THE ROLE OF POSTPROCESSOR IN THE TRANSLATION OF VECTOR GRAPHICS UNDERSTANDABLE TO THE CNC MACHINE CONTROLLER

Boris Pauković, Dražen Marinković
Pan-European University APEIRON Banjaluka
paukovic@hotmail.com, drazen.m.marinkovic@apeiron-edu.eu

Critical Review
https://doi.org/ 10.7251/JIT2001058P UDC: 621.9-52:004.388, 004.382:621.01

Abstract: The purpose of this paper is to show the post-processor role and importance in the creation of programming code which CNC machine controller can understand and proceed. Today, the use of CNC technology introduces the Industry 4.0 principles in the production, thus increasing the productivity and precision of the produced parts. It is very important to optimize all the production steps, from choosing the right CAD software for vectors drawing, defining tools and generating tool paths, creating and optimizing post-processor, to translating the tool paths in the programming code which controller of the CNC machine can understand, as well as educating the operators to be able to calibrate the machine and understand and run the CNC programming code properly. When all the abovementioned steps are correctly defined, the production can be optimized and best results are guaranteed.

Keywords: CNC technology, CAD/CAM software, postprocessor, CNC programming.

WHAT IS CNC PROGRAMMING

By defining the programming of CNC machines as creating commands that define pre-conceived actions to the control unit to obtain the final product, which in many cases is very expensive, we can conclude that it is very important that all steps are well analyzed and optimized. It is necessary to know the control unit that is located on the CNC machine and based on this information to enable the creation of program code.

There are two basic ways to create a program code:

1. Manual programming - the programmer writes the CNC program independently using commands known to the control unit. This way of programming is quite demanding, very slow and prone to errors because there is no simulation, which is a very important step, especially if the program will be executed on a CNC machine for the first time.
2. Automatic programming - this way you create drawings (2D or 3D models) in one of the CAD programs. The drawing is entered into the CAM program in which important parameters are defined such as: thickness of machining material, tool, starting point, position of the draft/model in the material, idle speed, speed of tool passing through the material, tool rotatin frequency, and with the help of a *processor* as a group of programs that determines the toolpath on the basis of input predefined parameters. In order for the control unit itself to successfully execute the pre-generated path, it is necessary to use the *postprocessor* as a compiler that converts the defined path into a language understandable to the control unit.

THE ROLE OF POSTPROCESSOR IN CREATING A PROGRAM CODE

A CNC postprocessor is a set of commands that converts a entered tool path into a language understandable to the control unit so that the tool can

move safely and consistently along a predefined path.

Most CAM software is designed to be independent of specific control units but to allow customization to each control unit. CAM software generates files that define tool paths and other necessary information to obtain the final product from the material, but this information is completely unusable on a specific CNC machine without the participation of postprocessors.

The main role of the postprocessor is to read the path from the files generated by the CAM software and convert it to G-code that will be understandable to the control unit.

It is important to emphasize that the translation into the program code for optimal use is not enough without knowing the mechanical components of the machine itself, because improper adjustment leads to mechanical failures that are usually very expensive and not covered by the manufacturer's warranty. Creating and optimizing postprocessors is not an easy task - it is necessary to have a very good command of programming skills and to be familiar with the principles of mechanical engineering, kinematics and mechatronics.

The following graphics represent a schematic representation of the process of generating program code from vector graphics and the role of postprocessor [2]:

What a postprocessor looks like - an example of translation of a program code

G-code is the most widely used language for numerical control today. The machine control unit uses the G-code to start the machine and execute the commands. The G-code was originally developed in 1958 at MIT (Massachusetts Institute of Technology) and is considered the industry standard for programming numerically controlled control units, known as RD-274. A finite number of basic control commands are available but depending on the software manufacturer can be expanded and specified according to customer needs.

A few basic commands which will be used in the representative postprocessor presented in this paper are:

G0 - linear movement that is simultaneous and coordinated for all axes that are present, in practice it is used to move and define the coordinates of the axles at idle.

Syntax example:

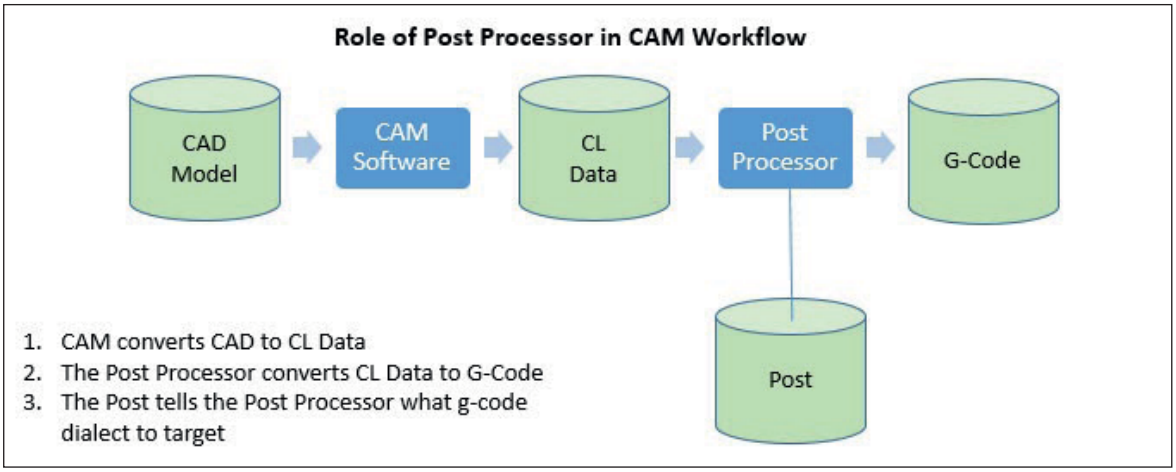
G0 [X] [Y] [Z] [F], where X, Y, Z are the coordinates of the three-axis CNC machine, F is the offset

Example of a program code (taken from a generated representative program code):

G0 X103.5887 Y94.4212 Z5.0000

G1 - linear motion that is simultaneous and coordinated for all axes present. In practice, it is used to move and define coordinate positions when the tool is in the material.

Figure 1: The role of postprocessor in the G-code creation process



Syntax example:

G1 [X] [Y] [Z] [F], where X, Y, Z are the coordinates of the three-axis CNC machine, F is the offset

Example of a program code (taken from a generated representative program code)

G1 Z-1.3333 F720.0
G1 X125.3425 F2520.0
G1 X116.7836 Y117.2038

G2 - Circular interpolation clockwise

Syntax example:

G2 [X] [Y] R [Radius] [F], where X, Y, Z are the coordinates of the three-axis CNC machine, R is the radius, F is the offset

Example of a program code (taken from a generated representative program code)

G2 X114.1735 Y124.8695R184.856
G2 X110.9580 Y114.1032R92.383

G3 - Circular interpolation counterclockwise

Syntax example:

G2 [X] [Y] R [Radius] [F], where X, Y, Z are the coordinates of the three-axis CNC machine, R is the radius, F is the offset

Example of a program code (taken from a generated representative program code)

G3 X187.3316 Y102.6650R38.223
G3 X192.2292 Y103.7325R20.928
G3 X194.6210 Y104.9136R11.432

There are also M-codes, with the help of which the control unit sends signals for starting the spindle, the direction of rotation of the spindle, measuring the tool, replacing the tool, etc.

M code	Description	Syntax example
M3	Start spindle rotation	M03 [S], where the variable S is the spindle rotation speed
M5	Stop spindle rotation	M5
M6	Tool replacement	M06 T [T], where the variable T is the number of tools in the tool holder
M30	Arrival at the starting point	M30

In the postprocessor, we define all the necessary parameters that are necessary for the G-code to be completely understandable to the control unit. Only then can the control unit execute all desired commands. The postprocessor consists of several parts:

1. Basic information - where the name of the postprocessor is most often defined (indicates the name of the control unit for which the postprocessor is in charge and will be displayed in the CAD/CAM software when selecting a postprocessor for G-code generation); extension of this unit of measurement:

Example:

```
DESCRIPTION = "OSAI OPEN CONTROL(*.cnc)"
FILE_EXTENSION = "cnc"
UNITS = mm
```

2. Defining the numbering of program blocks:

Example:

```
; Block numbering;
LINE_NUM_START = 10
LINE_NUM_INCREMENT = 10
LINE_NUM_MAXIMUM = 9999999
START = "N[N] G300 F4"
```

3. Defining the format of variables:

Example:

```
; Line numbering
FORMAT = [N|@|N|1.0]
; Spindle Speed
FORMAT = [S|@|S|1.0]
; Feed Rate
FORMAT = [F|#|F|1.1]
; Tool moves in x,y and z
FORMAT = [X|#|X|1.4]
FORMAT = [Y|#|Y|1.4]
FORMAT = [Z|#|Z|1.4]
```

4. Program header where it is defined which tool will be used first in machining, material dimensions, starting point, starting of the spindle and so on.

Example:

```
START = ";First tool:[T] [TOOLDESC]"
START = ";Material dimensions: X=[XSIZE],
    Y=[YSIZE], Z=[ZMATERIAL]"
START = "[N] G90"
START = "[N] G40"
START = "[N] G80"
START = "[N] (UAO,1)"
START = "[N] M06 T[T]"
START = "[N] M03 [S]"
```

5. Definition of movement:

Example:

```
;Linear section:
RAPID_RATE_MOVE = "[N] G0 [X] [Y] [Z]"
FIRST_FEED_RATE_MOVE = "[N] G1 [X] [Y] [Z] [F]"
FEED_RATE_MOVE = "[N] G1 [X] [Y] [Z]
; Arc Section:
CW_ARC_MOVE = "G2 [X] [Y]R[Radius] [F]"
CCW_ARC_MOVE = "G3 [X] [Y]R[Radius] [F]"
```

6. End of the program (spindle shutdown, return to starting point or predefined point with other possible options).

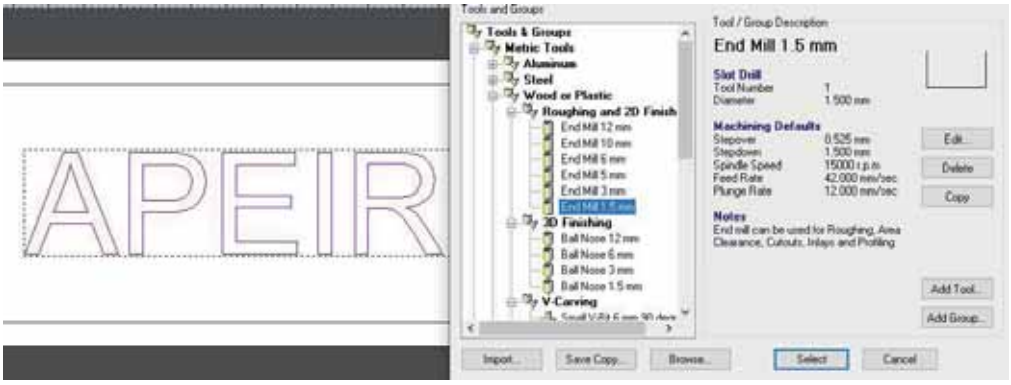
Example:

```
; End of file
END = "[N] G80"
END = "[N] M05"
END = "[N] M30"
```

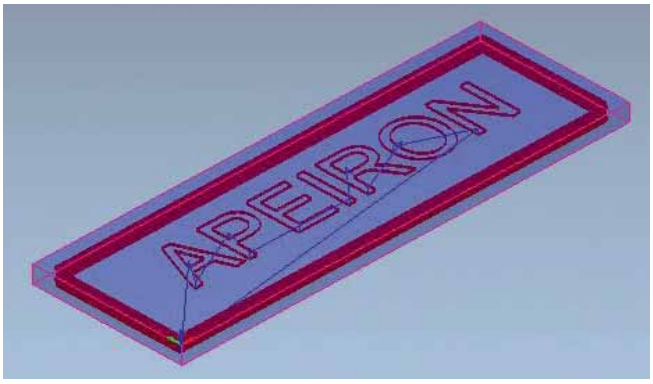
An example of a draft that has been converted into a G-code understandable to the control unit of the Italian manufacturer OSAI with the help of a postprocessor created on the basis of the previously described basic procedures.

1. Step: Use CAD software to create or import an existing vector graphic:

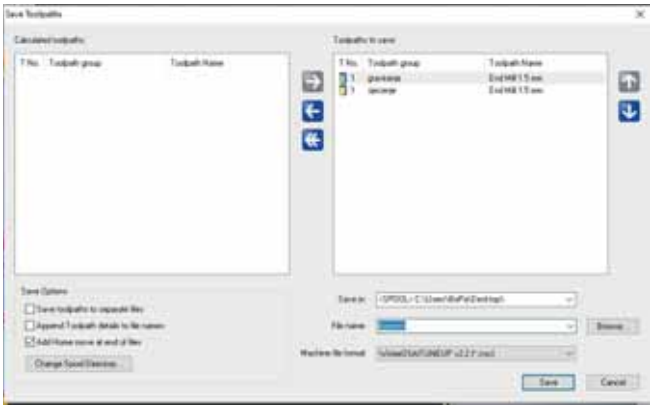
2. Select the tools:



3. Create a toolpath:



4. Generate a G-code with the help of a postprocessor:



Header and the first few lines of the code:

```
N10 G300 F4
;First tool 1 1.500 mm dia. slot drill
;Material dimensions : X=600.000, Y=200.000,
    Z=25.000
N40 G90
N50 G40
N60 G80
N70 (UAO,1)
N80 M06 T1
N90 M03 S15000
N100 M49
N110 h1
N120 G27
N130 G17
N140 G0 X103.5887 Y94.4212 Z5.0000
N150 G1 Z-1.3333 F720.0
N160 G1 X125.3425 F2520.0
N170 G1 X116.7836 Y117.2038
G2 X114.1735 Y124.8695R184.856
G2 X110.9580 Y114.1032R92.383
N200 G1 X103.5887 Y94.4212
N210 G1 Z-2.6667 F720.0
N220 G1 X125.3425 F2520.0
N230 G1 X116.7836 Y117.2038
G2 X114.1735 Y124.8695R184.856
G2 X110.9580 Y114.1032R92.383
N260 G1 X103.5887 Y94.4212
N270 G1 Z-4.0000 F720.0
N280 G1 X125.3425 F2520.0
N290 G1 X116.7836 Y117.2038
G2 X114.1735 Y124.8695R184.856
```

OSAI OPENCONTROL CONTROLLER HARDWARE STRUCTURE

OpenControl is a family of controllers designed by the Italian manufacturer OSAI. The most powerful configurations can support up to 64 digital axes. It is installed and controls numerous types of machines; CNC milling machine, sharpener, marble processing machine, laser cutters and other. Thanks to the "OPEN" philosophy, it is quite flexible and is very popular among manufacturers, as well as final users due to its simple and intuitive user interface. The user interface is known as "WinNBI" (Windows Network Based Interface) based on the Windows Embedded CE operating system, known for its stability and durability.

The Windows Embedded CE system in this environment retains the basic functionality of the Windows operating system with the ability to run the software required to manage the machine itself.

OpenControl systems are modular based and can be combined as needed. In the most widespread and popular setting, the system consists of an operator panel where parameters are manipulated and commands to start the machine. The following figure [2] shows the logical setup of the system consisting of a panel connected to an industrial computer, an OpenControl controller, a central part of the system that communicates using TCP/IP protocol with the industrial computer and further with the IO module using the standard fieldbus protocol.) and

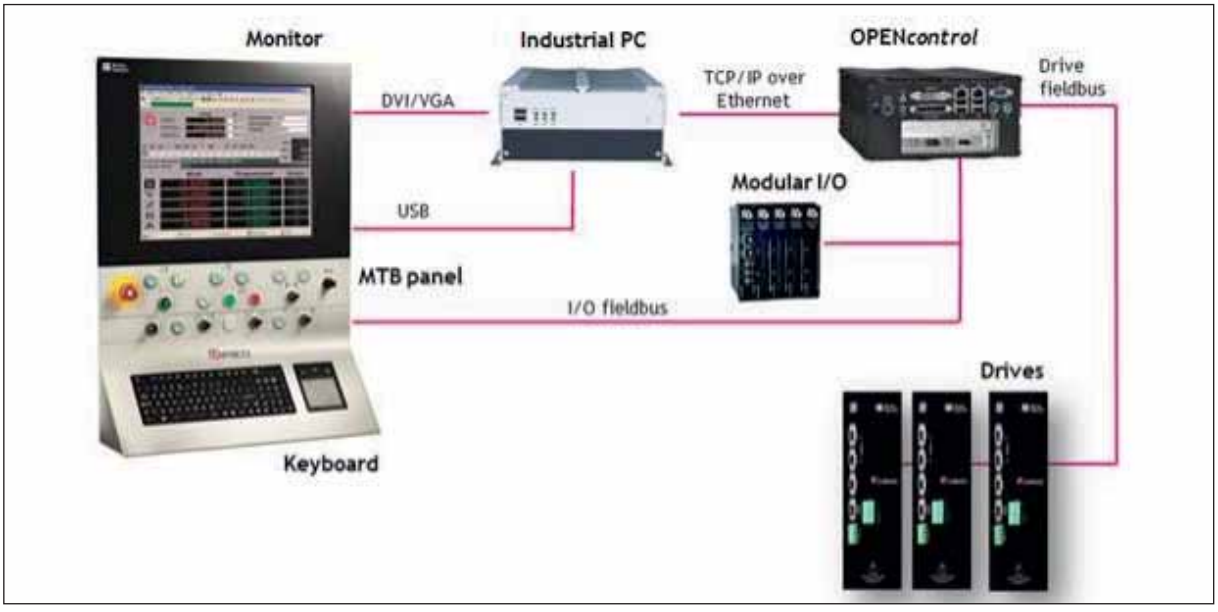


Figure 2: A display of OSAI OpenControl system

recently with the help of EtherCAT protocol (Ethernet for Control Automation Technology) - Ethernet based fieldbus protocol, created by the German manufacturer Beckoff Automation, standardized as IEC61158, offering great stability and reliability required by real-time computing and automation, with drivers that allow the machine to start and execute commands.

A practical example of motion optimization of a CNC machine for embossing using paramacro subroutines

Optimization of CNC machine movement is crucial when creating specific projects. The speed of production, along with the quality of the finished product with minimal wear of the components of the machine itself is crucial during installation and initial setup. This is often not taken into account in today's production.

The following example will show the basic functions of paramacro subroutines with a practical example where the processing speed is increased by up to 40% thus satisfying the demand of the final customer.

Based on the instructions for OSAI controller developers, we come to the conclusion that paramacro subroutines can be used in user-defined cycles and are called with a three-digit code.

Basic syntax of paramacro subroutine [1]:

Gn parameter-name-1 [value-1].... [parameter-name-n] [value-n]..... [string]

whereby:

n	a number from 300 to 998
parameter-name-1....parameter-name-n	letters (defined based on Table 2
value-1....value-n	can be a number or E parameter
string	a string of characters (maximum 99)

There are two basic groups of paramacro subroutines:

1. Non-modal paramacros (G300-G699)
2. Modal paramacros (G700 - G998)

The main difference is that modal paramacros can only be active in blocks that do not have M functions. No other function or code can be active or executed while the modal paramacro is active.

In this example, we will use the non-modal paramacro subroutine G300 that will be called in the postprocessor and will perform optimal acceleration of the axles in order to obtain the speed in operation:

```
;PARAMACRO G300
"1" ;SLOW
ERF=0.05
MDA=80
;VELOCITY LEAP
(CPA,W,X,92,100)
(CPA,W,Y,92,100)
(CPA,W,Z,92,100)
;WORK JRK
(CPA,W,X,16,10000)
(CPA,W,Y,16,10000)
(CPA,W,Z,16,20000)
;WORK DEJRK
(CPA,W,X,122,10000)
(CPA,W,Y,122,10000)
(CPA,W,Z,122,20000)
;SPLINE
TBS(X)=0.01
TBSG0(X)=0.2
TBS(Y)=0.01
TBSG0(Y)=0.2
TBS(Z)=0.01
TBSG0(Z)=0.2
(GTO,END)
;
```

In the above example, we see the difference in the parameters VELOCITY, JERK, DEJERK, which are directly responsible for the speed of the machine itself. The solution presented using subroutines represents an elegant change of essential parameters with small corrections within the program code of the postprocessor, while manual adjustment and

reprogramming would result in several hours of adjustment with a high possibility of errors that often cause major mechanical failures and production downtime. Also, the extension and additional increase of speed is enabled by adding new F parameters and expanding the subroutine in a simple way.

After the successful definition of paramacro subroutines, it is necessary to perform the integration with the postprocessor used for the generated program G-code recognizable by the CNC machine controller itself.

Most often, the paramacro subroutine is called and defined in the postprocessor header itself, as in the following example:

```
"2" ;FAST
ERF=0.15
MDA=80
;VELOCITY LEAP
(CPA,W,X,92,200)
(CPA,W,Y,92,200)
(CPA,W,Z,92,200)
;WORK JRK
(CPA,W,X,16,20000)
(CPA,W,Y,16,15000)
(CPA,W,Z,16,30000)
;WORK DEJRK
(CPA,W,X,122,20000)
(CPA,W,Y,122,15000)
(CPA,W,Z,122,30000)
;SPLINE
TBS(X)=0.05
TBSG0(X)=0.2
TBS(Y)=0.05
TBSG0(Y)=0.2
TBS(Z)=0.05
TBSG0(Z)=0.2
```

```
DESCRIPTION = "OSAITUNEUP v2.2 (*.cnc)"
;
FILE_EXTENSION = "cnc"
;
UNITS = mm
;
; Carriage return - line feed at end of each line
;
END_OF_LINE = "[13][10]"
;
; Block numbering
;
LINE_NUM_START = 10
LINE_NUM_INCREMENT = 10
LINE_NUM_MAXIMUM = 9999999
START = "N[N] G300 F2"; here we can see that the called paramacro is G300 and ;defined speed F2, FAST
;
; Set up default formatting for variables
; .....
```

CONCLUSION:

The development of CNC machines and affordability in recent years has enabled this technology to come into widespread use. Modernization of production of CNC machines enables greater efficiency and profitability. Machines are becoming more common in many industries. Postprocessor as one of the more important items in this system is credited with the proper transmission of information and the creation of commands that the machine understands. By using a postprocessor, errors in writing programs are minimized. Proper setup and optimization of the postprocessor allows you to get the most out of the machine, speed up production, increase productivity and thus profit.

REFERENCES:

Web Page

[3] https://www.cnccookbook.com/cnc-post-processor-cam/, [Accessed: 12. April 2020]

PDF Offprint

[4] Prima Electro SpA; Open Control Programming Manual, Document Number 45004607K, Release Ref. V3.1, Edition 04 February2013

[5] Prima Electro SpA; Open Control End User Manual, Document Number 4602H, Edition 00, June 2013.

Submitted: April 4, 2020
Accepted: June 8, 2020.

About the authors

Boris Paukovic was born in Sanski Most in 1988. He is currently a master candidate at the Pan-European University APEIRON at the College of Information Technologies.
From 2010 to 2012 he worked as an Information Technology Support Specialist at the company Computing SYSTEMS d.o.o in Banjaluka.
Since 2013 he has lived and worked in Leipzig, Germany and gained experience in the usage of information technologies in CNC Machinery and the administration of modern software solutions for architects.

Dražen M. Marinković was born in Banja Luka, Republika Srpska - Bosnia and Herzegovina. He finished elementary school in Prnjavor. He finished secondary school of Electrical Engineering in Prnjavor. He graduated at the Faculty of Information Technologies at Pan-European University "APEIRON" in Banja Luka. Master studies completed in 2015. Doctoral studies of the third degree enrolled in the academic 2015/2016. He lives and works in Prnjavor.

For citation

Boris Pauković, Dražen M. Marinković, The Role of Postprocessor in the Translation of Vector Graphics Understandable to the CNC Machine Controller, *JITA – Journal of Information Technology and Applications Banja Luka*, PanEuropien University APEIRON, Banja Luka, Republika Srpska, Bosna i Hercegovina, JITA 10(2020) 1:58-65, (UDC: 621.9-52:004.388, 004.382:621.01), (DOI: 10.7251/JIT2001058P), Volume 10, Number 1, Banja Luka, June 2020 (1-68), ISSN 2232-9625 (print), ISSN 2233-0194 (online), UDC 004