A picture containing text

Description automatically generated

Contents

[Overview 3](#_Toc125979011)

[Mapper Windows 4](#_Toc125979012)

[TDC Tags 5](#_Toc125979013)

[Modules 7](#_Toc125979014)

[Cross Communications 8](#_Toc125979015)

[CL - All 10](#_Toc125979016)

[CL On Tag 12](#_Toc125979017)

[Connections 13](#_Toc125979018)

[File References 14](#_Toc125979019)

[Tags On Map 15](#_Toc125979020)

[TDC Parameters 16](#_Toc125979021)

[Tags In CL 17](#_Toc125979022)

[File 18](#_Toc125979023)

[Connect 18](#_Toc125979024)

[Export 19](#_Toc125979025)

[Import 20](#_Toc125979026)

[TDC Map 21](#_Toc125979027)

[Modules 22](#_Toc125979028)

[Reducing Clutter 24](#_Toc125979029)

[Drawing settings 27](#_Toc125979030)

[Pan & Zoom 28](#_Toc125979031)

[Save Map 29](#_Toc125979032)

[Print Map 30](#_Toc125979033)

[Function Block Colors 31](#_Toc125979034)

[Function Block Shapes 33](#_Toc125979035)

[CL 34](#_Toc125979036)

[Multiple CL 34](#_Toc125979037)

# Overview

Mapper is a program that is similar to D3K. Where D3K’s intended purpose is to maintain TDC systems, Mapper’s intended purpose is to migrate TDC systems. Mapper shows point to point connections, maps logic blocks, maps JS files (TDC 2000), displays CL in an IDE format, formats TDC2000/3000 CL to show points that are not immediately obvious (e.g. add tagnames to $REGCTL CDS, packages, indirect hardware addresses), and much more.

The TDC Parser (located at 10.213.10.134) is used to generate the database for mapper. Copy all the xxx.DB files into the same folder of Mapper.exe.

# Mapper Windows

The mapper program is segregated into many smaller windows. These windows are hidden, unless pinned, on the edges of the app (top, bottom, left, right) and appear as tabs.

## TDC Tags

A screenshot of a computer

Description automatically generated

Graphical user interface, table

Description automatically generated

The “TDC Tags” window is where you can search for tags that were parsed from the EB files. When you click on a row in the table, it will pull up a diagram under the “Map” tab which displays all the interconnected nodes attached to this point.

There are several columns that you can filter on in this table which can be seen in the above screenshot.

* **Name**: this is the tag name typically defined in the EB files. There are several exceptions listed below about “BOX (Internal)” and “LOGICHG”.
* **PointType**: this is the type of tag defined in the EB files
* **Network**: this is the physical location the tag resides
* **PVALGID**: this is the parameter that defines the PV algorithm for TDC3000 points (REGPVNIM and REGAM)
* **CTLALGID**: this is the parameter that defines the PV algorithm for TDC3000 points (REGCLNIM and REGAM)
* **ALGIDDAC**: this is the parameter that defines the PV/OP algorithm for TDC2000 points (REGHG)

There are 2 “PointType” that can be added to this table that aren’t real TDC point types defined in EB files. Those are “BOX (Internal)” and “LOGICHG”.

A BOX point can be thought of as a container that holds potentially hundreds of tags (e.g. FLAGS, NUMERICS, TIMERS, etc). Tags in mapper that are listed as “BOX (Internal)” are references that are used in CL or logic connections. A “BOX (Internal)” tag is only created if the parser finds that box reference being used in logic or CL.s

In TDC2000, you could not define logic blocks in EB files but could create logic blocks in JS files. A “LOGICHG” tag is created when there are JS files that have been parsed. These are similar to LOGICNIM blocks but are the HG equivalent.

## Modules

A screenshot of a computer

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

This a list of all the TDC Tag containers. See ***Modules*** for more information on how to create/delete/modify. The TDC parser automatically creates many modules for you, but they are not perfect and will need to be looked at.

## Cross Communications

A screenshot of a computer

Description automatically generated

Graphical user interface, application

Description automatically generated

This is a useful table if the entire LCN is not being migrated, in particular, it’s good for quickly identifying AM tags that should and shouldn’t be migrated. The table lists all the parameter connections (i.e. not CL connections) between 2 tags that occur on different networks. Clicking on a row will pull up a map of the tags.

Consider the following scenario based on the screenshot above.

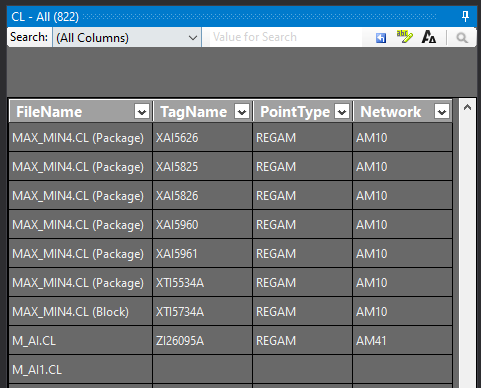
* All of UCN6 is being migrated,
* AM points are being migrated on a case-by-case basis.
* HWY1 is not being migrated

You can quickly filter “in scope” tags based off of “Source Network”. From that list, you could further filter “Target Network” to see all AM tags. Assuming there are no other connections to these tags or CL involved, you can quickly see that “Target Tag” FD4462 should probably be “in scope” and FY4116A should probably not be in scope

## CL - All

A screenshot of a computer

Description automatically generated



This window displays all the parsed CL files. Clicking on a cell will pull up a new tab or jump to an existing tab that displays the CL in a formatted manner. This table allows you to quickly find “in scope” and “out of scope” CL by filtering on “Network”. The columns are defined as follows:

* **FileName**: this is the name of the CL file
* **TagName**: the CL is linked to this tag i.e. the CL program runs inside of this tag
* **PointType**: the type of tag that the CL is linked to
* **Network**: the physical location of the tag that the CL is linked to

There is a row in this table for every CL file. In addition to this, if a CL program is a PACKAGE, each tag the CL file links to is also listed in this table. The parser searches for parameters in the EB file named PKGNAME(i). The value of the PKGNAME parameter is a CL file. This allows the parser to figure out how CL is linked to TDC tags when they are not explicitly defined. You will see “(Package”) appended to the end of “FileName” if the PKGNAME parameter is found on a tag. Sometimes the PKGNAME parameter is not defined in the EB exports for some reason (perhaps a bad export, but I’m not sure). In this situation, if the CLSLOTS.ZB file was defined during parsing, the parser will check this file to see how tags are linked to CL. If the parser discovers a link in this file and not on the PKGNAME parameter, then you will see “(Block”) appended to the end of “FileName”.

CL files that have no tag defined likely do not need to be explicitly migrated. These are often parameter lists or unlinked (broken) CL.

## CL On Tag

A screenshot of a computer

Description automatically generated

Graphical user interface, application

Description automatically generated

This table displays the CL that references the selected tag. The selected tag is shown in parenthesis and appended to the end of “Name” column e.g. (FI24010G) is selected tag in screenshot above. Whenever your selected tag changes (click a tag on the “Map”, click a tagname in a CL file, etc) then this table will update. Clicking on a cell in will pull up the CL file.

## Connections

A screenshot of a computer

Description automatically generated

Graphical user interface, text, application

Description automatically generated

This table displays the parameter connections that references the selected tag. Whenever your selected tag changes (click a tag on the “Map”, click a tagname in a CL file, etc) then this table will update. The “Connection” column is connected to the “Parameter” column which can be visually seen via the “Map”.

## File References

A screenshot of a computer

Description automatically generated

Graphical user interface, table

Description automatically generated

This table displays miscellaneous files that reference the selected tag. Whenever your selected tag changes (click a tag on the “Map”, click a tagname in a CL file, etc) then this table will update. This data is only available if a D3K.mdb file was included during parsing. The table displays all the Free format log (FFL), Group, and Schematic (graphic) references.

## Tags On Map

A screenshot of a computer

Description automatically generated

Graphical user interface, application, table

Description automatically generated

This table shows all the tags that are in the current “Map”. Clicking on a cell zooms into the node of the “Map”. Use the scroll wheel to zoom back out to a reasonable view from there.

## TDC Parameters

A screenshot of a computer

Description automatically generated

A screenshot of a phone

Description automatically generated with medium confidence

This table displays all the parameters defined in the EB files of the selected tag. Whenever your selected tag changes (click a tag on the “Map”, click a tagname in a CL file, etc) then this table will update.

## Tags In CL

A screenshot of a computer

Description automatically generated

Table

Description automatically generated

This table shows all the tags that are defined in the open CL file.

# File

There are several options under “File” in the toolbar such as “Connect”, “Import”, “Export”

## Connect

Graphical user interface, text, application

Description automatically generated

You can have several DB files stored in different locations that Mapper can open.

## Export

Graphical user interface, application

Description automatically generated

You can connect any application that is capable of opening a SQLITE DB file to the “.DB” file of mapper. However, you can also use the “Export” button to easily export the data to excel. Checking multiple boxes will add sheets to the spreadsheet.

Table

Description automatically generated

## Import

Graphical user interface, application

Description automatically generated

You can bulk edit changes from an exported file and then re-import the spreadsheet. There is a column called “Delete” that you can set to true/false if you want to remove something. If you want to add a row, just fill in all the necessary columns. This is useful to bulk edit tags in modules.

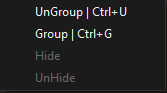
# TDC Map

A screenshot of a computer

Description automatically generated

## Modules

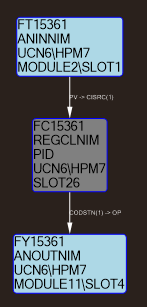


The TDC parser automatically creates modules and assigns tags to these modules. However, you may need to tweak these modules. You use the “Group” and “Ungroup” button to do this. You can also use the context menu by right clicking anywhere on the map or the shortcuts listed on the context menu.  


Select the tags that you want to ungroup (blue outline indicates tag is selected) then press “Ungroup”

A picture containing text, monitor

Description automatically generated

The result looks like this  


You can then select tags however you want and click “Group”, fill in the name you’d like for the module then click “Ok”. You could for instance, change the module to this  


Out of all the selected tags, if 1 of them has a defined module, all the tags will be merged into that same module. If none of the tags have a module or if there are tags in multiple modules, then you will be prompted for a new module name.

## Reducing Clutter



The “Max Nodes” textbox and “Direct Connections Only” checkbox control how many nodes are displayed on the map. Often times you don’t need to see the entire map because it just makes it more difficult to digest the information you can see.

Changing the value in “Max Nodes” will start from the selected tag and then do a breadth width first (BFW) search to find the closest nodes. See below screenshot for example.  
Chart, radar chart

Description automatically generated

“Direct Connections Only” will only show tags that are directly connected to the selected tag via parameter connections.

Tags that are hiding connections have a red outline.  
A picture containing text, sign, different

Description automatically generated

There are several key differences in how “Mapper” creates a diagram of interconnected nodes compared to DOC3000. The main reason for the discrepancies is a “Map” can become saturated with tags that often don’t affect one another. Below the discrepancies you will find 2 screenshots that are mapping the same tag.

* Mapper only displays a tag once whereas D3K could display the same tag numerous times. See D3K screenshot below, all green tags are duplicates.
* The “LOGICNIM” can be thought of as a container for many more tags. Mapper automatically breaks apart “LOGICNIM” tags (“Map Logic Block” in D3K) into separate logic gates i.e. you will never see a “LOGICNIM” tag on a map, however, you will see the logic gate on the map (see tan tags on Mapper screenshot.
* CL isn’t displayed as nodes on a map in Mapper but is displayed in D3K. D3K displays CL files with oval nodes in a map. Mapper indicates a tag has CL on it by using an oval shape. It’s a small difference but the outcome is a much less cluttered map

Diagram

Description automatically generated

A picture containing timeline

Description automatically generated

## Drawing settings



Graphical user interface, text, application

Description automatically generated

You can change the algorithm for how a map is displayed on a map. You would likely only need to do this for very large maps and you are probably better off simply reducing the “Max Nodes”

## Pan & Zoom



You can use the selected buttons to reset zoom, zoom in, zoom out, zoom to selection, and pan. Alternatively, you can scroll up to zoom in, scroll down to zoom out, and click and hold scroll wheel to pan.

## Save Map



This feature does not work at this time. Plans to save node location if user wants to manually manipulate map.

## Print Map



Prints the map to PDF or printer.

## Function Block Colors

Text

Description automatically generated with medium confidence

* Black container is module
* Blue is IO
* Gray is point type that isn’t IO

A picture containing text, sign, different

Description automatically generated

* Red outline is missing connections



* Tan is not real TDC tag as defined by EB files. It is auxiliary logic.

A green sign with black text

Description automatically generated with medium confidence

* Green is SM address

A red sign with black text

Description automatically generated with medium confidence

* Red is missing tag

## Function Block Shapes

A picture containing text, sign, electronics

Description automatically generated

* Oval indicates tag appers in CL
* Rectangle is no CL on tag

# CL

See “Help” toolbar in Mapper, then click CL for an interactive guide of the most important elements of the program.

## Multiple CL

You can open as many CL as you want and they will be added to the top tab. You can also drag tabs so that you have 2 CL files side by side.

Graphical user interface

Description automatically generated

Graphical user interface, text

Description automatically generated