

# Server API

This article serves as the main reference point for all myCNC material related to the myCNC Server API. It will be expanded at a full manual.

The server API allows to control movement and machine coordinates remotely, via a series of commands. A list of such commands is presented below for jog control, etc.

## Server API, Jog control

Command “SetJog” is used for Jog control through Sever API. Command syntax is “SetJog Parameter”. Parameters listed in table below:

| Parameter | Description                          |
|-----------|--------------------------------------|
| x+        | start jog X+ direction               |
| x-        | start jog X- direction               |
| y+        | start jog Y+ direction               |
| y-        | start jog Y- direction               |
| z+        | start jog Z+ direction               |
| z-        | start jog Z- direction               |
| a+        | start jog A+ direction               |
| a-        | start jog A- direction               |
| b+        | start jog B+ direction               |
| b-        | start jog B- direction               |
| c+        | start jog C+ direction               |
| c-        | start jog C- direction               |
| x+y+      | start jog in 2 axes X+,Y+ directions |
| x-y-      | start jog in 2 axes X-,Y- directions |
| x+y-      | start jog in 2 axes X+,Y- directions |
| x-y+      | start jog in 2 axes X-,Y+ directions |
| xstop     | stop jog in X axis                   |
| ystop     | stop jog in Y axis                   |
| zstop     | stop jog in Z axis                   |
| astop     | stop jog in A axis                   |
| bstop     | stop jog in B axis                   |
| cstop     | stop jog in C axis                   |
| xystop    | stop jog in X, Y axes                |

## Server API, Pendant control emulation

Server API has a command to simplify Pendant control emulation. This command can be used for writing third-patry Pendant control device drivers.

## SetPendant

**Command:** SetPendant **Format:** SetPendant Parameter1;Parameter2

- Parameter1 - Emulated Hand wheel position. It's supposed the handwheel has resolution of "100", such that the position range can be (0...99)
- Parameter2 - 16-bit value, high byte is Step selector, low byte is Axis selector.
- Step selector values can be set through the following format:
  - 0 → "x0.001" (1 wheel step = 0.001 unit)
  - 1 → "x0.01" (1 wheel step = 0.01 unit)
  - 2 → "x0.1" (1 wheel step = 0.1 unit)
  - 3 → "x1" (1 wheel step = 1 unit)
- Axis selector values:
  - 0 → "OFF"
  - 1 → "X"
  - 2 → "Y"
  - 3 → "Z"
  - 4 → "A"
  - 5 → "B"
  - 6 → "C"

| Example              | Description                                                                          |
|----------------------|--------------------------------------------------------------------------------------|
| SetPendant 50;0x0300 | Axis=OFF, Mul=1, Wheel=50 - initial position                                         |
| SetPendant 50;0x0301 | Axis=X, Mul=1, Wheel=50 - Turn ON X axis                                             |
| SetPendant 55;0x0301 | Axis=X, Mul=1, Wheel=55 - Rotate wheel +5 "clicks", will move X +5*1.0=5.0 mm        |
| SetPendant 55;0x0302 | Axis=Y, Mul=0.01, Wheel=55 - Switch to Y axis                                        |
| SetPendant 65;0x0302 | Axis=Y, Mul=0.01, Wheel=65 - Rotate wheel +10 "clicks", will move Y +10*0.01=+0.1 mm |
| SetPendant 65;0x0302 | Axis=Z, Mul=0.1, Wheel=65 - Switch to Z axis                                         |
| SetPendant 50;0x0302 | Axis=Z, Mul=0.1, Wheel=50 - Rotate wheel -15 "clicks", will move Z -15*0.1=-1.5 mm   |

## Hardware outputs (binary, DAC, PWM) manipulation through Server API

There are 3 ways to manipulate myCNC control board Hardware outputs through Server API:

1) Direct manipulation through commands "SetHWBinaryOutput", "SetHWDAC", "SetHWPWM" 2) Manipulation through running PLC procedure with Hardware Outputs manipulation. 3) Manipulation through running short G/M-codes program

### Direct Manipulation

#### Set HW Binary Output

**Command:** SetHWBinaryOutput

**Syntax:** SetHWBinaryOutput <Port number> <Port value>

Port Number is number binary port. Its range is 0 to 159. Set Port value to “1” to turn ON binary output (relay, open collector), “0” to turn OFF the binary output.

| Example                | Description                |
|------------------------|----------------------------|
| SetHWBinaryOutput 0 1  | Turn ON binary output #0   |
| SetHWBinaryOutput 1 1  | Turn ON binary output #1   |
| SetHWBinaryOutput 23 1 | Turn ON binary output #23  |
| SetHWBinaryOutput 0 0  | Turn OFF binary output #0  |
| SetHWBinaryOutput 1 0  | Turn OFF binary output #1  |
| SetHWBinaryOutput 19 0 | Turn OFF binary output #19 |

### Set HW PWM

*Command:* SetHWPWM

*Syntax:* SetHWPWM <PWM channel> <PWM value>

PWM channel is the number of the PWM pin. Software range is 0 to 3, however the number of PWM channels depends on the myCNC Ethernet controller model. Controllers ET1, ET3, ET5 each contain 3 PWM pins.

| Example         | Description                                                       |
|-----------------|-------------------------------------------------------------------|
| SetHWPWM 0 2000 | Set value “2000” to PWM#0 (PWM1). Max PWM value is 4095           |
| SetHWPWM 0 0    | Set value “0” to PWM#0 (PWM1). Max PWM value is 4095              |
| SetHWPWM 2 3500 | Set value “3500” to PWM#2 (PWM3). Max PWM value is 4095           |
| SetHWPWM 1 4095 | Set value “4095” (Maximum) to PWM#1 (PWM2). Max PWM value is 4095 |

### Set HW DAC

*Command:* SetHWDAC

*Syntax:* SetHWDAC <DAC channel> <DAC value>

DAC channel is number of the DAC pin. Software range is 0 to 1, however real number of DAC channels depends on myCNC Ethernet controller model. Controllers ET1 and ET3 each contain 1 DAC pins, while the ET5 controller contains 6 DAC pins.

| Example         | Description                                                |
|-----------------|------------------------------------------------------------|
| SetHWDAC 0 1000 | Assign value “1000” to DAC#0 (DAC1). Max DAC value is 4095 |
| SetHWDAC 0 0    | Assign value “0” to DAC#0 (DC1). Max DAC value is 4095     |
| SetHWDAC 1 4095 | Assign value “4095” (Maximum) to DAC#1 (DAC2)              |

### Manipulation through PLC procedure

Server API command “RunPLC” loads and runs PLC procedure on the controller end with a given parameter. We can run a PLC procedure that manipulates Hardware outputs to get a similar result. Normally every profile configuration contains PLC procedures for Hardware outputs manipulation :

**M62**

*PLC procedure:* M62 *Syntax:* M62 <Parameter> M62 allows the user to turn ON a binary output pin with a given number (Parameter=Port number)

| Example | Description               |
|---------|---------------------------|
| M62 1   | Turn ON binary output #1  |
| M62 0   | Turn ON binary output #0  |
| M62 15  | Turn ON binary output #15 |

**M63**

*PLC Procedure:* M63 *Syntax:* M63 <Parameter>

M63 allows the user to turn OFF a binary output pin with a given number (Parameter=Port number)

| Example | Description                |
|---------|----------------------------|
| M63 1   | Turn OFF binary output #1  |
| M63 0   | Turn OFF binary output #0  |
| M63 15  | Turn OFF binary output #15 |

**DAC**

*Syntax:* DAC <Parameter>

The low 12 bits (11...0) contain the value of DAC to be written, high bits (14..12) contain the DAC channel.

| Example    | Description                                                     |
|------------|-----------------------------------------------------------------|
| DAC 0      | Set value "0" to DAC#0 (DAC1). Max DAC value is 4095            |
| DAC 0x0600 | Set value "0x600" (1536) to DAC#0 (DAC1). Max DAC value is 4095 |
| DAC 0x1400 | Set value "0x400" (1024) to DAC#1 (DAC2). Max DAC value is 4095 |
| DAC 0x1000 | Set value "0" to DAC#1 (DAC2). Max DAC value is 4095            |

**PWM**

*Syntax:* PWM <Parameter>

The low 12 bits (11...0) contain the value of PWM to be written, high bits (14..12) contain the PWM channel number.

| Example    | Description                                                     |
|------------|-----------------------------------------------------------------|
| PWM 0      | Set value "0" to PWM#0 (PWM1). Max DAC value is 4095            |
| PWM 0x0600 | Set value "0x600" (1536) to PWM#0 (PWM1). Max PWM value is 4095 |
| PWM 0x1400 | Set value "0x400" (1024) to PWM#1 (PWM2). Max PWM value is 4095 |
| PWM 0x2000 | Set value "0" to PWM#2 (PWM3). Max PWM value is 4095            |

Examples of running these PLC procedures through the Server API are shown table below:

| Binary input manipulation |                   |
|---------------------------|-------------------|
| Example                   | Description       |
| RunPLC M62 1              | Turn ON output#1  |
| RunPLC M62 0              | Turn ON output#0  |
| RunPLC M63 3              | Turn OFF output#3 |
| RunPLC M63 5              | Turn OFF output#5 |

| PWM control       |                                        |
|-------------------|----------------------------------------|
| Example           | Description                            |
| RunPLC PWM 0x1350 | Write to PWM#1 (PWM2) a value of 0x350 |
| RunPLC PWM 0x250  | Write to PWM#0 (PWM1) value 0x250      |
| RunPLC PWM 0x2800 | Write to PWM#2 (PWM3) value 0x800      |
| RunPLC PWM 0x0    | Write to PWM#0 (PWM1) value 0          |
| RunPLC PWM 0x1000 | Write to PWM#1 (PWM2) value 0          |
| RunPLC PWM 0x2000 | Write to PWM#2 (PWM3) value 0          |

| SetHWDAC          |                                   |
|-------------------|-----------------------------------|
| Example           | Description                       |
| RunPLC DAC 0x1570 | Write to DAC#1 (DAC2) value 0x570 |
| RunPLC DAC 0x800  | Write to DAC#0 (DAC1) value 0x800 |
| RunPLC DAC 0x0    | Write to DAC#0 (DAC1) value 0     |
| RunPLC DAC 0x1000 | Write to DAC#1 (DAC2) value 0     |

## Manipulation through running G-codes

Server API command “SinglePlay” will run a G-code line (or a number of G-code lines separated by the “;” symbol). It's possible to control binary outputs by running M-functions “M62” and “M63” (similar to the method described above). P-parameter sets the binary output number to control.

*M-code: M62 Syntax: M62 P<number>*

This will turn ON the binary output pin with the given number.

| Example | Description               |
|---------|---------------------------|
| M62 P1  | Turn ON binary output #1  |
| M62 P0  | Turn ON binary output #0  |
| M62 P15 | Turn ON binary output #15 |

*Syntax: M63 P<number>*

This will turn OFF the binary output pin with the given number.

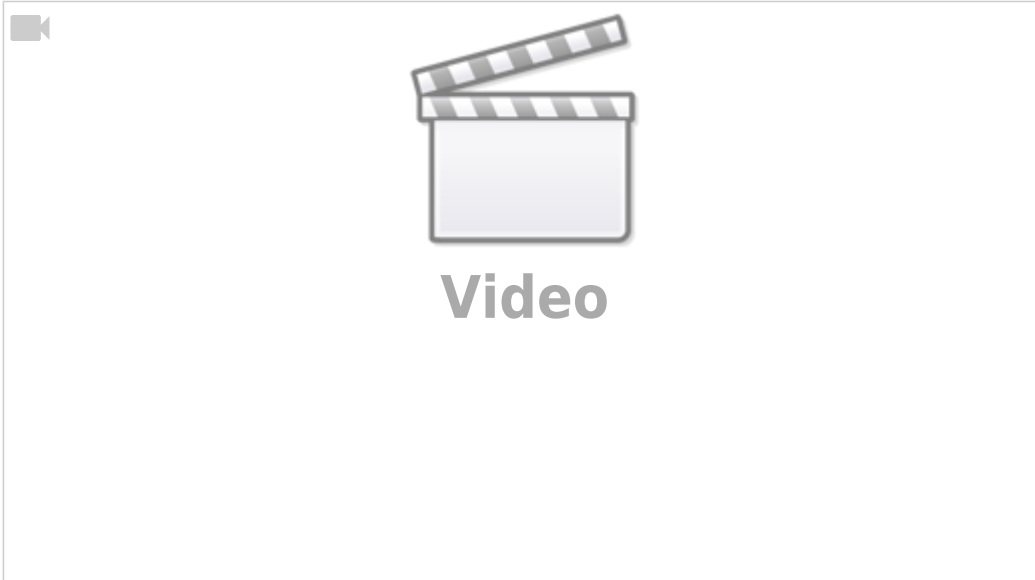
| Example | Description                |
|---------|----------------------------|
| M63 P1  | Turn OFF binary output #1  |
| M63 P0  | Turn OFF binary output #0  |
| M63 P15 | Turn OFF binary output #15 |

Any custom PLC function can be written and assigned to M-code, then loaded through the CNC Server API.

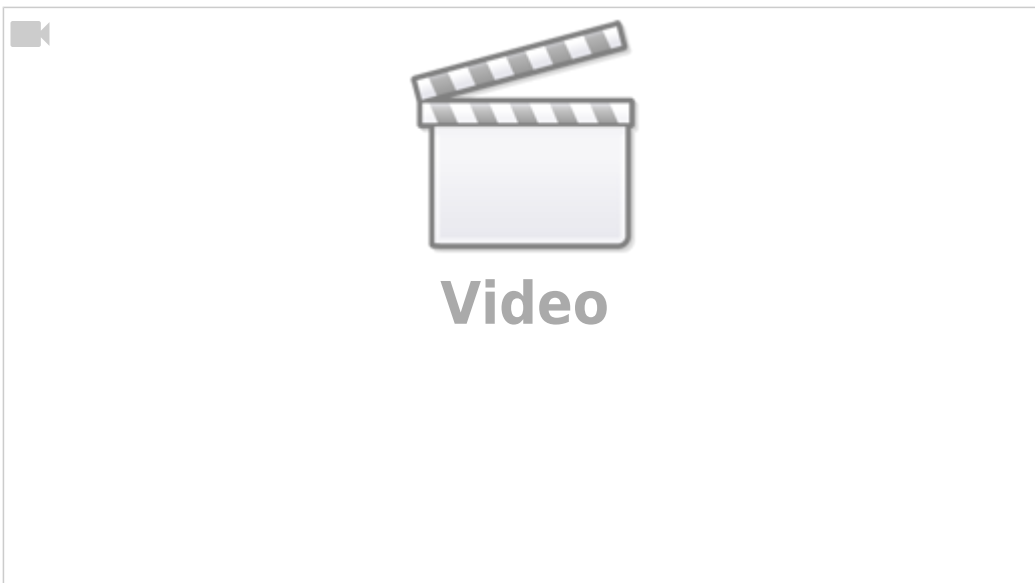
## Demo Videos

The following videos show examples of using the server API to control a myCNC system (older versions of the software are used, however the principle is the same):

ProgramFileOpen and SetScale:



Hardware Binary Outputs:



Set Scale and Rotation Angle:





Video



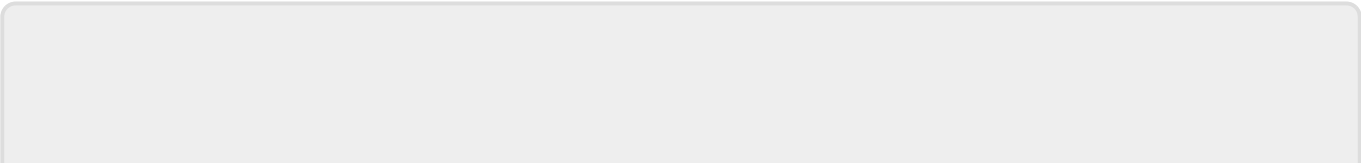


Video





Video



From:

<http://docs.pv-automation.com/> - **myCNC Online Documentation**

Permanent link:

[http://docs.pv-automation.com/mycnc/server\\_api?rev=1571761123](http://docs.pv-automation.com/mycnc/server_api?rev=1571761123)

Last update: **2019/10/22 12:18**

