



# DELTA TAU

## Power PMAC-NC 2014

Motion Commander Foundation  
©2014 Greene & Morehead Engineering, Inc.



# Software User Manual

## Power PMAC-NC 2014



Delta Tau Data Systems, Inc.  
April, 2015

**COPYRIGHT INFORMATION**

Software: © 2014 Delta Tau Data Systems, Inc. All rights reserved.

Software User Manual: © 2014 Delta Tau Data Systems, Inc. All rights reserved.

Motion Commander Foundation: © 2012-2014 Greene & Morehead Engineering, Inc. All rights reserved.

This document contains proprietary information of Delta Tau Data Systems, Inc. The information contained herein is not to be used by or disclosed to third parties without the express written permission of an officer of Delta Tau Data Systems, Inc.

**TRADEMARK ACKNOWLEDGMENT**

Windows, Visual Studio and .NET Framework are registered trademarks of Microsoft Corporation. MTConnect is a registered trademark of the MTConnect Institute. Other brands, product names, company names, trademarks and service marks are the properties of their respective holders.

**REVISION HISTORY**

<b>Version</b>	<b>Date</b>	<b>Description</b>
1.0	4/1/2015	Initial release

# Power PMAC-NC 14™ - Software User Manual

---

## Contents

Introduction .....	5
Requirements.....	7
Installation .....	7
Deployment.....	8
The Power PMAC Project .....	8
Hardware Key.....	9
Configuration File.....	9
Runtime Operation .....	10
Main Screen .....	10
Currently Loaded Program Display .....	11
Soft Panel .....	11
Login Display .....	12
Time/Program Elapsed Time display.....	12
Message Log Slider View.....	12
Full Screen Mode .....	13
Vertical Button Bar.....	13
Tabbed View Screen Selections .....	13
Program Editor.....	14
Run Screen .....	15
MDI Screen.....	17
Manual Mode Screen.....	18
Work Offset Screen.....	19
Tool Offset Screen.....	20
Alarms Screen .....	21
Machine View .....	22
Users .....	24
Foreign Language Support .....	24
Skins .....	25
Ctrl and Shift Keys .....	25
NC Files.....	26
The NC File Parser .....	26

NC File Configuration .....	26
Subprograms .....	27
Native PMAC Commands and Expressions .....	27
G and M-Code Groups.....	28
Mid-Program Start .....	29
Fixed Cycles .....	29
Using M99 to Repeat the Main Program .....	30
The NC Program Queue .....	30
NC File Comments.....	30
NC File Size Limitations .....	30
Aliasing <code>#define</code> and <code>#include</code> .....	31
Customizing the Application .....	32
Private Labeling.....	32
Messages.xml.....	33
External Assemblies .....	33
The Visual Studio Project Template.....	34
Appendix A. The Source Files .....	35
Appendix B. The Configuration File.....	37
Appendix C. Turbo PMAC Support .....	38
The Turbo PMAC Project.....	38
Appendix D. Source Code Exclusions .....	39
Appendix E. Send1 Command List.....	40
Appendix F. Included G & M Codes.....	43

# Power PMAC-NC™ - Software User Manual

## Introduction

The *Power PMAC-NC 14* HMI (PPNC14) is a host PC application for Delta Tau Power PMAC controlled CNC machines. This document is the Software User Manual for the *Power PMAC-NC 14* application. It contains information about how to use the software, what features the software includes, and also describes what can be customized.



- Supports standard RS-274 style G-code programs as well as native Power PMAC Programs.
- Split screen Subprogram visualization with embedded and external subprograms supported.
- Configurable for 1-10 axes, type of application, and machine/velocity units.
- Software and Hardware Control Panel support built in.
- Secure SSH/SFTP communications with Power PMAC.
- Built-in Power PMAC command terminal and Linux terminal.
- Colorized NC file editor optimized for large files.
- NC file Execution Queue for remote or unattended machine automation.
- Real-time Execution Monitoring including Subprograms.
- Mid-Program Start Capability.
- User Login system with Definable Feature Access.
- Built in Foreign Language Translation.
- MTConnect 1.1 Agent for supervisory data collection.
- Fully portable application deployment (no installation required < 5MB total file size!).
- Fully customizable with the *Software Development Kit (SDK)*.

- External assembly (plugin) system for custom Screens, Code Groups and other data.
- Customizable color schemes and login screens for OEM branding.

## Requirements

The PPNC14 program is compatible with Windows 7 or newer (64-bit or 32-bit).

The application requires .NET 4.0 and the Visual C++ 2010 runtime libraries. The application will install these components automatically. The following links can be used for manual installation of the same libraries.

*Microsoft .NET Framework 4 (Web Installer)*


<http://www.microsoft.com/en-us/download/details.aspx?id=17851>

*Microsoft Visual C++ 2010 Redistributable Package (x64)*

<http://www.microsoft.com/en-us/download/details.aspx?id=14632>

*Microsoft Visual C++ 2010 Redistributable Package (x86)*


<http://www.microsoft.com/en-us/download/details.aspx?id=5555>

 Some versions of the PPNC14 program also support the Turbo PMAC and are 32-bit applications, requiring the x86 version of the Visual C++ Redistributable - even on 64-bit Windows systems.

## Installation

The *PPNC14 Runtime Software* is distributed via a private GitHub repository and on media directly from Delta Tau Data Systems. In order to access the online repository sign up for a [free GitHub account](#) and give your account name to **Delta Tau Data Systems** when you purchase the Runtime. You will be given read-only access to the repository. Install [GitHub for Windows](#) on your development PC, log in, and "Clone" the repository. You will want to "Sync" occasionally to insure you have the latest release version.



 It is highly recommended that you make working copies of both the latest release of *PPNC14* Runtime and your working copy avoid losing your changes when you Sync. If a Sync fails for any reason, simply delete the entire "GitHub\PowerPmacNc14-Runtime" folder and Clone again.

## Deployment

The *PPNC14* HMI can be deployed by simply copying the "PowerPmacNc14-Runtime" folder to any location on your machine. The folder may be renamed if desired. Your distribution must include the files shown below. (Files not shown in this list may be deleted without affecting the application.)

Name	Type	Size
Languages	File folder	
MessageLogViewer.exe	Application	52 KB
PowerPmacNC.exe	Application	974 KB
PowerPmacNC.ini	Configuration settings	3 KB
DynamicDataDisplay.dll	DLL File	350 KB
ICSharpCode.AvalonEdit.dll	DLL File	612 KB
MCF.CustomControls.dll	DLL File	12 KB
MCF.DeltaTau.dll	DLL File	51 KB
MCFoundation.dll	DLL File	1,084 KB
Microsoft.WindowsAPICodePack.dll	DLL File	104 KB
Microsoft.WindowsAPICodePack.Shell.dll	DLL File	530 KB
Renci.SshNet.dll	DLL File	450 KB
Routrek.Granados.dll	DLL File	136 KB
SecureDongle_Control32.dll	DLL File	111 KB
SecureDongle_Control64.dll	DLL File	146 KB
PowerPmacNC.pdb	Program Debug Database	194 KB
DeviceMembers.xml	XML Document	107 KB
Messages.xml	XML Document	14 KB
PowerPmacNC_Settings.xml	XML Document	21 KB


## The Power PMAC Project

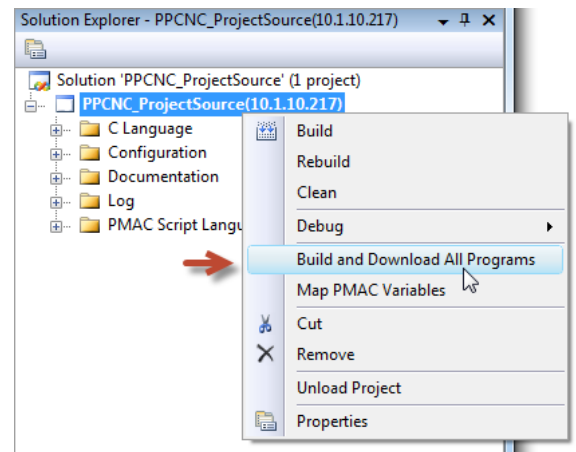
The *PPNC14* application requires the Power PMAC controller be configured with its source code counterpart to enable proper functionality and handshaking. This Power PMAC project is included with your product.

The Power PMAC project will be located in the "PowerPmacNc14-Runtime\PMAC Source Code\PowerPMAC" folder. Make a working copy of this directory before you download the project to the controller.

Open the "PPCNC\_ProjectSource.PowerPmacSuite\_sln" solution file in the *Power PMAC IDE*, right-click and select "Build and Download" as shown. Look for the "Download Successful" message in the Output window.

```
Download Successful.  
Total Project Download time = 13.057 sec  
Total Project Build and Download time = 24.212 sec
```

 After downloading the project, use the Terminal window to issue a "**save**" command to copy the project to nonvolatile flash memory, then issue a "\$\$\$" command to reset the controller.



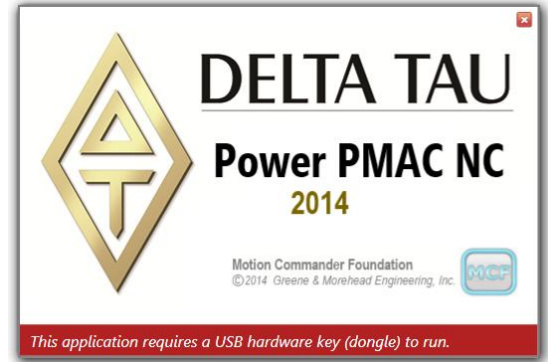


At this point the Power PMAC controller is now ready to work with the *PPNC14* program in a virtual mode. Actual machine functionality will require the appropriate integration of the motors, I/O, safety systems, etc. The default Power PMAC code is used as a starting point for all machine integrations which will utilize the PPNC14 software.

## Hardware Key

This application requires a USB hardware key (dongle) to run. You will receive hardware keys when you purchase copies of the PPNC14 program from Delta Tau Data Systems.

The hardware key is compatible with all versions of Windows and does not require a driver to be pre-installed.



## Configuration File

The application reads the "PowerPmacNC.ini" configuration file in its exe directory at start-up to obtain its configuration data. A *Reference* copy of this file is included in the project for convenience. Copy "Reference PowerPmacNC.ini" to the exe directory, rename it to "PowerPmacNC.ini", and edit it to specify your machine type, axis definitions, units, velocity units, and other important parameters. The configuration file is well commented for convenience.

```
[Machine Constructor]
; TODO: Specify from one to ten axis labels separated by commas.
; Axis labels can be more than one character but must be short. Suggest two characters max.
AxisLabels=X,Y,Z,A,B

; TODO: Specify motor numbers separated by commas (for status monitoring).
; The first motor number will be used to monitor the status of the first axis, etc.
MotorNumbers=1,2,3,4,5

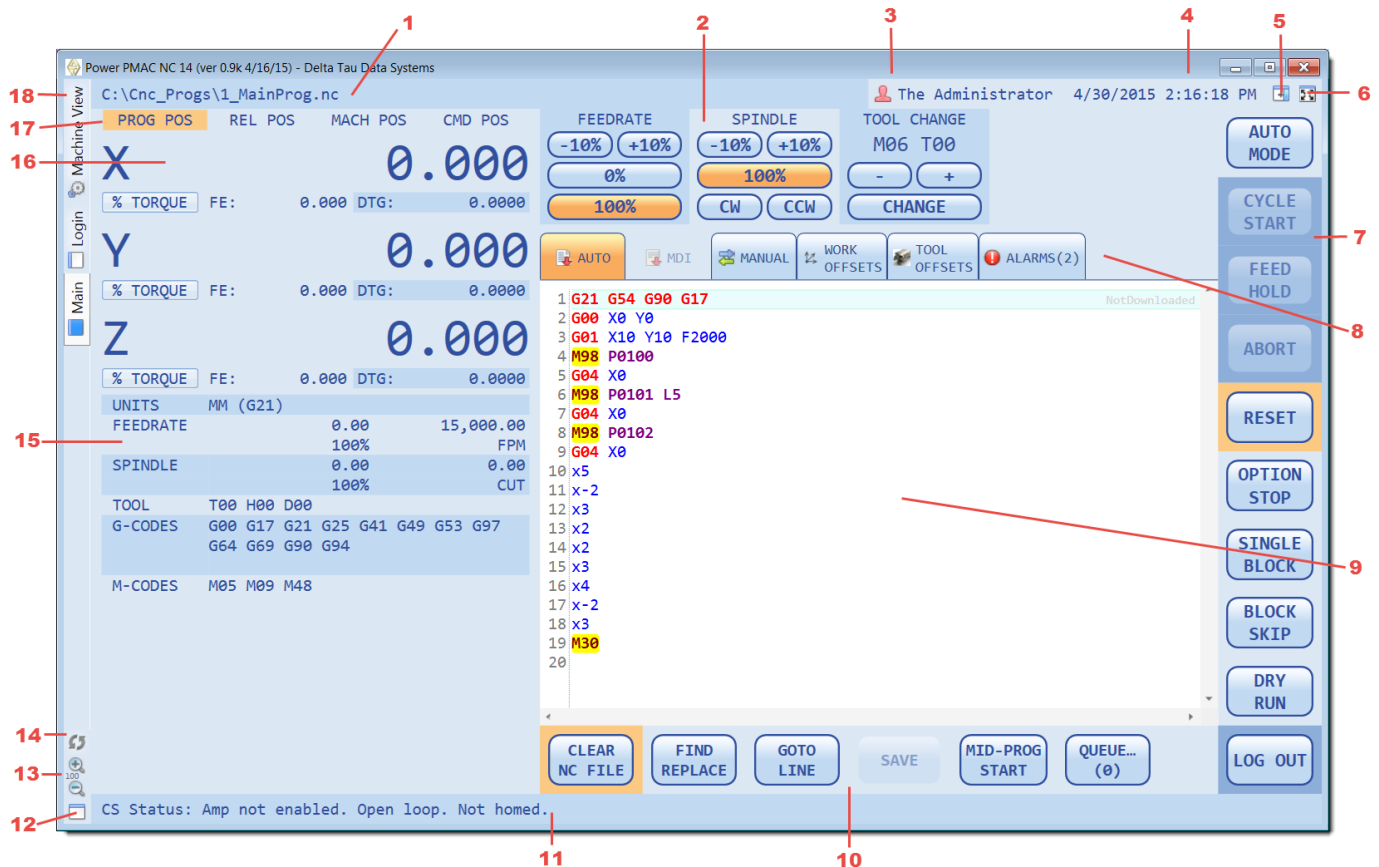
; TODO: Specify native length units (INCH or MM) and decimal places of precision (0-6).
NativeLengthUnits=MM
NativeLengthDecimalPlaces=3

; TODO: Select the controller (PowerPmacController, TurboPmacController or MockController)
Controller=PowerPmacController
```

# Runtime Operation

## Main Screen

The Main Screen serves as the base of operation during runtime. It is a modern feature rich implementation of a traditional CNC interface console with many added features specifically optimized for the Windows environment. Operators will find this screen intuitive and easy to use.

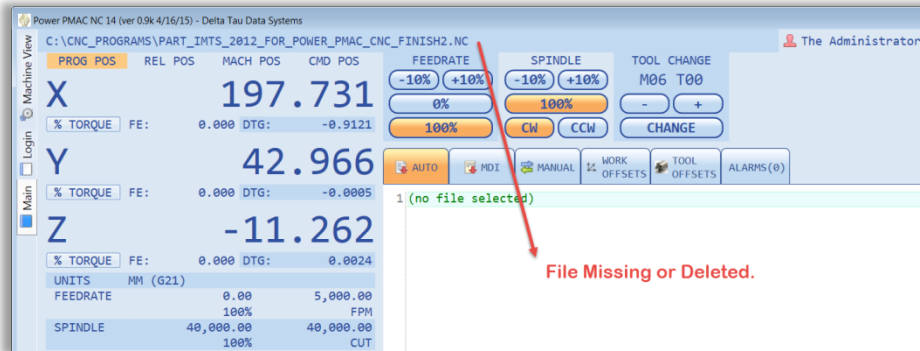


1. Currently loaded program and path display.
2. Soft Control Panel.
3. Login display, shows current user.
4. Time/Program Elapsed Time display.
5. Message log slider view.
6. Full Screen mode.
7. Vertical Button Bar - Main software operator controls.
8. Tabbed view screen selections.
9. Program editor and run screen.
10. Multi-Level Horizontal Button Bar - NC file editor.
11. Message status bar.
12. Dual-Screen pop out control.
13. HMI scaling controls.
14. HMI watchdog indicator.
15. NC parameter display window.
16. Axis parameter display window.
17. Position mode selector buttons.
18. Machine view selection.

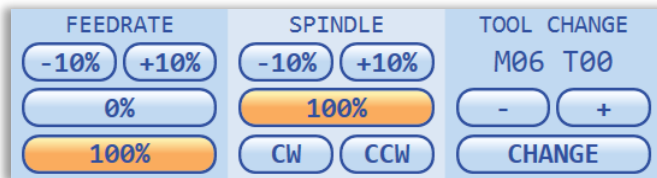
## Currently Loaded Program Display

C:\CNC\_PROGRAMS\PART\_IMTS\_2012\_ORGSHAPE\_CNC\_FINISH.NC (1:280,831 lines. Download Complete)

The currently loaded part program will display in the upper left hand corner of the HMI. The part program name, path, and number of lines are displayed. Additionally the first line to execute after a cycle start will be displayed. This can be useful for mid-program starts. If the file being downloaded is large the display will show the HMI is in process downloading. The PPNC14 will load the last file loaded on application startup. If the last file has been subsequently deleted the file name and path will be displayed, but the PPNC14 editor window will be empty as shown below.




## Soft Panel



The soft panel is used to display and change Feedrate Override, Spindle Override, as well as control tool changes. In general, the Soft Panel is displayed only when a hardware control panel is not present. In certain situations the system integrator may find it useful to include both for specialty applications.

The soft panel can be added or removed from the Main Screen by the following code in the Power PMAC project:

```
sendl "PendantConnected"           // Hides the Soft Panel
sendl "PendantDisconnected"        // Displays the Soft Panel
```

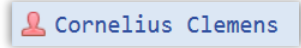
 The default PMAC project includes code to automatically show/hide this panel depending on whether a hardware pendant is present (see ppnc\_hmimonitor.plc in the Power PMAC project).

The feedrate override can be adjusted in increments of 10%, or set to either 0% or 100% immediately using the buttons provided. If the machine is currently at 0% or 100%, the button will illuminate accordingly.

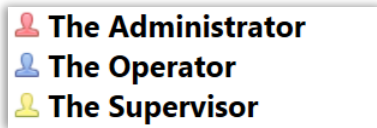
The spindle override can be adjusted similarly. Additionally there are modal buttons for CW/CCW spindle direction.

The Tool Change mechanism allows the operator to set the desired tool, and then initiate a tool change directly from the Soft Panel.

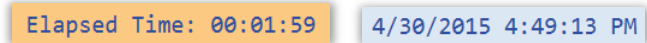
### Login Display



The Login Display will show the currently logged in machine operator and their user access level shown by color. There are three access levels as shown below.

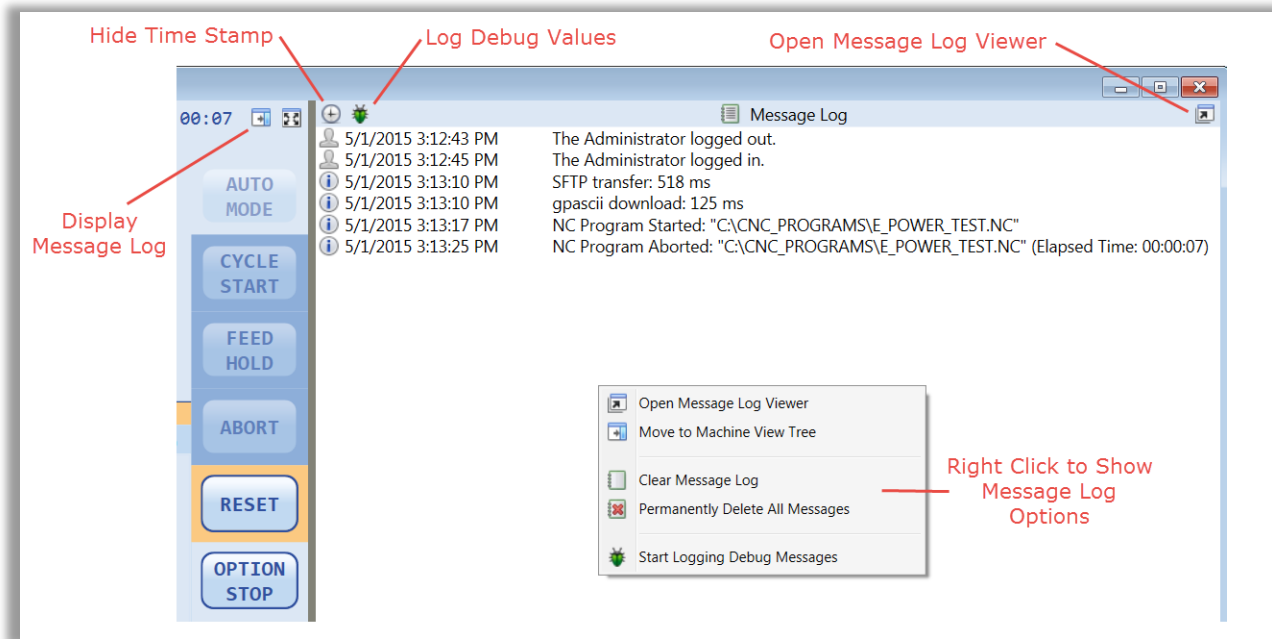


### Time/Program Elapsed Time display




The Time/Program Elapsed time display will toggle between real computer time and program elapsed time during program execution. In addition to this the automatic message log will keep track of program start time, end time, overall elapsed time, and actual elapsed time (non-feedhold time).

### Message Log Slider View

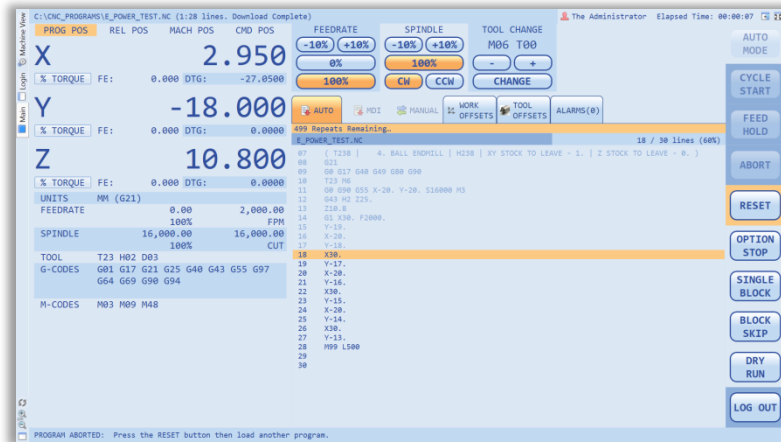


The Message Log Slider view allows access to the Message Log directly from the PPNC14 Main Screen. The message log can be expanded and contracted as necessary for the operator to view relevant information. The time stamp can be hidden if the operator prefers a more compact view of the messages. A Debug logging tool is included which will show

every command, status, etc., being sent back and forth between the control and the HMI. This should only be used for troubleshooting purposes. The Message Log window can be cleared by selecting this option by right clicking in the message area and selecting this option.

 Clearing the messages from the Message Log window does not delete the information from the ongoing message log utility.

## Full Screen Mode



Full Screen mode will extend the PPNC14 to the video resolution boundaries. The Windows task bar will be covered by the bottom portion of the PPNC14 application. The operator can either use the Window button or ALT-TAB to move to other applications if desired. If Full Screen mode is selected it will be retained through application shut down.

## Vertical Button Bar

The main operator control buttons and mode display will be found in the Vertical Button Bar. These buttons will illuminate or fade depending on the operating mode and functionality. If a button is faded it cannot be used. The top mode button can be used to switch modes and view the current mode of operation.

## Tabbed View Screen Selections

These tabs display the main screens which the operator utilizes during normal operation of the machine. The first three screens, AUTO, MDI, and MANUAL will switch the operational mode when selected.



Offset	X	Y	Z
G54	0.1569	0.0488	-0.0332
G55	0.3937	0.3937	0.3937
G56	23.4885	-36.7734	-27.4106
G57	-0.9744	-36.7734	-27.4106
G58	8.0465	-36.7734	-27.4106
G59	-0.3937	-0.3937	-0.3937
G54.1 P1	0.3937	0.3937	0.3937
G54.1 P2	0.7874	0.7874	0.7874
G54.1 P3	8.0465	-36.7734	-27.4106
G54.1 P4	23.4885	-36.7734	-27.4106
G54.1 P5	23.4885	-36.7734	-27.4106
G54.1 P6	23.4885	-36.7734	-27.4106
G54.1 P7	23.4885	-36.7734	-30.6509
G54.1 P8	2.1654	-36.7734	-27.4106
G54.1 P9	0.5906	0.5906	-0.5906
G54.1 P10	23.4885	-36.7734	-27.4106

SET WORK OFFSETS: (Manual Mode Only)

X Y Z

Tool Index	Tool Length	Tool Wear	Tool Diameter	Diameter	Wear
Tool 1	0.9843	0.0000	0.0197	0.0000	0.0000
Tool 2	0.9497	0.0000	0.0197	0.0000	0.0000
Tool 3	1.0364	0.0000	0.0295	0.0000	0.0000
Tool 4	0.5988	0.0000	0.0492	0.0000	0.0000
Tool 5	0.5604	0.0000	0.0394	0.0000	0.0000
Tool 6	0.8661	0.0787	0.0492	0.0000	0.0000
Tool 7	0.4853	0.0000	0.0591	0.0000	0.0000
Tool 8	0.7126	0.0000	0.0787	0.0000	0.0000
Tool 9	0.4853	0.0000	0.0492	0.0000	0.0000
Tool 10	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 11	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 12	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 13	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 14	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 15	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 16	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 17	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 18	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 19	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 20	0.0000	0.0000	0.0000	0.0000	0.0000
Tool 21	0.0000	0.0000	0.0000	0.0000	0.0000

SET TOOL OFFSETS: (Manual Mode Only)

SET TOOL LENGTH

CS Error Status: Mismatch between # of motors used and # in lookahead buffer  
5/2/2015 4:56:01 PM

## Program Editor

The PPNC14 includes a powerful editor which serves as the Run Screen as well. In edit mode NC codes are colorized by their code type (G, M, T, D, Comment, S-code, etc.). All programs lines are automatically pre-pended with line numbers which are used by the program for line display and mid-program starts. These auto-assigned line numbers do not conflict with CAM generated "N" line numbers in any way.

```

1 %
2 00000(MOLD_PUMPKIN1)
3 (DATE=DD-MM-YY - 24-10-13 TIME=HH:MM - 11:20)
4 (MATERIAL - ALUMINUM MM - 2024)
5 ( T1 | | H1 | XY STOCK TO LEAVE - 1. | Z STOCK TO LEAVE - 0. )
6 G21
7 G0 G17 G40 G49 G80 G90
8 T1 M6
9 G0 G90 G54 X103.738126 Y30.758289 A0. S20000 M5
10 G43 H1 Z15.
11 Z5.
12 G1 Z.673184 F0.
13 X81.692036
14 G0 Z5.
15 Z15.
16 X75.699911 Y32.748606
17 Z5.
18 G1 Z.673184
19 X109.703385
20 X114.401006 Y34.738924
21 Y71.704024
  
```

The download status of the part program is displayed in the upper right corner of the program editor. The following standard features are supported by the Program Editor:

- Clear NC File
- Find Replace
- Goto Line Number
- Save

- Save As
- Mid-Program Start
- NC File Queue

## Run Screen

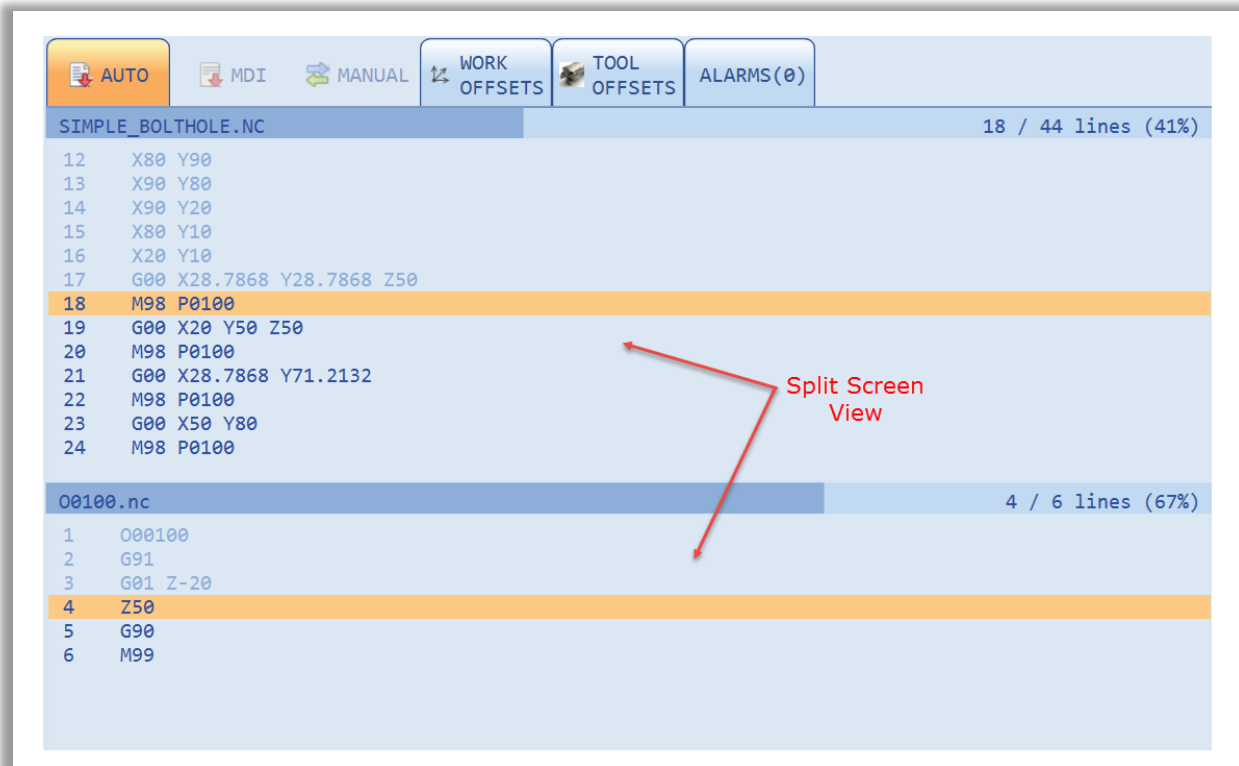
During Run Mode the editor screen will change background colors and the text will show as a lighter color once executed. The current executing line will be highlighted by the horizontal indicator as shown below. The NC program progress indicator will display the part program name and progress both a horizontal bar graph, current line number over total lines, as well as percent of lines executed.

**NC Program Progress Indicator**

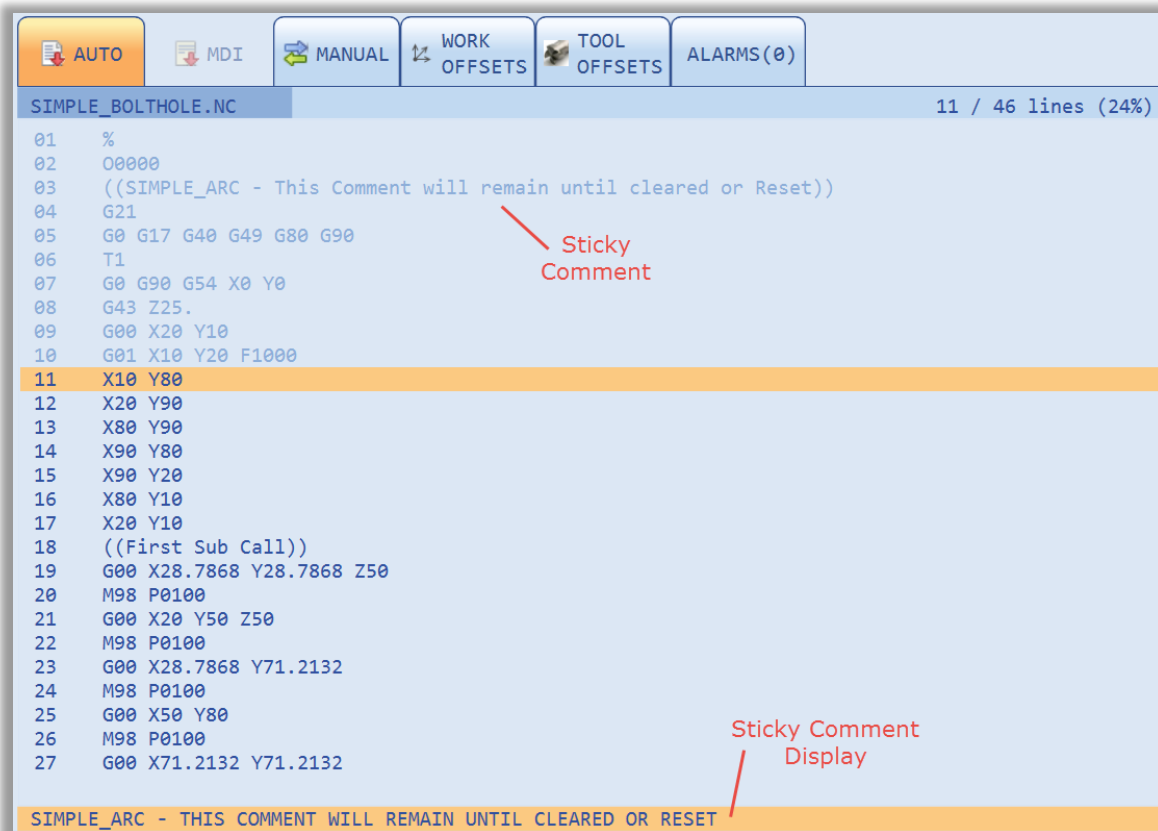
Line Number	X	Y	Z
05457	X97.224043		Z-7.056464
05458	X96.78334		Z-7.099091
05459	X96.462557		Z-7.13278
05460	X95.72831		Z-7.196708
05461	X95.015357		Z-7.243871
05462	X94.30953		Z-7.27246
05463	X93.602462		Z-7.282964
05464	X92.894846		Z-7.275522
05465	X92.213436		Z-7.249896
05466	X90.297045		Z-7.16981
05467	X88.438397		Z-7.147357
05468	X86.717618		Z-7.101384
05469	X85.482606		Z-7.003583
05470	X85.096087		Z-6.948877
05471	X84.231684		Z-6.831556
05472	X83.814411		Z-6.750881
05473	X82.928515		Z-6.58458
05474	X82.410423		Z-6.471533
05475	X81.5234		Z-6.282643
05476	X81.510454		Z-6.281341
05477	X80.082531		Z-5.973269
05478	X78.678064		Z-5.644969
05479	X77.26279		Z-5.2747

Current Line Indicator

During Sub-Program calls the Run Mode screen will morph into a split screen view simultaneously showing line tracking for both the main program as well as the sub-program.



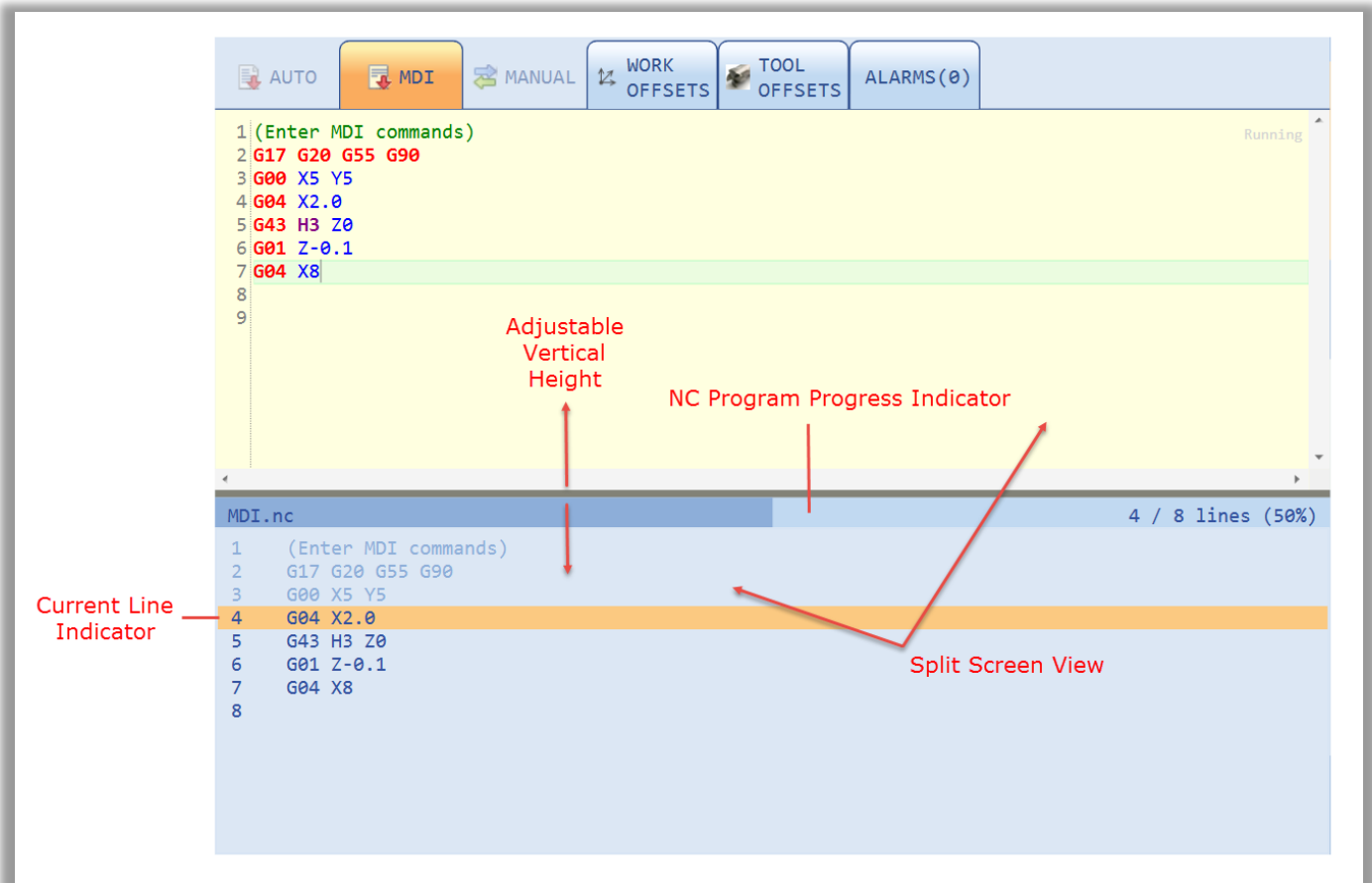
During Run Mode the NC program monitor supports “Sticky Comments”. Sticky comments are designated by using double parenthesis. The sticky comment will display at the bottom of the Run Mode screen until a subsequent sticky comment is encountered, or the program finishes. This can be a powerful feature for annotating NC files with operator instructions.





## MDI Screen

The MDI Mode screen is a split screen view which includes an MDI editor on top, with an execution monitor on the bottom. The vertical height of these screens is adjustable by dragging the split bar up or down. MDI programs are downloaded to the PMAC when a Cycle Start is executed. The application will automatically sense if the program is modified. If a subsequent Cycle Start is executed the PPNC14 will re-download the MDI program to the control buffer.




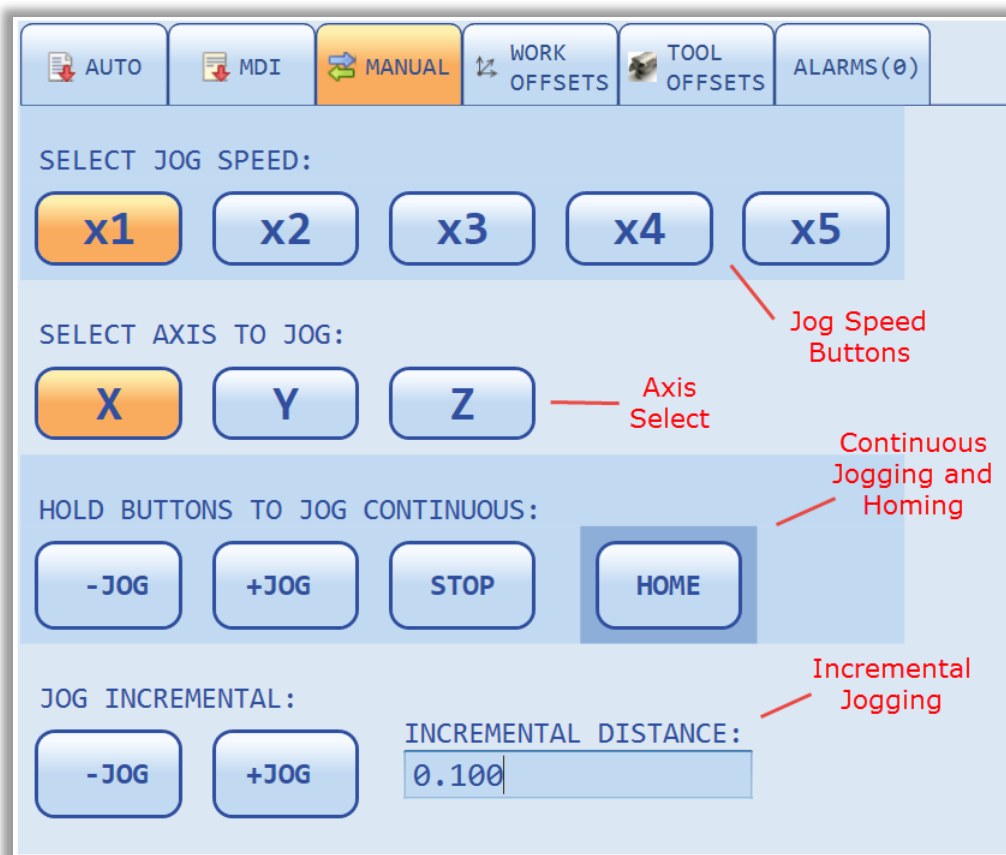
## Manual Mode Screen

When a hardware control panel is not present, a software Manual screen may be enabled. By default the Manual screen tab will be enabled. The operator will find jog speed select buttons, axis select buttons, continuous jogging buttons, home button, and incremental jog buttons. The incremental jog distance is an operator parameter which can be input. There are two configurations available for the jog speed select buttons, five button, and three button, depending on the integrators preference.

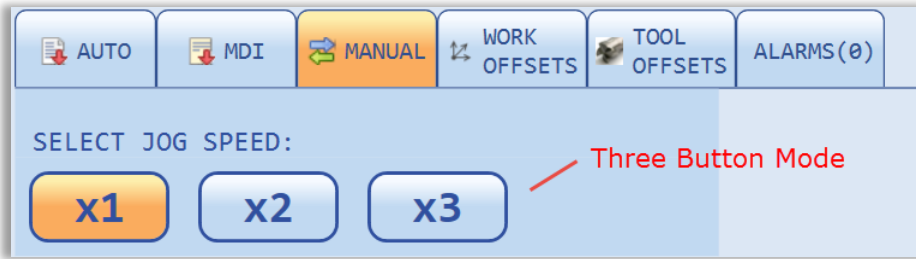
The Manual mode screen may be disabled completely if the integrator wishes to rely on a hardware control panel. This is done by the following code in the Power PMAC project:

```
sendl "HideManual" // Hides the Manual Screen and Tab  
sendl "ShowManual" // Shows the Manual Screen and Tab
```

 The default PMAC project includes code to automatically show/hide this panel depending on whether a hardware pendant is present (see ppnc\_hmimonitor.plc in the Power PMAC project).



Three button mode:

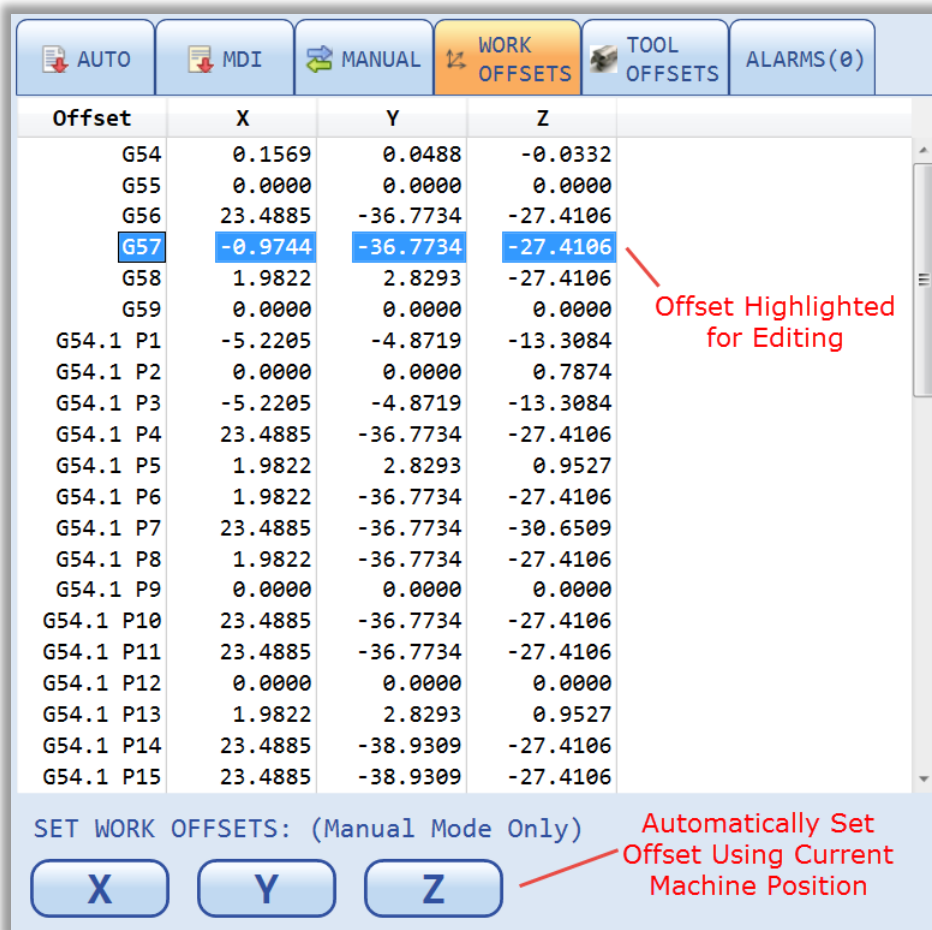



### Work Offset Screen


The Work Offset screen displays and allows modification of the coordinate system offset values. The values can be modified manually directly in the input boxes or can be set automatically by using the *Set Work Offsets* buttons at the bottom of the screen. When the buttons are used the current machine position will be queried and used as the offset. The offsets can be modified only while in manual mode.

When using the automatic work offset buttons, multiple offsets can be modified simultaneously by selecting more than one work offset row at a time.

The number of auxiliary G54.1 Px offsets is configurable in the PowerPmacNC.ini file.



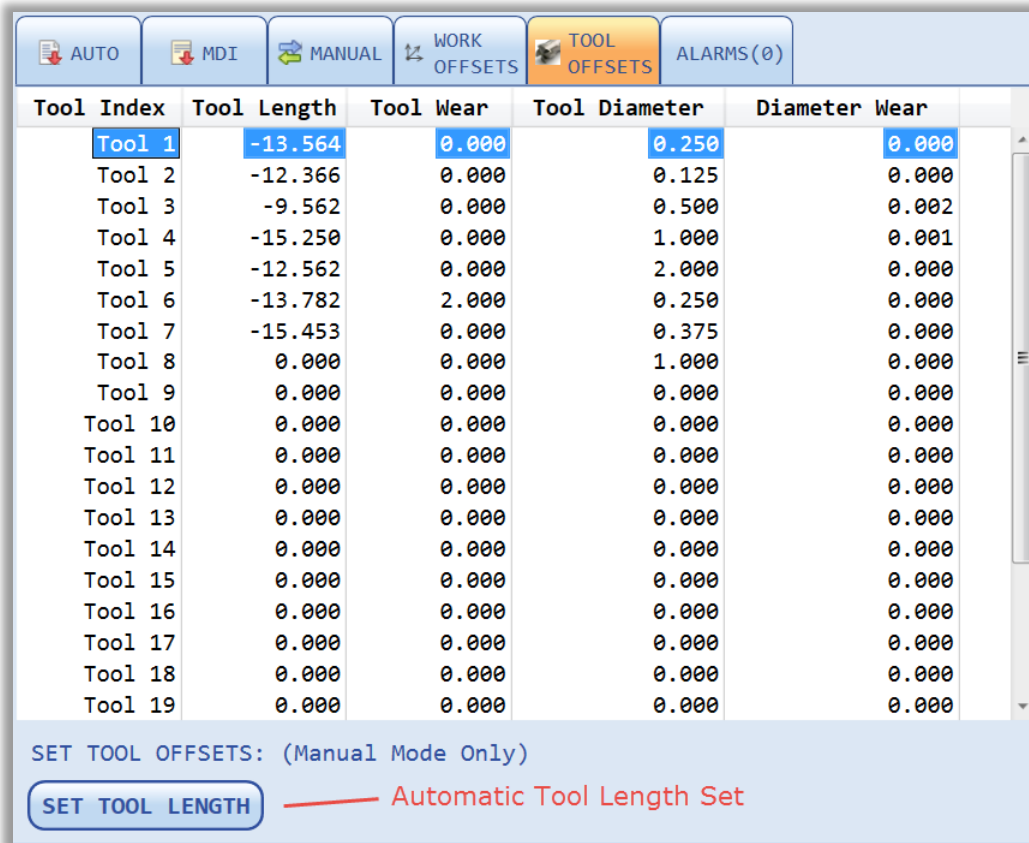
 The PPNC14 software includes a mechanism so the programmatic setting of offsets can be achieved directly from the controller. This allows for the simple integration of touch probes or other automated systems for this purpose (see ppnc\_worktooloffset.plc in the Power PMAC project).

 The actual data from the Work Offset table is saved in the PowerPMACSettings.xml file. In some special cases this file can be manually configured for special tooling or setups. This xml file cannot be edited while the application is running. Care should be taken whenever modifying settings files and a backup should be made prior to doing any modification.

### Tool Offset Screen

The Tool Offset screen displays and allows modification of the Tool offset parameters (Length, Length Wear, Diameter, Diameter Wear). The values can be modified manually directly in the input boxes or can be set automatically by using the *Set Tool Length* button at the bottom of the screen (length only). When the button is pressed the current machine position will be queried and used as the tool offset. The offsets can be modified only while in manual mode.

When using the automatic tool offset button, multiple offsets can be modified simultaneously by selecting more than one work offset row at a time.





The screenshot shows the 'TOOL OFFSETS' screen in a software interface. At the top, there are several mode buttons: AUTO, MDI, MANUAL, WORK OFFSETS, TOOL OFFSETS (highlighted), and ALARMS(0). Below these is a table with the following columns: Tool Index, Tool Length, Tool Wear, Tool Diameter, and Diameter Wear. The table contains 19 rows of tool data. At the bottom of the screen, there is a status bar that reads 'SET TOOL OFFSETS: (Manual Mode Only)' and a button labeled 'SET TOOL LENGTH' with a red arrow pointing to the text 'Automatic Tool Length Set'.

Tool Index	Tool Length	Tool Wear	Tool Diameter	Diameter Wear
Tool 1	-13.564	0.000	0.250	0.000
Tool 2	-12.366	0.000	0.125	0.000
Tool 3	-9.562	0.000	0.500	0.002
Tool 4	-15.250	0.000	1.000	0.001
Tool 5	-12.562	0.000	2.000	0.000
Tool 6	-13.782	2.000	0.250	0.000
Tool 7	-15.453	0.000	0.375	0.000
Tool 8	0.000	0.000	1.000	0.000
Tool 9	0.000	0.000	0.000	0.000
Tool 10	0.000	0.000	0.000	0.000
Tool 11	0.000	0.000	0.000	0.000
Tool 12	0.000	0.000	0.000	0.000
Tool 13	0.000	0.000	0.000	0.000
Tool 14	0.000	0.000	0.000	0.000
Tool 15	0.000	0.000	0.000	0.000
Tool 16	0.000	0.000	0.000	0.000
Tool 17	0.000	0.000	0.000	0.000
Tool 18	0.000	0.000	0.000	0.000
Tool 19	0.000	0.000	0.000	0.000

SET TOOL OFFSETS: (Manual Mode Only)

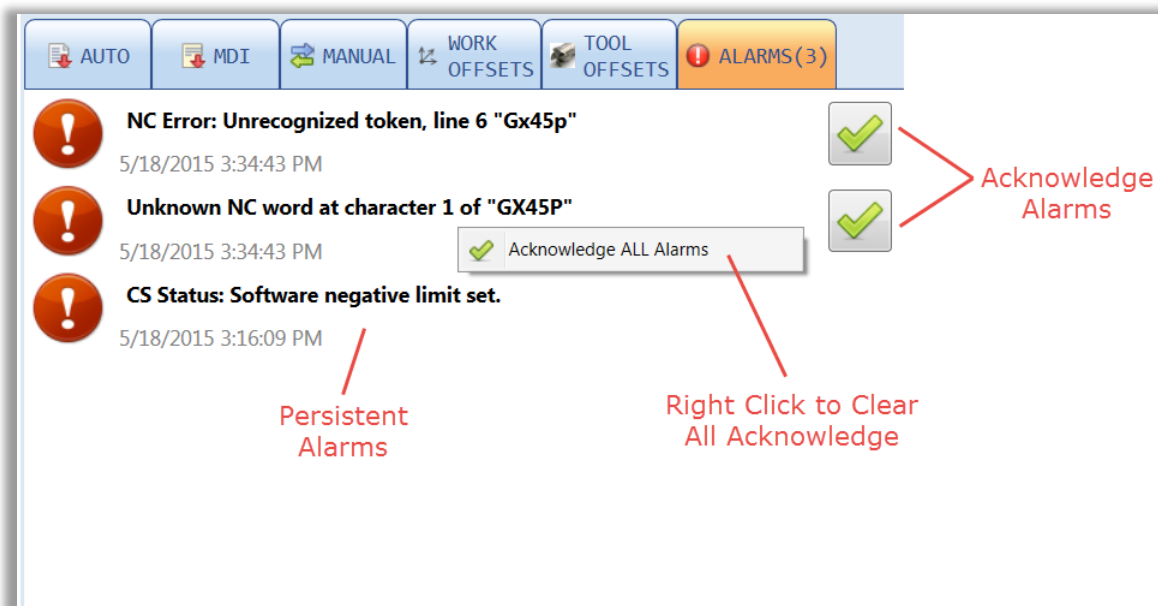
SET TOOL LENGTH — Automatic Tool Length Set

 The PPNC14 software includes a mechanism so the programmatic setting of offsets can be achieved directly from the controller. This allows for the simple integration of touch probes or other automated systems for this purpose (see ppnc\_worktooloffset.plc in the Power PMAC project).

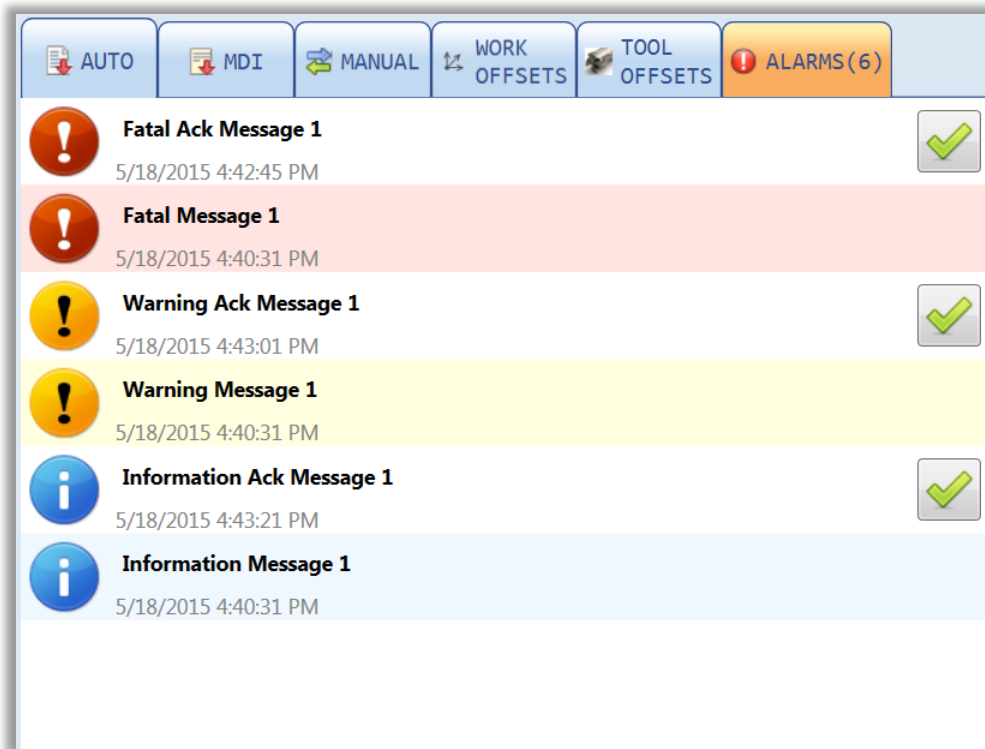
 The actual data from the Tool Offset table is saved in the PowerPMACSettings.xml file. In some special cases this file can be manually configured for special tooling or setups. This xml file cannot be edited while the application is running. Care should be taken whenever modifying settings files and a backup should be made prior to doing any modification.

## Alarms Screen

The Alarm screen will display any active alarms, warning, messages, etc. Any alarm or message which is displayed will also be sent to the message log with a time and date stamp. In general, there are two types of messages, persistent, and acknowledge. The persistent messages will remain active until the underlying fault is cleared. Acknowledge messages will appear with a check box to the right of the message. These messages can be cleared by clicking on the check to box to acknowledge the message. If there are multiple acknowledge messages present they can all be cleared by right clicking in the Alarms screen and selecting the “Acknowledge ALL Alarms” option.

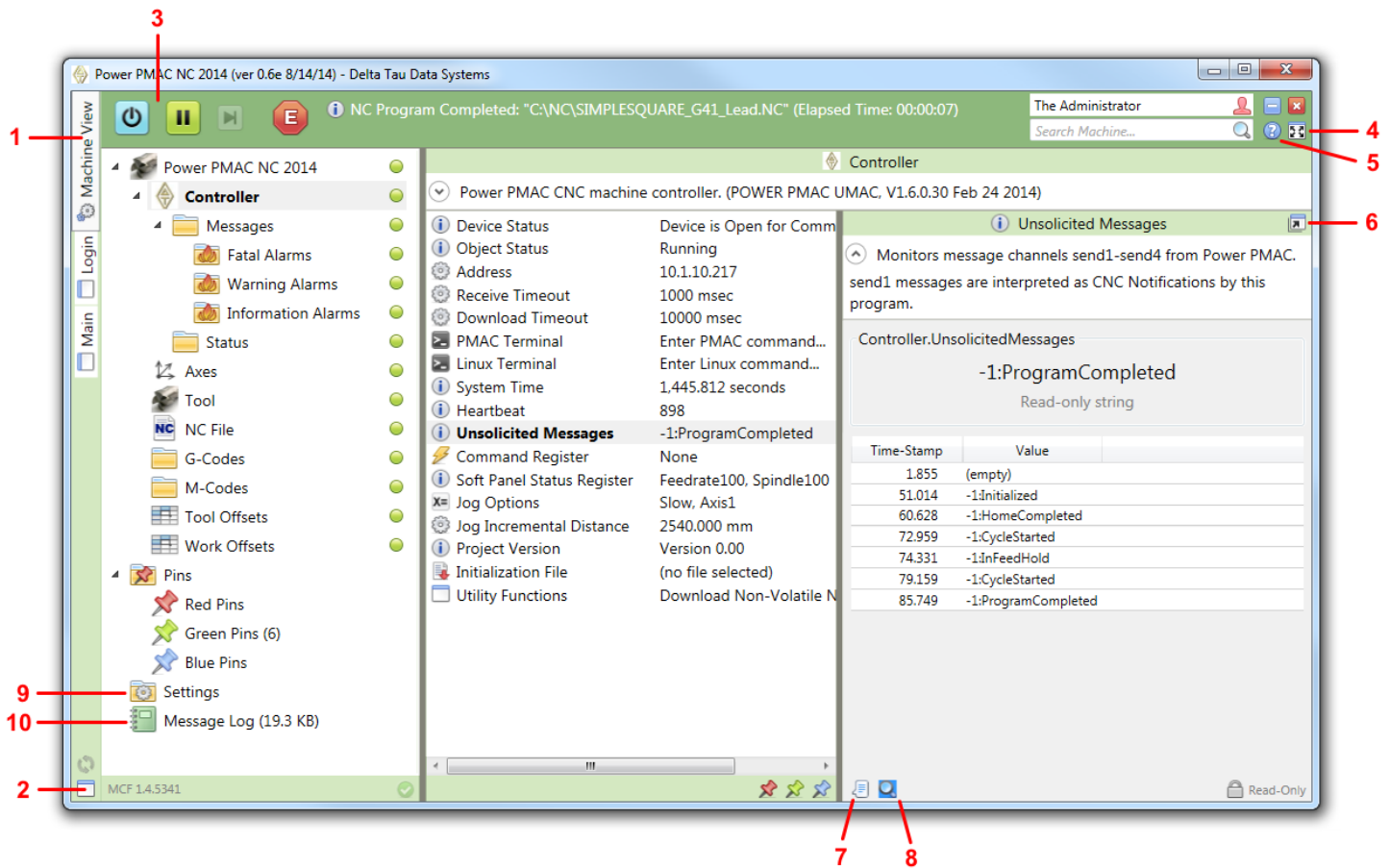


There are six types of custom messages which the machine builder may choose to utilize. Three persistent and three acknowledge. They are Fatal, Warning, and Message. They exist for persistent and acknowledge formats as shown below:




## Machine View

*Motion Commander Foundation* applications feature *Machine View*, an Explorer-style view of the data shared between the host PC program and the motion controller. Machine View also includes powerful tools for logging and plotting this data, controlling the runtime state of the application, editing settings, viewing the message log, etc.



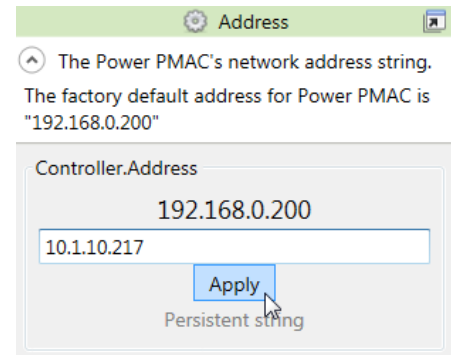
1. Navigate between Machine View and the User Interface.
2. Move the User Interface to a Separate Window.
3. Open/Close Communications, control the update cycle.
4. Toggle Full Screen mode.
5. Open the About Box for version information.
6. Open the member page in a Separate Window.
7. Log changes in value to the Message Log.
8. Open the time-stamped list of changes in value.
9. Edit the program Settings.
10. View the Message Log.

 Logging changes in value (7) of multiple variables allows the technician to view the interplay of these variables as the machine is operated. This is a very powerful diagnostic tool.

Message Log			
	8/16/2014 12:13:26 PM	205.060	Controller.CommandRegister=CycleStart (120)
	8/16/2014 12:13:26 PM	205.064	Controller.UnsolicitedMessages=-1:CycleStarted
	8/16/2014 12:13:26 PM		NC Program Started: "C:\NC\SIMPLESQUARE_G41_Lead.NC"
	8/16/2014 12:13:26 PM	205.090	Controller.CommandRegister=None (0)
	8/16/2014 12:13:34 PM	213.070	Controller.UnsolicitedMessages=-1:ProgramCompleted
	8/16/2014 12:13:34 PM		NC Program Completed: "C:\NC\SIMPLESQUARE_G41_Lead.NC" (Elapsed Time: 00:00:08)

Start the Power PMAC-NC 14 program, log in as the Administrator, and navigate to Machine View via the tab along the left edge of the window.

Select the **Controller** object to specify the IP address of your Power PMAC, then click the "Go Online" and "Run" buttons to open communication.



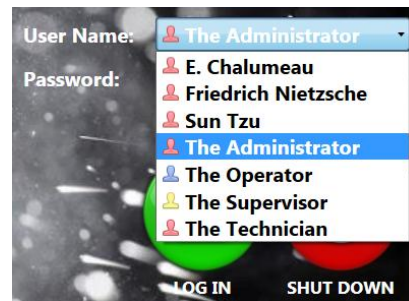
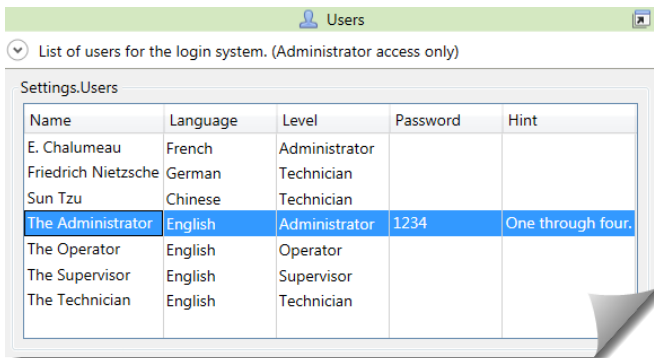
Navigate to Machine View to change application **Settings** and manage the list of Users. In particular, you will most likely want to change the "Start Up" setting to "Go Online and Run at Login".

## Users

The user login system supports four access levels.

- Operator**      Access to the Operator screens but no access to Machine View
- Supervisor**    Access to the Operator screens and read-only access to Machine View
- Technician**    Unrestricted access to all UI pages and Machine View (except the list of Users)
- Administrator**    Unrestricted access including the list of Users

Log in as the Administrator and navigate to Machine View Settings to manage the list of Users.



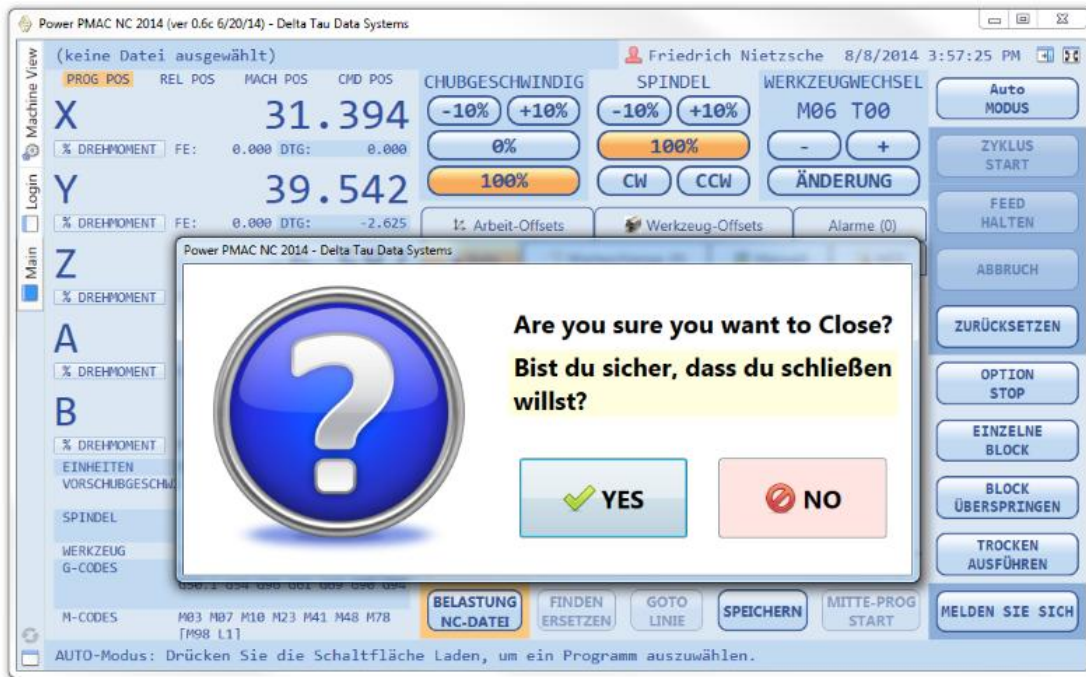
## Foreign Language Support

The Power PMAC-NC 14 program includes a sophisticated foreign language translation system. Each user's language is specified in his or her login profile. When a foreign language user logs in, Machine View text and selected UI Page text is displayed in the user's language. The "Language Support" setting provides runtime control of the language translation system.

Initial translations are obtained from the *Microsoft Translator* web service. Therefore, an internet connection will be required the first time that the foreign language user logs in. The translated text will be saved to a file named "*Languages\PowerPmacNC\_Language\_xx.txt*" where xx is one of 38 language codes. These files store native language strings and their foreign language translations, enabling the foreign language text to be edited by a human translator.

- English:      Are you sure you want to Close?
- German:      Bist du sicher, dass du schließen willst?






## Skins

The "Skin" setting may be used to change the color scheme of the main operator screen.

The "SkyBlue" and "MidnightBlue" skins are shown.



## Ctrl and Shift Keys

 If the **Ctrl** key is pressed while the **LOG IN** button is clicked then the user will be logged in but communications will not be opened. This feature is useful for working off line without a controller connected.

If the **Shift** key is pressed while the program is closed then the program will close immediately change the "Are you sure" dialog box or the splash screen. This feature saves time during development.

## NC Files

The Power PMAC-NC 14 program parses the selected NC file and generates a temporary file named "NcProgram.pmc" for downloading to the controller. The main NC program is enclosed in "open prog n/close" statements for loading into the PMAC program buffer. Any existing block numbers are stripped and a block number is prepended to each line for execution monitoring as shown in this example.

The NC file	The "NcProgram.pmc" download
1 <b>G21</b>	open prog 100
2 <b>G0 G17 G40 G49 G80 G90</b>	N1G21
3 <b>T1</b>	N2G00G17G40G49G80G90
4 <b>G0 G90 G54 X0 Y0</b>	N3T1
5 <b>G1 Z0. F1500</b>	N4G00G90G54X0Y0
6 <b>G41 X25 Y25</b>	N5G01Z0.F1500
7 <b>G1 X50</b>	N6G41X25Y25
8 <b>Y50</b>	N7G01X50
9 <b>X25</b>	N8Y50
10 <b>Y25</b>	N9X25
11 <b>G40 X0 Y0</b>	N10Y25
12 <b>G00 X-5 Y-5</b>	N11G40X0Y0
13 <b>M30</b>	N12G00X-5Y-5
	M30
	close

### The NC File Parser

- Wraps the NC in "open prog n/close" for downloading
- Prepends block numbers for execution tracking
- Strips comments (NC-style and C-style) and normalizes line endings
- Detects volatile subprogram calls and includes them in the download
- Tracks volatile subprogram file time-stamps and re-downloads if modified
- Detects nonvolatile subprogram calls and uploads them from the controller
- Throws an error if any called subprogram is not found
- Detects native PMAC commands and expressions and passes them unmodified
- Tracks modal G and M-codes and D/F/H/S/T values for Mid-Program Starts
- Supports the Block Skip option
- Supports Fixed Cycles
- Supports Line and File Prepend and Append features
- Warns if closing M30 or M99 is absent
- Supports #define and #include style parameter aliasing and substitution

### NC File Custom Pre-Parser

PPNC14 includes a user customizable pre-parser. This feature allows the machine builder to add custom logic to the parser. If the custom pre-parser is enabled it will call the custom pre-parser method for every NC line and evaluate it before it is downloaded to the PMAC control. The PPNC14 SDK is required to make changes to the pre-parser.

### NC File Configuration

The application's "PowerPmacNC.ini" configuration file includes a section for NC files. These are the default values:

```
[NC Files]
; Main and MDI program numbers are absolute.
; Subprogram numbers start at the specified base address.
MainProgramNumber=100
MdiProgramNumber=99
CoordinateSystem=1
SubprogramFolder="C:\NC"
```

```

; To disallow subprograms, set range to (0,-1)
NonvolatileSubprogramMin=0
NonvolatileSubprogramMax=99
VolatileSubprogramMin=100
VolatileSubprogramMax=199
SubprogramBaseAddress=5000

```

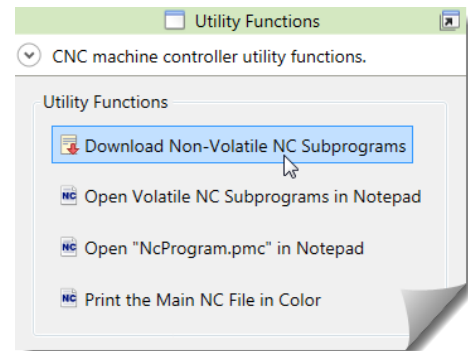
## Subprograms

Subprograms can be called by 'O' program number or by path as shown in this example. Subprograms may call other subprograms with the one restriction that *nonvolatile* subprograms may only call other *nonvolatile* subprograms.

"Volatile" subprograms exist in NC files and are downloaded along with the main program. "Nonvolatile" subprograms exist on the controller and must be downloaded via a utility included in the Power PMAC-NC 14 program and saved.

(Main Program) ... M98 P0101 M98 P0002 L5 // repeat count M98 (C:\myfolder\myname.nc) ... M30  O0101 (subprogram #101) ... M99  O0102 (subprogram #102) ... M99	(C:\NC\O0103.nc) O0103 (subprogram #103) ... M99  (C:\myfolder\myname.nc) O0104 (subprogram #104) ... M99	(Nonvolatile Subprogram File) O0001 (subprogram #001) ... M99  O0002 (subprogram #002) ... M99
---	---	---

"Volatile" subprograms are always downloaded together with the main program via a temporary file named "NcPrograms.pmc". "Nonvolatile" subprograms are downloaded via a utility included in the Power PMAC-NC 14 program. An NC file may contain more than one nonvolatile subprogram. The *Power PMAC IDE* must be used to SAVE after nonvolatile subprograms are downloaded. Volatile and nonvolatile subprogram 'O' numbers must be within the ranges specified in the configuration file. These ranges may be set to (0, -1) to disallow subprograms of either type.



Volatile subprograms called by path may reside in any folder under any filename, and their 'O' program numbers must be specified as the first line of the NC program. Volatile subprograms called by 'O' program number may reside in the main program's NC file (after the closing M30) or in the Subprogram Folder specified in the configuration file. Volatile subprogram NC files in the Subprogram Folder must be named "Onnnn.nc" where "nnnn" is the 'O' program number.

All subprograms will be loaded into PMAC buffers at the specified Subprogram Base Address plus 'O' program number.

## Native PMAC Commands and Expressions

The NC file parser detects native PMAC *command lines* and passes them unmodified. Tokens that identify a line as a native PMAC command are "if, else, while, do, goto" or the presence of an equals sign '=' or curly bracket {}. In addition, lines with a pipe '|' as the first character will be treated as native PMAC commands. The pipe will be stripped.

```

N100:
P103 = P24 + cos(P102)*P18
do

```

```

{
  G01 Y0.01 F2500
  | P779++
}
while (P779 <= P780)
if (P101 > P11) goto 100

```

PMAC jump labels such as "N100:" will pass, while NC-style block numbers such as "N100" will be stripped.

Native PMAC *expressions* can be enclosed in square brackets within a line of NC as shown in this example:

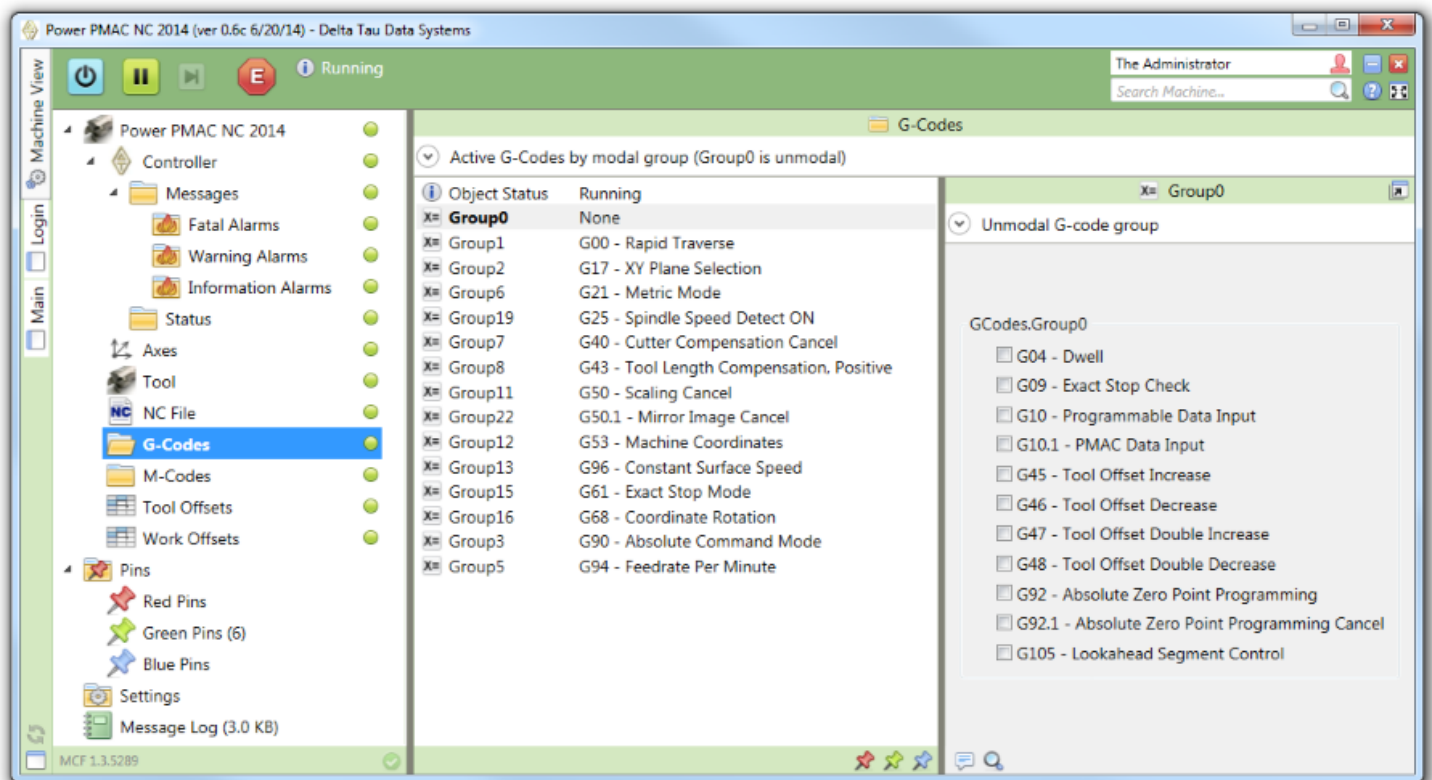
```

G54.1 P[-1+Q511*2] (1ST PART OFFSET)
X[cos (P30)] Y[P32]

```

## G and M-Code Groups

The Power PMAC-NC 14 program includes support for the common G and M-code groups as shown in Machine View. Group 0 is "unmodal" and the rest are "modal" (the codes in the group are mutually exclusive). Group 6 switches the application between English (INCH) and Metric (MM) modes.



Each G and M-code group is linked to a PMAC variable specified in "DeviceMembers.xml".

```

<Member Name="GCodes.Group0" Getter="P200" TimeBetweenUpdates="500" />
<Member Name="GCodes.Group1" Getter="P201" TimeBetweenUpdates="500" />
<Member Name="GCodes.Group2" Getter="P202" TimeBetweenUpdates="500" />
<Member Name="GCodes.Group6" Getter="P206" TimeBetweenUpdates="500" /> ...
<Member Name="MCodes.SubprogramGroup" Getter="P308" TimeBetweenUpdates="500" />

```

Refer to "GCodes.cs" and "MCodes.cs" in the *PowerPmacNC* project for the numerical values of each code. For example, if P206=1 then G21 of Group 6 is active.

```

/// <summary>G-code Group 6 enumeration.</summary>

```

```

public enum EGroup6
{
    /// <summary>Inch Mode</summary>
    [Description("G20 - Inch Mode")]
    G20,
    /// <summary>Metric Mode</summary>
    [Description("G21 - Metric Mode")]
    G21,
};

```

Refer to "CustomCodeGroups.cs" in the *CustomExamples* project to learn how to add custom G and M-code groups to the Power PMAC-NC 14 program. Conversely, G and M-code groups that are not required by the application may be removed by specifying them in the application's "PowerPmacNC.ini" configuration file. (Group0, Group6, ProgramGroup and SubprogramGroup are used by the program and may not be removed.)

```

[Machine Constructor]
ExtraneousGroups=Group11,Group22,ThreadingGroup,GearRangeGroup,BAxisGroup

```

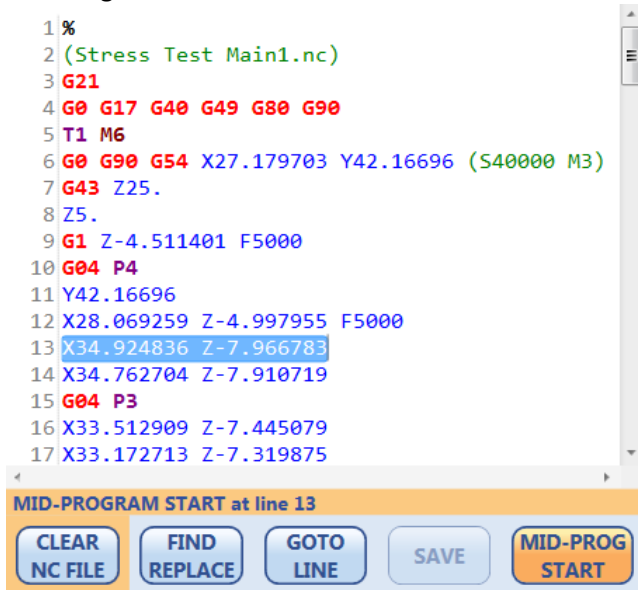
### Mid-Program Start

The line of NC where the cursor is located will become the *Mid-Program Start line* when the "MID-PROG START" button is pressed. Pressing the button a second time clears the Mid-Program Start function.

When the CYCLE START button is pressed, the NC file will be scanned down to the Mid-Program Start line and a block of NC will be generated which sets all of the modal G and M-codes and D/F/H/S/T values encountered in the scan.

Mid-Program Starts from within subprograms are not supported at this time.

Mid-Program Start at line 13



The "NcProgram.pmc" download

```

open prog 100
G21G01G17G40G43G90G54F5000T1M06
N13X34.924836Z-7.966783
N14X34.762704Z-7.910719
N15G04P3
N16X33.512909Z-7.445079
N17X33.172713Z-7.319875
...
M30
close

```

### Fixed Cycles

Fixed Cycles ("Canned Cycles") supported include G73, G74, G76, and G81 through G89. G80 cancels the fixed cycle function, as shown in the following example:

The NC File

The Resulting Parser Output

```

3 G99 G81 X29.726016 Y-5.826994 R2.5 Z-17.5 F4000 N3G99G81X29.726016Y-5.826994R2.5Z-17.5F4000
4 X31.393792 Y-6.587248 N4G99G81X31.393792Y-6.587248R2.5Z-17.5F4000
5 X33.072664 Y-7.281055 N5G99G81X33.072664Y-7.281055R2.5Z-17.5F4000
6 X34.762704 Y-7.910719 N6G99G81X34.762704Y-7.910719R2.5Z-17.5F4000
7 G80 X47.062818 Y-10.726689 N7G80X47.062818Y-10.726689
8

```

## Using M99 to Repeat the Main Program

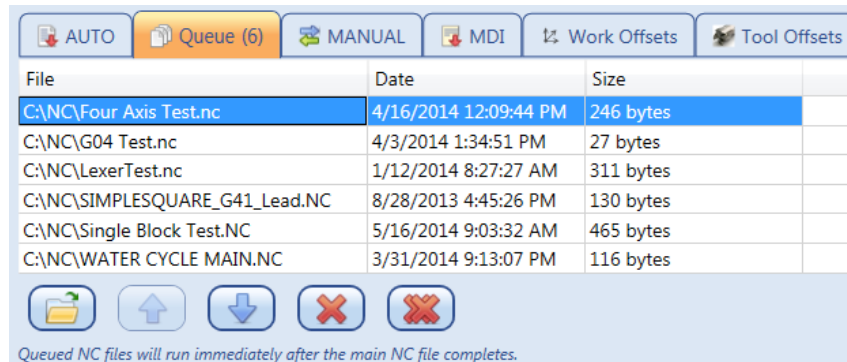
The main NC program will be repeated in an infinite loop if it is ended with M99 instead of the usual M30. A repeat count may be specified using the 'L' parameter. "M99 L5" will cause the main NC program to be repeated five times.

## The NC Program Queue

The first NC file loaded will always appear in the editor. Additional NC files will be loaded into the *Queue* as shown. The Queue includes tools for re-ordering and deleting files.



Drag-and-drop from *Windows Explorer* may be used to load multiple NC files.



Queued NC files will be downloaded and executed immediately after the initial program completes without the need for further CYCLE START commands.

## NC File Comments

Comments (displayed in green in the editor) are stripped before the NC program is downloaded. NC-style () comments, C-style /\*\*/ comments and CPP-style // comments are supported, as shown in the example.

C-style /\*\*/ comments may not span multiple lines.

Subprogram paths are also surrounded by parentheses in M98 calls as shown in the example.

```

12 X34.924836 Z-7.966783
13 M98 P0109 // CPP style comment
14 X33.072664 /*C style comment*/ Z-7.281055
15 M98 P0109 (NC style comment)
16 X31.393792 Z-6.587248
17 M98 (C:\NC\00105.nc)
18 X29.726016 Z-5.826994

```

## NC File Size Limitations

Informal testing indicates that NC files larger than around 100 MB will cause the text editor to run out of memory and crash the program.

## Aliasing `#define` and `#include`

PPNC14 supports `#define` aliasing and `#include` type file headers.

```
[NC Files]
; Option to allow #define macro substitutions in NC files.
AllowMacros=true
```

```
#define Lock P100
#define LockSafetyDoor Lock=1
#define UnlockSafetyDoor Lock=0
#define #5000 P200
...
G1 Z-2.5 F4000.0
LockSafetyDoor
X-8.003 Y9.593
```


```
#include "MyMacros.nc"
...
G1 Z-2.5 F4000.0
LockSafetyDoor
X-8.003 Y9.593
```



## Customizing the Application

There are three ways that the Power PMAC-NC 14 program can be customized to suit a particular machine.

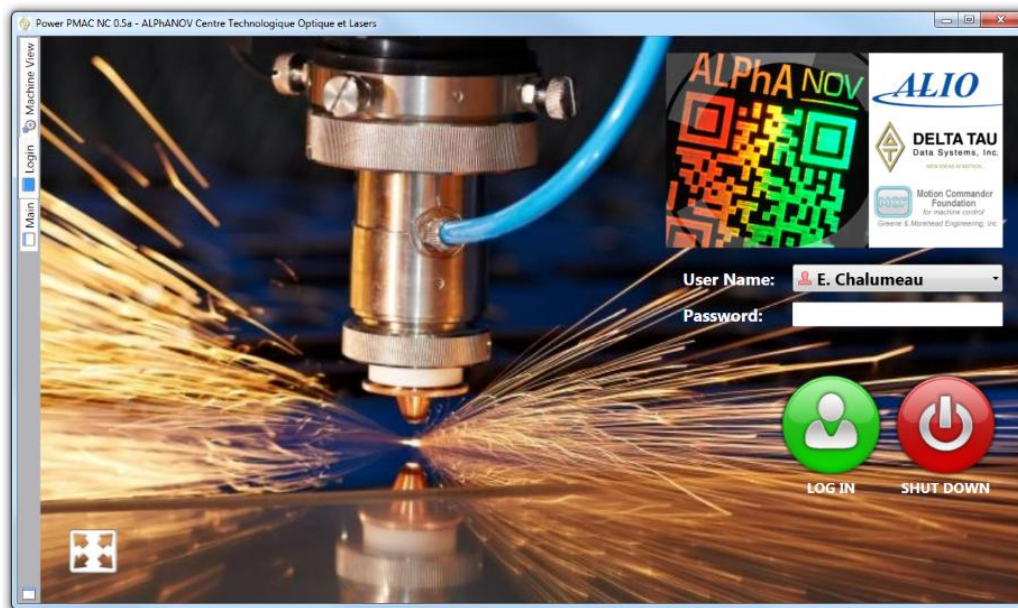
1. *Configure* the application by editing "PowerPmacNC.ini" and "Messages.xml".
2. *Extend* the application by creating external assemblies (plugin DLL's).
3. *Modify* the application by editing its source code.

 The configuration file and external assemblies are the preferred methods for customization because you'll be able to upgrade the main application without the need to merge your source code changes back in. If you do need to modify the main application, you'll want to clearly mark your edits with block-start and block-end comments to make merges easier in the future.

## Private Labeling

You may add the following lines to your "PowerPmacNC.ini" file to private-label the program and specify your own splash image and login screen background. Images should be PNG or JPEG format and must be located in the exe directory. The splash image should be approximately 500x300 pixels and the login image should be around 1000x700.

```
[Private Label]
CompanyName="ALPhANOV Centre Technologique Optique et Lasers"
SplashImage="LaserSplashImage.png"
LoginImage="LaserLoginImage.jpg"
```



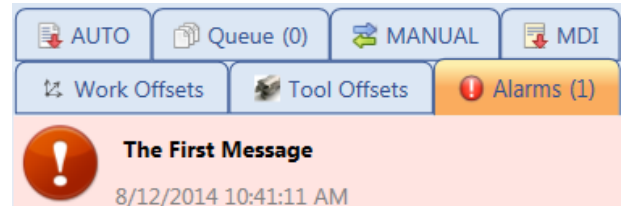


## Messages.xml

The Power PMAC-NC 14 program includes three bitwise message registers (Fatal, Warning and Information) that the controller can use to display messages to the operator. The message strings are specified in the "Messages.xml" file in the exe directory. A *Reference* copy of this file is included in the project for convenience.

```
<FatalMessages>
  <!-- Alarms activate on the rising edge of the bits and deactivate on the falling edge. -->
  <Message Bit="0" Description="The First Message">The is the first fatal message.</Message>
  <Message Bit="1" Description="The Second Message">The is the second fatal message.</Message>
</FatalMessages>
```

"Fatal" messages are marked with a red icon and are flashed in the status bar, while "Warning" and "Information" messages are marked with yellow or blue icons and are highlighted in the status bar.



Each message register is linked to a PMAC variable specified in "DeviceMembers.xml".

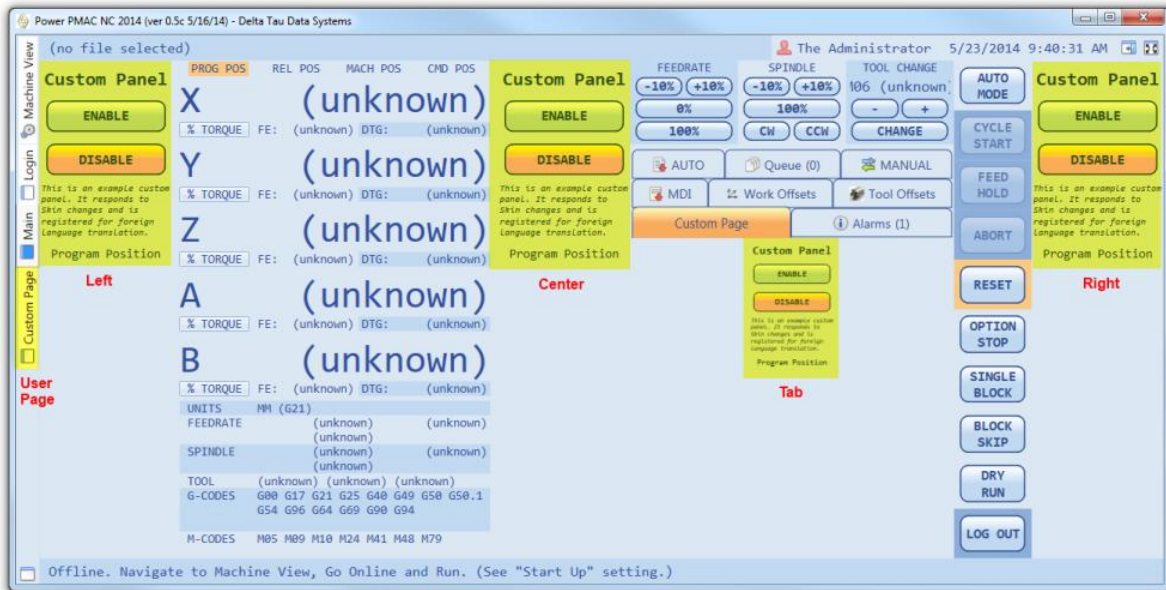
```
<Member Name="Controller.Messages.FatalMessages" Getter="M100" TimeBetweenUpdates="500" />
<Member Name="Controller.Messages.WarningMessages" Getter="M120" TimeBetweenUpdates="500" />
<Member Name="Controller.Messages.InformationMessages" Getter="M140" TimeBetweenUpdates="500" />
```

## External Assemblies

A C#/WPF class library project called "CustomExamples" is included with the SDK as a template for your custom plugin DLL's. The source files are well commented for convenience. A custom object adds its own device members to the Machine View hierarchy and also attaches "Changed" handlers to members created by the main program. A custom WPF Panel responds to UI Skin changes and is registered for foreign language translation. Simply add the following lines to your "PowerPmacNC.ini" file to test the custom plugin DLL.

```
[External Assemblies]
Object="..\..\..\CustomExamples\bin\Debug\CustomExamples.dll;CustomExamples.CustomObject"
CenterCustomFrame="..\..\..\CustomExamples\bin\Debug\CustomExamples.dll;CustomExamples.PageCustom"
CodeGroups="..\..\..\CustomExamples\bin\Debug\CustomExamples.dll;CustomExamples.CustomCodeGroups"
```

Your custom WPF Pages can be displayed by the main program in five different locations as shown in the illustration. Panels designed for the left, center and right columns should be tall and narrow, panels designed for the main screen tab area should be square, and panels designed to be full-screen *User Pages* may be much larger and more complex. Custom panels are hosted inside WPF *Viewboxes* so that they will be sized to fit the available screen area.



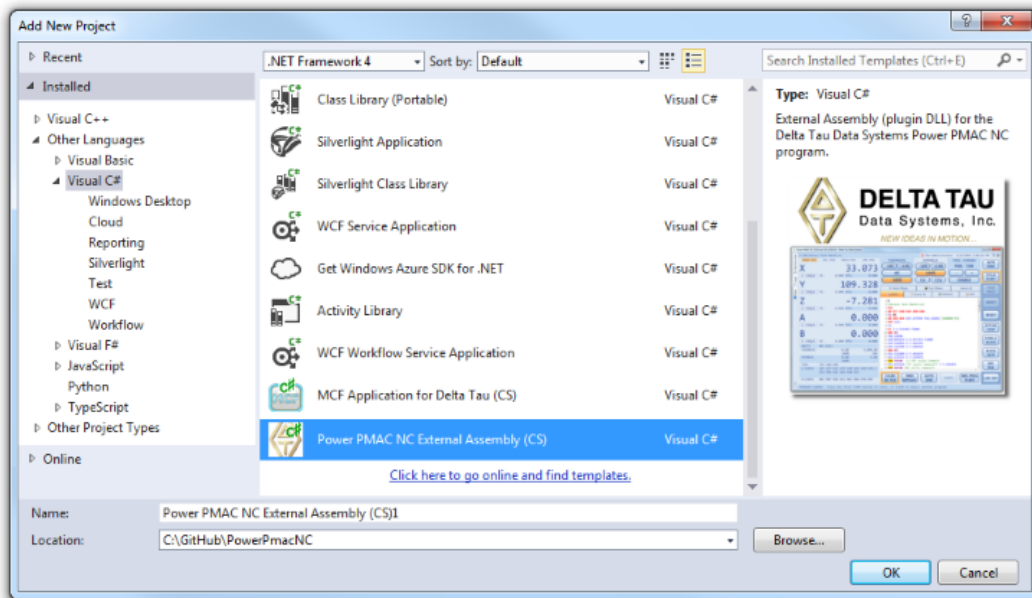
## The Visual Studio Project Template

A Visual Studio project template is provided that will add a *Power PMAC-NC 14 External Assembly* project to your solution. To use this template, simply put a copy of the included "Power PMAC-NC 14 External Assembly (CS).zip" file in your "Documents\Visual Studio 201x\Templates\ProjectTemplates\Visual C#" folder.

1. In Visual Studio, right-click on your "PowerPmacNC" solution and select "Add|New Project".
2. In the Visual C# templates, select "Power PMAC-NC 14 External Assembly (CS)".
3. Give it a name and start customizing!
4. Add your external assembly specification to your application's "PowerPmacNC.ini" file.

[External Assemblies]

CenterCustomFrame="mypath\MyExternalAssembly.dll;MyExternalAssembly.MyCustomPage"



## Appendix A. The Source Files

### GitHub\PowerPmacNcRelease

This folder will be copied to your PC when you “Clone” the repository from either the *GitHub* website or the *GitHub for Windows* application. You will want to use the *GitHub for Windows* application to periodically “Sync” to update your local copy to the most recently released version.



It is highly recommended that you make working copies of both the *PowerPmacNC* Visual Studio solution and the Power PMAC project in order to avoid losing your edits when you Sync. If a Sync fails for any reason, simply delete the entire “GitHub\PowerPmacNcRelease” folder and Clone again.

README.md, Banner.png, .gitattributes, .gitignore  
The Git repository configuration files. Do not edit.

GitHub\PowerPmacNcRelease\PowerPmacNC  
Source code for the Power PMAC-NC 14 C# application. (See detail below.)

GitHub\PowerPmacNcRelease\PMAC Source Code\PowerPMAC  
The Power PMAC project that works together with the host PC application to run NC files. The *Delta Tau Power PMAC IDE* will be used to download this project to the controller.

GitHub\PowerPmacNcRelease\PMAC Source Code\TurboPMAC  
The Turbo PMAC project that works together with the host PC application to run NC files. The *Delta Tau PWin32-PRO2 PMAC Executive Program* will be used to download this project to the controller.

GitHub\PowerPmacNcRelease\PowerPmacNC Demo Build  
A demonstration version of the Power PMAC-NC 14 program intended for marketing and training purposes.

GitHub\PowerPmacNcRelease\CustomExamples  
This example C# project produces an external (plugin) DLL that can be loaded by the Power PMAC-NC 14 program to extend its functionality with custom operator panels, variables, and G/M-code groups.

### GitHub\PowerPmacNcRelease\PowerPmacNC

Source code for the *Power PMAC-NC 14* application. This is a .NET 4.0/C#/WPF application based on the *Motion Commander Foundation* .NET framework. Visual Studio 2010 or newer (Express or Professional) can be used to build this project.

PowerPmacNC.sln, PowerPmacNC.csproj, DeltaTau.ico  
The Visual Studio solution and project files, and the application icon.

Main.cs  
The entry point for MCF-based applications, including the *Machine* definition.

Enumerations.cs  
The enumerations used by the application.

PowerPmacController.cs, TurboPmacController.cs  
*Power PMAC* and *Turbo PMAC* CNC controller support classes.

DeviceMembers.xml  
The PMAC variable assignments and update rates for each device member in the *Machine* definition.

GCodes.cs, MCodes.cs

The standard G and M-code group definitions. Custom G and M-code groups should be added via external (plugin) DLL's instead of directly editing these files.

PageLogin.xaml, PageLogin.xaml.cs

The user login WPF page. Alternative background images may be specified in the application's configuration file.

PageMain.xaml, PageMain.xaml.cs

The main operator panel WPF page. Most of the process logic that operates the CNC machine is in the code-behind of this WPF page.

Reference PowerPmacNC.ini

The application reads the "PowerPmacNC.ini" configuration file in its exe directory at start-up to obtain its configuration data (machine type, axis definitions, units, and other important parameters). A *Reference* copy of this file is included in the project for convenience.

Reference PowerPmacNC\_Settings.xml

The application saves the values of all persistent variables in the "PowerPmacNC\_Settings.xml" file in its exe directory. A *Reference* copy of this file is included in the project for convenience.

Reference Messages.xml

The app includes three bitwise message registers (Fatal, Warning and Information) that the controller can use to display messages to the operator. The message strings are specified in the "Messages.xml" file in the exe directory. A *Reference* copy of this file is included in the project for convenience.

SecureDongle\_Control32.dll, SecureDongle\_Control64.dll

The hardware key (dongle) driver libraries for both 64-bit Windows and 32-bit Windows.

GitHub\PowerPmacNcRelease\PowerPmacNC\Properties

Standard C# project directory that contains assembly information.

GitHub\PowerPmacNcRelease\PowerPmacNC\References

This folder contains the MCF libraries and other DLL's required by the application.

GitHub\PowerPmacNcRelease\PowerPmacNC\SupportClasses

This folder contains C# and WPF support classes for the application.

GitHub\PowerPmacNcRelease\PowerPmacNC\Skins

This folder contains the user interface "Skins" WPF resource dictionaries.

GitHub\PowerPmacNcRelease\PowerPmacNC\Images

This folder contains image files for the application. Do not edit these images. Alternative background images may be specified in the application's configuration file.

GitHub\PowerPmacNcRelease\PowerPmacNC\Languages

This folder contains the foreign language files. MCF generates a language file the first time that a foreign language user logs in. This file may then be edited by a skilled human translator to refine the machine translations. For example, the German translation of FEEDRATE must be abbreviated to display correctly.

English: FEEDRATE

German: VORSCHUBGESCHWINDIGKEIT

## Appendix B. The Configuration File

```
;
; "PowerPmacNC.ini" - Configuration file for the Power PMAC-NC 14 2014 program.
; This file is read by PowerPmacNC.exe at startup and must be in the exe directory.
; This file will NOT be overwritten by MCF and should be well commented.
;

[Machine Constructor]
; TODO: Specify the machine type (Mill or Laser)
MachineType=Mill

; TODO: Specify from one to ten axis labels separated by commas.
; Axis labels can be more than one character but they must be short. Suggest two characters max.
AxisLabels=X,Y,Z,A,B

; TODO: Specify motor numbers separated by commas (for status monitoring).
; The first motor number will be used to monitor the status of the first axis, etc.
MotorNumbers=1,2,3,4,5

; TODO: Specify the application's native length units (INCH or MM) and decimal places of precision (0-6).
NativeLengthUnits=INCH
NativeLengthDecimalPlaces=4

; TODO: Select the controller (PowerPmacController, TurboPmacController or MockController)
Controller=PowerPmacController

; TODO: Specify quantity of tool offsets (0 min, 25 max)
ToolOffsets=25

; TODO: Specify quantity of G54.1 work offsets (0 min, 48 max)
G541=25

; TODO: List G and M-code group names that are NOT required by the application (separated by commas).
; Note: Group0, Group6, ProgramGroup and SubprogramGroup may not be removed.
;ExtraneousGroups=Group11,Group22,ThreadingGroup,GearRangeGroup,BAxisGroup

[NC Files]
; Main and MDI program numbers are absolute.
MainProgramNumber=100
MdiProgramNumber=99
CoordinateSystem=1
SubprogramFolder="C:\NC"
; Subprogram numbers are relative to the specified Subprogram Base Address.
SubprogramBaseAddress=5000
; To disallow subprograms, set range to (0,-1)
NonvolatileSubprogramMin=0
NonvolatileSubprogramMax=99
VolatileSubprogramMin=100
VolatileSubprogramMax=199


[External Assemblies]
; TODO: Specify the assembly path and type name in quotes separated by a semicolon
; Object="path to DLL;FObject class name"
; UserPage="path to DLL;Page class name"
; CustomTab="path to DLL;Page class name"
; LeftCustomFrame="path to DLL;Page class name"
; CenterCustomFrame="path to DLL;Page class name"
; RightCustomFrame="path to DLL;Page class name"
; CodeGroups="path to DLL;class name"
; Example: Object="..\..\..\MySolution\MyProject\bin\Debug\MyProject.dll;MyNamespace.MyFObjectClass"
; Example: CenterCustomFrame="..\..\..\MySolution\MyProject\bin\Debug\MyProject.dll;MyNamespace.MyWpfPage"
; Example: CodeGroups="..\..\..\MySolution\MyProject\bin\Debug\MyProject.dll;MyNamespace.MyClass"

[Private Label]
; Optional private labeling. Images should be PNG or JPEG format and must be in the exe directory.
; Splash image should be around 500x300 pixels and login image should be around 1000x700.
;CompanyName="My Company Name"
;SplashImage="MySplashImage.png"
;LoginImage="MyLoginImage.jpg"
```

## Appendix C. Turbo PMAC Support

If your version of the Power PMAC-NC 14 program includes Turbo PMAC support then you may specify that controller in the application's "PowerPmacNC.ini" file.

```
[Machine Constructor]
; TODO: Select the controller (PowerPmacController, TurboPmacController or MockController)
Controller=TurboPmacController
```

 The Delta Tau Pro2 (*PcommServer*) communications library must be installed on the PC and properly configured. If your Turbo PMAC is not device 0 then the device number may be changed in the Power PMAC-NC 14 program by selecting the **Controller** object in *Machine View*.

### The Turbo PMAC Project

The "TurboPMAC\Configuration" folder contains a configuration file that can be downloaded to prepare your Turbo PMAC for use with the Power PMAC-NC 14 program.

"GitHub\PowerPmacNcRelease\PMAC Source Code\TurboPMAC\Configuration\TurboNcPlus\_Build.cfg"

Open this file in a text editor and uncomment the "tpnc\_umacdemobox.cfg" include line for a Demo Box test system or "tpnc\_virtualmotors.cfg" for a controller-only test setup running software-simulated motors.


```
//Comment in for UMAC Demo Box
//#Include "tpnc_umacdemobox.cfg"
//Comment in for virtual motors 1-8
//#Include "tpnc_virtualmotors.cfg"
```

Run the *PEWin32-PRO2 PMAC Executive Program* and use the Terminal window to issue a "\$\$\$\*\*" command to initialize the controller, then select the "Configure|M-variables" menu option and click the "Download Suggested M-variables" button. Now select the "Backup|Restore Configuration" menu option to download the "TurboNcPlus\_Build.cfg" configuration file.



Verify no warnings or errors in the Results window.

```
Device 0-> Total Warnings: 0
Device 0-> Total Errors: 0
Device 0-> END.
```

 After downloading the configuration, use the Terminal window to issue a "**SAVE**" command to copy to nonvolatile flash memory, then issue a "\$\$\$" command to reset the controller.

The Turbo PMAC controller is now ready to work with the Power PMAC-NC 14 program!

## Appendix D. Source Code Exclusions

### Included in the Source Code version:

- The WPF operator screens (Login, Main, and future screens) and associated logic (code-behind).
- The logic that defines the operation of the NC application.
- The device member definitions and G and M-code definitions.
- CS and Motor status monitoring.
- The NC file editor (AvalonEdit).
- Application skins support.
- Initialization file configuration support.
- The Mock Controller.
- Example External Assembly projects that demonstrates how to add custom panels and device members.

### NOT Included in the Source Code version:

- The MCF source code (Machine View, runtime engine, logging, alarming, foreign language support, MTConnect, etc)
- The low-level Power PMAC communications (sending commands, downloading files, etc)
- The NC file parsing (support for execution monitoring, subprograms, mid-tape start, etc)

## Appendix E. Send1 Command List

The following is a list of the “send1” commands used by the Power PMAC to communicate status and requests to the PPNC14:

```
send1 "initialized"  
// HMI Handshake - Send after initialization code has completed.  
  
send1 "resetcompleted"  
// HMI Handshake - Send after reset sequence code has completed.  
  
send1 "homecompleted"  
// HMI Handshake - Send after initialization code has completed.  
  
send1 "cyclestarted"  
// HMI Handshake - Send after cycle start.  
  
send1 "infeedhold"  
// HMI Handshake - Send after feedhold.  
  
send1 "programcompleted"  
// HMI Handshake - Send after program completes (See M2 and M30).  
  
send1 "programfailed"  
// HMI Handshake - Send after a program fails for any reason.  
  
send1 "programaborted"  
// HMI Handshake - Send after an operator issued abort.  
  
send1 "estoppressed"  
// HMI Status Request – Puts HMI into Estopped condition.  
  
send1 "estopreleased"  
// HMI Status Request – Puts HMI into Estop Clear condition.  
  
send1 "workoffsetsset"  
// HMI Read/Write Request – Initiates read of CS offset variables and writes to xml settings file.  
  
send1 "tooloffsetsset"  
// HMI Read/Write Request – Initiates read of tool offset variables and writes to xml settings file.  
  
send1 "pendantconnected"  
// HMI Status Request – Hardware Pendant connected, hide soft panel.  
  
send1 "pendantdisconnected"  
// HMI Status Request – Hardware Pendant not available, show soft panel.  
  
send1 "hidemanual"  
// HMI Status Request – Do not show Manual Mode Tab (soft jog screen).  
  
send1 "showmanual"  
// HMI Status Request – Show Manual Mode Tab (soft jog screen).  
  
send1 "manualsubmodenone"  
// HMI Status Request – Show “Manual Mode” only in mode button.  
  
send1 "manualsubmodecontinuous"  
// HMI Status Request – Show “Manual Mode - Cont” in mode button.
```



```
send1 "manualsubmodehandle"  
// HMI Status Request – Show “Manual Mode - Hand” in mode button.  
  
send1 "manualsubmodehome"  
// HMI Status Request – Show “Manual Mode - Home” in mode button.  
  
send1 "requestautomode"  
// HMI Request – Change to Auto Mode.  
  
send1 "requestmanualmode"  
// HMI Request – Change to Manual Mode.  
  
send1 "requestmdimode"  
// HMI Request – Change to MDI Mode.  
  
send1 "requestcyclestart"  
// HMI Request – Request Cycle Start.  
  
send1 "requestfeedhold"  
// HMI Request – Request Feedhold.  
  
send1 "requestabort"  
// HMI Request – Request Abort.  
  
send1 "requestreset"  
// HMI Request – Request Reset.  
  
send1 "requestoptionstop"  
// HMI Request – Toggle Option Stop.  
  
send1 "requestsingleblock"  
// HMI Request – Toggle Single Block.  
  
send1 "requestblockskip"  
// HMI Request – Toggle Block Skip.  
  
send1 "requestspindlecw"  
// HMI Request – Toggle Spindle CW.  
  
send1 "requestspindleccw"  
// HMI Request – Toggle Spindle CCW.  
  
send1 "requesttoolchangeplus"  
// HMI Request – Increment Manual Tool Change positive.  
  
send1 "requesttoolchangeminus"  
// HMI Request – Increment Manual Tool Change negative.  
  
send1 "requestjogspeed1"  
// HMI Request – Select Jog Speed1.  
  
send1 "requestjogspeed2"  
// HMI Request – Select Jog Speed2.  
  
send1 "requestjogspeed3"  
// HMI Request – Select Jog Speed3.  
  
send1 "requestjogspeed4"  
// HMI Request – Select Jog Speed4.
```

```

send1 "requestjogspeed5"
// HMI Request – Select Jog Speed5.

send1 "requestjog1"
// HMI Request – Request Axis 1 jog.

send1 "requestjog2"
// HMI Request – Request Axis 2 jog.

send1 "requestjog3"
// HMI Request – Request Axis 3 jog.

send1 "requestjog4"
// HMI Request – Request Axis 4 jog.

send1 "requestjog5"
// HMI Request – Request Axis 5 jog.

send1 "requestjog6"
// HMI Request – Request Axis 6 jog.

send1 "requestjog7"
// HMI Request – Request Axis 7 jog.

send1 "requestjog8"
// HMI Request – Request Axis 8 jog.

send1 "requestjog9"
// HMI Request – Request Axis 9 jog.

send1 "requestjog10"
// HMI Request – Request Axis 10 jog.

send1 "requesthome"
// HMI Request – Request Home Sequence.

send1 "jogging"
// HMI Status Request – Disable manual controls while incremental jogging.

send1 "jogstopped"
// HMI Status Request – Send jog stopped status.

send1 "canceled"
// HMI Handshake – Status Bar operation cancel sequence complete.

send1 "fatalmessage=xxxx"
// HMI Fatal Ack Message – Sends xxxx text to HMI as fatal message.

send1 "warningmessage=xxxx"
// HMI Warning Ack Message – Sends xxxx text to HMI as warning message.

send1 "informationmessage=xxxx"
// HMI Information Ack Message – Sends xxxx text to HMI as information message.

send1 "logmessage=xxxx"
// HMI Log Only Message – Sends xxxx text to HMI as Log Only message.

```

## Appendix F. Included G & M Codes

The following is a list of G & M codes included with the default project. It is important to note the user can add custom G & M codes as necessary to complete their requirements.

```
// G00 Rapid move mode declaration
// G01 Linear move mode declaration
// G02 Clockwise circle move mode declaration
// G03 Counter Clockwise circle move mode declaration
// G04 Dwell for time of F, P, or X value in seconds
// G09 Exact stop (non-modal)
// G17 XY plane declaration for circles and radius comp
// G18 ZX plane declaration for circles and radius comp
// G19 YZ plane declaration for circles and radius comp
// G20 Set English (inch) mode
// G21 Set metric (mm) mode
// G28 Return to Reference Point
// G40 Cutter radius compensation cancel
// G41 Cutter radius compensation on left
// G42 Cutter radius compensation on right
// G43 Tool Length Offset
// G44 Tool Length Offset (Minus Direction)
// G49 Tool Length Compensation Cancel
// G50 Cancel scaling
// G50.1 Disable mirroring
// G51 Set scaling factors
// G51.1 Enable mirroring
// G53 set machine coordinate system
// G54 - G59 set work coordinate system 1
// G54.1 Px set auxiliary work coordinate system offsets
// G61 set exact stop mode
// G64 set cutting mode
```

```
// G68 coordinate system rotation
// G69 Cancel rotation
// G73 High Speed Peck Drilling Cycle - Short Retract
// G76 Fine Boring Cycle
// G83 Peck Drilling Cycle - Long Retract
// G90 absolute move mode
// G90.1 absolute move mode (IJK Abs Arc Center)
// G91 incremental move mode
// G91.1 incremental move mode (IJK Inc Arc Center)
// G92 position set mode
// G93 Inverse Time
// G94 Feed Per Minute
// G98 Return Initial Level
// G99 Return R Level

// M0 - Feed Hold
// M1 - Option Stop
// M2 - Stop with Rewind
// M3 - Spindle CW
// M4 - Spindle CCW
// M5 - Spindle Stop
// M6 - Tool Change
// M30 - Stop with Rewind
// M98 - Sub-Program Call
// M99 - Return From Sub-Routine
```