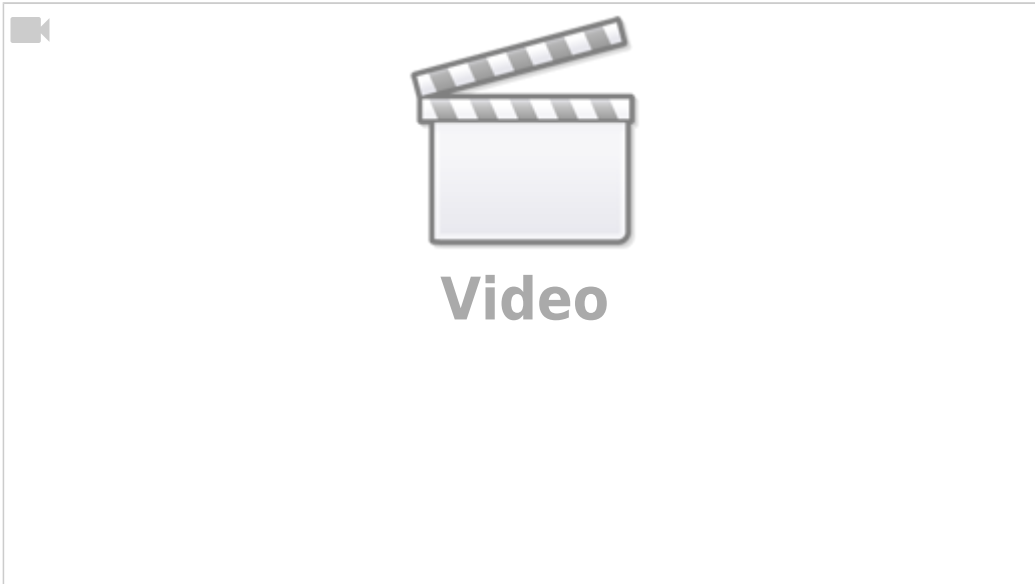


Eparam

Video manual on eparam usage is available below:



If an M-code (macro) is running from a G-code program, 16-bit integer parameters P and L are sent to the PLC procedure in the **eparam** variable.

P-parameter is in a low word of the eparam value and L-parameter is in high word of eparam value. To decode P and L parameters from **eparam** 2 simple lines of code can be used in the necessary PLC procedure:

```
P=eparam&0xFFFF; //P-parameter  
L=eparam>>16; //L-parameter
```

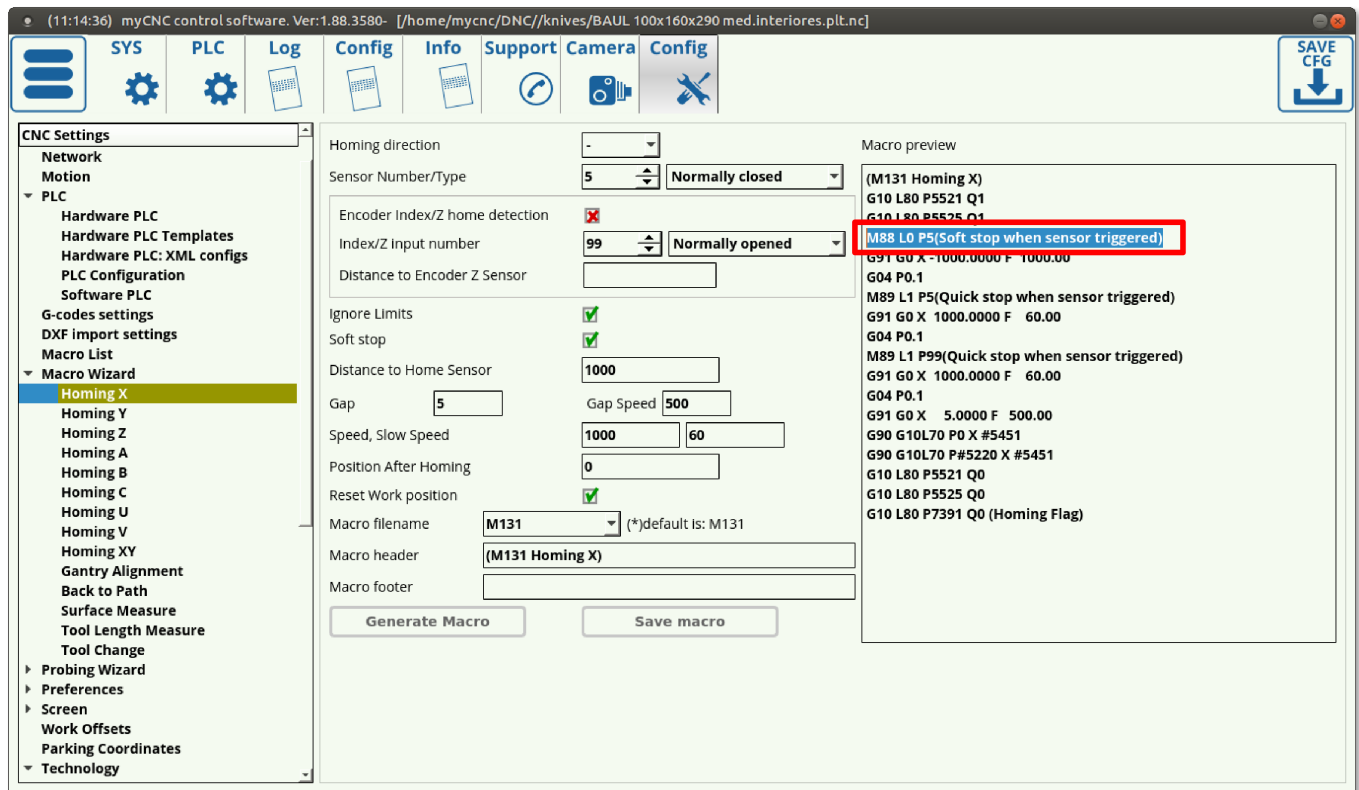
Eparam is often used in situations where it is necessary to send some value from the G-code or macro to a hardware PLC command which will use and incorporate this value. A good example of such a macro is the G-code for the X-axis homing procedure, since it relies on calling on a couple of Hardware PLC commands in order to perform the homing procedure:

Homing X

```
(M131 Homing X)  
G10 L80 P5521 Q1  
G10 L80 P5525 Q1  
M88 L0 P5(Soft stop when sensor triggered)  
G91 G0 X -1000.0000 F 1000.00  
G04 P0.1  
M89 L1 P5(Quick stop when sensor triggered)  
G91 G0 X 1000.0000 F 60.00  
G04 P0.1  
M89 L1 P99(Quick stop when sensor triggered)  
G91 G0 X 1000.0000 F 60.00  
G04 P0.1  
G91 G0 X 5.0000 F 500.00  
G90 G10L70 P0 X #5451
```

```
G90 G10L70 P#5220 X #5451
G10 L80 P5521 Q0
G10 L80 P5525 Q0
G10 L80 P7391 Q0 (Homing Flag)
```

Line 4 is using the L and P parameters to send an eparam value to the M88 macro.



Below is the code for the M88 macro itself:

M88.plc

```
#include common.const.h
//Watch on given sensor number, Soft Stop if sensor triggered
//used for homing, surface measure, tool length measure etc

#define input  var00
#define state  var01

main ()
{

message=PLCCMD_TRIGGER3_OFF;
textit=timer+2;do{timer++;}while(timer<textit);
message=PLCCMD_TRIGGER4_OFF;
textit=timer+2;do{timer++;}while(timer<textit);

input=eparam&0xFFFF; //P-parameter
state=eparam>>16; //L-parameter
```

```
timer=0;

message=PLCCMD_MOTION_CONTINUE;
texit=timer+30;do{timer++;}while(timer<texit);

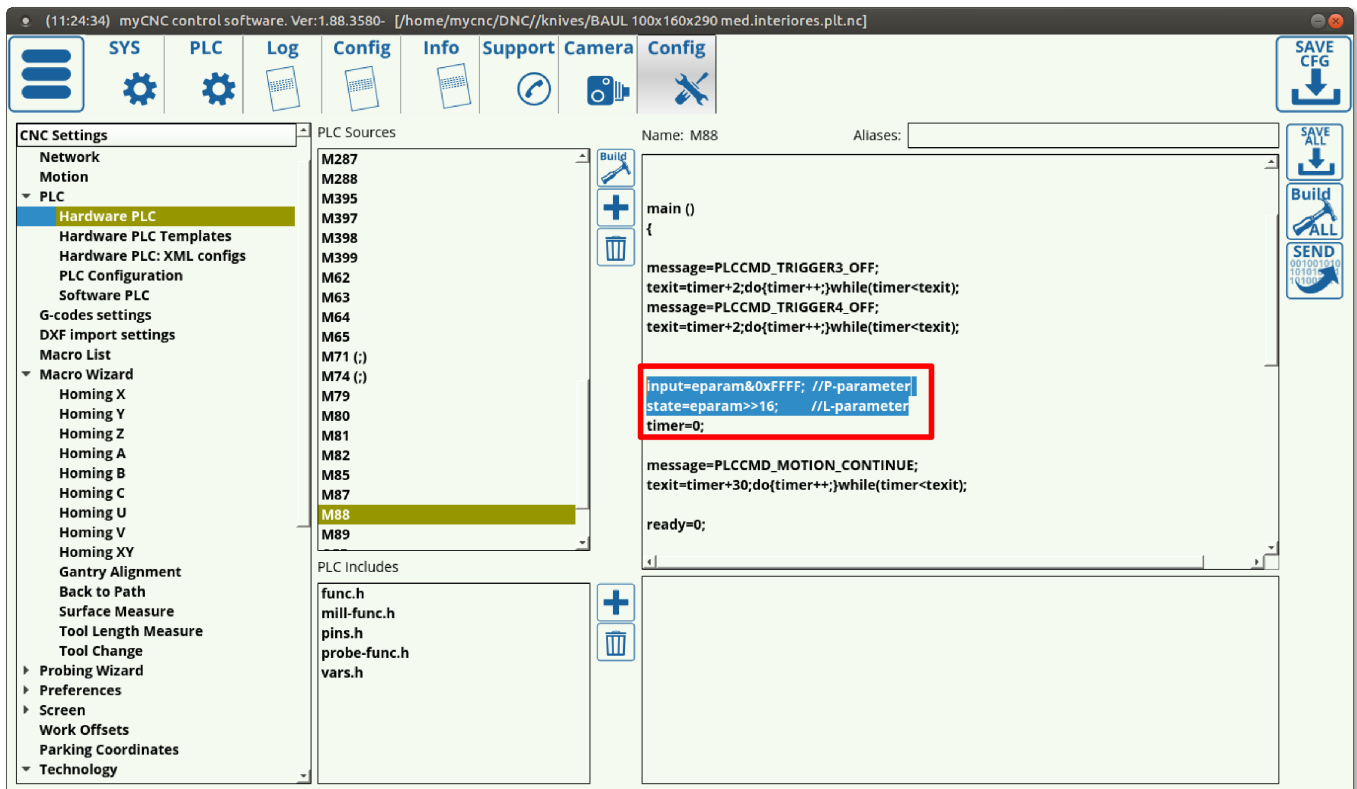
ready=0;

do
{
    timer++;
    a=portget(input);
    if (state==0)
    {
        if (a==0) {ready=1;};
    };
    if (state!=0)
    {
        if (a!=0) {ready=1;};
    };
}while(ready==0);

//message=PLCCMD_MOTION_SKIP;
message=PLCCMD_MOTION_SOFT_SKIP;
texit=timer+2;do{timer++;}while(timer<texit);

exit(99);
};
```

In this M88 code, you can see the eparam variable being “imported” into the macro that has been summoned by the G-code command. In this particular case, this allows to set the state and the necessary input number to monitor in order to initiate a soft stop when a particular sensor is triggered.



M03

A unique case of using the eparam value can be seen in the M03/SPN macros (spindle control). The M03 macro code is presented below:

M03

```
//Turn on Spindle clockwise
#include pins.h
#include vars.h
main()
{
    timer=0;
    proc=plc_proc_spindle;

    val=eparam;
    if (val>0xffff) {val=0xffff;};
    if (val<0) {val=0;};

    dac01=val;

    portclr(OUTPUT_CCW_SPINDLE);
    portset(OUTPUT_SPINDLE);

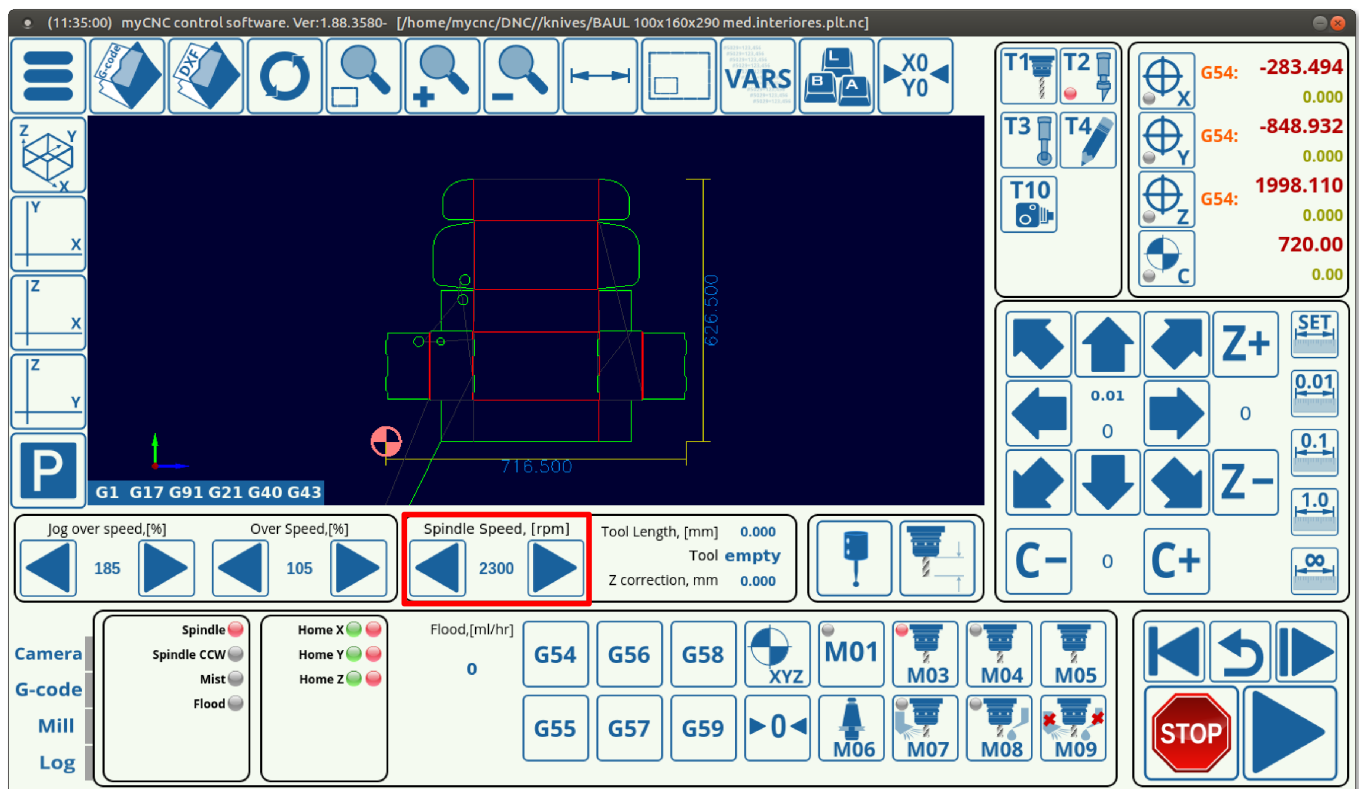
    gvarset(7370,1);//Spindle State
    timer=30;do{timer--;}while (timer>0); //
    gvarset(7371,eparam);//Spindle Speed Mirror register
    timer=30;do{timer--;}while (timer>0); //
```

```
//gvarset(7372,0);//Mist State
//timer=30;do{timer--;}while (timer>0); //
//gvarset(7373,0);//Flood State
//timer=30;do{timer--;}while (timer>0); //

//delay after spindle started
timer=spindle_on_delay;
do{timer--;}while (timer>0); //delay for Spindle reach given speed

exit(99);    //normal exit
};
```

The eparam value here is taken from the spindle speed indicator on the main screen of the myCNC X1366 series profiles:



This value is taken from the field above and converted to a necessary output value using the settings provided in the Settings > Config > Technology > Mill/Lathe > Spindle configuration dialog:

myCNC control software. Ver:1.88.3580- [I:/home/mycnc/DNC//knives/BAUL 100x160x290 med.interiores.plt.nc]

CNC Settings

- Gantry Alignment
- Back to Path
- Surface Measure
- Tool Length Measure
- Tool Change
- Probing Wizard
- Preferences
- Screen
- Work Offsets
- Parking Coordinates
- Technology
 - Plasma Cutting
 - Gas/Oxyfuel
 - Cutcharts
 - THC
 - Mill/Lathe
 - Spindle**
 - Tools
 - ATC Pots
 - Lathe
 - Multi Head
 - Laser control
 - Tangential Knife
 - Special Purpose
- Camera
- 5 axes RTCP
- Panel/Pendant
- Hardware
- Advanced
 - Profile
 - Debug
 - UI Settings

Spindle Speed, [rpm] (Min, Max, Step) 100 24000 100

Spindle Overspeed, [%] (Min, Max, Step) 1 100 1

Encoder channel: Not used

Encoder pulses per revolution: 1

Voltage offset, units:

Voltage ratio, units: 1

RS485/Modbus communication ☒

Speed ratio (modbus):

RS485 speed: 9600

Connection: 8 N 1

Inverter Address ☒ 7 ☒ -1 ☒ -1 ☒ -1

Inverter Modbus address should be 16 or more. Addresses 0...15 reserved for Non-Modbus devices.

Messages: Exceptions:

Write registers

WR/Operate ☒ 8192

WR/Frequency ☒ 8193

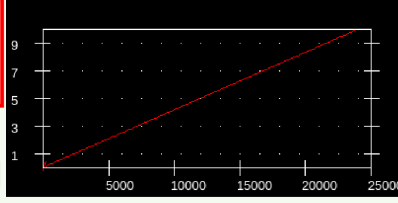
Read registers

RD/Drive Status ☒ 8448

RD/Fault Content ☒ 8449

RD/Frequency reference ☒ 8450

RD/Output frequency ☒ 8451



From:

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

Permanent link:

<http://docs.pv-automation.com/plc/plc/eparam>Last update: **2021/07/02 11:19**