```
%Konstantinos Vilaetis 2012
%Flow Rate calculation function
%Inputs are:
%DT: temperature difference in network in C
%dmnd: Demand profile daily, 1000btu/day
%
%Outputs:
%kg/s daily

function [kgspV TprV TsrV kgssV Tps0V Tpr0 dmndWthV maxWth HXArea] =
flowcalc(dmnd,PeakUnitDmnd,Tps,Tss,Tsr0,Tabsmin,com_units,res_units,WellFlowR
ate,Tpinch,maxdmnd)

% for i = 1:365
%     dmnd(i) = max(220000*(1-sin(pi()*i/365)),5000);
% end
% Tps = 85;
% Tpr = 30;
% Tss = 60;
% Tsr0 = 40;
% Tpinch = 5;
% maxdmnd = 300000;
% WellFlowRate = 40;

% U0 = 5000;
% Tss0 = Tss;
% Tps0 = Tps;
% Tpr0 = Tsr0+Tpinch;
% kgspV=dmnd./(3.412*24*4.2.*(Tps-(Tsr0+Tpinch)));
% Tps0V(1:365,1) = Tps;
% dmndWthV = (dmnd*1000*1055)/86400;
% maxWth = (maxdmnd*1000*1055)/(86400);
% TsrV(1:365,1) = Tsr0;
% TprV(1:365,1) = Tpr0;
% kgssV = 10;
% LMTDhx0 = ((Tps0-Tss0)-(Tpr0-Tsr0))/log((Tps0-Tss0)/(Tpr0-Tsr0));
% HXArea = maxWth/(U0*LMTDhx0);


%Design parameters for HX
Tps0 = Tps;                                             %Design primary
supply temp [C]
Tss0 = Tss;                                             %Design secondary
supply temp [C]
Tsr0 = Tsr0;                                            %Design secondary
return temp [C]
Tpr0 = Tsr0 + Tpinch;                                   %Design primary
return temp [C]
kgsp0 = WellFlowRate;                                   %Design primary mass
flow [kg/s]
kgss0 = (kgsp0*4200*(Tps0-Tpr0))/(4200*(Tss0-Tsr0));    %Design secondary
mass flow [kg/s]
Ti0 = 20;                                               %Indoor design
temperature
```

```
U0 = 5000;                                              %Design U-value
[W/m2*C]
Cu = U0*(1/(kgsp0^0.7)+1/(kgss0^0.7));                  %Constant for heat x-
fer coefficient estimation, calculated from design conditions and optimum U-
value
maxWth = (maxdmnd*1000*1055)/(86400);                   %Max (design)
community demand [Wth]
maxWthV(1:365,1) = maxWth;                              %Max power vector
(calc tool) [W]
LMTDhx0 = ((Tps0-Tss0)-(Tpr0-Tsr0))/log((Tps0-Tss0)/(Tpr0-Tsr0));    %Design
heat exchanger LMTD [C]
LMTDr0 = (Tss0-Tsr0)/log((Tss0-Ti0)/(Tsr0-Ti0));        %Design radiator LMTD
[C]
dmndWthV = (dmnd*1000*1055)/86400;                      %Demand vector
(average power each day of year) [W]
DesignWth = max(dmndWthV);                              %Design condition
power (if sized for average peak) [W]
DesignWthV(1:365,1) = DesignWth;                        %Design power vector
(calc tool)
HXArea = maxWth/(U0*LMTDhx0);                          %Req'd heat exchnager
size (if sized for extreme peak) [m2]
%HXArea = DesignWth/(U0*LMTDhx0);                       %Req'd heat
exchanger size (if sized for average peak) [m2]


LMTDrV = ((dmndWthV./maxWthV).^1.3)*LMTDr0;            %Daily radiator LMTD
vector at non-design loads (if designed for extreme peak)
%LMTDrV = ((dmndWthV./DesignWthV).^1.3)*LMTDr0;         %Daily radiator LMTD
vector at non-design loads (if designed for average peak)
Tss0V(1:365,1) = Tss0;                                  %Daily secondary
supply temp vector
TsrV = zeros(365,1);                                    %Setup secondary
return temp vector
kgssV = zeros(365,1);                                   %Setup secondary mass
flow vector
Ti0V(1:365,1) = Ti0;
Uvect = zeros(365,1);
kgspV = zeros(365,1);
TprV = zeros(365,1);
Tps0V(1:365,1) = Tps0;


%crazyassfunction = @(x) dmndWthV.*((kgssV.*(Tss0V-TsrV)./(Tps0V-x)).^-
0.7+kgssV.^-0.7).*log(((Tps0V-Tss0V)./(x-TsrV)))-(Tps0V-Tss0V-
x+TsrV)*Cu*HXArea;




    TsrV = max(Ti0V-LMTDrV.*lambertw(-((Tss0-Ti0).*exp((Ti0V./LMTDrV)-
(Tss0V./LMTDrV)))./LMTDrV),Ti0+2);   %Secondary return temp vector
    kgssV = dmndWthV./(4200*(Tss0V-TsrV));       %Daily secondary mass flow
vector
    %TprV(i) = fzero(crazyassfunction,30);
    %syms x
```

```
    %TprV(i) = solve(dmndWthV(i)*((kgssV(i)*(Tss0-TsrV(i))/(Tps0-x))^-
0.7+kgssV(i)^-0.7)*log((Tps0-Tss0)/(x-TsrV(i)))==Cu*HXArea*(Tps0-Tss0-
x+TsrV(i)),x);
    %kgspV(i) = dmndWthV(i)/(4200*(Tps0-TprV(i)));



x0(1:365,1) = TsrV+Tpinch;
%x = sym('x',[365 1]);
%TprV = solve(dmndWthV.*((kgssV.*(Tss0V-TsrV)./(Tps0V-x)).^-0.7+kgssV.^-
0.7).*log(((Tps0V-Tss0V)./(x-TsrV)))==(Tps0V-Tss0V-
x+TsrV)*Cu*HXArea,x,'IgnoreAnalyticConstraints',true);
crazyassfunction = @(x) dmndWthV.*((kgssV.*(Tss0V-TsrV)./(Tps0V-x)).^-
0.7+kgssV.^-0.7).*log(((Tps0V-Tss0V)./(x-TsrV)))-(Tps0V-Tss0V-
x+TsrV)*Cu*HXArea;
%TprV = fzero(crazyassfunction,x0);
TprV = fsolve(crazyassfunction,x0);
TprV(TprV<(min(TsrV+2))) = (min(TsrV+2));
kgspV = dmndWthV./(4200*(Tps0V-TprV));

end
```