# Beginners Guide to PLX DAQ v2 by Net^Devil

(Revision 1)

## Table of content



## 1. About the program and its use

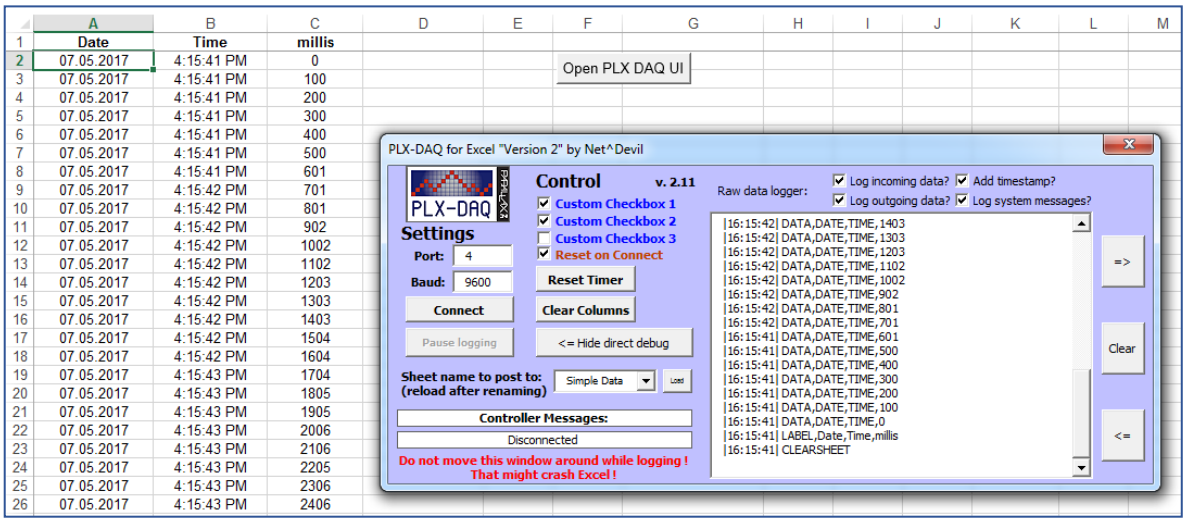
PLX DAQ v2 is a program used to establish an easy communication between Microsoft Excel on a Windows Computer and any device that supports serial port protocol. It was intentionally written to allow communication between Arduino and Excel.

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:




| | A | B | C |
|---|---|---|---|
| 1 | Date | Time | millis |
| 2 | 07.05.2017 | 4:15:41 PM | 0 |
| 3 | 07.05.2017 | 4:15:41 PM | 100 |
| 4 | 07.05.2017 | 4:15:41 PM | 200 |
| 5 | 07.05.2017 | 4:15:41 PM | 300 |
| 6 | 07.05.2017 | 4:15:41 PM | 400 |
| 7 | 07.05.2017 | 4:15:41 PM | 500 |
| 8 | 07.05.2017 | 4:15:41 PM | 601 |
| 9 | 07.05.2017 | 4:15:42 PM | 701 |
| 10 | 07.05.2017 | 4:15:42 PM | 801 |
| 11 | 07.05.2017 | 4:15:42 PM | 902 |
| 12 | 07.05.2017 | 4:15:42 PM | 1002 |
| 13 | 07.05.2017 | 4:15:42 PM | 1102 |
| 14 | 07.05.2017 | 4:15:42 PM | 1203 |
| 15 | 07.05.2017 | 4:15:42 PM | 1303 |
| 16 | 07.05.2017 | 4:15:42 PM | 1403 |
| 17 | 07.05.2017 | 4:15:42 PM | 1504 |
| 18 | 07.05.2017 | 4:15:42 PM | 1604 |
| 19 | 07.05.2017 | 4:15:43 PM | 1704 |
| 20 | 07.05.2017 | 4:15:43 PM | 1805 |
| 21 | 07.05.2017 | 4:15:43 PM | 1905 |
| 22 | 07.05.2017 | 4:15:43 PM | 2006 |
| 23 | 07.05.2017 | 4:15:43 PM | 2106 |
| 24 | 07.05.2017 | 4:15:43 PM | 2205 |
| 25 | 07.05.2017 | 4:15:43 PM | 2306 |
| 26 | 07.05.2017 | 4:15:43 PM | 2406 |

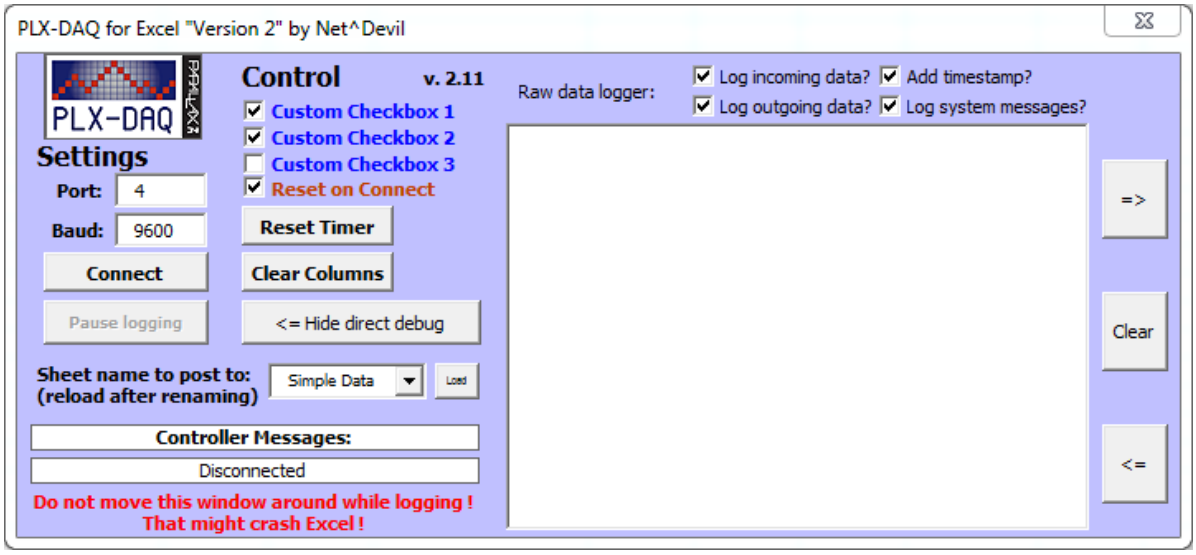
## 2. How to use

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:




a. Port: set to Arduino port (same as in Arduino IDE => Tools => Port, e.g., 4 for COM4
b. 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)
c. Connect: connects to your Arduino and starts logging
d. Pause logging/resume logging: when connected will pause the logging of data
e. Reset Timer: will set the Timer to 0. The Timer can be used to measure how long Excel is already logging
f. Clear Columns: will delete all logged data from the sheet. Won't clear the labels of the columns
g. 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
h. 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
i. 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 ☺
j. 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
k. 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.

l. Log incoming data? checkbox: information received from Arduino will be displayed in the direct debug window. Hint: disable if you experience performance issues
m. Log outgoing data? checkbox: information send to your Arduino will be displayed in the direct debug window. Hint: disable if you experience performance issues
n. Log system messages? checkbox: information from Excel will be display in the direct debug window (e.g., errors) . Hint: disable if you experience performance issues
o. Add timestamp? checkbox: will add a timestamp to every information logged in the direct debug window. This is handy for debugging.
p. =>: will increase the size of the direct debug window to a certain maximum
q. <=: will decrease the size of the direct debug window to a certain minimum
r. Clear: will clear all information in the direct debug window

# 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:

a. Basic setup and communication:
Commands here are used to format the sheet to log to and to send data to the sheet

b. 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

c. Excel workbook commands:
Commands here are used to control the logging process or even saving workbooks in window

d. Miscellaneous commands:
Everything that is not really crucial or does not have any benefit (anymore)

# 5. The Arduino part - Basic setup and communication

a. CLEARSHEET:
This command clears all data the ActiveSheet (including labels!). It should be the first command on every sketch.
Syntax: `Serial.println(“CLEARSHEET”);`

b. CLEARDATA:
This commands clears only logged data on the ActiveSheet (starting at row 2)
Syntax: `Serial.println("CLEARDATA");`

c. 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!");`

d. 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)

You can add the keyword `AUTOSCROLL_20` to your string. This information will make PLX DAQ to automatically scroll your Excel window down with every new line of data that is received. The number (e.g., `20`) indicates how many lines should be shown above the latest line that is scrolled to automatically.

i. Syntax 1 – this can be used to make you code more readable but works for static info only:
```
Serial.println("DATA,DATE,TIME");
```
ii. Syntax 2 – this can be used to include function calls or variables in the same line:
```
Serial.println( (String) "DATA,DATE,TIME," + millis() );
```
iii. 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

a. 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”);
```

**PLEASE NOTE !!** ➔ there might occur problems at higher baud rates with the CELL,SET command. It might happen that values are not transferred correctly – instead “0” might be transmitted. In these cases please either decrease your baud rate or insert small `delay` before to your CELL,SET command (e.g., `delay(3);`). Typically values between 3 and 100 should do the trick. In case you make excessive use of CELL,SET or CELL,GET please increase the delay value even further in case you continue receiving wrong values.

b. 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.

Use `Serial.readStringUntil(10);`
or `Serial.readStringUntil(10).toInt();`
or `Serial.readStringUntil(10).toFloat();`

The number `10` is important as ASCII char 10 represents the end of a line to read.

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.readStringUntil(10).toInt();
  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”);
```

**PLEASE NOTE !!** ➔ there might occur problems at higher baud rates with the CELL,GET command. It might happen that values are not transferred correctly – instead “0” might be transmitted. In these cases please either decrease your baud rate or insert small `delay` before to your CELL,GET command (e.g., `delay(3);`). Typically values between 3 and 100 should do the trick. In case you make excessive use of CELL,SET or CELL,GET please increase the delay value even further in case you

continue receiving wrong values.

c. 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 `Serial.readStringUntil(10).toInt()` command afterwards in Arduino)
Syntax:
```
Serial.println("ROW,SET,2");
```
Syntax:
```
Serial.println("ROW,GET");
myRow = Serial.readStringUntil(10).toInt();
```

d. 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.readStringUntil(10).toInt();
```

e. 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");`

f. 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

a. 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");`

b. 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");`

c. 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");`

d. 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

a. 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");`

b. 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");`

c. 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");`

d. GETRANDOM

Will return a random number from Excel to Arduino. This is useful because Arduino can't create true random numbers. Arduino's random function requires prior initialization by randomSeed(value). If value is not unique (e.g., user input) the number sequence return by Arduino's random will always be the same. To get a unique value for randomSeed's value you can therefore query Excel.

Syntax:

```
Serial.println("GETRANDOM,-31313,32323");
int rndseed = Serial.readStringUntil(10).toInt();
randomSeed(rndseed);
Serial.println( (String) "1st number: " + random(0, 50));
Serial.println( (String) "2nd number: " + random(0, 50));
Serial.println( (String) "3rd number: " + random(0, 50));
```

## 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.

## 10. Upgrading PLX DAQ v2 Excel versions

If you want to use the new version of PLX DAQ v2 but want to keep all your configured sheets in old Excel version of PLX DAQ try the following steps to copy all sheets, references and formulas easily to the new version.
**PLEASE NOTE !!** ➔ it is NOT possible to copy the sheet "Simple Data" from the old version to the new version. Excel would automatically copy the old version of PLX DAQ as well ….

a. Download new PLX DAQ Excel and open, open your Excel as well.
b. In your old Excel select first sheet, then hold shift key and select last sheet, right click first sheet and select "Move or Copy"
c. Select new PLX DAQ Excel in the dropdown, select "move to end", select "Create a copy" and press OK.
d. It should copy all selected sheets to the new version.

## 11. Full blown demo

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)

```
/*
 * Demo sketch for new "PLX DAQ v2"
 * including most all common commands to be used
 */
```

```
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,AUTOSCROLL_20"
    Serial.println( (String) “DATA,DATE,TIME,TIMER,” + i++ + “,” +
                        millis() + “,AUTOSCROLL_20”);

    // 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
      if(i==150)
        Serial.println("BEEP");

    // read a value (in this case integer) from a sheet by name
      if(i==200)
        //Serial.println("CELL,GET,E4"); ==> active sheet in Excel
        Serial.println("CELL,GET,FROMSHEET,Simple Data,E,4"); //named sheet
        int readvalue = Serial.readStringUntil(10).toInt();
        // result displayed in Excel DirectDebugWindow to double check
        Serial.println( (String) "Value of cell E4 is: “ + 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.readStringUntil(10).toInt();
    // 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");
    }
  }

// get a true random number from the computer
  if(i==300)
  {
    Serial.println("GETRANDOM,-4321,12345"); // between -4321 to 12345
    int rndseed = Serial.readStringUntil(10).toInt();
    Serial.println( (String) "Got random value '" + rndseed + "'" );
    // Note: this information is not posted to Excel: "DATA" is missing
    // instead this information can be seen in the direct debug window
  }

// and now resume logging
  if(i==350)
  {
    Serial.println("CUSTOMBOX2,GET");
    int resumelogging = Serial.readStringUntil(10).toInt();
    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.readStringUntil(10).toInt()) {
          Serial.println("SAVEWORKBOOKAS,450-Lines-File");
          Serial.println("FORCEEXCELQUIT");
        }
        else
          Serial.println("No forced Excel quit requested!");
      }
}
```