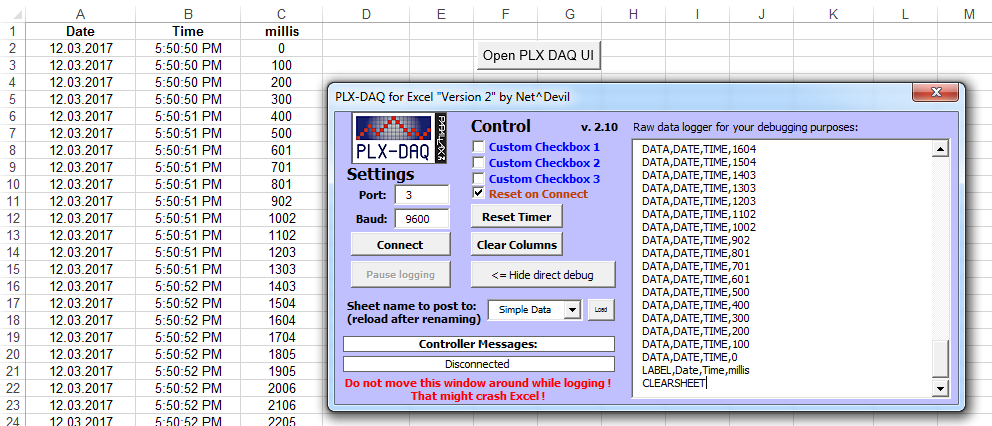
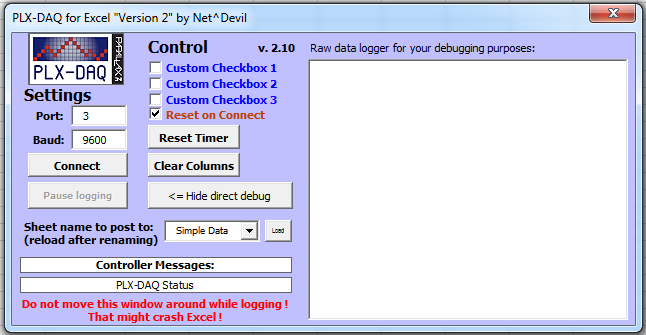
**Beginners Guide to PLX DAQ v2 by Net^Devil**

1. About the program and its use  
     
   PLX DAQ v2 is a program used to establish an easy communication between Microsoft Excel on a Windows Computer and any device that supports serial port protocol. It was intentionally written to allow communication between Arduino and Excel.  
     
   You can, for example, measure temperature data with your Arduino, send the results to Excel every 10 seconds, printed the data on a sheet and draw a graph with all information. All communication will be done by Serial.println commands just like the commands you use to send from Arduino to monitor in your Arduino IDE Serial Monitor.  
     
   The output of  
     
   void loop() {  
    Serial.println( (String) "DATA,DATE,TIME," + millis() );  
    delay(100);  
   }  
     
   looks like the following in Excel:  
   
2. How to use  
     
   The program uses two parts to work: the special Microsoft Excel Spreadsheet with the PLX DAQ v2 UI and commands plus any Arduino device that sends special commands for communication. The latest Excel Spreadsheet can be downloaded on the Arduino forum (Link: <http://forum.arduino.cc/index.php?topic=437398.msg3013761#msg3013761> ; please always download the latest version!), the Arduino code can be written by everyone themselves with below guideline.
3. The Excel UI part  
     
   After opening the Excel Spreadsheet please allow running the macro (yellow warning message at the top). Afterwards you can see the PLX DAQ v2 UI. In case not please click the button “Open PLX DAQ UI” on the “Simple Data” sheet.  
     
   The UI offers the following options:  
     
     
   1. Port: set to Arduino port (same as in Arduino IDE => Tools => Port, e.g., 3 for COM3
   2. Baud: set to the baud rate you run your Arduino on (e.g., 9600 if you are using Serial.begin(9600); in your Arduino code)
   3. Connect: connects to your Arduino and starts logging
   4. Pause logging/resume logging: when connected will pause the logging of data
   5. Reset Timer: will set the Timer to 0. The Timer can be used to measure how long Excel is already logging
   6. Clear Columns: will delete all logged data from the sheet. Won’t clear the labels of the columns
   7. Display/Hide direct debug: will show or hide the text field on the right. The Direct Debug Window can be used to manually monitor commands received by PLX DAQ v2 in Excel
   8. Sheet name to post to: this will list all sheets in the Excel workbook. Whatever sheet you select in the dropdown the logged data will be posted to it. This sheet will be referred to as the “ActiveSheet” throughout this document.  
      Note: after adding / deleting sheets please press the small “Load” button on the left side of the dropdown box for the sheet list to be updated
   9. Controller Messages: in the field below the most recent commands and status information will be shown. Most likely the information is changing way too fast for you to read, thus use the Direct Debug Window ☺
   10. Reset on Connect: the checkbox should be ticked at all time. If ticket the first command from Excel to Arduino will be to restart, thus your code starts from the beginning as well. This way you can have a fresh session. If you want to connect to your Arduino without restarting it just untick the box
   11. “Custom Checkbox 1/2/3”: these can be used to control your Arduino during run in any way you want. There are commands to label the Checkboxes by your Arduino and to query the state of the boxes. You could for example label one box “Measure humidity as well?” and check on demand in Excel if you want your Arduino to measure humidity with a second sensor next to only measure e.g., temperature. There are special commands Arduino can use to query the status of the checkboxes. More detail on these can be found below.
4. The Arduino part - overview  
     
   For PLX DAQ v2 to work correctly your Arduino needs to send specially formatted commands. All commands need to be send from Arduino to the PC using the Serial.println commands. These commands can include parameters, variables and functions to send to as well. These parameters need to be separated by commas. This can be done like this:  
     
   Serial.println( (String) "DATA,DATE,TIME," + millis() );  
     
   These commands can be split up into different categories:  
   1. Basic setup and communication:  
      Commands here are used to format the sheet to log to and to send data to the sheet
   2. Specific communication and manipulation:  
      Commands here are used to work with further parameters, jump on or between sheets, and using checkboxes for “communication” to your Arduino
   3. Excel workbook commands:  
      Commands here are used to control the logging process or even saving workbooks in window
   4. Miscellaneous commands:  
      Everything that is not really crucial or does not have any benefit (anymore)
5. The Arduino part - Basic setup and communication:  
   1. CLEARSHEET:  
      This command clears all data the ActiveSheet (including labels!). It should be the first command on every sketch.  
      Syntax: Serial.println(“CLEARSHEET”);
   2. CLEARDATA:  
      This commands clears only logged data on the ActiveSheet (starting at row 2)  
      Syntax: Serial.println(“CLEARDATA”);
   3. LABEL:  
      With this command you can set the labels for the top most row of the ActiveSheet  
      Syntax: Serial.println(“LABEL,1st Column,2nd Column,Third one!”);
   4. DATA:  
      This is the most basic and crucial command of PLX DAQ v2. It is used to send data from your Arduino to Excel and have it printed on the ActiveSheet. You can send anything you want but you should make sure you split the data up by commas and match the number of columns you defined with the LABEL command.  
      The reserved code words DATE, TIME and TIMER will be recognized by PLX DAQ and will be replaced with values.  
      DATE will be switched to the current Windows computer’s date (e.g., 12.03.2017)  
      TIME will be switched to the current Windows computer’s time (e.g., 18:17:42)  
      TIMER will be switched to the time the logging is already active (e.g., 1,365 seconds)
      1. Syntax 1 – this can be used to make you code more readable but works for static info only:  
         Serial.println("DATA,DATE,TIME”);
      2. Syntax 2 – this can be used to include function calls or variables in the same line:  
         Serial.println( (String) "DATA,DATE,TIME," + millis() );
      3. Syntax 3 – this is a way to present the code with rather more lines but with a better readability if you use many variables in one line. Please note that only the print command is used and println is only used at the end to send the fully build up string:  
         Serial.print("DATA,DATE,TIME,”);  
         Serial.print(myVariableA);  
         Serial.print(",”);  
         Serial.print(millis());  
         Serial.print(",”);  
         Serial.println(myVariableB);
6. The Arduino part - Specific communication and manipulation:  
   1. CELL,SET  
      By this command you can set the value of any cell in the Excel workbook with any value you want. This can either be done on the ActiveSheet or on any other sheet.  
      Syntax to set a value on the ActiveSheet:  
      Serial.println("CELL,SET,C9,MyValue”);   
      Syntax to set a value on a named sheet somewhere in the workbook:  
      Serial.println("CELL,SET, ONSHEET,AnySheet,C,9,MyValue”);
   2. CELL,GET  
      With this command values from the Excel sheet (either ActiveSheet or named sheet) can be queried and read by Arduino. For example you could put a value for the delay duration in cell J9, query it every loop iteration and control how long your Arduino should pause between each loop. The value in J9 can be changed by you during runtime at any time.  
      **Please note:** you need to read the value on your Arduino. This can be done by Integer or String. You should take care to only send integers when you are reading integers and vice versa.   
      Either use Serial.parseInt(); or Serial. readString();  
        
      With the following command you could read the value in J9 for the example above:  
      void loop() {  
       int myDelayValue;  
       Serial.println("CELL,GET,J9");  
       myDelayValue = Serial.parseInt();  
       Serial.println( (String) "Read value is: " + myDelayValue);  
       delay(myDelayValue);  
      }  
        
      To read a value from any other named sheet please use the following syntax:  
      Serial.println("CELL,GET,FROMHEET,AnySheet,C,9”);
   3. ROW  
      You can set the row on which data should be logged to manually. Thus after e.g., having logged 20 data sets you can reset the row counter to “2” and start from the beginning again. Or you can read the row that is currently logged to (please note you need a Serial.parseInt() command afterwards in Arduino)  
      Syntax: Serial.println("ROW,SET,2”);  
      Syntax: Serial.println("ROW,GET”); myRow = Serial.parseInt();
   4. CUSTOMBOX1 / CUSTOMBOX2 / CUSTOMBOX3  
      You can set a label to the checkboxes on the PLX DAQ UI, set the values to the checkboxes (ticked or not ticked) and read the value of the checkboxes (into bool).  
      Syntax for example:  
      Serial.println("CUSTOMBOX1,LABEL,Measure humidity as well?”);  
      Serial.println(“CUSTOMBOX1,GET”);  
      myBoolValue = Serial.parseInt();
   5. CLEARRANGE  
      Other than Cleardata or Clearsheet this command will only clear a certain range on the ActiveSheet. Data will be deleted in a rectangle from top left to bottom right.  
      Syntax: Serial.println(“CLEARRANGE,B,10,D,20”);
   6. RESETTIMER  
      It will reset the timer that is used to count the time PLX DAQ is already logging  
      Syntax: Serial.println(“RESETTIMER”);
7. The Arduino part - Excel workbook commands:  
   1. PAUSELOGGING / RESUMELOGGING / STOPLOGGING  
      Basically these commands explain themselves. Use to pause PLX DAQ to post logged data to the sheet. Of course PLX DAQ will still listen to incoming commands (to recognize RESUMELOGGING after PAUSELOGGING) but it won’t be printed to ActiveSheet. In case you want to see what information is received use the Direct Debug Window. STOPLOGGING will completely stop the logging process. No more data can be received afterwards! You have to restart the logging process manually by clicking the connect button!  
      Syntax: Serial.println(“PAUSELOGGING”);
   2. SAVEWORKBOOK  
      Will just save the workbook by its current name. This is useful if you are logging for a long time and want to save your data every now and then.  
      Syntax: Serial.println(“SAVEWORKBOOK”);
   3. SAVEWORKBOOKAS  
      Will save the workbook by any given name. The new workbook will be saved in the same folder as the currently open workbook (except if you include subfolders in the name to be saved).  
      Syntax: Serial.println(“SAVEWORKBOOKAS,MyNewWorkbookName”);  
      Syntax: Serial.println(“SAVEWORKBOOKAS,Subfolder\Workbookname”);
   4. FORCEEXCELQUIT  
      This is a heavy command! It will force quit Excel.  
      !! PLEASE NOTE TO SAVE YOUR WORKBOOK FIRST !!  
      Syntax: Serial.println(“FORCEEXCELQUIT”);
8. The Arduino part - Miscellaneous commands:  
   1. BEEP  
      Will simply make a beep noise in Excel. Good if you want to be notified by your Arduino e.g., after a certain threshold is past on a value you measure (room temperature or other)  
      Syntax: Serial.println(“BEEP”);
   2. MSG  
      Can be used to put a string on the Controller Message label on the PLX DAQ UI  
      Syntax: Serial.println(“MSG,Put your text here”);
   3. DONE  
      Will flush the Serial port on Excel side. Can be used to send data which might be in the buffer on Excel  
      Syntax: Serial.println(“DONE”);
9. Doing your own stuff:  
     
   Each and every one – including you (!!) – is allowed and should and can see the full source code of PLX DAQ v2, read it, learn from it and adjust it in any way you want or need. The source code can be viewed by pressing Alt+F11 in Excel. I deeply hope I managed to write the code as clear as possible. Generally, all you need to know is that in the sub “DataReady” you can add new commands. Simply copy & paste an old one, give it your name and go for it!  
     
   You could also use the CustomDevPoints that I included in the code. These functions are called at different times during the run. E.g., every time a new line is read from PLX DAQ (thus after each Serial.println();), after a new line with DATA is read (thus after each Serial.println(“DATA,…..”);) or after this data is added to the sheet.

Last but not least here is a fully working demo sketch that is using most of the above mentioned commands and should give you a fair overview on how to program your Arduino to communicate with Excel using PLX DAQ v2.

Enjoy !!

And thanks for reading and sharing.

Greetings

Jonathan Arndt (aka Net^Devil)

/\*

\* very basic sketch for PLX DAQ test

\* for new "Version 2" of PLX DAQ

\*/

int i = 0;

void setup() {

// open serial connection

Serial.begin(9600);

//Serial.println("CLEARDATA"); // clears starting at row 2

Serial.println("CLEARSHEET"); // clears starting at row 1

// define 5 columns named "Date", "Time", "Timer", "Counter" and "millis"

Serial.println("LABEL,Date,Time,Timer,Counter,millis");

// set the names for the 3 checkboxes

Serial.println("CUSTOMBOX1,LABEL,Stop logging at 250?");

Serial.println("CUSTOMBOX2,LABEL,Resume log at 350?");

Serial.println("CUSTOMBOX3,LABEL,Quit at 450?");

// check 2 of the 3 checkboxes (first two to true, third to false)

Serial.println("CUSTOMBOX1,SET,1");

Serial.println("CUSTOMBOX2,SET,1");

Serial.println("CUSTOMBOX3,SET,0");

}

void loop() {

// simple print out of number and millis

// output "DATA,DATE,TIME,TIMER,4711,13374"

Serial.print("DATA,DATE,TIME,TIMER,");

Serial.print(i++); Serial.print(",");

Serial.println(millis());

// clear some cells in Excel (rectangle range from B10 to D20)

if(i==100)

Serial.println("ClearRange,B,10,D,20");

// do a simple beep in Excel on PC and read value from named sheet

if(i==150)

{

Serial.println("BEEP");

//Serial.println("CELL,GET,E4"); ==> active sheet in Excel

Serial.println("CELL,GET,FROMSHEET,Simple Data,E,4"); //named sheet

int readvalue = Serial.parseInt();

// result displayed in Excel DirectDebugWindow to double check

Serial.print("Value of cell E4 is: ");

Serial.println(readvalue);

}

// check value of custombox1 on PLX DAQ in Excel and if

// checkbox is checked then send the command to pause logging

if(i==250)

{

Serial.println("CUSTOMBOX1,GET");

int stoplogging = Serial.parseInt();

// this information can be seen in the

// direct debug window on PLX DAQ in Excel

Serial.print("Value of stoplogging/checkbox is: ");

Serial.println(stoplogging);

if(stoplogging) {

Serial.println("PAUSELOGGING");

}

}

// same thing for resume logging

if(i==350)

{

Serial.println("CUSTOMBOX2,GET");

int resumelogging = Serial.parseInt();

if(resumelogging) {

Serial.println("RESUMELOGGING");

}

}

// post to specific cells on defaul sheet and named sheet

if(i==400)

{

// default sheet active in PLX DAQ Excel

Serial.println("CELL,SET,G10,400 test 1 string");

// named sheet available in PLX DAQ Excel

Serial.println("CELL,SET,ONSHEET,Simple Data,G,11,400 test 2”);

}

// and for forced quit of Excel with saving the file first

if(i==450)

{

Serial.println("CUSTOMBOX3,GET");

if(Serial.parseInt()) {

Serial.println("SAVEWORKBOOKAS,450-Lines-File");

Serial.println("FORCEEXCELQUIT");

}

}

}