} \times p$

$\text{expectedEw}: \text{real} \times \text{real} \rightarrow \text{real}$

$\text{expectedEw}(Ec, Ep) \equiv Ec - Ep$

$\text{sum}: \text{array of reals} \rightarrow \text{real}$

$\text{sum}(a) \equiv +(i \in [0..|a| - 2]) (a[i] + a[i + 1])$

$\text{error}: \text{real} \times \text{real} \rightarrow \text{real}$

$$\text{error}(\text{exp}, \text{act}) \equiv \frac{|\text{exp}-\text{act}|}{\text{act}} \times 100$$

12 MIS of Plotting Module

12.1 Module

plot

12.2 Uses

N/A

12.3 Syntax

12.3.1 Exported Access Programs

Name	In	Out	Exceptions
plot	array of reals, array of reals, array of reals, array of reals, array of reals, string	-	-

12.4 Semantics

12.4.1 State Variables

plotFilename: string

12.4.2 Environment Variables

directory: The current directory of files from which the program is run.

12.4.3 Assumptions

The input arrays are all of the same size.

12.4.4 Access Routine Semantics

plot(*t*, *Tw*, *Tp*, *Ew*, *Ep*, *filename*): transition: *directory*: writes a .png file named *plotFilename* containing the graphs of the simulation results.
exception: none

13 MIS of Output Module

13.1 Module

output

13.2 Uses

parameters (5)

13.3 Syntax

13.3.1 Exported Constants

max_width: integer

13.3.2 Exported Access Program

Name	In	Out	Exceptions
output	string, array of reals, array of reals, array of reals, array of reals, array of reals, array of reals, parameters	-	-

13.4 Semantics

13.4.1 State Variables

outFilename: string

13.4.2 Environment Variables

directory: The current directory of files from which the program is run.

13.4.3 Access Routine Semantics

output(*params*, *t*, *Tw*, *Tp*, *Ew*, *Ep*, *ETot*, *filename*): transition: *directory*: writes a .txt file named *outFilename* containing the input parameters, calculated parameters, and results of the simulation.
exception: none

14 Appendix

Table 2: Possible Exceptions

Message ID	Error Message
badLength	Error: Tank length must be > 0
badDiam	Error: Tank diameter must be > 0
badPCMVolume	Error: PCM volume must be > 0
badPCMAndTankVol	Error: PCM volume must be $<$ tank volume
badPCMArea	Error: PCM area must be > 0
badPCMDensity	Error: ρ_p must be > 0
badMeltTemp	Error: T_{melt} must be > 0 and $< T_c$
badCoilAndInitTemp	Error: T_c must be $> T_{\text{init}}$
badCoilTemp	Error: T_c must be > 0 and < 100
badPCMHeatCapSolid	Error: C_{ps} must be > 0
badPCMHeatCapLiquid	Error: C_{pl} must be > 0
badHeatFusion	Error: H_f must be > 0
badCoilArea	Error: A_c must be > 0
badWaterDensity	Error: ρ_w must be > 0
badWaterHeatCap	Error: C_w must be > 0
badCoilCoeff	Error: h_c must be > 0
badPCMCoeff	Error: h_p must be > 0
badInitTemp	Error: T_{init} must be > 0 and < 100
badFinalTime	Error: t_{final} must be > 0
badInitAndMeltTemp	Error: T_{init} must be $< T_{\text{melt}}$