```
%Nirav Patel and Faisal Sayed
%ECE 4760 Final Project
%with Prof. Bruce Land and Prof Brandon Hencey

function varargout = WirelessDAQ(varargin)
% WIRELESSDAQ M-file for WirelessDAQ.fig
%      WIRELESSDAQ, by itself, creates a new WIRELESSDAQ or raises the existing
%      singleton*.
%
%      H = WIRELESSDAQ returns the handle to a new WIRELESSDAQ or the handle to
%      the existing singleton*.
%
%      WIRELESSDAQ('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in WIRELESSDAQ.M with the given input arguments.
%
%      WIRELESSDAQ('Property','Value',...) creates a new WIRELESSDAQ or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before WirelessDAQ_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to WirelessDAQ_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help WirelessDAQ

% Last Modified by GUIDE v2.5 26-Apr-2012 14:44:06

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @WirelessDAQ_OpeningFcn, ...
                   'gui_OutputFcn',  @WirelessDAQ_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before WirelessDAQ is made visible.
function WirelessDAQ_OpeningFcn(hObject, eventdata, handles, varargin)
```

```
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to WirelessDAQ (see VARARGIN)

% Choose default command line output for WirelessDAQ
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
set(handles.slider1,'Value',24);   %set default values for slider
set(handles.sliderVal,'String','24');%set default values for Display for slider
setappdata(handles.sliderVal,'sliderVal','24'); %save the value of temperature
% UIWAIT makes WirelessDAQ wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = WirelessDAQ_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

%-------------------------Data Acquisition Part---------------------------%

%---User Sets the Iteration value in the edit box---%
function iterationVal_Callback(hObject, eventdata, handles)
% hObject    handle to iterationVal (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of iterationVal as text
%        str2double(get(hObject,'String')) returns contents of iterationVal as a double

iterations=get(hObject,'string');   %get the user entered value
index=strfind(iterations,'.');
if(isempty(index) && str2double(iterations)> 0) %check for non negetive integer input
    %save if qualifies the requirements
    setappdata(handles.iterationVal,'itrVal', get(hObject,'String'));
    %save ('data.mat','iterations'); %save the demanded data values in a file
else
    %throw an error message if not compatible
    msgbox('Incompatible Input. Please enter an Integer Value greater than 0.','Input↙
Error','Warn');
end

% --- Executes during object creation, after setting all properties.
```

```
function iterationVal_CreateFcn(hObject, eventdata, handles)
% hObject    handle to iterationVal (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

%---User sets the interval between consecutive reads ---%

function readInterval_Callback(hObject, eventdata, handles)
% hObject    handle to readInterval (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of readInterval as text
%        str2double(get(hObject,'String')) returns contents of readInterval as a double
val=get(hObject,'String');
%fprintf(val);
if(isempty(strfind(val,'.')) && str2double(val)>=200) %check for non negetive integer↙
input
    %save if qualifies the requirements
    setappdata(handles.readInterval, 'interVal',val);
else
    %throw an error message if not compatible
    msgbox('Incompatible Input. Please enter integer values from 200 to 32676.','Input↙
Error','Warn');
end



% --- Executes during object creation, after setting all properties.
function readInterval_CreateFcn(hObject, eventdata, handles)
% hObject    handle to readInterval (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% ------Starts Data Acquisition--------%

function AcquireData_Callback(hObject, eventdata, handles)
% hObject    handle to AcquireData (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%resets all the communication devices using the serial port
instrreset;
clc;    %clear command window
%clear all;  %clear old versions of variables
try             %try-catch error detection mechanism starts
fid=load('serial.mat','s'); %load the serial object created in SerialOpen
%load the number of iterations entered by the user
no_iterations = getappdata(handles.iterationVal,'itrVal');
%itrVal=str2double(no_iterations);
%s=serial('COM1');   % the communication port
%save('Data.mat','s');
%set(s,'terminator','CR'); %configure the terminating character in the communication
fopen(fid.s);   %open the serial port

%-----init variables-----%

%data=[];    %collects all the incoming serial data
iteration=1; %init iteration
raw_data=zeros(1000,8);  %init the data matrix
Power=zeros(str2double(no_iterations),1);
a=0;    %total number of bytes received
Vref = 3.3; %Voltage reference for ADC
VScale=9.2; %Voltage divider scaling factor for voltage sensor
IScale=14.7/10;%Voltage divider scaling factor for current sensor

%get the interval value from the interval object
interval=getappdata(handles.readInterval,'interVal');
%Send the Data transmission command to the controller
fprintf(fid.s,'r');
set(fid.s,'terminator','LF');
%send the number of iterations
fprintf(fid.s,no_iterations);
pause(0.2);
%send the interval between iterations
fprintf(fid.s,interval);
%Update the status
set(handles.Status,'String','Receiving...');
pause(2);

while(1)    % check for incoming data
  pause(str2double(interval)/1000);
  b=fid.s.bytesavailable();     %find the bytes available
  j=1;
  if(b>0)   %if available
    a=a+b;  %increment the global count of bytes
    out=fscanf(fid.s);  %scan the buffer
    x=strfind(out,'~'); %find out the start address of the start/end bytes for each data↙
value
%    fprintf('%d',x);
    if(isempty(x))  %if there exisits the desired identifiers
```

```
            fprintf('%s\n','Not found');
        else
            fprintf('%s\n','found');
           for i=1:2:(numel(x)-1) %loop around based on number of start bytes we have
           %y=out(1,x(i):x(i+1));
           out(x(i))=' ';   %insert a space instead of the identifiers
           out(x(i+1))=' ';
           y=out(1,x(i):x(i+1));
           temp=str2double(y);   %extract the string and converted it to number
           if(rem(i,2)~=0)
                raw_data(iteration,j)=temp; %add it to the raw data value
                j=j+1;
           end
         %  temperature(iteration,1)=temp;   %add the value to the column array
           end
        end
       %temperature(iteration,:)=out(x(1));
       %data=horzcat(data,out); %keep on adding the data into a single character array
       % out=[];
        b=0;
        %Calculate the voltage and current in the peltier loop
        V=raw_data(iteration,6)*Vref*VScale/1024;
        I_vout=raw_data(iteration,5)*Vref*IScale/1024;
        I=-36.067*I_vout^2 + 255.8*I_vout -423.16;
        %calculate the power consumed in the peltier in watts
        Power(iteration,1)=V*I;
        %update the power value on the GUI display
        set(handles.Power,'String',num2str(Power(iteration),5));
        if(iteration==str2double(no_iterations)) %loop for the number of iterations defined by↙
the user
           set(handles.Status,'String','Received Successfully');
           break;
        end
        iteration=iteration+1; %increment the counter after each successful receive
    else           %if no data is available, wait for 10 seconds and check again
          pause(10);
          if(fid.s.bytesAvailable()==0) %display an error message if no data after 10s
              msgbox('No Data Received','TimeOut Error','Error');
              set(handles.Status,'String','Reception Unsuccessful');
              break;
          end
    end
end
fclose(fid.s);  %close the serial port
save ('Data.mat','raw_data','Power'); %save the required data variables

catch err4          %Catch error messages anywhere in the above code
      % display(err4.identifier);
        %Handle the error by updating the status
        if(strcmp(err4.identifier,'MATLAB:serial:fopen:opfailed'))
            set(handles.Status,'String','Serial Port Not Open');
        else if(strcmp(err4.identifier,'MATLAB:load:couldNotReadFile'))
        set(handles.Status,'String','serial.mat not found. Please open Serial Port.');
        end
    end
end

%----------------------End of Data Acquisition Part----------------------%


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
value=get(hObject,'Value'); %get the value of the slider for a setpoint
%value=round(value);         %round the values to remove any decimals
%save('Data.mat','value');   %save the value of setpoint
%save the value of slider in the object property
setappdata(hObject,'slider1',value);
%update the value in the display
set(handles.sliderVal,'String',num2str(value,3));

% --- Executes during object creation, after setting all properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


function sliderVal_Callback(hObject, eventdata, handles)
% hObject    handle to sliderVal (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of sliderVal as text
%        str2double(get(hObject,'String')) returns contents of sliderVal as a double
value=str2double(get(hObject,'String'));    %check for user entered value
if (isnumeric(value) && ...     %check if it is numeric
   value >= get(handles.slider1,'Min') && ...%check if it is within bounds of the sliders
   value <= get(handles.slider1,'Max'))
   set(handles.slider1,'Value',value);  %update the slider position based on user value
   %store the data in the application space of the display object
   setappdata(hObject,'sliderVal',get(hObject,'String'));
   %save('Data.mat','value');    %save the value
end
```

```
% --- Executes during object creation, after setting all properties.
function sliderVal_CreateFcn(hObject, eventdata, handles)
% hObject    handle to sliderVal (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in setPoint.
function setPoint_Callback(hObject, eventdata, handles)
% hObject    handle to setPoint (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
instrreset;
clc;
%load the value of temperature desired from the workspace of slider1
value=getappdata(handles.slider1,'slider1');
try            %try-catch command for error handling
fid=load('serial.mat','s');
fopen(fid.s);   %open the serial port
value=value*10;
setpoint=num2str(value,3);
%fprintf(setpoint);
set(handles.Status,'String','Transmitting');
%transmit the command to update the setpoint
fprintf(fid.s,'w');
set(fid.s,'terminator','LF'); %configure the terminating character in the communication
%transmit the setpoint
fprintf(fid.s,setpoint);
pause(1);
% check for the acknowledgement sent by the controllers
        if(fid.s.bytesavailable()>0)
            status=fscanf(fid.s);
            index=strfind(status,'OK');
            %msgbox(status,'Status','warn');
            if(isempty(index))
                set(handles.Status,'String','Data Transmission Failed');
            else
                set(handles.Status,'String','Data Transmitted Successfully');
            end
        else
            set(handles.Status,'String','Data Transmission Failed');
        end
fclose(fid.s);
catch err3      %error detection and handling  by the catch error statement
```

```
    if(strcmp(err3.identifier,'MATLAB:serial:fopen:opfailed')||strcmp(err3.↙
identifier,'MATLAB:load:couldNotReadFile'))
        set(handles.Status,'String','Serial Port Not Open');
    end
end

%--------------Update the activity of GUI--------------%
function Status_Callback(hObject, eventdata, handles)
% hObject    handle to Status (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Status as text
%        str2double(get(hObject,'String')) returns contents of Status as a double


% --- Executes during object creation, after setting all properties.
function Status_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Status (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

%------------------------Serial Comm functions-------------------------%

%-----Get the serial port value-----%
function commSelect_Callback(hObject, eventdata, handles)
% hObject    handle to commSelect (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of commSelect as text
%        str2double(get(hObject,'String')) returns contents of commSelect as a double
com=get(hObject,'String');
%save the serial port value in the workspace of the Object
setappdata(handles.commSelect, 'comval', com);


% --- Executes during object creation, after setting all properties.
function commSelect_CreateFcn(hObject, eventdata, handles)
% hObject    handle to commSelect (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in serialOpen - Open serial Port---%
function serialOpen_Callback(hObject, eventdata, handles)
% hObject    handle to serialOpen (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
instrreset; %resets the devices communicating on the serial port
clc;    %clear command window
try     %try-catch function for error handling
%load the com port value
COMPORT=getappdata(handles.commSelect,'comval');
s=serial(COMPORT);   %Set the communication port
set(s,'terminator','CR'); %configure the terminating character in the communication
fopen(s);   %open the serial port
save('serial.mat','s');
set(handles.Status,'String','Serial Port Opened Successfully');
catch err1  %handle the errors and update the status message
    %display(err1.identifier);
    if (strcmp(err1.identifier,'MATLAB:serial:fopen:opfailed') ||...
        strcmp(err1.identifier,'MATLAB:serial:serial:invalidPORT') ||...
        strcmp(err1.identifier,'MATLAB:badfid_mx'))
        set(handles.Status,'String','Failed to open Serial Port. Enter Valid COMPORT.');
    end
end




% --- Executes on button press in serialClose-Close the serial Port ---%
function serialClose_Callback(hObject, eventdata, handles)
% hObject    handle to serialClose (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
try
serial=load('serial.mat','s');
fclose(serial.s);
delete(serial.s);
clear serial;
delete serial.mat;
set(handles.Status,'String','Serial Port Closed Successfully');
clear all;
catch err
    %display(err.identifier);
    if (strcmp(err.identifier,'MATLAB:load:couldNotReadFile'))
        set(handles.Status,'String','Serial.mat does not exist. Port is already closed');
    end
```

```
end


%--------------------End of Serial Comm Functions---------------------%

%--------------------Set Destination Node Addrress--------------------%
%------Accept User input for MSBytes------%
function destAddMSB_Callback(hObject, eventdata, handles)
% hObject    handle to destAddMSB (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of destAddMSB as text
%        str2double(get(hObject,'String')) returns contents of destAddMSB as a double
%check for compatible input address
if(length(get(hObject,'String'))==8)
    %save the address MSB in the workspace of the Object
    setappdata(hObject,'highbytes',get(hObject,'String'));
else
    msgbox('Incompatible Input. Please enter 8 Hex Characters','Input Error','Warn');
end



% --- Executes during object creation, after setting all properties.
function destAddMSB_CreateFcn(hObject, eventdata, handles)
% hObject    handle to destAddMSB (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


%------Accept User input for LSBytes------%
function destAddLSB_Callback(hObject, eventdata, handles)
% hObject    handle to destAddLSB (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of destAddLSB as text
%        str2double(get(hObject,'String')) returns contents of destAddLSB as a double
%check for compatible input address
if(length(get(hObject,'String'))==8)
    %save the address LSB in the workspace of the Object
    setappdata(hObject,'lowbytes',get(hObject,'String'));
else
    msgbox('Incompatible Input. Please enter 8 Hex Characters','Input Error','Warn');
```

```
end


% --- Executes during object creation, after setting all properties.
function destAddLSB_CreateFcn(hObject, eventdata, handles)
% hObject    handle to destAddLSB (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

%----Sets the user entered destination address----%
% --- Executes on button press in setDestAddr.
function setDestAddr_Callback(hObject, eventdata, handles)
% hObject    handle to setDestAddr (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
instrreset; %resets the serial devices
clc;        %clear command line
%load the serial port object
try
fid=load('serial.mat','s');
fopen(fid.s);   %open the serial port
set(fid.s,'terminator','CR'); %configure the terminating character in the communication
%form the required string to set the address in the Zigbee
msb=horzcat('ATDH',getappdata(handles.destAddMSB,'highbytes'));
lsb=horzcat('ATDL',getappdata(handles.destAddLSB,'lowbytes'));

fwrite(fid.s,'+++');
pause(1);
%fprintf(fscanf(fid.s));
if(fid.s.bytesavailable()>0)
    status=fscanf(fid.s);
    index=strfind(status,'OK');
    %msgbox(status,'Status','warn');
    if(isempty(index))
        set(handles.Status,'String','Failed to Enter Command Mode');
    else
        set(handles.Status,'String','Entering Command Mode');
        fprintf(fid.s,msb);
        pause(0.1);
        fprintf(fid.s,lsb);
        pause(0.1);
        fprintf(fid.s,'ATWR');
        pause(1);
        if(~isempty(strfind(fscanf(fid.s),'OK')))
            set(handles.Status,'String','Destination Address Changed Successfully');
        end
    end
end
catch err
   if(strcmp(err.identifier,'MATLAB:serial:fopen:opfailed')||strcmp(err.↙
identifier,'MATLAB:load:couldNotReadFile'))
        set(handles.Status,'String','Serial Port Not Open. Open the Serial Port.');
    end
end
%-------------------------Set Gain Parameters----------------------------%
function Kp_Callback(hObject, eventdata, handles)
% hObject    handle to Kp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Kp as text
%        str2double(get(hObject,'String')) returns contents of Kp as a double
val=get(hObject,'String');
index=strfind(val,'.');
if(isempty(index) && ~isnan(str2double(val))) %check for non integer input
    setappdata(hObject,'kp',str2double(val));
else
    msgbox('Incompatible Input. Please enter integer values.','Input Error','Warn');
end

% --- Executes during object creation, after setting all properties.
function Kp_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Kp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Ki_Callback(hObject, eventdata, handles)
% hObject    handle to Ki (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Ki as text
%        str2double(get(hObject,'String')) returns contents of Ki as a double
val=get(hObject,'String');
index=strfind(val,'.');
if(isempty(index) && ~isnan(str2double(val))) %check for non integer input
    setappdata(hObject,'ki',str2double(val));
else
```

```
    msgbox('Incompatible Input. Please enter integer values.','Input Error','Warn');
end



% --- Executes during object creation, after setting all properties.
function Ki_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Ki (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in setGains.
function setGains_Callback(hObject, eventdata, handles)
% hObject    handle to setGains (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
instrreset;
clc;
try
fid=load('serial.mat','s');
fopen(fid.s);   %open the serial port

%load the Ki and Kp values saved in their respective object workspace
Kp=getappdata(handles.Kp,'kp');
Ki=getappdata(handles.Ki,'ki');

Kp=round(Kp);
Ki=round(Ki);
    set(handles.Status,'String','Transmitting');
    %send the command
    fprintf(fid.s,'g');
    set(fid.s,'terminator','LF'); %configure the terminating character in the↙
communication
    %send Kp value
    fprintf(fid.s,num2str(Kp));
    pause(0.1);
    %send Ki value
    fprintf(fid.s,num2str(Ki));
    pause(1);
    %check the acknowledgement sent by the controller
    if(fid.s.bytesavailable()>0)
        status=fscanf(fid.s);
        index=strfind(status,'OK');
        %msgbox(status,'Status','warn');
        if(isempty(index))
            set(handles.Status,'String','Failed to set Parameters');
        else
            set(handles.Status,'String','Parameters Set');
        end
    else
        set(handles.Status,'String','Failed to Set Parameters');
    end
fclose(fid.s);
catch err
    if(strcmp(err.identifier,'MATLAB:serial:fopen:opfailed')||strcmp(err.↙
identifier,'MATLAB:load:couldNotReadFile'))
        set(handles.Status,'String','Serial Port Not Open. Open the Serial Port.');
    end
end
%--------------------------End of gain settings--------------------------%


%------------------------------Plot Functions----------------------------%
% --- Executes on button press in plot1.
function plot1_Callback(hObject, eventdata, handles)
% hObject    handle to plot1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%load all the data variables
data= load ('Data.mat','raw_data');

iterations=getappdata(handles.iterationVal,'itrVal');
iterations=str2double(iterations);
interval=getappdata(handles.readInterval,'interVal');
interval=str2double(interval)/1000;
temperature1=data.raw_data(1:iterations,1)/10;
endTime=interval*iterations;
%calculate the time axis
time=0:interval:endTime-1;
figure(1);
plot(time,temperature1);
title('Transient Response of the Peltier Module');
xlabel('Time(seconds)');
ylabel('Peltier Outlet Temperature(C)');

% --- Executes on button press in plot2.
function plot2_Callback(hObject, eventdata, handles)
% hObject    handle to plot2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data= load ('Data.mat','raw_data');
iterations=getappdata(handles.iterationVal,'itrVal');
iterations=str2double(iterations);
interval=getappdata(handles.readInterval,'interVal');
interval=str2double(interval)/1000;
temperature2=data.raw_data(1:iterations,2)/10;
```

```matlab
endTime=interval*iterations;
time=0:interval:endTime-1;

figure(2);
plot(time,temperature2);
title('Transient Response of the Heat Exchanger');
xlabel('Time(seconds)');
ylabel('Condensor Outlet Temperature(C)');


% --- Executes on button press in plotPower.
function plotPower_Callback(hObject, eventdata, handles)
% hObject    handle to plotPower (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data= load ('Data.mat','Power');
iterations=getappdata(handles.iterationVal,'itrVal');
iterations=str2double(iterations);

interval=getappdata(handles.readInterval,'interVal');
interval=str2double(interval)/1000;
endTime=interval*iterations;
time=0:interval:endTime-1;
figure(3);
plot(time,data.Power);
title('Power consumed by the Peltier Module');
xlabel('Time(seconds)');
ylabel('Power(Watts)');

%-------------------------End of Plot Functions---------------------------%

%---------------------------Menu Bar Functions----------------------------%

function file_Callback(hObject, eventdata, handles)
% hObject    handle to file (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% -----save the enter values to a file -----%
function saveSettings_Callback(hObject, eventdata, handles)
% hObject    handle to saveSettings (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA
iterations=getappdata(handles.iterationVal,'itrVal');
interval=getappdata(handles.readInterval,'interVal');
temperature=getappdata(handles.sliderVal,'sliderVal');
kp=getappdata(handles.Kp,'kp');
ki=getappdata(handles.Ki,'ki');
save('Parameters.mat','iterations','interval','temperature','kp','ki');
set(handles.Status,'String','Settings Saved');
```

```
% ----- Load the saved values from file -----%
function loadSettings_Callback(hObject, eventdata, handles)
% hObject    handle to loadSettings (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
load Parameters.mat;
setappdata(handles.iterationVal,'itrVal',iterations);
set(handles.iterationVal,'String',iterations);
setappdata(handles.readInterval,'interVal',interval);
set(handles.readInterval,'String',interval);
setappdata(handles.slider1,'slider1',temperature);
set(handles.slider1,'Value',str2double(temperature));
setappdata(handles.sliderVal,'sliderVal',temperature);
set(handles.sliderVal,'String',temperature);
setappdata(handles.Kp,'kp',kp);
set(handles.Kp,'String',num2str(kp));
setappdata(handles.Ki,'ki',ki);
set(handles.Ki,'String',num2str(ki));
set(handles.Status,'String','Settings Loaded');

% --------------------------------------------------------------------
function help_Callback(hObject, eventdata, handles)
% hObject    handle to help (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Topics_Callback(hObject, eventdata, handles)
% hObject    handle to Topics (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
open('Troubleshooting.pdf');

% --------------------------------------------------------------------
function about_Callback(hObject, eventdata, handles)
% hObject    handle to about (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
text = fileread('About.txt');
msgbox(text,'About','help');
% --------------------------------------------------------------------
%-----------------------End of Menu Bar Functions-------------------------%



function Power_Callback(hObject, eventdata, handles)
% hObject    handle to Power (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Power as text
%        str2double(get(hObject,'String')) returns contents of Power as a double


% --- Executes during object creation, after setting all properties.
function Power_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Power (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```