

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

Brooks MacLachlan and Spencer Smith

November 7, 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	9
6.5.4	Local Functions	9
7	MIS of Input Verification Module	10
7.1	Module	10
7.2	Uses	10
7.3	Syntax	11
7.3.1	Exported Access Programs	11
7.4	Semantics	11
7.4.1	Environment Variables	11
7.4.2	Assumptions	11
7.4.3	Access Routine Semantics	12
8	MIS of Temperature ODEs Module	13
8.1	Module	13
8.2	Uses	13
8.3	Syntax	14
8.3.1	Exported Access Programs	14
8.4	Semantics	14
8.4.1	State Variables	14
8.4.2	Assumptions	14
8.4.3	Access Routine Semantics	15
9	MIS of ODE Solver Module	15
9.1	Module	15
9.2	Uses	15
9.3	Syntax	15
9.3.1	Exported Constants	15
9.3.2	Exported Access Programs	16
9.4	Semantics	16
9.4.1	State Variables	16
9.4.2	Access Routine Semantics	16

10 MIS of Energy Module	16
10.1 Module	16
10.2 Uses	16
10.3 Syntax	17
10.3.1 External Access Programs	17
10.4 Semantics	17
10.4.1 State Variables	17
10.4.2 Assumptions	17
10.4.3 Access Routine Semantics	18
10.4.4 Local Functions	18
11 MIS of Output Verification Module	19
11.1 Module	19
11.2 Uses	19
11.3 Syntax	19
11.3.1 Exported Access Programs	19
11.4 Semantics	19
11.4.1 State Variables	19
11.4.2 Environment Variables	19
11.4.3 Local Variables	19
11.4.4 Assumptions	19
11.4.5 Access Routine Semantics	20
11.4.6 Local Functions	20
12 MIS of Plotting Module	21
12.1 Module	21
12.2 Uses	21
12.3 Syntax	21
12.3.1 Exported Access Programs	21
12.4 Semantics	21
12.4.1 State Variables	21
12.4.2 Environment Variables	21
12.4.3 Assumptions	21
12.4.4 Access Routine Semantics	22
13 MIS of Output Module	22
13.1 Module	22
13.2 Uses	22
13.3 Syntax	22
13.3.1 Exported Constants	22
13.3.2 Exported Access Program	22
13.4 Semantics	22
13.4.1 State Variables	22

13.4.2 Environment Variables	22
13.4.3 Access Routine Semantics	23
14 Appendix	23

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 $(-\infty, \infty)$
natural number	\mathbb{N}	a number without a fractional component in $[1, \infty)$
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. In addition, SWHS 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 Short Name

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 Short Name

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 Short Name

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. It should contain the numeric values for each input parameter in order, each on a new line. The order is the same as in the table in R1 of the SRS. Any comments should be denoted with a '#' symbol.

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], calcVt(post(*params.L*), post(*params.diam*)),
calcMw(post(*params.Vp*), post(*params.rho_w*),
post(*params.Vt*)), calcTauw(post(*params.Mw*),
post(*params.C_w*), post(*params.hc*),
post(*params.Ac*)), calcEta(post(*params.hp*),
post(*params.Ap*), post(*params.hc*), post(*params.Ac*)),
calcMp(post(*params.rho_p*), post(*params.Vp*)),
calcTaups(post(*params.Mp*), post(*params.C_ps*),
post(*params.hp*), post(*params.Ap*)),
calcTaupl(post(*params.Mp*), post(*params.C_pl*),
post(*params.hp*), post(*params.Ap*)),
calcEpmeltinit(post(*params.C_ps*), post(*params.Mp*),
post(*params.Tmelt*), post(*params.Tinit*)),
calcEpmelt3(post(*params.Hf*), post(*params.Mp*)),
calcMwnoPCM(post(*params.rho_w*), post(*params.Vt*)),
calcTauwnoPCM(post(*params.Mw_noPCM*),
post(*params.C_w*), post(*params.hc*), post(*params.Ac*)),
where *param* is the array of parameters obtained from the
input file *s*

exception: none

6.5.4 Local Functions

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

$$\text{calcVt}(L, \text{diam}) \equiv \pi \times L \times \left(\frac{\text{diam}}{2}\right)^2$$

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

$$\text{calcMw}(Vp, \text{rho}_w, Vt) \equiv \text{rho}_w \times (Vt - Vp)$$

calcTauw: $\text{real} \times \text{real} \times \text{real} \times \text{real} \rightarrow \text{real}$
 $\text{calcTauw}(Mw, C_w, hc, Ac) \equiv \frac{Mw \times C_w}{Ac \times hc}$

calcEta: $\text{real} \times \text{real} \times \text{real} \times \text{real} \rightarrow \text{real}$
 $\text{calcEta}(hp, Ap, hc, Ac) \equiv \frac{hp \times Ap}{hc \times Ac}$

calcMp: $\text{real} \times \text{real} \rightarrow \text{real}$
 $\text{calcMp}(\rho_p, Vp) \equiv \rho_p \times Vp$

calcTaups: $\text{real} \times \text{real} \times \text{real} \times \text{real} \rightarrow \text{real}$
 $\text{calcTaups}(Mp, C_ps, hp, Ap) \equiv \frac{Mp \times C_ps}{hp \times Ap}$

calcTaupl: $\text{real} \times \text{real} \times \text{real} \times \text{real} \rightarrow \text{real}$
 $\text{calcTaupl}(Mp, C_pl, hp, Ap) \equiv \frac{Mp \times C_pl}{hp \times Ap}$

calcEpmeltinit: $\text{real} \times \text{real} \times \text{real} \times \text{real} \rightarrow \text{real}$
 $\text{calcEpmeltinit}(C_ps, Mp, T_{melt}, T_{init}) \equiv C_ps \times Mp \times (T_{melt} - T_{init})$

calcEpmelt3: $\text{real} \times \text{real} \rightarrow \text{real}$
 $\text{calcEpmelt3}(Hf, Mp) \equiv Hf \times Mp$

calcMwnoPCM: $\text{real} \times \text{real} \rightarrow \text{real}$
 $\text{calcMwnoPCM}(\rho_w, Vt) \equiv \rho_w \times Vt$

calcTauwnoPCM: $\text{real} \times \text{real} \times \text{real} \times \text{real} \rightarrow \text{real}$
 $\text{calcTauwnoPCM}(Mw_noPCM, C_w, hc, Ac) \equiv \frac{Mw_noPCM \times C_w}{hc \times Ac}$

7 MIS of Input Verification Module

7.1 Short Name

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(params):` transition: *win*: (error is thrown \Rightarrow Prints error message)
 exceptions: *exc* := ($params.L \leq 0 \Rightarrow \text{badLength} \mid params.diam \leq 0 \Rightarrow \text{badDiam} \mid params.Vp \leq 0 \Rightarrow \text{badPCMVolume} \mid params.Vp \geq params.Vt \Rightarrow \text{badPCMAndTankVol} \mid params.Ap \leq 0 \Rightarrow \text{badPCMArea} \mid params.rho_p \leq 0 \Rightarrow \text{badPCMDensity} \mid params.Tmelt} \leq 0 \vee params.Tmelt \geq params.Tc \Rightarrow \text{badMeltTemp} \mid params.Tc \leq params.Tinit \Rightarrow \text{badCoilAndInitTemp} \mid params.Tc \geq 100 \vee params.Tc \leq 0 \Rightarrow \text{badCoilTemp} \mid params.C_{ps} \leq 0 \Rightarrow \text{badPCMHeatCapSolid} \mid params.C_{pl} \leq 0 \Rightarrow \text{badPCMHeatCapLiquid} \mid params.Hf} \leq 0 \Rightarrow \text{badHeatFusion} \mid params.Ac} \leq 0 \Rightarrow \text{badCoilArea} \mid params.rho_w \leq 0 \Rightarrow \text{badWaterDensity} \mid params.C_w} \leq 0 \Rightarrow \text{badWaterHeatCap} \mid params.hc} \leq 0 \Rightarrow \text{badCoilCoeff} \mid params.hp} \leq 0 \Rightarrow \text{badPCMCoeff} \mid params.Tinit} \leq 0 \vee params.Tinit \geq 100 \Rightarrow \text{badInitTemp} \mid params.tfinal} \leq 0 \Rightarrow \text{badFinalTime} \mid params.Tinit} \geq params.Tmelt \Rightarrow \text{badInitAndMeltTemp}$) See Appendix (14) for the complete list of exceptions and associated error messages.

verify_recommended(*params*): transition: *win*: (Warning is thrown \Rightarrow Prints warning message)

exceptions: *exc* := (*params.L* < 0.1 \vee *params.L* > 50 \Rightarrow warnLength | *params.diam*/*params.L* < 0.002 \vee *params.diam*/*params.L* > 200 \Rightarrow warnDiam | *params.Vp* < *params.Vt* $\times 10^{-6}$ \Rightarrow warnPCMVol | *params.Vp* > *params.Ap* \vee *params.Ap* > (2/0.001) \times *params.Vp* \Rightarrow warnVolArea | *params.rho_p* \leq 500 \vee *params.rho_p* \geq 20000 \Rightarrow warnPCMDensity | *params.C_ps* \leq 100 \vee *params.C_ps* \geq 4000 \Rightarrow warnPCMHeatCapSolid | *params.C_pl* \leq 100 \vee *params.C_pl* \geq 5000 \Rightarrow warnPCMHeatCapLiquid | *params.Ac* > $\pi \times (\textit{params.diam}/2)^2$ \Rightarrow warnCoilArea | *params.rho_w* \leq 950 \vee *params.rho_w* > 1000 \Rightarrow warnWaterDensity | *params.C_w* \leq 4170 \vee *params.C_w* \geq 4210 \Rightarrow warnWaterHeatCap | *params.hc* \leq 10 \vee *params.hc* \geq 10000 \Rightarrow warnCoilCoeff | *params.hp* \leq 10 \vee *params.hp* \geq 10000 \Rightarrow warnPCMCoeff | *params.tfinal* \leq 0 \vee *params.tfinal* \geq 86400 \Rightarrow warnFinalTime) None of these exceptions terminate the program. See Appendix (14) for the complete list of exceptions and associated warning messages.

8 MIS of Temperature ODEs Module

8.1 Short Name

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 Short Name

ODE Solver Module

9.2 Uses

N/A

9.3 Syntax

9.3.1 Exported Constants

$MaxStep$: natural number

N : natural number

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
solve	function, array of reals, array of reals, function, real, real	array of reals (N of them)	ODE_BAD_INPUT, ODE_MAXSTEP, ODE_ACCURACY

9.4 Semantics

9.4.1 State Variables

results: array of reals (N of them)

9.4.2 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: $exc :=$ (Invalid input parameters \Rightarrow ODE_BAD_INPUT | $MaxStep$ steps taken and no solution found \Rightarrow ODE_MAXSTEP | $reltol$ and $abstol$ not satisfied for a step \Rightarrow ODE_ACCURACY)

10 MIS of Energy Module

10.1 Short Name

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 Short Name

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: expEPCM , expEWat ,
 errorWater , errorPCM ,
 $\text{win} := (\forall i \in [1..|t| - 1])$
 $(\text{expectedEp}(\text{traprule}(\text{delta}(t[i - 1], t[i]), Tw[i], Tp[i],$
 $Tw[i - 1], Tp[i - 1]), \text{params})),$
 $(\forall i \in [1..|t| - 1]) (\text{expectedEw}$
 $(\text{expectedEc}(\text{traprule}(\text{delta}(t[i - 1], t[i]),$
 $\text{params.Tc}, Tw[i], \text{params.Tc}, Tw[i - 1]),$
 $\text{params}), \text{post}(\text{expEPCM}))),$
 $\text{error}(\text{sum}(\text{post}(\text{expEWat})),$
 $Ew[|Ew| - 1]),$
 $\text{error}(\text{sum}(\text{post}(\text{expEPCM})),$
 $Ep[|Ep| - 1]), (\text{errorWater} >$
 $\text{ConsTol} \vee \text{errorPCM} >$
 $\text{ConsTol} \Rightarrow \text{Prints warning}$
 $\text{message(s)})$
exception: $(\text{errorWater} > \text{ConsTol} \Rightarrow$
 $\text{warnWaterError} \mid \text{errorPCM} >$
 $\text{ConsTol} \Rightarrow \text{warnPCMErrors})$
These exceptions do not terminate the program.

11.4.6 Local Functions

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

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

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

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

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

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

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

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

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

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

sum: array of reals \rightarrow real
 $\text{sum}(a) \equiv \sum_{i=0}^{|a|-1} a[i]$

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

12 MIS of Plotting Module

12.1 Short Name

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 Short Name

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, 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 $> T_{init}$
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 $< T_{melt}$

ODE_ACCURACY	<i>reltol</i> and <i>abstol</i> were not satisfied by the ODE solver for a given solution step.
ODE_BAD_INPUT	Invalid input to ODE solver
ODE_MAXSTEP	ODE solver took <i>MaxStep</i> steps and did not find solution
warnLength	Warning: It is recommended that $0.1 \leq L \leq 50$
warnDiam	Warning: It is recommended that $0.002 \leq D/L \leq 200$
warnPCMVOL	Warning: It is recommended that V_p be $\geq 0.0001\%$ of V_t
warnVolArea	Warning: It is recommended that $V_p \leq A_p \leq (2/0.001) * V_p$
warnPCMDensity	Warning: It is recommended that $500 < \rho_p < 20000$
warnPCMHeatCapSolid	Warning: It is recommended that $100 < C_{ps} < 4000$
warnPCMHeatCapLiquid	Warning: It is recommended that $100 < C_{pl} < 5000$
warnCoilArea	Warning: It is recommended that $A_c \leq \pi * (D/2) \wedge 2$
warnWaterDensity	Warning: It is recommended that $950 < \rho_w \leq 1000$
warnWaterHeatCap	Warning: It is recommended that $4170 < C_w < 4210$
warnCoilCoeff	Warning: It is recommended that $10 < h_c < 10000$
warnPCMCoeff	Warning: It is recommended that $10 < h_p < 10000$
warnFinalTime	Warning: It is recommended that $0 < t_{final} < 86400$
warnWaterError	Warning: There is greater than $x\%$ relative error between the energy in the water output and the expected output based on the law of conservation of energy. (Where x is the value of <i>ConsTol</i>)
warnPCMError	Warning: There is greater than $x\%$ relative error between the energy in the PCM output and the expected output based on the law of conservation of energy. (Where x is the value of <i>ConsTol</i>)
