```
function ConPow = PID(CurTime, SetpointTemp, FurnaceTemp)
global Up Ui Ud DerPts Pzero;
global PreviousTime PreviousSetpointT PreviousFurnaceT IntegralSetpointT
global IntegralFurnaceT;
global DTime DSetTemp DFurnTemp DPointsCounter
global FirstTimeIntegral
%Proportional
Proportional = FurnaceTemp - SetpointTemp;
%Integral
if FirstTimeIntegral==0
    FirstTimeIntegral=1; Integral=0;
else
    AreaS= (SetpointTemp+PreviousSetpointT)*(CurTime-PreviousTime)/2;
    IntegralSetpointT = IntegralSetpointT +AreaS;
    AreaF = (FurnaceTemp+PreviousFurnaceT)*(CurTime-PreviousTime)/2;
    IntegralFurnaceT= IntegralFurnaceT+AreaF;
    Integral = IntegralFurnaceT-IntegralSetpointT;
    %Wait until the second call and after to calculate the integral
    %Update IntegralFurnaceT, IntegralSetpointT, and Integral here...
end
PreviousTime=CurTime; PreviousSetpointT=SetpointTemp; PreviousFurnaceT=FurnaceTemp;
%Derivative
if DPointsCounter < DerPts
    DPointsCounter=DPointsCounter+1;
    DTime(DPointsCounter)=CurTime;
    DSetTemp(DPointsCounter)=SetpointTemp;
    DFurnTemp(DPointsCounter)=FurnaceTemp;
    Derivative=0;
    %Fill DTime, DSetTemp, and DFurnTemp arrays here...
else
    for i=1:DerPts-1
        DTime(i)=DTime(i+1);
        DSetTemp(i)=DSetTemp(i+1);
        DFurnTemp(i)=DFurnTemp(i+1);
    end
    DTime(DerPts)=CurTime;
    DSetTemp(DerPts)=SetpointTemp;
    DFurnTemp(DerPts)=FurnaceTemp;

    sumx=0; sumx2=0; sumxy=0; sumy=0;sumy1=0; sumxy1=0;
    for j=1:DerPts
        sumx=sumx+DTime(j);
        sumx2=sumx2+(DTime(j))^2;
        sumxy=sumxy+(DTime(j)*DSetTemp(j));
        sumy=sumy+(DSetTemp(j));
        sumy1=sumy1+(DFurnTemp(j));
        sumxy1=sumxy1+(DTime(j)*DFurnTemp(j));
    end
    slopeS = (sumxy-(sumx*sumy)/DerPts)/(sumx2-(sumx)^2/DerPts);
    slopeF = (sumxy1-(sumx*sumy1)/DerPts)/(sumx2-(sumx)^2/DerPts);
    Derivative=slopeF-slopeS;
    %Wait until after DerPts is collected to calculate the derivative
    %Shift data so most recent DerPts are loaded into the array...
    %Calculate least-squares slopes and calculate Derivative...
end
%ControlFunction
ConPow = Pzero - Proportional*Up - Integral*Ui - Derivative*Ud;
fprintf('%f, %f, %f, %f\n',CurTime, -Proportional*Up, -Integral*Ui, -Derivative*Ud)
```

```
if ConPow>100 %Protection to keep the control power in the correct range
    ConPow=100;
elseif ConPow<0
    ConPow=0;
end
```