

Multi-threaded Python Pin Chart Instructions

Table of Contents

Table of Contents.....	1
Objective	2
Published Date	2
Contact Information.....	2
What's New?	2
No Need for RSLinx Classic.....	2
Python Version.....	2
Speed	2
Flexibility	2
Installation	2
Step 0: Remove Previously Installed Software	2
Step 1: Install Required Software	3
Step 2: Python Scripts	3
Excel Pin Chart Use	6
Excel Data Sheet Rules & Limitations	6
Importing / Exporting: Depreciated.....	6
PLC Operation Screenshot	6
Uploading.....	6
Downloading.....	7
Excel Sheet Formatting	7
Main Program Sheet	7
Pinchart Sheet(s).....	7
Further Information	9
Updating Python Code.....	9
64-Bit Office	10
Upcoming Features.....	10
Testing-to-date and bug reporting	10
Contributing	10

Objective

To describe and guide a user through the installation / setup process to get the pin chart utility to work.

Published Date

December 18th, 2020 By Chris Panici

Updated – January 25th, 2021 by Chris Panici

Updated – April 1st, 2021 by Chris Panici (added section on merged cells)

Contact Information

Chris Panici

(920) 540-6344

cpanici@gotoccs.com

What's New?

No Need for RSLinx Classic

The new system utilizes a pure python library called pycomm3. This library serializes and deserializes raw socket packets natively without the need for an OPC interface.

Python Version

The new system supports python 3.x (64-bit). Python 2.x (32-bit) is no longer supported and will give errors if the script tries to run with that version.

Speed

The speed of the PLC operations has increased dramatically. This firstly is due to the communication change and the elimination of the OPC server requirement, and secondly to some excel interface code optimization.

Flexibility

Some data types and configuration variables got added to make the spreadsheets more flexible. The added configuration variables allow the user to adjust the number of columns in between PLC DATA columns.

Installation

Step 0: Remove Previously Installed Software

The following packages need to get removed if the previous version of the pin chart was installed.

1. Python 2.7 (32-bit) – Uninstall this package using Add/Remove programs if it is not being used for anything else. This package can be left on the users computer, however, the user should make sure that all of the pinning chart references are moved to the new version.

2. OpenOPC – Uninstall this package using Add/Remove programs

Step 1: Install Required Software

The first step is to install the required software. It is recommended to install them in-order listed in the steps.

These software packages can be found on the CCS Public drive and are in the following folder “1.0 Install Packages”. There are three software packages that are required.

Software Package 1 - Python 3.9 (64-bit) <https://www.python.org/downloads>

If you have other versions of python that is ok, you'll just have to be sure that when you run the script it is pointed to the correct python path.

- Run the msi file and follow the wizard.
- **IMPORTANT:** Install for ALL USERS. There is a check box for this in the wizard and checking it will allow it to install to C:\Program Files\Python39\ directory.
- Install it at the default location C:\Program Files\Python39\

Software Package 2 - Git bash (if not installed already)

<https://github.com/git-for-windows/git/releases/download/v2.29.2.windows.3/Git-2.29.2.3-64-bit.exe>

This pulls the updated code from Github. This is somewhat optional however you can always grab the updated code with this tool.

- Run the EXE file and follow the wizard to install it to the computer.

If you do not install this package, you will have to download and copy the script files manually from the github repository. It is recommended that this package is installed.

Step 2: Python Scripts

The second step is to load the python scripts onto the local machine through the Github repository.

The scripts are required for the pinchart to work, and it is recommended that they be placed in the following location C:\CCS\. The location of the scripts can change if desired, but for this tutorial it is assumed that they are in the recommended folder C:\CCS\.

- 2.1 Uninstall the previous script (if it was installed) by removing it from the C:\CCS\MTPyPinchart. This is an optional step as the new script will go in a new folder.
- 2.2 To download the python scripts from Github, navigate to the directory that the scripts will be installed in. In this case, C:\CCS\.
- 2.3 Launch a Git Bash session by right-clicking within the folder and clicking on “Git Bash Here” as seen below in Figure 1.

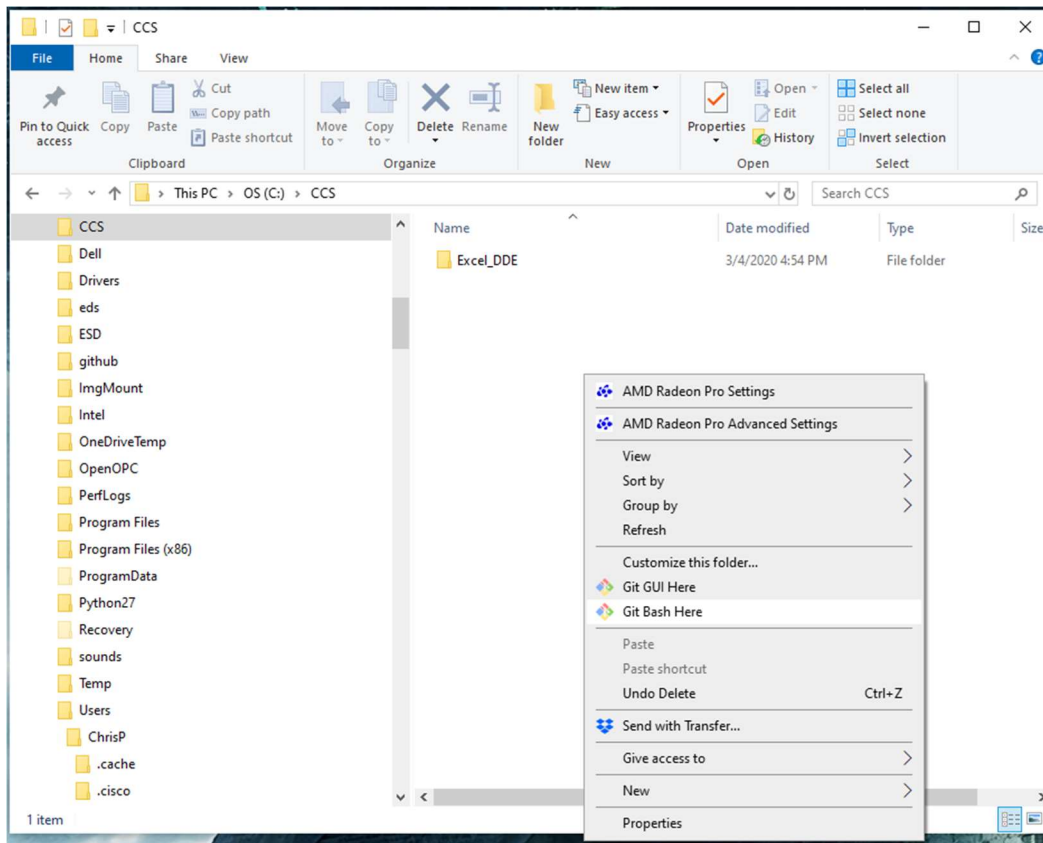


Figure 1: Launching a Git Bash

2.4 Once the bash session comes up, run the following command:

git clone https://github.com/ccscpanici/MTPyPinchart

*****NOTE: DO NOT REPLACE ANY PART OF THE HTML LINK***

Github may prompt for login information; if that happens reach out to cpanici@gotoccs.com for the credentials:

The output that comes back should be the same as seen in Figure 2 below. The following folder will be created: C:\CCS\MTPyPinchart.

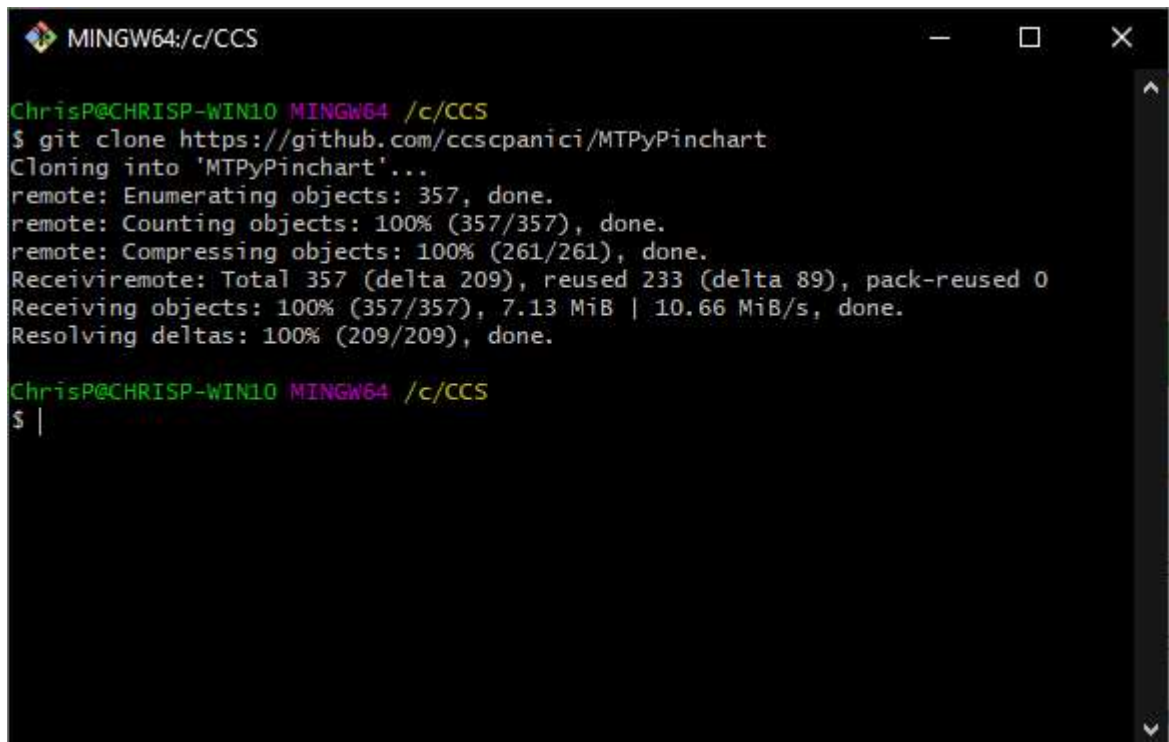
A screenshot of a Git Bash terminal window. The title bar at the top reads 'MINGW64:/c/CCS'. The terminal shows a user named 'ChrisP@CHRISP-WIN10' in a 'MINGW64' environment at the '/c/CCS' directory. The user has executed the command '\$ git clone https://github.com/ccscpanici/MTPyPinchart'. The output shows the cloning process: 'Cloning into 'MTPyPinchart'...', 'remote: Enumerating objects: 357, done.', 'remote: Counting objects: 100% (357/357), done.', 'remote: Compressing objects: 100% (261/261), done.', 'Receivremote: Total 357 (delta 209), reused 233 (delta 89), pack-reused 0', 'Receiving objects: 100% (357/357), 7.13 MiB | 10.66 MiB/s, done.', and 'Resolving deltas: 100% (209/209), done.'. The prompt '\$ |' is visible at the bottom.

Figure 2: Git Bash Output

- 2.5 Launch a Windows command prompt and navigate to the script folder by typing in the following command

```
cd C:\CCS\MTPyPinchart
```

- 2.6 Run the following command to install the numpy binary wheel.

```
"C:\Program Files\Python3.9\Scripts\pip.exe" install numpy-1.19.4-cp39-cp39-win_amd64.whl
```

- 2.7 Run the following command in the command prompt to install the python library dependencies. For this step, the computer must be connected to the internet.

```
"C:\Program Files\Python39\Scripts\pip.exe" install -r requirements.txt
```

There should not be any errors that prevent things for being installed. If there is a message at the end of the command with the list of packages that got installed than it should work. Some errors are "normal" as long as it says that the modules were installed on the last couple lines of the output.

- 2.8 The script should come with a sample excel file to get started with. It will be located in the same directory that the script is in: i.e. C:\CCS\MTPyPinchart

At this point everything is installed and ready to use the new style pin charts. The proceeding sections is information on how to use the spreadsheet and what the system looks for.

sheets. The upload sheet(s) button does a similar function, but for only the specified sheets. In Figure 4, there is only one sheet specified in the sheets list cell to upload or download. A user can specify additional sheets by adding them as a comma separated list. "PINCHART-PROC, PINCHART-CIP, PINCHART-SHORTWASH" would be an acceptable comma separated list. The sheet list cell is used for BOTH the uploading and downloading of specific sheets.

Downloading

A download operation reads the data inside the excel workbook and writes the data into the controller. Any data that was in the controller tags will get overwritten with the data that is in the workbook.

There are two buttons for downloading: The "Download" button, which downloads all of the sheets, and the "Download Sheet(s)" button which downloads the sheets listed in the sheet list cell.

When the download or export button is clicked, the pinchart downloads all the pinchart sheets in the workbook. The system acts much like the upload / import process in that a command window pops up and gives the user information about what the system is doing, which is normal operation. Any errors in the process should be reported. In general, if an error occurs it is usually before it downloads anything. Another common error is for the OPC connection to be broken, meaning that the system can't access the tag inside the controller.

Excel Sheet Formatting

There are certain formatting and flag placements inside the excel pinchart that the user should be aware of. Use the following sections for reference and to troubleshoot any formatting errors.

Main Program Sheet

This is the sheet that holds the configuration data and calls the scripts. It also does some "error checking" regarding files and folders.

IP Address – This is the IP address of the controller that the user needs to transact with

Slot Number – This is the slot number of chassis. For compact logix, or if the connection used is going directly to the processor, the user should enter in a 0. If the connection used involves an intermediary communication card (i.e. 1756-EN2T) the ip address would point to the communication card, then the slot number would be the slot of the processor.

Pinchart Sheet(s)

Pinchart Formatting

There is some formatting within the pinchart workbook that needs to be in place for the pinchart to work correctly. The different items are described in the below paragraphs.

Pinchart Sheet Name

To be able to upload or download a pinchart sheet it must have the test "pinchart" in the name of it and is not case sensitive. For example, "CIP A PINCHART" will download but "CIP A" will not. It is not case sensitive so "Cheese Recipe-1 Pinchart" will upload and download.

Header Row Flag

\$HeaderRow\$ - This is the header row flag. This flag must be put on the header row, so the system knows where to find the DATA column flags. The data flags are the following "INT DATA", "DINT DATA",

“FLOAT DATA”, and “STRING DATA”. If no columns within the row marked as the header row exists, then no data will be transferred during upload / download.

Data Type / Data Flags

Within the pinchart sheets there are some flags to be aware of. Any data that needs to be uploaded / downloaded must have the following inside the header column. The header row is marked with the \$HeaderRow\$ flag. The below items are the data flags.

FLOAT DATA

STRING DATA

DINT DATA

DINT-32 DATA - Must also have 32-columns to the left of this. When this data type is uploaded it overwrites data in the columns to the left/right of it based on the configuration data offset value.

INT DATA

INT-16 DATA - Must also have 16-columns to the left of this. When this data type is uploaded it overwrites data in the columns to the left/right of it based on the configuration data offset value.

The columns without these flags **DO NOT** get downloaded. When the INT and DINT data types are uploaded, the system fills in the “bit level” data to the left of the INT DATA or DINT DATA column. These DATA columns are formula columns that calculate the value of the bit-wise data.

Columns can be hidden and will not affect the functionality of the upload/download.

Configuration Data

On the main program sheet, there is some configuration data fields and values; they are outlined and described below:

\$Config\$	Key	Value	Description						
	ADDRESS OFFSET		1 Column offset from data value						
	DATA ROW OFFSET		2 Row offset from \$HeaderRow\$ Flag						
	DINT-32 BITS OFFSET		-32 Column offset from data value. Column should be MSB						
	INT-16 BITS OFFSET		-16 Column offset from data value. Column should be MSB						
	TITLE ROW OFFSET		-1 RESERVED						

Figure 5: Configuration Data

A sample configuration data can be seen in Figure 5 above. Keep in mind, these numbers are relative to the \$HeaderRow\$ or data flags used in the data sheets (i.e. DINT DATA, STRING DATA...). Positive number values for these fields are to the right / down, negative numbers are left/up, relatively.

ADDRESS OFFSET – The offset where the system will look for the PLC word address relative to the associated value data column. In the example above the PLC word address column would be one (1) column to the right of the associated data value column.

DATA ROW OFFSET – The row in which the actual data starts. In the example above the data starts two (2) rows down from the \$HeaderRow\$ flag. This makes it possible to put additional non uploaded/downloaded data between the header row and actual PLC data.

DINT-32 BITS OFFSET – The column in which the system will begin filling out the ones and zeros to compose the word. This makes it possible to put additional non uploaded/downloaded data between the value column and the “bits” data columns. In the example above, the system will overwrite the data 32-columns to the left of the value column.

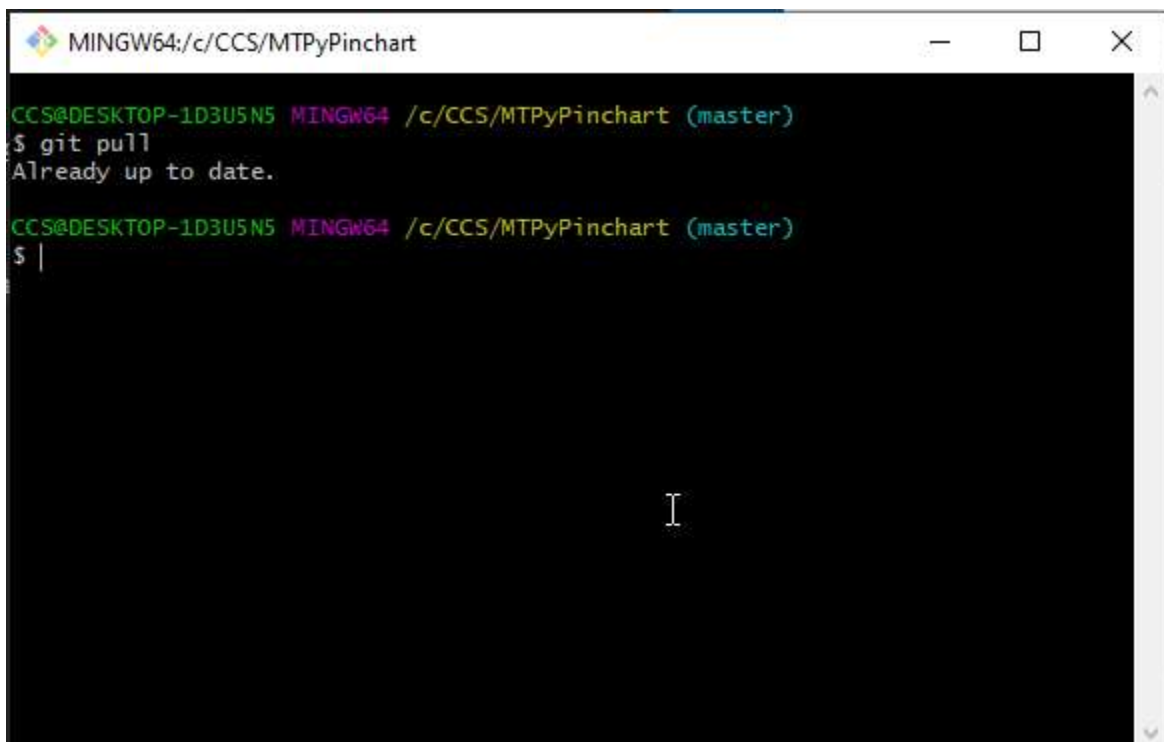
INT-16 BITS OFFSET – This field works much like DINT-32 BITS OFFSET, but instead will write in 16-bits instead of 32. In the above example, the system will upload and write in sixteen (16) columns to the left of the associated data value column.

Further Information

Updating Python Code

If there is an update to the python code and the user wants to install the new code the following git bash command should be run within the script folder as shown in Figure 6 below. If you need the login credentials to perform this action, they will be the same as in step #2. Please note that the folder should be C:\CCS\MTPyPinchart, then run the following command:

git pull

A screenshot of a MINGW64 terminal window titled "MINGW64:/c/CCS/MTPyPinchart". The terminal shows the following text: "CCS@DESKTOP-1D3U5N5 MINGW64 /c/CCS/MTPyPinchart (master)", "\$ git pull", "Already up to date.", "CCS@DESKTOP-1D3U5N5 MINGW64 /c/CCS/MTPyPinchart (master)", and "\$ |". The cursor is positioned at the end of the last line.

```
MINGW64:/c/CCS/MTPyPinchart
CCS@DESKTOP-1D3U5N5 MINGW64 /c/CCS/MTPyPinchart (master)
$ git pull
Already up to date.
CCS@DESKTOP-1D3U5N5 MINGW64 /c/CCS/MTPyPinchart (master)
$ |
```

Figure 6: Update the Code

64-Bit Office

The pin chart tool will work on 32-bit or 64-bit office. No changes to the VBA code behind the scenes need to change and the python script runs the same code; this is a CPU/OS agnostic script.

Upcoming Features

1. Formatting macros
2. Logging – log file for system instead of leaving command prompt open.

Testing-to-date and bug reporting

The new system has been tested on a limited number of scenarios and only controllogix controllers. The code runs the same regardless of PLC but has only been tested on a controllogix processor. The file import/export has been tested on a limited number of SLC programs.

If a bug is discovered the procedure would be to snap a picture of the command output and send it to me (cpanici@gotoccs.com). I can review the exception and work to resolve it.

Contributing

Anyone can contribute to fix bugs. If you would like to be involved let me know and I can get you set up.