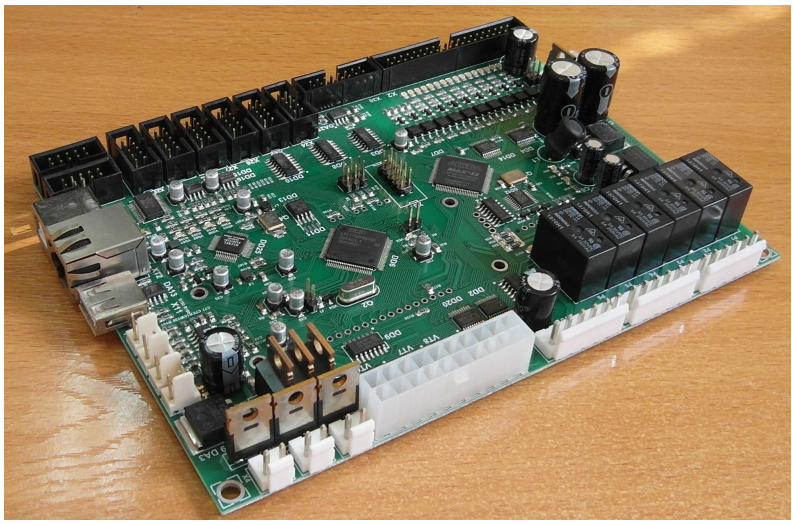
myCNC control & software.

MyCNC-ET1 CNC control board

User manual (rev 0.14)

last edited 2013-11-14



myCNC
http://www.bevelcutting.com
email: sk@bevelcutting.com
4048 Carling Ave PO Box 72074 Kanata North, Kanata ON K2K2P4, Canada

Using This Manual

This user manual provides information for proper installation of the myCNC-ET1 controller and operation of machining centers based on myCNC control software.

WARNING: Machinery in motion can be dangerous! It is the responsibility of the user to design effective error handling and safety protection as part of the machinery. myCNC shall not be liable or responsible for any incidental or consequential damages

Table of content

1 Introduction

The myCNC-ET1 is “all-in-one” 6 axes motion controller and PLC controller. The controller communicates with HMI software via the LAN/Ethernet. Performance capability of the controller includes:

- 6 channels STEP/DIR outputs with maximum pulse frequency is 3MHz;
- Motion controller processing time is 82us ;
- Virtual machine processing time for integrated PLC controller and peripheral units (relay outputs, opto-isolated inputs, PWM outputs, DAC output, ADC inputs) is 1ms;
- 1 Mbytes flash memory for PLC program, motion program and parameters storage for maximum flexibility (more than 100 000 write cycles).

myCNC-ET1 controller drives up tp 6 motor drivers (stepper or servo which accepts step/dir signals as inputs). Modes of motion include jogging, point-to-point positioning and contouring. Stand-alone working mode (offline working without computer) is available. Electronic gearing is available via myCNC control software. Several motion parameters can be specified including acceleration and deceleration rates.

For synchronization with outside events, the myCNC-ET1 control board provides reach peripherals:

- 16 opto-isolated inputs;
- 6 relay outputs;
- 9 transistor key (open collector) outputs (24V, 100mA max)
- 3 Power PWM outputs (24V DC, 3 Amps max);
- 4 ADC inputs (0...5V);
- 1 DAC output (0...10V);
- 2 channel RS485 (one of them is used for THC module connection);

Committed digital inputs can be configured as abort, jog, start and home events or can be flexibly used while machining process via PLC controller.

For communication between controller and HMI software specially designed binary half-duplex master-slave protocol is used.

To prevent system damage during machine operation, the myCNC software programming interface and myCNC-ET1 controller provide many error handling features. These include software and hardware limits, automatic shut-off on excessive error, user-definable abort input.

2 MyCNC-ET1 Functional units

The myCNC-ET1 circuit can be divided into the following functional elements:
CPU unit – ARM Cortex M3 based microcomputer (512k Flash, 64k RAM) 100MHz ;

- Main processing unit - All-in-one Motion Controller (MC) and Programmable Logic Controller (PLC);
- Motor interface based on Altera FPGA programmable logic;

- 6 channel Stepper motor drivers;
- 3 channel Power PWM (24V 3A), programmable via PLC;
- 16 opto-isolated digital inputs, programmable via PLC;
- 6 relay outputs, programmable via PLC;
- 9 transistor key (open collector) outputs (24V, 100mA max)
- USB slave with integrated USB-to-serial converter for diagnostic/ programming/ configuring;
- USB Host connector (to connect USB flash stick and run nc-programs directly from flash disks);
- LAN/Ethernet controller for HMI connection;

2.1. Main processing unit section

The main processing unit of the myCNC-ET1 is a 32-bit NXP ARM Cortex-M3 series microcontrollers with 512 kbytes internal Flash memory, 1Mbytes external flash memory and 64 kbytes RAM memory. The RAM provides memory for variable storage and application programs. The external flash memory provides non-volatile storage of variables, motion and PLC programs, and arrays. Internal flash memory contains the myCNC-ET1 firmware. On the project official website there is available up-to-date version of firmware to reflash the board.

2.2. Motor Interface

Altera FPGA programmable logic installed on myCNC-ET1 controllers provides high speed motor STEP/DIR motor interface, PWM control, relays and binary inputs control with integrated digital filtering.

2.3. Communication

For communication myCNC-ET1 controllers with HMI (myCNC software) is used LAN/Ethernet interface (10base-T Ethernet port).

For communication myCNC-ET1 controller with the Torch Height Controller myTHC-RU01 is used integrated RS485 bus port (communication speed is 115kbaud).

For programming and configuring the device is used USB slave interface with integrated USB-to-serial converter. Communication parameters for connection is 115200-N-8-1

2.4. myCNC peripherals.

myCNC-ET1 controller board contains wide range of peripherals (PWM, binary Inputs, relay and transistor keys outputs, DAC output, ADC inputs). There are available operations that synchronized with motion and asynchronous operations that run immediately.

All peripherals are available via PLC reserved variables or functions.
In a table below there are PLC variable names and peripherals that assigned to it.

PWM, ADC and DAC are unsigned 12-bit values. Complete range for these variables is 0…4095 (in hex 0…0xfff)

Table 1. PLC variables for access to ADC, DAC and PWM

PLC variable	Peripherals
pwm01	PWM1
pwm02	PWM2
pwm03	PWM3
pwm04	Reserved for PWM4
adc01	ADC1
adc02	ADC2
adc03	ADC3
adc04	ADC4
dac01	DAC
dac02	Reserved for DAC2

Digital inputs available via PLC function:

```
getport(int port_number)
//read digital input nr. (port_number+1); return value is 0 if
input pin open, 1 if closed;
```

Digital outputs available via PLC functions:

```
portset(int port_number); //turn on relay nr. (port_number+1)
portclr(int port_number); //turn off relay nr. (port_number+1)
```

Samples:

1. Turn on relay nr.3 and turn off relay nr.1
   ```
   portset(2); // turn on relay output nr.3
   portclr(0); // turn off relay output nr.1
   ```
2. if input pin nr.5 is open turn off PWM channel nr. 1, otherwise turn on it on a half of maximum;
   ```
   if (portget(4)==0)
   pwm=0;
   else
   pwm01=2048;
   ```
3. PWM channel nr. 2 repeats value on ADC channel nr.1
   ```
   pwm02=adc01;
   ```

2.5. System Elements

The myCNC-ET1 is part of a motion control system which includes PC control with installed myCNC software, servo or stepper motor drivers, motors and power supply.

3 myCNC-ET1 connection.

3.1. MyCNC-ET1 outline.

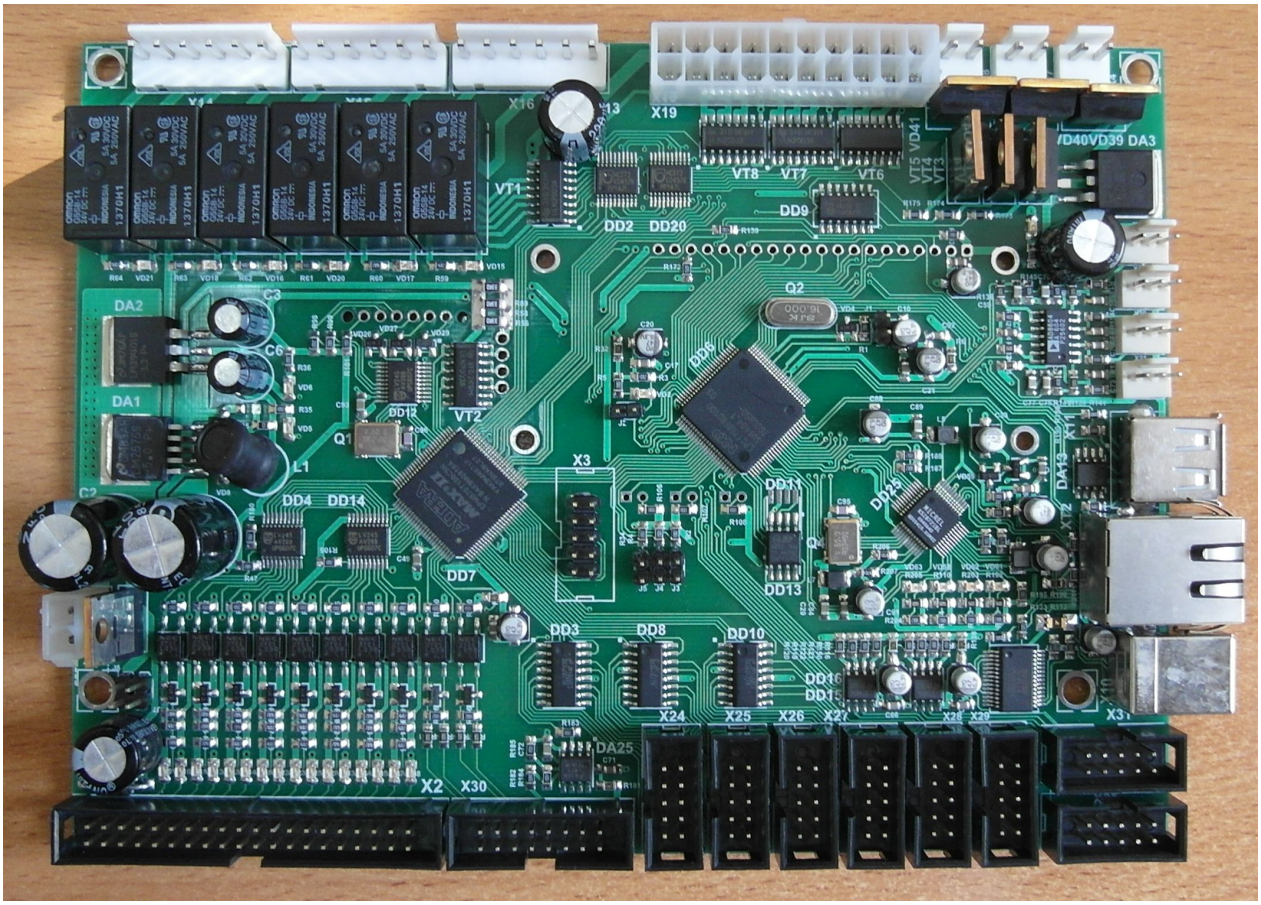


Figure 1. myCNC-ET1 board (revision 2).

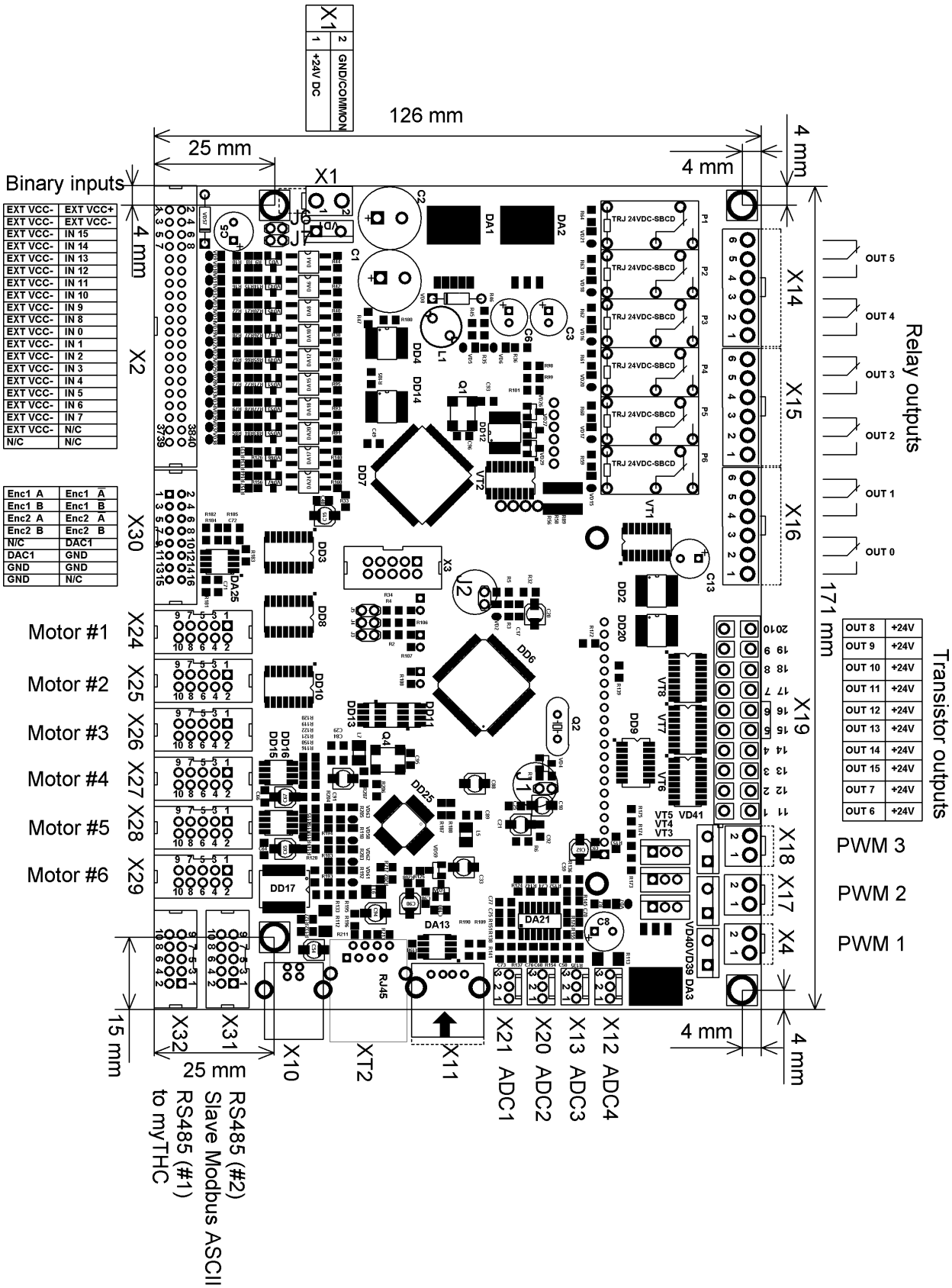


Figure 2. myCNC-ET1 (rev 2) board outline

Table 2. myCNC-ET1 peripherals. Address-to-pin assosiations.

Software name	Connector	Pin numbers	Comments
Relay outputs Выходы			
OUT 0	X16	1-3	
OUT 1	X16	4-6	
OUT 2	X15	1-3	
OUT 3	X15	4-6	
OUT 4	X14	1-3	
OUT 5	X14	4-6	
Transistor outputs			
OUT 6	X19	1	
OUT 7	X19	2	
OUT 8	X19	10	
OUT 9	X19	9	
OUT 10	X19	8	
OUT 11	X19	7	
OUT 12	X19	6	
OUT 13	X19	5	
OUT 14	X19	4	
OUT 15	X19	3	
Binary inputs (optoisolated)			
IN 0	X2	22	
IN 1	X2	24	
IN 2	X2	26	
IN 3	X2	28	
IN 4	X2	30	
IN 5	X2	32	
IN 6	X2	34	
IN 7	X2	36	
IN 8	X2	20	
IN 9	X2	18	
IN 10	X2	16	
IN 11	X2	14	
IN 12	X2	12	
IN 13	X2	10	
IN 14	X2	8	
IN 15	X2	6	
ADC inputs			
ADC 1	X21		
ADC 2	X20		
ADC 3	X13		
ADC 4	X12		
PWM outputs			
PWM 1	X4		
PWM 2	X17		
PWM 3	X18		

3.2. Power supply.

The board 24V DC power supply. Connector X1 is used for control power supply 24V DC connection.

The second connector can be used as external power supply connector.

Table 3. myCNC-ET1. Power supply. X1 pin description.

Pin nr.	Description
1	24V DC
2	GND / COMMON

3.3. Motor Interface.

myCNC-ET1 board contains 6 connectors for connection motor drivers – X24- X29.
Step/Dir motor interface is made with 5V linear driver chip DS34C87 (RS422 compatible).

X24 for motor 1;
X25 for motor 2;
…
X29 for motor 6;
Pin connection is described it a table below.

Table 4. myCNC-ET1 motor interface. Connectors X24-X29 pin description

Pin Nr.	Pin description
1	GND/COMMON
2	GND/COMMON
3	GND/COMMON
4	GND/COMMON
5	STEP
6	~STEP
7	DIR
8	~DIR
9	Not connected
10	Not connected

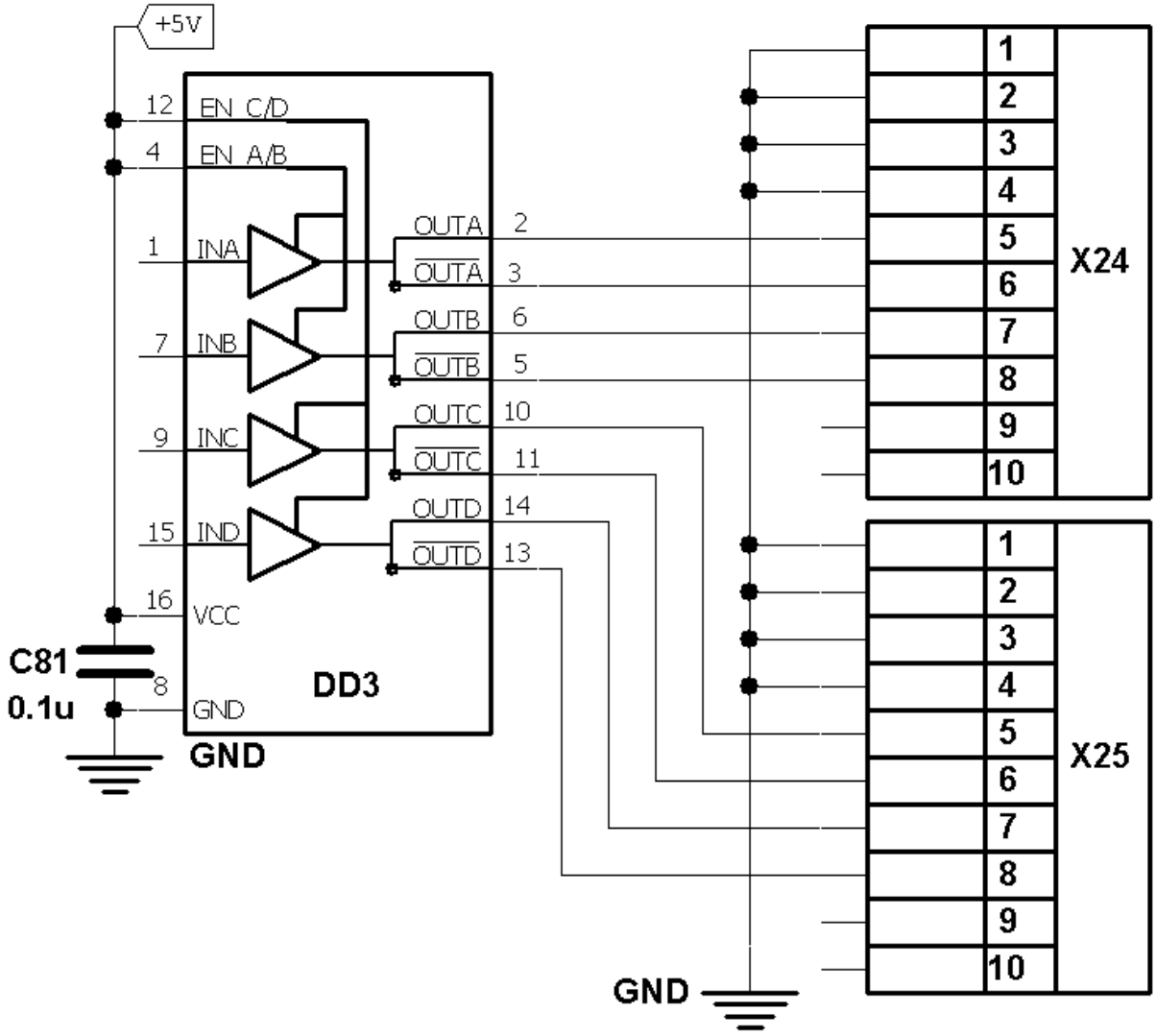


Figure 3. STEP/DIR interface schematic design for myCNC-ET1 control board.

3.4. Opto-isolated inputs.

The myCNC-ET1 contains 16 opto-isolated inputs for connection any type sensors, keys, switchers. Schematic design is shown on the figure below.

For LED power supply can be used internal +24V DC power supply or external power supply 12…24V DC (For complete digital input optical isolation).

Configuration for internal/external power supply is shown on a table below.

Table 5. Settings for digital inputs power supply.

External power supply 12…24V DC for digital inputs	J6 & J7 are open;
Internal power supply 24V DC for digital inputs	J6 & J7 are closed

Table 6. X2 connector – binary inputs – pins description.

	Description
1,2	Ext VCC + (+24V DC if J6 closed)
3,4	Ext VCC- (GND/COMMON if J7 closed)
5	Ext VCC- (GND/COMMON if J7 closed)
6	Input 15
7	Ext VCC- (GND/COMMON if J7 closed)
8	Input 14
9	Ext VCC- (GND/COMMON if J7 closed)
10	Input 13
11	Ext VCC- (GND/COMMON if J7 closed)
12	Input 12
13	Ext VCC- (GND/COMMON if J7 closed)
14	Input 11
15	Ext VCC- (GND/COMMON if J7 closed)
16	Input 10
17	Ext VCC- (GND/COMMON if J7 closed)
18	Input 9
19	Ext VCC- (GND/COMMON if J7 closed)
20	Input 8
21	Ext VCC- (GND/COMMON if J7 closed)
22	Input 0
23	Ext VCC- (GND/COMMON if J7 closed)
24	Input 1
25	Ext VCC- (GND/COMMON if J7 closed)
26	Input 2
27	Ext VCC- (GND/COMMON if J7 closed)
28	Input 3
29	Ext VCC- (GND/COMMON if J7 closed)
30	Input 4
31	Ext VCC- (GND/COMMON if J7 closed)
32	Input 5
33	Ext VCC- (GND/COMMON if J7 closed)
34	Input 6
35	Ext VCC- (GND/COMMON if J7 closed)
36	Input 7
37, 38	Not connected
39, 40	Not connected

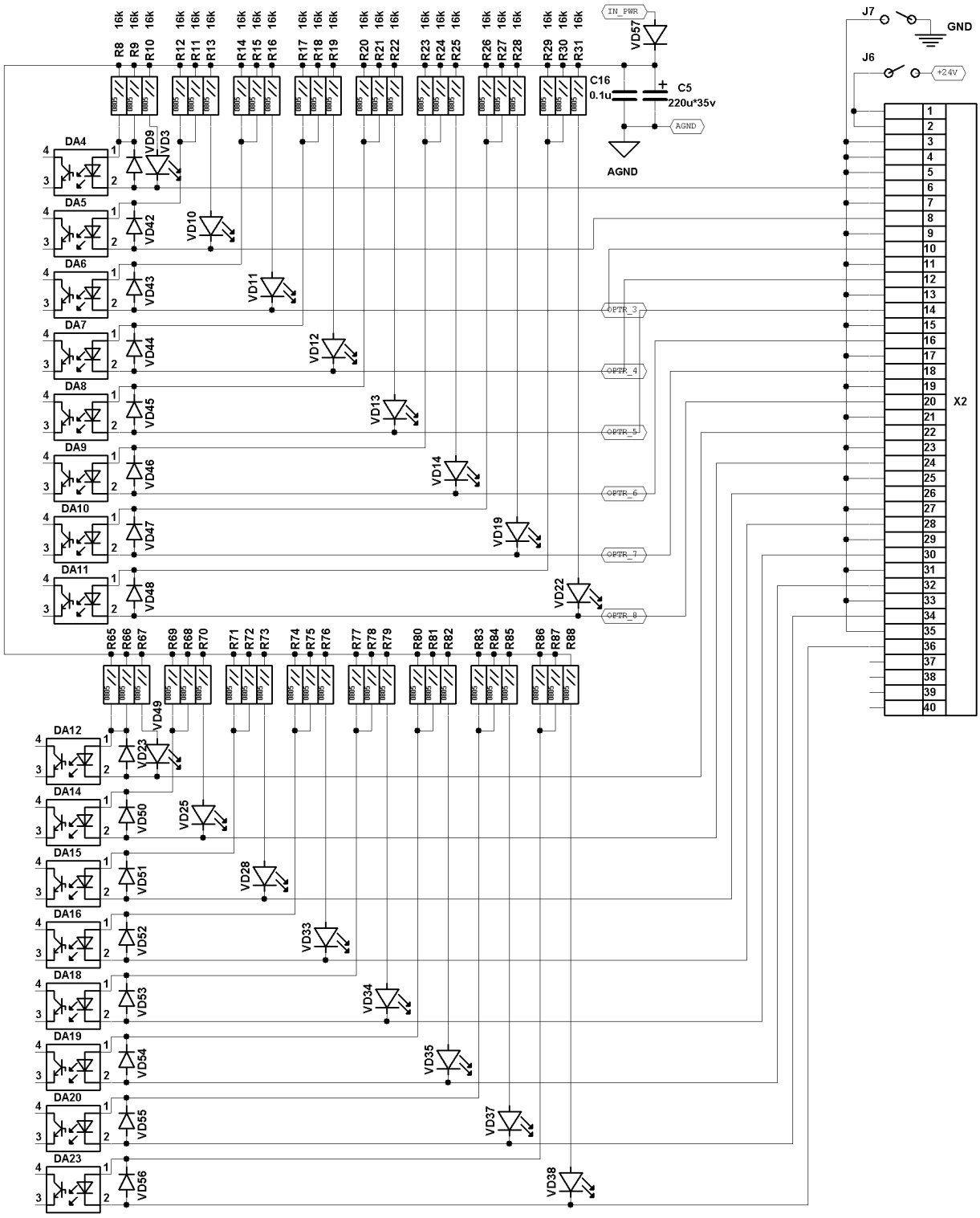


Figure 4. Opto-isolated inputs schematic design (myCNC-ET1).

3.5. PWM outputs, DAC (Spindle speed control) output.

The myCNC-ET1 board contains 3 channel PWM power outputs.
PWM outputs are “open drain”. Internal power supply +24V DC is used for output transistor keys. Connectors X4, X17, X18 are used for PWM outputs.

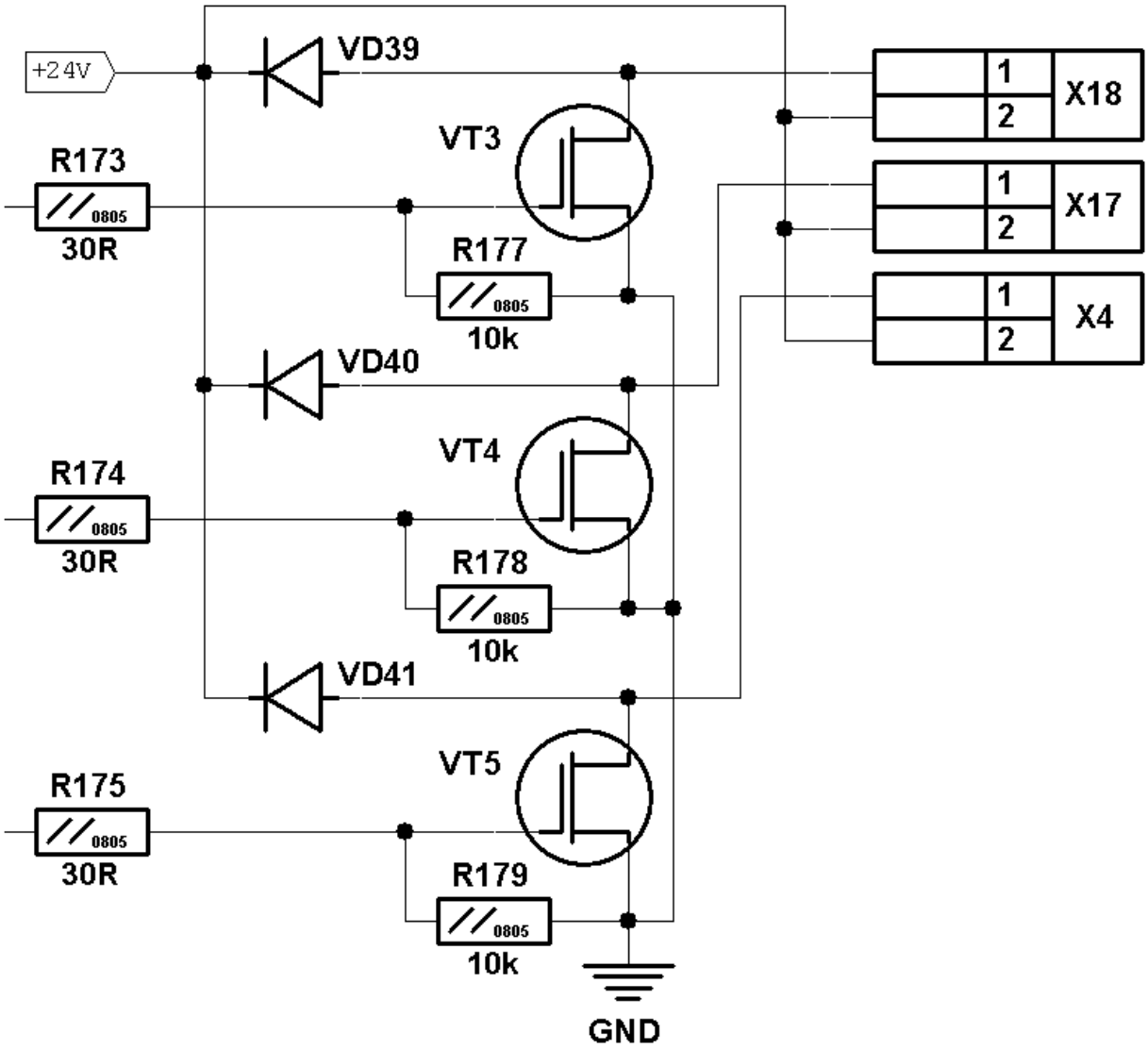


Figure 5. Power PWM outputs schematic design.

Table 7. PWM outputs. X4, X17, X18 connectors pin description.

Pin nr.	Description
1	PWM x
2	+24V DC

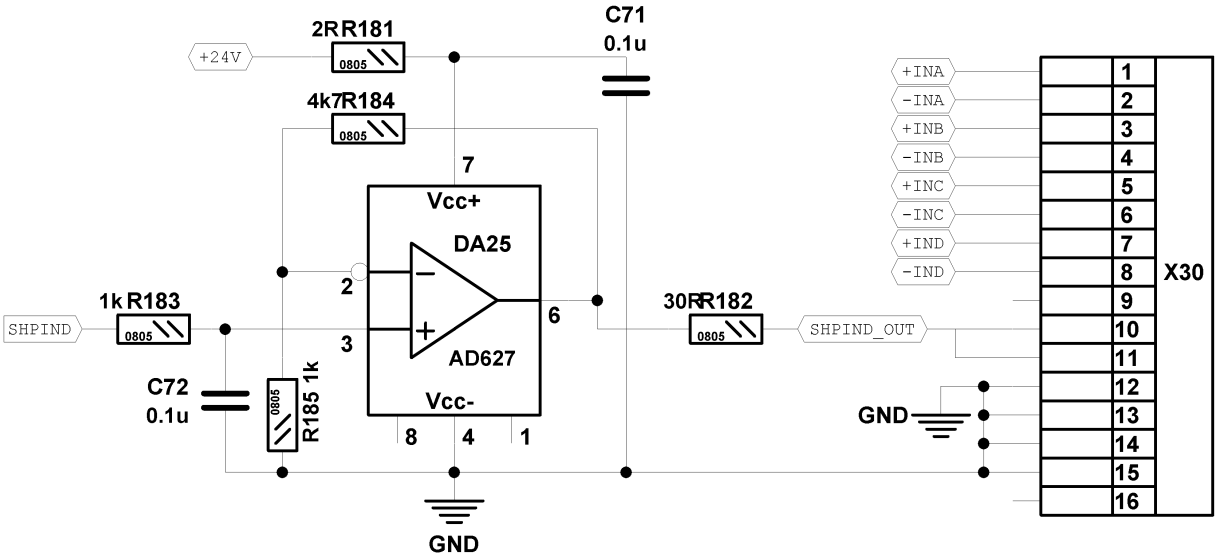


Figure 6. DAC output schematic design.

Table 8. DAC output, X30 connector pinout.

Pin nr.	Description
9, 10	DAC output (0-10V)
12, 13, 14, 15, 16	GND

3.6. Encoder inputs.

MyCNC-ET1 controller contains 4 extra inputs, which can be programmed as

- 2x decoder of incremental encoder (may be used as pendant hand wheel decoder);
- 4 extra binary inputs (general purpose inputs);
- pulse width meter (special purpose);
- frequency meter (height sensing input for sensors with frequency output).

Extra inputs behaviour depends on the board FPGA logic programming firmware and can't be reprogrammed by customer. Please contact with as to define features you need for this pins.

Table 9. Encoder inputs (additional binary inputs), X30 connector pinout.

Pin #	Description	Software Input #
1	+IN A (Encoder 1, A channel, positive input)	Input #19
2	-IN A (Encoder 1, A channel, negative input)	
3	+IN B (Encoder 1, B channel, positive input)	Input #18
4	-IN B (Encoder 1, B channel, negative input)	
5	+IN C (Encoder 2, A channel, positive input)	Input #17
6	-IN C (Encoder 2, A channel, negative input)	
7	+IN D (Encoder 2, B channel, positive input)	Input #16
8	-IN D (Encoder 2, B channel, negative input)	

Handwheel encoder connection to myCNC-ET1 control board shown on a table below.

Table 10. Handwheel encoder connection to ET1 board.

Handwheel Pin	X30 pin #	Description	Comments
-	1	+5V DC	
A	2	-IN A	Input #19
-	3	+5V DC	
B	4	-IN B	Input #18
0V	15,16	GND	
VCC	-	+5V DC	

Encoder inputs schematic shown on a picture below.

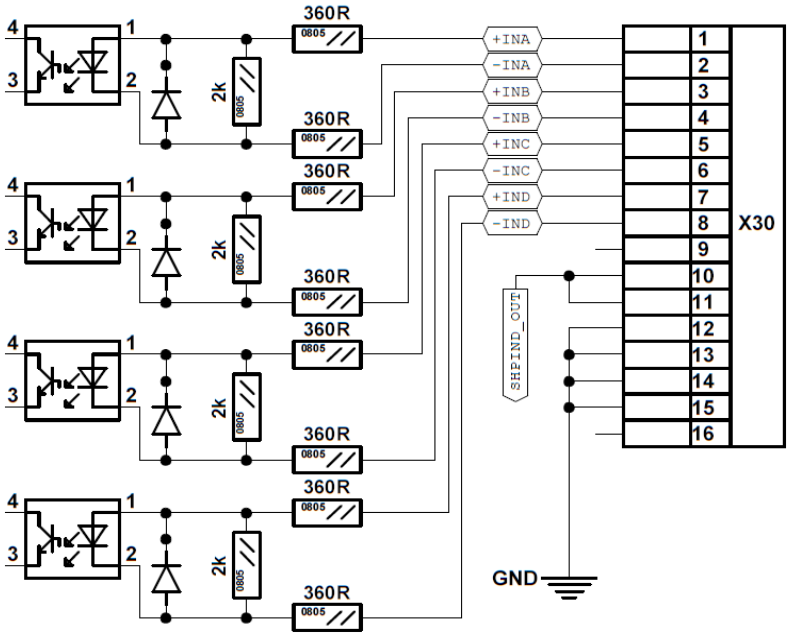


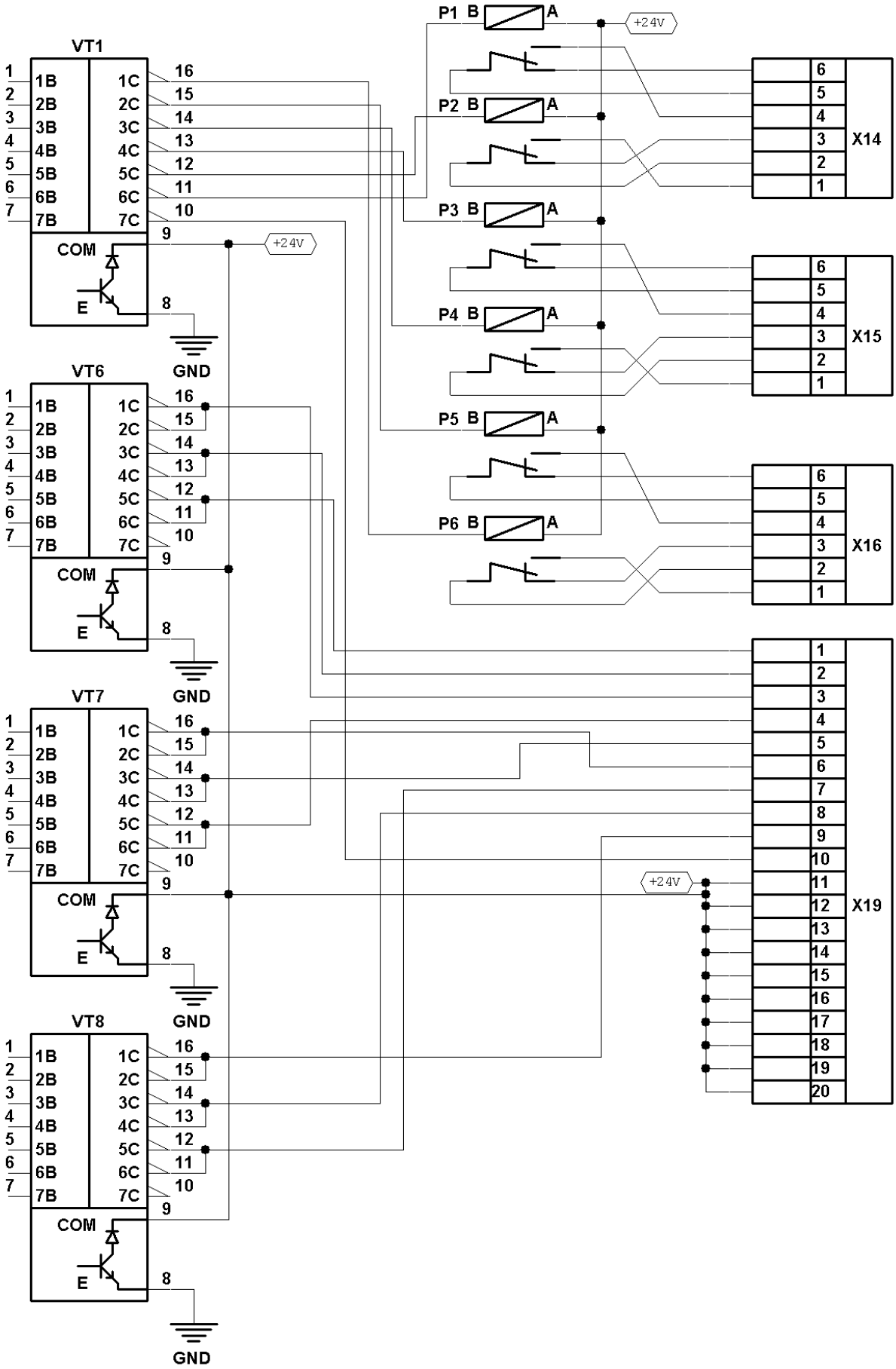
Figure 7. Encoder inputs schematic design.

3.7. Relays and transistors outputs.

The myCNC-ET1 board contains 6 relays (with normally open & closed contacts available) and 9 transistor key outputs (open collector). Connectors X14-X16 are used for relays connection and connector X19 is used for open collector outputs.

Table 11. X19 connector pin description – transistor open collector outputs.

Pin nr.	Description
1	Output 6
2	Output 7
3	Output 15
4	Output 14
5	Output 13
6	Output 12
7	Output 11
8	Output 10
9	Output 9
10	Output 8
11-20	+24V DC



Schematic of relays outputs and connectors is shown on a figure below.

Figure 8. Relay outputs & open collector outputs schematic design.

3.8. RS485 bus connectors.

ET1 control board contains 2 RS485 bus connectors. RS485 might be configured as Modbus-ASCII server, IPG fiber laser interface, THC board interface.

Connectors X31, X32 are used for RS485. X32 connector schematic is shown on a picture below. X31 connectors has similar pinouts.

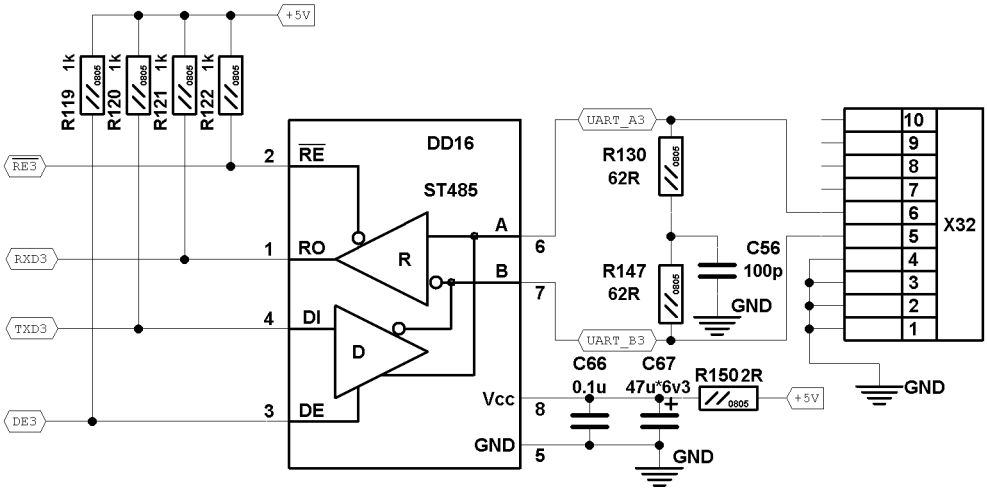


Figure 9. RS485 connectors (X31, X32) schematic design.

Pin description for X31, X32 connector is shown below.

Table 12. RS485 X31, X32 pin description.

Pin nr.	Description
1, 2, 3, 4	GND/COMMON
5	RS485 - B
6	RS485 - A
7, 8	Not used
9, 10	Not used (+5V DC optional)

3.9. ADC inputs.

MyCNC-ET1 control board contains 4 Analog-to-Digital converter (ADC) inputs. ADC input voltage range is 0...5V.

Connectors X12,X13, X20, X21 are used for connection ADC inputs to ET1 control board. Pin connection described in a table below. Connectors contains DC power supply output pin, so external active sensors can be connected & powered from myCNC-ET1 control board. Depends on DA3 - voltage regulator chip installed on myCNC-ET1 board output supply voltage can be 5V (7805 chip) or 12V (7812 chip).

Voltage rgulator chip DA3 shown on a picture below.

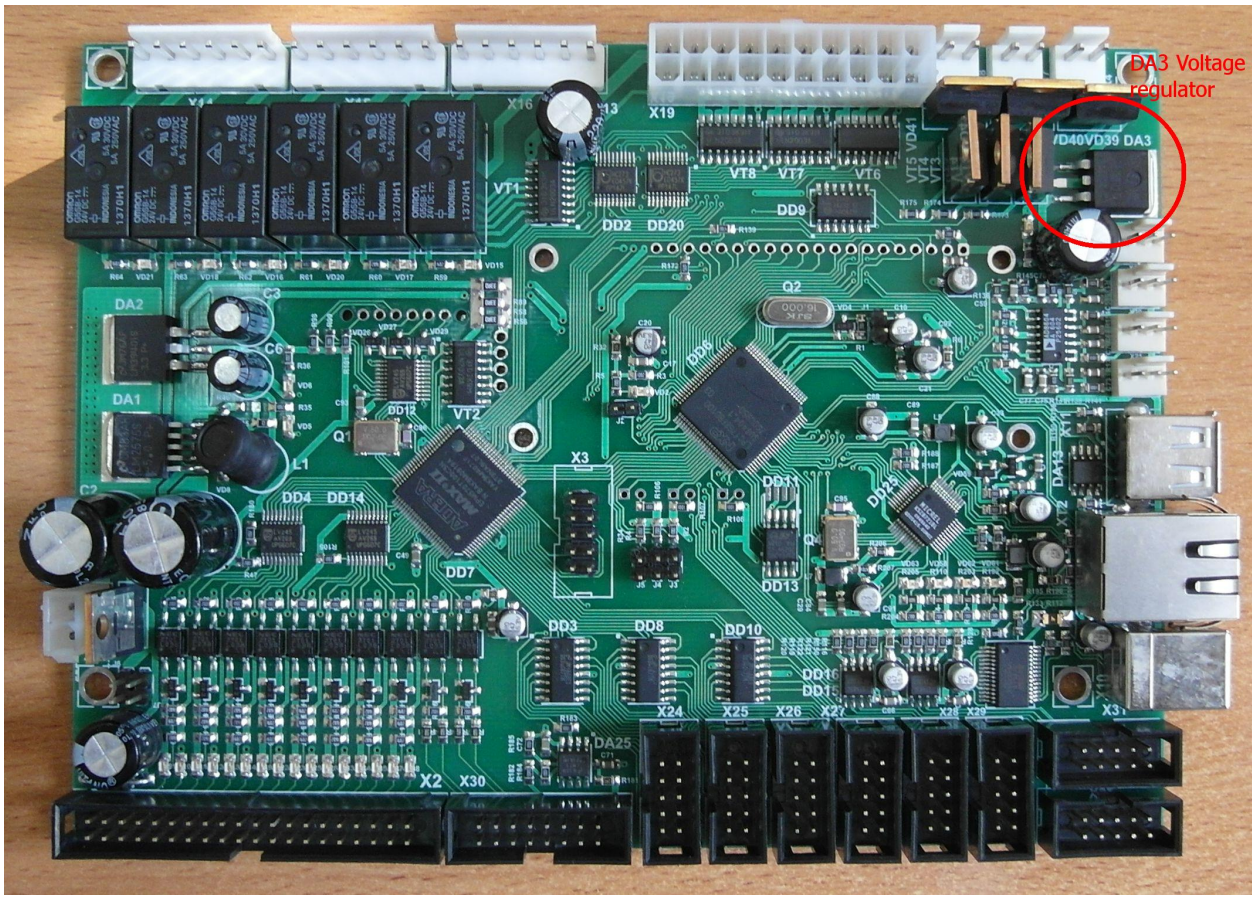


Figure 10. Voltage regulator chip for X12, X13, X20, X21 output power supply.

Table 13. ADC inputs. X12, X13, X20,X21 connectors pin description.

Pin nr.	Description
1	+12V (+5V) DC supply output
2	ADC input
3	GND (Common)

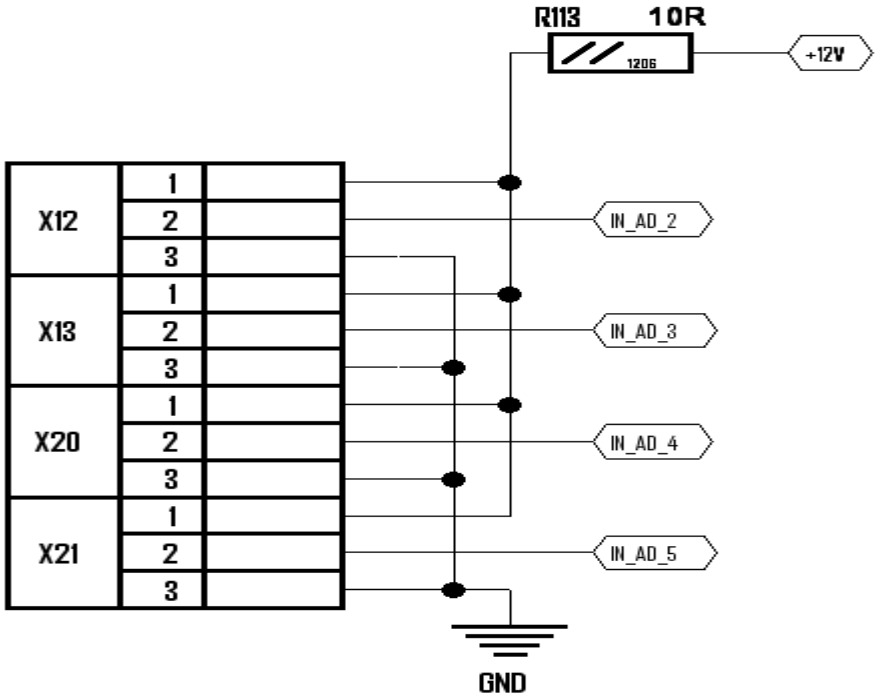


Figure 11. ADC inputs pinout schematic.

Typical using of ADC inputs are:

- Joystick connection for Jog motion;
- Pressure sensors connection for automatic gas console;
- Height sensing control;
- Analogue end switchers;

3.10. Connection to the control PC.

MyCNC-ET1 control board is connected to PC HMI through LAN/Ethernet connector XT2. Standard Ethernet cable can be used for connection. The control board may be connected either directly to the control PC or through Ethernet Switch/HUB to Local Network.

For reflashing, configuration and siagnistic is used USB slave connector X10.

3.11. USB flash stick connection.

MyCNC-ET1 control board contains USB Host controller on-board. This intefrace is used for USB flash disk drives connection to the controller. The controller software accept FAT16/FAT32 file systems on USB flash. For stand-alone mode ET1 control may load and run nc-programs directly from USB disks without control PC.

3.12. MyCNC-ET1 connection examples.

3.12.1 Motor interface connection examples.

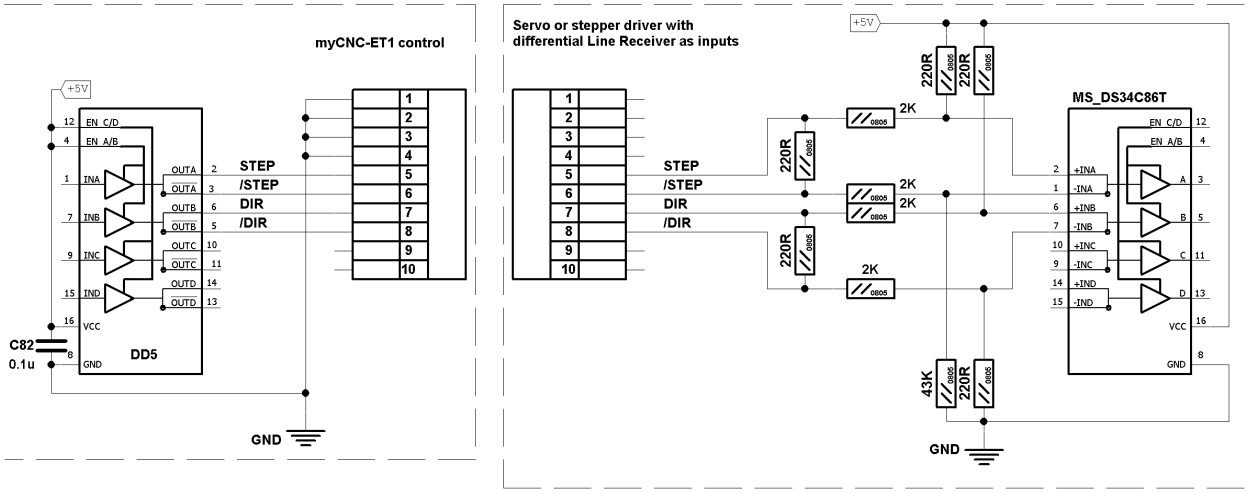


Figure 12. Connection servo or stepper driver with differential line driver as inputs.

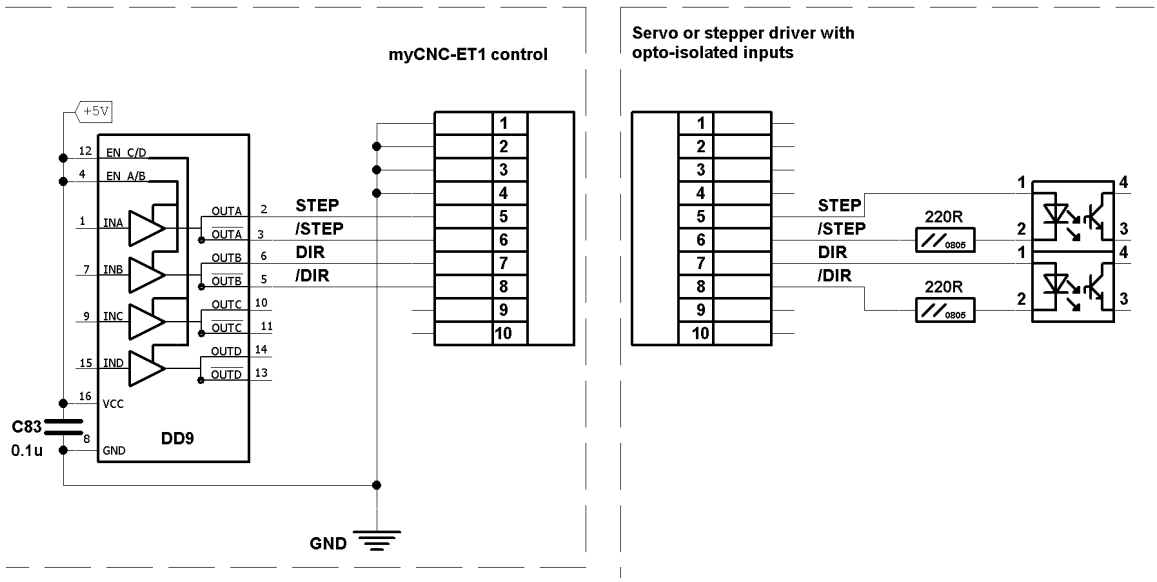


Figure 13. Connection servo or stepper driver with opto-isolated inputs.

4 Configuration and diagnostic.

4.1. Connection the myCNC-ET1 controller to diagnostic channel.

For programming, reflashing of configuring the ET1 controller should be connected to PC through USB-slave connector.

USB-to-serial converter chip FT232RL is installed on the ET1 controller for serial communication with Host PC. Drivers for this converter should be installed on the Host. Driver for Linux OS is called «ftdi_sio» and included into the Linux Kernel.

For Windows machines the drivers can be installed from myCNC folder:
myCNC/driver.usb-to-serial

or directly from the manufacturers website:
http://www.http://www.ftdichip.com/Drivers/D2XX.htm

For communication with myServo-A01 servodriver terminal software like
hyperterminal for MS Windows
or
minicom for Linux, can be used.

Serial connection is used with parameters 115200 8N1.

4.2. PC to driver command format.

Command to the driver is text line that begins with symbol "#" and ends with <CR/LF>. Text line is case sensitive. There are available Set, Print and Debug commands.

4.3. Set commands.

Table 14. Set LAN/Ethernet commands (myCNC-ET1 controller).

Command format.	Command description.
#SLE {A}<CR/LF> Possible value for {A} is 0,1	[S]et [L]AN [E]nable – turn on/off LAN/Ethernet interface. Automatically save settings in flash memory. Change will take effect after restart the driver. {A} – "1" LAN enable {A} – "0" LAN disable Examples: #SLE 0 turns LAN interface off; #SLE 1 turns LAN interface on
#SLA {A} {B} {C} {D}<CR/LF> Possible value for {ABCD} are 0...255	[S]et [L]AN [A]ddress – set LAN/Ethernet address of the board as given values ABCD: IP Addr ={A}.{B}.{C}.{D}

	Automatically save settings in flash memory. Change will take effect after restart the driver. Example: #SLA 192 168 4 78<CR/LF> sets IP address to 192.168.4.78
#SLG {A} {B} {C} {D}<CR/LF> Possible value for {ABCD} are 0...255	[S]et [L]AN [G]ateway – set LAN/Ethernet gateway as given values ABCD: GW Addr ={A}.{B}.{C}.{D} Automatically save settings in flash memory. Change will take effect after restart the driver. Example: #SLG 192 168 4 1<CR/LF> sets GW address to 192.168.4.1

Table 15. Set PID regulator commands (myCNC-ET1 controller).

Command format.	Command description.
#SP{A}P {B}<CR/LF> Possible value for {A} is 0...2 Possible value for {B} is 0...32000	[S]et [P]id regulator number {A} [P]roportional coeff to given value {B}. Change takes effect immediately. New value is NOT stored in the flash memory. Special command need to store settings in the flash. Examples: #SP1P 300 set PID number 1, Proportional coefficient to value 300
#SP{A}I {B}<CR/LF> Possible value for {A} is 0..2 Possible value for {B} is 0...32000	[S]et [P]id regulator number {A} [I]ntegral coeff to given value {B}. Change takes effect immediately. New value is NOT stored in the flash memory. Special command need to store settings in the flash. Examples: #SP2I 25 set PID number 2, Integral coefficient to value 25
#SP{A}0 {B}<CR/LF> Possible value for {A} is 0..2 Possible value for {B} is 0...32000	[S]et [P]id regulator number {A} K[0] coeff to given value {B}. Change takes effect immediately. New value is NOT stored in the flash memory. Special command need to store settings in the flash. Examples: #SP20 500 set PID number 2, K0 coefficient to value 500
#SP{A}K {B}<CR/LF> Possible value for {A} is 0..2	[S]et [P]id regulator number {A} Linear coefficient [K] to given value {B}. Change takes effect immediately. New value is NOT

Possible value for {B} is 0...32000	stored in the flash memory. Special command need to store settings in the flash. Examples: #SP0K 450 set PID number 0, K linear coefficient to value 25
#SP{A}H {B}<CR/LF> Possible value for {A} is 0..2 Possible value for {B} is 0...32000	[S]et [P]id regulator number {A} [H]ysteresis coeff to given value {B}. Change takes effect immediately. New value is NOT stored in the flash memory. Special command need to store settings in the flash. Examples: #SP2H 40 set PID number 2, Hysteresis coefficient to value 25
#SP{A}< {B}<CR/LF> Possible value for {A} is 0..2 Possible value for {B} is 0...4095	[S]et [P]id regulator number {A} [<] minimum control range to given value {B}. Minimum level of PID Output control will be limited with given value. Change takes effect immediately. New value is NOT stored in the flash memory. Special command need to store settings in the flash. Examples: #SP2< 100 set PID number 2 Output control value minimum range to 100
#SP{A}> {B}<CR/LF> Possible value for {A} is 0..2 Possible value for {B} is 0...4095	[S]et [P]id regulator number {A} [>] maximum Output control range to given value {B}. Maximum level of PID Output control will be limited with given value. Change takes effect immediately. New value is NOT stored in the flash memory. Special command need to store settings in the flash. Examples: #SP0> 3100 set PID number 0 Output control value maximum range to 3100
#SP{A}1 {B}<CR/LF> Possible value for {A} is 0..2 Possible value for {B} is 0...32000	[S]et [P]id regulator number {A} [1] Integral Part minimum range to given value {B}. Minimum of Integral Part for given PID regulator will be limited with given value multiplied to (-1). Change takes effect immediately. New value is NOT stored in the flash memory. Special command need to store settings in the flash. Examples:

	#SP11 2000 set PID number 1 Minimum limit for Integral Part to (-2000)
#SP{A}2 {B}<CR/LF> Possible value for {A} is 0..2 Possible value for {B} is 0...32000	[S]et [P]id regulator number {A} [2] Integral Part maximum range to given value {B}. Maximum of Integral Part for given PID regulator will be limited with given value. Change takes effect immediately. New value is NOT stored in the flash memory. Special command need to store settings in the flash. Examples: #SP12 2500 set PID number 1 Maximum limit for Integral Part to (2500)
#SP{A}X <CR/LF> Possible value for {A} is 0..2	[S]et [P]id regulator number {A} [X] rest of config parameters to default values and save all PID configuration to Flash memory. Change takes effect immediately. Examples: #SP1X set PID number 1 rest of variables to default value and save the Configuration.
#SP{A}W {B} <CR/LF> Possible value for {A} is 0..2 Possible value for {B} is 0..2	[S]et [P]id regulator number {A} all the values from configuration of PID number {B} and save the config to Flash memory. Change takes effect immediately. Examples: #SP1W2 — Load a complete configuration to PID number 1 from the PID number 2 and save the Configuration to Flash memory.
#SP{A}D {B}<CR/LF> Possible value for {A} is 0..2 Possible value for {B} is 0..1000	[S]et [P]id regulator number {A} Debug level to given value {B}. While B=15 Modbus debug is turned on. Examples: #SP1D15 set PID number 1 Debug ON.

Table 16. Print command (myCNC-ET1 controller).

Command format.	Command description.
#PP <CR/LF>	[P]rint [P]ID regulator data. Print current PID coefficients Example: #PP

Table 17. Debug commands (myCNC-ET1 controller).

Command format.	Command description.
#DP {A} <CR/LF> Possible value range for {A} is 0...3	[D]ebug [P]ID regulator level. Turn on PID control diagnostic with given level or turn off it Give value- 0 – turn off diagnostic; 1 – print position error, speed; 2 – print position error, motor voltage, speed; 3 – print position error, speed error, motor voltage, speed; Examples: #DP0 turn off diagnostic #DP1 turn on PID diagnostic with level 1

5 RS485 Modbus interface.

5.1. Introduction.

MyCNC-ET1 control board provides Modbus/ASCII interface for access to Peripherals and Motion controller features of the controller.

Modbus ASCII protocol implemented through RS485 and Ethernet. Physical specification of connection is:
1) for RS485 - 115200-n-8-1
2) for Ethernet — UDP, port 4230

Modbus server service is implementer in ET1 by default. Optional client features can be activated through Ethernet connection with myCNC control software.

Implemented commands listed in tables below.

Table 18. RS485 Modbus IO access commands (myCNC-ET1).

	Description	Value, range	Address
20	Read binary inputs.	16-bit value	0x800
21	Read/Write binary outputs (relay & open collector outputs)	16-bit value	0x804
22	Read ADC values (ADC1... ADC4).	16-bit value (low 12 bit is actual ADC value)	0x808, 0x809, 0x80A, 0x80B
23	Read/Write DAC outputs value (DAC1, DAC2).	16-bit value (low 12 bit is actual avlue)	0x80C, 0x80D
24	Read/Write PWM outputs value(PWM1... PWM4).	16-bit value (low 12 bit is actual avlue)	0x810, 0x811, 0x812, 0x813

Table 19. RS485 Modbus Positioning commands — Motion control access through Modbus (myCNC-ET1).

	Description	Value, range	Address
1	Motion speed set (pulses per sec), 32-bit format : 24.8	32-bit value, (0x814 — hi word, 0x815 — lo word)	0x814, 0x815
2	Acceleration set for motion command (pulses per sec^2), integer value	32-bit value, (0x816 — hi word, 0x817 — lo word)	0x816, 0x817
3	Absolute position/Increment for axis X	32-bit value, (0x820 — hi word, 0x821 — lo word)	0x820, 0x821
4	Absolute position/Increment for axis Y	32-bit value, (0x822 — hi word, 0x823 — lo word)	0x822, 0x823
5	Absolute position/Increment for axis Z	32-bit value, (0x824 — hi word, 0x825 — lo word)	0x824, 0x825
6	Absolute position/Increment for axis A	32-bit value, (0x826 — hi word, 0x827 — lo word)	0x826, 0x827
7	Absolute position/Increment for axis B	32-bit value, (0x828 — hi word, 0x829 — lo word)	0x828, 0x829
8	Absolute position/Increment for axis C	32-bit value, (0x82A — hi word, 0x82B — lo word)	0x82A, 0x82B
9	Start motion in given axes.	16-bit value (6 bit binary fields). «1» in the fiels means moving in the given axis, «0» - ignore increment, no motion. Bit 0 — axis X Bit 1 — axis Y Bit 2 — axis Z Bit 3 — axis A Bit 4 — axis B Bit 5 — axis C Bit 15 - «1» - absolute programming, «0» - incremental programming.	0x830
10	Set pulse/dir configuration	16 bit value 1) bit0...bit7 (8 bit value) - pulse width 0- 0.16us …. 7- 3.5us 2) bit8...bit15 (8 bit value) — pulse format 0,2 — reserved 3 — STEP/DIR 1 — STEP CW/ STEP CCW	0x840
11	Set Hard Limits	16-bit value 1) bit0...bit9 (10 bit value) input port number 2) bit10...bit13 (4 bit value) axis 0 - axis X 1 - axis Y	0x842

		2 - axis Z 3 - axis A 4 - axis B 5 - axis C 6...15 — axes 7...15 (reserved) 3) bit 14 — normally opened/closed state 4) bit 15 — negative/positive limits «0» - limit set for negative direction «1» - limit for positive direction Example: a) Write 0x4001 - set Limit switch for axis X, negative direction, input number 1, opened input triggers the limit. b) Write 0x8402 - set Limit switch for axis Y, positive direction, input number 2, closed input triggers the limit.	
12	Multiplier ratio for Job X coordinate	32-bit value, fixed point format 8.24 (0x850 — hi word, 0x851 — lo word)	0x850, 0x851
13	Multiplier ratio for Job Y coordinate	32-bit value, fixed point format 8.24 (0x852 — hi word, 0x853 — lo word)	0x852, 0x853
14	Multiplier ratio for Job Z coordinate	32-bit value, fixed point format 8.24 (0x854 — hi word, 0x855 — lo word)	0x854, 0x855
15	Multiplier ratio for Job A coordinate	32-bit value, fixed point format 8.24 (0x856 — hi word, 0x857 — lo word)	0x856, 0x857
16	Multiplier ratio for Job B coordinate	32-bit value, fixed point format 8.24 (0x858 — hi word, 0x859 — lo word)	0x858, 0x859
17	Multiplier ratio for Job C coordinate	32-bit value, fixed point format 8.24 (0x85A — hi word, 0x85B — lo word)	0x85A, 0x85B
18	Multiplier ratio for Encoder X coordinate	32-bit value, fixed point format 8.24 (0x850 — hi word, 0x851 — lo word)	0x860, 0x861
19	Multiplier ratio for Encoder Y coordinate	32-bit value, fixed point format 8.24 (0x852 — hi word, 0x853 — lo word)	0x862, 0x863
20	Multiplier ratio for Encoder Z coordinate	32-bit value, fixed point format 8.24 (0x854 — hi word, 0x855 — lo word)	0x864, 0x865
21	Multiplier ratio for Encoder A coordinate	32-bit value, fixed point format 8.24 (0x856 — hi word, 0x857 — lo word)	0x866, 0x867
22	Multiplier ratio for Encoder B coordinate	32-bit value, fixed point format 8.24 (0x858 — hi word, 0x859 — lo word)	0x868, 0x869
23	Multiplier ratio for Encoder C coordinate	32-bit value, fixed point format 8.24 (0x85A — hi word, 0x85B — lo word)	0x86A, 0x86B
24	Read current Position X	32-bit value (0x880 — hi word, 0x881 — lo word)	0x880, 0x881
25	Read current Position Y	32-bit value (0x882 — hi word, 0x883 — lo word)	0x882, 0x883

26	Read current Position Z	32-bit value (0x884 — hi word, 0x885 — lo word)	0x884, 0x885
27	Read current Position A	32-bit value (0x886 — hi word, 0x887 — lo word)	0x886, 0x887
28	Read current Position B	32-bit value (0x888 — hi word, 0x889 — lo word)	0x888, 0x889
29	Read current Position C	32-bit value (0x88A — hi word, 0x88B — lo word)	0x88A, 0x88B
30	Read current motion code	8-bit value reflects Motion control current code. Codes list : 3 — Idle (wait) 18 — Linear interpolation 19 — Arc/Spiral interpolation 28 — Positioning command 109/'m' — Manual/Jog motion 116/'t' — PLC procedure running	0x900
31	Stop running program	Stop program	0x901
32	Run Homing procedure	Runs PLC procedure «HOME»	0x902
33	Reset Work/Machine coordinates	Reset work/machine coordinates. 8-bit bit field. «1» tells reset coordinates in thi axis, «0» - leave unchanged. Bit 0 — reset X coordinate; Bit 1 — reset Y coordinate; Bit 2 — reset Z coordinate; Bit 3 — reset Z coordinate;	0x903
34	Set Virtual Machine Variable #0 (var00)	32-bit value (0x980 — hi word, 0x981 — lo word)	0x980, 0x981
35	Set Virtual Machine Variable #1 (var01)	32-bit value (0x982 — hi word, 0x983 — lo word)	0x982, 0x983
36	Set Virtual Machine Variable #2 (var02)	32-bit value (0x984 — hi word, 0x985 — lo word)	0x984, 0x985
37	Set Virtual Machine Variable #3 (var03)	32-bit value (0x986 — hi word, 0x987 — lo word)	0x986, 0x987
38	Set Virtual Machine Variable #4 (var04)	32-bit value (0x988 — hi word, 0x989 — lo word)	0x988, 0x989
39	Set Virtual Machine Variable #5 (var05)	32-bit value (0x98A — hi word, 0x98B — lo word)	0x98A, 0x98B
40	Set Virtual Machine Variable #6 (var06)	32-bit value (0x98C — hi word, 0x98D — lo word)	0x98C, 0x98D
41	Set Virtual Machine Variable #7 (var07)	32-bit value (0x98E — hi word, 0x98F — lo word)	0x98E, 0x98F
42	Set Virtual Machine Variable #8 (var08)	32-bit value (0x990 — hi word, 0x991 — lo word)	0x990, 0x991
43	Set Virtual Machine	32-bit value	0x992,

	Variable #9 (var09)	(0x992 — hi word, 0x993 — lo word)	0x993
44	Set Virtual Machine Variable #10 (var10)	32-bit value (0x994 — hi word, 0x995 — lo word)	0x994, 0x995
45	Set Virtual Machine Variable #11 (var11)	32-bit value (0x996 — hi word, 0x997 — lo word)	0x996, 0x997
46	Set Virtual Machine Variable #12 (var12)	32-bit value (0x998 — hi word, 0x999 — lo word)	0x998, 0x999
36	Set Virtual Machine Variable #13 (var13)	32-bit value (0x99A — hi word, 0x99B — lo word)	0x99A, 0x99B
47	Set Virtual Machine Variable #14 (var14)	32-bit value (0x99C — hi word, 0x99D — lo word)	0x99C, 0x99D
48	Set Virtual Machine Variable #15 (var15)	32-bit value (0x99E — hi word, 0x99F — lo word)	0x99E, 0x99F