HPWHsim Usage

Interface

Using the HPWH simulation in your own C++ code is quite simple. The HPWH.hh header contains all the necessary functions and definitions.

1. First, declare a HPWH object.

2. Use either of the member functions HPWHinit\_preset or HPWHinit\_file to intialize the hpwh object. Descriptions of these two functions are provided later in this document.

3. Run the simulation. There are two functions used for this. The primary function is runOneStep,

which takes arguments for the water inlet temperature in Celsius, the volume of water to be withdrawn this step in liters, the ambient temperature in Celsius, the external (evaporator) temperature in Celsius, the Demand Response status, and the length of time in minutes of this step.

RunNSteps takes an additional argument, the number of steps to run, and calls runOneStep that many times, aggregating the outputs as appropriate and reassigning them to the usual variables.

Note: Although using runOneStep with 20 minutes per step and runNSteps with 20 steps at 1 minute per step both run 20 minute of simulation, the results will be far different. 1 minute is the recommended length of time per step.

There is an alternate version of runOneStep that takes four inputs. This version uses stored variables for the inlet temperature and the minutes per step. The stored variables must be set using the setting functions setInletT and setMinutesPerStep; they do not change unless these functions are called.

4. Get the outputs. The hpwh object holds all of the outputs from the step (or steps, for runNsteps) that was run last. These are accessed through the following functions:

getTankNodeTemp – returns the temp of the specified node

getNthSimTcouple – the nodes are averaged to result in six equally spaced points, representing the 6 thermocouples used in lab measurements. Returns the Nth thermocouple.

getNthHeatSourceEnergyInput – returns the energy (electrical) input into the Nth heat source

getNthHeatSourceEnergyOutput – returns the energy output (heat) from the Nth heat source

getNthHeatSourceRunTime – returns the length of time the Nth heat source ran last step. Should not be longer than minutes per step, although the sum of all heat source run times can be, due to concurrently running heat sources

isNthHeatSourceRunning – returns whether or not the Nth heat source is still running

getOutletTemp – returns the average temp of outlet water over the last step – 0 for no draws

getEnergyRemovedFromEnvironment – returns the energy removed from the environment

getStandbyLosses – returns the energy lost from the water through the tank walls

Many of these functions have two forms. The first, with fewer arguments (or no arguments), returns the value in the default units, which are metric. The other form takes an argument of type UNITS, which is an enum that specifies which units will be used for the output value.

Model Specification

HPWHinit\_presets allows for a fully specified water heater model to be selected and initialized. The choices are enumerated by the enum MODELS. Currently the available choices are:

These are used for testing and should not be expected to provide realistic results

MODELS\_restankNoUA, MODELS\_restankHugeUA, MODELS\_restankRealistic, MODELS\_basicIntegrated, MODELS\_externalTest

These models are based on real tanks and measured lab data

MODELS\_Voltex60

MODELS\_Voltex80

MODELS\_GEGeospring

MODELS\_SandenGAU

HPWHinit\_file allows for a run time specification of all parameters and values used to run a HPWH. The function takes a filename argument and processes the text in this file to calculate parameters for the HPWH model. The specification is based on keywords and values. As described here, the quoted strings represent the keys, and %d, %f, etc. are the values.

%d – an integer

%f – a decimal number-of-showers

%s – a string for units - gal, L, F, C, kJperHrC

%b – the string “true” or “false”

“#” - if this starts a line, that line is a comment and will be skipped

“” - blank lines will also be skipped

"numNodes %d" – the number of nodes

"volume %d %s" – the tank volume, and its units

"UA %d %s" – the UA of the tank and its units, currently only specifiable as kJperHrC

"depressTemp %b" – whether or not the temperature depression ability should be used

"mixOnDraw %b" – whether or not the bottom of the tank should mix when draws occur

"setpoint %f %s" – the setpoint of the tank

"verbosity %s" – the verbosity; accepts "silent", "reluctant", "typical", or "emetic"

"numHeatSources %d" – the number of heat sources

The follow commands apply specifically to heat sources and must be preceded (in the same line) by:

“heatsource %d" where the integer given refers to one of the heat sources.

**Heat Source Parameters**

"isVIP %b" – if the heat source is a VIP or not

"isOn %b" – if the heat source is currently running

"type %s" – the type of the heat source: "resistor" or "compressor"

"coilConfig %s" – the configuration of the heat source's coils: "wrapped", "submerged", or "external"

"condensity %d %d %d %d %d %d %d %d %d %d %d %d" – the twelve values of the condensity

"T1 %d %s" – the first temperature for the heat source's COP and input power curves, and the units

"T2 %d %s" – the second temperature for the heat source's COP and input power curves, and the units

The following are the coefficients for the input power and COP curves – units are not specifiable

"inPowT1const %f”

"inPowT1lin %f"

"inPowT1quad %f"

"inPowT2const %f"

"inPowT2lin %f"

"inPowT2quad %f"

"copT1const %f"

"copT1lin %f"

"copT1quad %f"

"copT2const %f"

"copT2lin %f"

"copT2quad %f"

"hysteresis %f %s" – the hysteresis (see Tips and Tricks) and units

"backupSource %d" – the number of the heat source that is the backup to this heat source

"onlogic %s %f %s" – a logical specifier for heat source engaging conditions. The first string chooses the logic and the number chooses the setpoint, followed by its units. Currently available on-logics are: "topThird", "bottomThird", and "standby"

**Example**: “heatsource 1 onlogic bottomThird 40 F”

"offlogic %s %f %s" – a logical specified for heat source turn-off conditions. The first string chooses the logic and the number chooses the setpoint, followed by its units. Currently available off-logics are: "lowT", "lowTreheat", "bottomNodeMaxTemp", "bottomTwelthMaxTemp", "largeDraw"

**Example**: “heatsource 1 offlogic largeDraw 66 F”